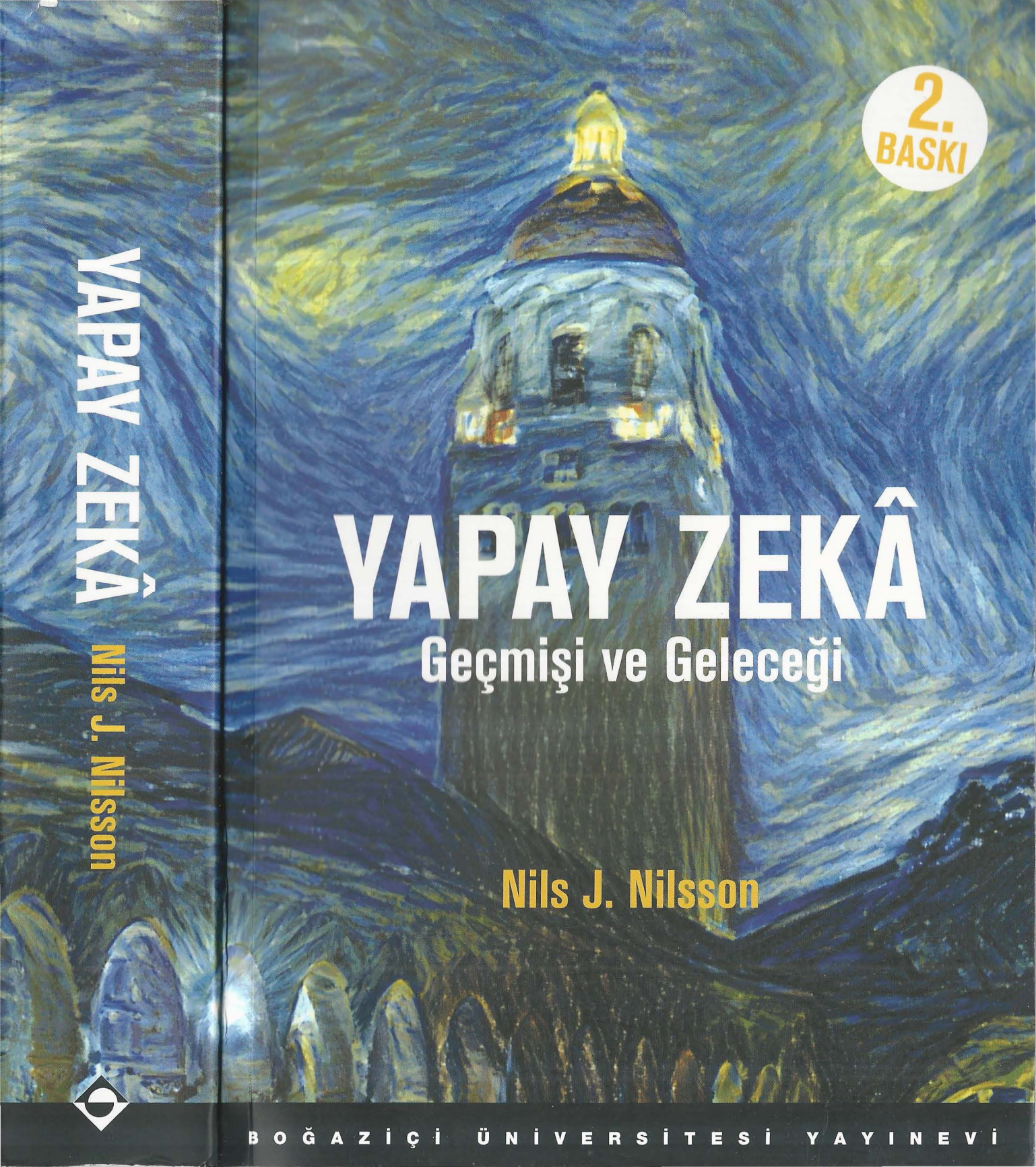

2. BASKI
YAPAY ZEKÂ
Geçmişi ve Geleceği
Nils J. Nilsson
YAPAY ZEKÂ
Nils J. Nilsson
BOĞAZİÇİ ÜNİVERSİTESİ YAYINEVİ

YAPAY ZEKÂ

Mehmet Doğan 1976 Bursa doğumlu. ODTÜ Biyoloji mezunu. NYU Biyoloji'de master yaptı. Serbet çevirmenlik yapıyor. Başlıca çevirileri: *Genom*, (Matt Ridley) 2007, *Gen Çeviktir*, (Matt Ridley) 2009; *Haklı Savaş Haksız Savaş* (Michael Walzer), 2010; *Neo-Liberal Genetik* (Susan McKinnon), 2010. *Boş Sayfa* (Steven Pinker) 2010. *Çift Görmek* (Peter Pesic) 2010. e-posta: memeddogan@gmail.com

YAPAY ZEKÂ

GEÇMİŞİ VE GELECEĞİ

NILS J. NILSSON

Çeviren: Mehmet Doğan

BOĞAZİÇİ
ÜNİVERSİTESİ
YAYINEVİ

Nils J. Nilsson
The Quest for Artificial Intelligence
A History of Ideas and Achievements
© Nils J. Nilsson, 2010.

Yapay Zekâ
Geçmişi ve Geleceği

© BÜTEK A.Ş. 2011. Tüm hakları saklıdır.

ISBN 978-605-68425-1-1

Bütek Boğaziçi Eğitim Turizm Teknopark
Uygulama ve Dan. Hiz. San. Tic. A.Ş.
Rumeli Hisarı Mahallesi, Boğaziçi Üniversitesi
Güney Kampüs No: 11/2 Sarıyer/İstanbul
Telefon: (0212) 287 03 12-13

Yönetim Yeri:
Boğaziçi Üniversitesi Yayınevi
Boğaziçi Üniversitesi Kuzey Kampüs
ETA B Blok, 5. Kat, No: 705
Sarıyer/İstanbul

bupress@boun.edu.tr
www.bupress.org
www.bounyayin.com
Telefon ve faks: (0212) 257 87 27
Sertifika No: 10821

Genel Yayın Yönetmeni: Murat Gülsoy
Yönetici Editör: Ergun Kocabıyık
Yayına Hazırlayan: Ethem Alpaydın
Kapak: Kerem Yeğin
Baskı: Acar Basım ve Cilt Sanayi Ticaret A.Ş.
Beysan Sanayi Sitesi, Birlik Cad., No: 26
Acar Binası, Haramidere, İstanbul
Telefon: (0212) 422 18 00
Sertifika No: 11957

Birinci Basım: Temmuz 2018

Bu eserin bütünüyle veya kısmen fotokopisinin çekilmesi, mekanik ya da elektronik araçlarla çoğaltılması, kopyalanarak internette ya da herhangi bir veri saklama cihazında bulundurulması, 5846 Sayılı Fikir ve Sanat Eserleri Kanunu'nun hükümlerine aykırıdır ve hak sahiplerinin maddi ve manevi haklarının çiğnenmesi anlamına geldiği için suç oluşturmaktadır.

Eşim ve en yakın dostum
Grace McConnell Abbott'a ithafen

İÇİNDEKİLER

Önsöz, 13

BİRİNCİ KISIM: **TOMURCUKLAR**

1 Düşler ve Düş Kuranlar, 19

2 İpuçları, 27
- 2.1 Felsefeden ve Mantıktan Gelenler, 27
- 2.2 Yaşamın Kendisinden Gelenler, 34
- 2.3 Mühendislikten Gelenler, 48

İKİNCİ KISIM: **İLK KEŞİFLER** 1950'LER VE 1960'LAR

3 Toplantılar, 75
- 3.1 Öğrenen Makineler Oturumu, 75
- 3.2 Dartmouth Yaz Projesi, 80
- 3.3 Düşünce Süreçlerinin Mekanikleşmesi, 85

4 Örüntü Tanıma, 91
- 4.1 Yazı Karakteri Tanıma, 92
- 4.2 Sinir Ağları, 95
- 4.3 İstatistiki Yöntemler, 106
- 4.4 Havadan Keşif Seferlerinde Örüntü Tanıma Uygulamaları, 109

5 İlk Sezgisel Programlar, 115
- 5.1 Mantık Kuramcısı ve Sezgisel Arama, 115
- 5.2 Geometri Teoremlerini Kanıtlamak, 120
- 5.3 Genel Problem Çözücü, 124
- 5.4 Oyun Oynayan Programlar, 127

6 Anlamsal Gösterimler, 134
- 6.1 Geometrik Benzerlik Problemlerini Çözmek, 134
- 6.2 Bilgi Saklamak ve Soruları Yanıtlamak, 137
- 6.3 Anlamsal Ağlar, 139

7 Doğal Dil İşleme, 143
- 7.1 Dilbilimde Düzeyler, 143
- 7.2 Yapay Çeviri, 148
- 7.3 Soru Yanıtlama, 153

8 1960'ların Altyapısı, 157
- 8.1 Programlama Dilleri, 157
- 8.2 İlk YZ Laboratuvarları, 159
- 8.3 Araştırma Desteği, 163
- 8.4 Tamamen Hazır Olmak ve Yeni Doğrultular, 166

ÜÇÜNCÜ KISIM: **ÇİÇEKLENME**
1960'LARIN ORTASINDAN 1970'LERİN ORTASINA

9 Bilgisayarla Görme, 171
9.1 Biyolojiden Gelen İpuçları, 173
9.2 Yüz Tanıma, 175
9.3 Üç Boyutlu Katı Nesneler İçin Bilgisayarla Görme, 176
10 "El-Göz" Araştırmaları, 190
10.1 MIT'de, 190
10.2 Stanford'da, 192
10.3 Japonya'da, 196
10.4 Edinburgh'un "FREDDY"si, 197
11 Bilgi Gösterimi ve Uslamlama, 200
11.1 Simgesel Mantıkta Tümdengelim, 201
11.2 Durum Kalkülüsü, 205
11.3 Mantık Programlama, 206
11.4 Anlamsal Ağlar, 208
11.5 Komut Dosyaları ve Çerçeveler, 211
12 Hareketli Robotlar, 216
12.1 SRI Robotu Shakey, 216
12.2 Stanford Arabası, 238
13 Doğal Dil İşlemede İlerleme, 240
13.1 Yapay Çeviri, 240
13.2 Anlama, 241
14 Oyun Oynamak, 254
15 Dendral Projesi, 258
16 Konferanslar, Kitaplar, Ödenekler, 263

DÖRDÜNCÜ KISIM: **UYGULAMALAR VE UZMANLAŞMALAR**
1970'LERDEN 1980'LERİN İLK YARISINA

17 Konuşma Tanıma ve Anlama Sistemleri, 271
17.1 Konuşma İşleme, 271
17.2 Konuşma Anlama Çalışma Ekibi, 274
17.3 DARPA Konuşma Anlama Araştırma Programı, 275
17.4 Konuşma Tanımada Sonraki Çalışmalar, 288
18 Danışma Sistemleri, 290
18.1 SRI'nın Bilgisayar Tabanlı Danışmanı, 290
18.2 Uzman Sistemler, 296
19 Sorguları ve Sinyalleri Anlamak, 314
19.1 Düzenek, 314

19.2 Bilgisayar Sistemlerine Doğal Dil Erişimleri, 318
19.3 HASP/SIAP, 326

20 Bilgisayarla Görme İlerleme, 331
20.1 Çizgileri Belirlemenin Ötesi, 331
20.2 Görüntüde Nesnelerin Bulunması, 337
20.3 DARPA'nın Görüntü Anlama Programı, 343

21 Yükseliş Dönemi, 347

BEŞİNCİ KISIM: **"YENİ NESİL" PROJELER**

22 Japonlar Heyacan Yaratıyor, 353
22.1 Beşinci Nesil Bilgisayar Sistemleri Projesi, 353
22.2 Japon Projesinin Kimi Etkileri, 358

23 DARPA'nın Stratejik Hesaplama Programı, 363
23.1 Stratejik Hesaplama Planı, 363
23.2 Ana Projeler, 367
23.3 YZ Teknoloji Tabanı, 375
23.4 Değerlendirme, 381

ALTINCI KISIM: **PERDE ARASI**

24 Hız Tümsekleri, 385
24.1 Çeşitli Gözlemcilerin Görüşleri, 385
24.2 Ölçek Sorunları, 406
24.3 Kabul Edilmiş Yetersizlikler, 413
24.4 "YZ Kışı", 416

25 Anlaşmazlıklar ve Alternatif Paradigmalar, 418
25.1 Mantık Hakkında, 418
25.2 Belirsizlik, 419
25.3 "Yamama Sistem", 421
25.4 Davranışlar Hakkında, 423
25.5 Beyin Tarzı Hesaplama, 429
25.6 Evrimi Taklit Etmek, 433
25.7 YZ'nin Hedeflerini Daraltmak, 437

YEDİNCİ KISIM: **BÜYÜYEN CEPHANELİK**
1980'DEN İTİBAREN

26 Uslamlama ve Gösterim, 441
26.1 Tekdüze Olmayan ya da Feshedilebilir Uslamlama, 441
26.2 Nitel Uslamlama, 445
26.3 Anlamsal Ağlar, 448

27 Uslamlamaya ve Gösterime Yönelik Başka Yaklaşımlar, 461
27.1 Kısıt Sağlama Problemlerini Çözmek, 461
27.2 Önermeli Mantık Kullanarak Problemlerin Çözülmesi, 465
27.3 Metinlerin Vektör Olarak Gösterilmesi, 473
27.4 Gizli Anlamsal Çözümleme, 476

28 Bayes Ağları, 481
28.1 Olasılıkların Ağlarda Gösterimi, 481
28.2 Bayes Ağlarının Otomatik Kurulması, 489
28.3 Olasılıksal İlişki Modelleri, 493
28.4 Zamansal Bayes Ağları, 497

29 Yapay Öğrenme, 501
29.1 Bellek Tabanlı Öğrenme, 502
29.2 Örnek Tabanlı Uslamlama, 505
29.3 Karar Ağaçları, 507
29.4 Sinir Ağları, 516
29.5 Gözetimsiz Öğrenme, 523
29.6 Pekiştirmeli Öğrenme, 526
29.7 Geliştirme, 536

30 Doğal Diller ve Doğal Manzaralar, 541
30.1 Doğal Dil İşleme (DDİ), 541
30.2 Bilgisayarla Görme, 549

31 Zeki Sistem Mimarileri, 571
31.1 Hesaplama Mimarileri, 573
31.2 Bilişsel Mimariler, 590

SEKİZİNCİ KISIM: **MODERN YZ: BUGÜNÜ VE YARINI**

32 Sıradışı Uygulamalar, 603
32.1 Oyunlar, 603
32.2 Robot Sistemleri, 613

33 Yapay Zekâ Her Yerde, 628
33.1 Hanelerimizde YZ, 629
33.2 İleri Sürücü Yardım Sistemleri, 630
33.3 Haritalarda Rota Bulmak, 631
33.4 Belki Bunu da Beğenirsiniz, 632
33.5 Bilgisayar Oyunları, 633

34 Akıllı Aletler, 636
34.1 Tıpta, 636
34.2 Çizelgelemede, 639
34.3 Otomatik Alışverişte, 640
34.4 İş Uygulamalarında, 641
34.5 Dillerin Çevirisinde, 642

34.6 İcatları Otomatikleştirmede, 643
34.7 Yüz Tanımada, 644

35 Arayış Sürüyor, 647
35.1 Laboratuvarlarda, 648
35.2 İnsan Düzeyinde Yapay Zekâya Doğru, 662
35.3 Özetlerken, 676

Dizin, 679

ÖNSÖZ

Yapay zekânın (YZ), herkesin üstünde anlaştığı bir tanımı yok, fakat YZ tarihi hakkında yazan kişinin aklında bir tür tanım olmak zorunda. Yapay zekâ, bence, makinelere zekâ kazandırmaya adanmış etkinliktir; zekâysa, bir varlığın kendi ortamında doğru düzgün ve olan biteni öngörerek işlev görmesini sağlayan niteliktir. Bu tanıma göre pek çok varlık (insanlar, hayvanlar, kimi makineler) zekidir. "Akıllı kamera" benzeri makineler ve pek çok hayvan, varlıkların çeşitli zekâ derecelerine göre sıralandığı geniş yelpazenin ilkel ucunda yer alır. Öteki uçtaysa insanlar var; insanlar mantık yürütebilir, koydukları hedeflere ulaşabilir, bir dili anlayıp o dilde cümle oluşturabilir, duyusal girdileri algılayıp tepki verebilir, matematik teoremlerini ispatlayabilir, zorlu oyunlar oynayabilir, bilgi sentezleyip özetleyebilir, sanat ve müzik eserleri yaratabilir, hatta tarihçe yazabilir. "Doğru düzgün ve öngörüyle işlev görmek", çevre şartlarına bağlı olarak pek çok farklı beceri gerektirdiği için, kendi içinde sürekliliği kesilmeyen birkaç zekâ yelpazemiz var aslında. Bu sebeplerden ötürü, YZ'yi oluşturan unsurlarla ilgili görece cömert bir görüş benimsiyorum. Demek ki bu konu hakkındaki tarihçem, yeri gelince biraz kontrol mühendisliği, biraz elektrik mühendisliği, biraz istatistik, biraz dilbilim, biraz mantık, biraz da bilgisayar bilimi içerecek.

YZ'yle ilgili başka tarihçeler yazıldığı oldu, fakat zaman ilerliyor, tıpkı YZ gibi; dolayısıyla artık yeni bir tarihçe yazmanın vakti geldi. Yapay zekâ arayışında elli yıl boyunca yer aldım; bu elli yıl benim bütün meslek yaşantımı ve bu alanın bütün ömrünü kapsar. Bu arayışın başlangıcından bugününe kadar olan öyküsünü "içeriden birinin" anlatması bana iyi bir fikir gibi geldi.

Aklımda üç tür okur var. Biri, bilimsel konulara ilgi duyan ve belki YZ'yi merak eden zeki ama sade okur. Belki ilkiyle örtüşen öbür okur topluluğu, teknik ya da mesleki alanlarda çalışanlardan meydana geliyor; bunların bir sebeple YZ konusunu öğrenmesi gerekiyor ve bu alan hakkında eksiksiz bir tablo işlerine yarayacaktır: YZ eskiden neredeydi, şimdi nerede, bundan sonra nereye gidiyor olabilir. Bu iki

okur zümresine de, kitapta karmaşık matematik formülleri ve bilgisayar jargonu olmayacağına, bol bol grafik sunacağıma, YZ programlarının ve tekniklerinin nasıl işlediğini berrak bir şekilde açıklamak için elimden geleni yapacağıma söz veriyorum. (Ayrıca, YZ alanında çalışmış insanların fotoğraflarını da koydum. Bu seçki bir parça rasgele oluşturuldu; bu alanda yapılan çalışmalar önem sırasına göre sıralanmadı.)

Üçüncü topluluk, YZ araştırmacılarından, öğrencilerinden, öğretmenlerinden oluşuyor; YZ'nin denediği, işe yaramış ya da yaramamış şeyleri bilmek, ayrıca tarihsel bilgileri ve başka türlü malumatı içeren sağlam bir kaynağa ulaşmak bu kesim için faydalı olacaktır. Bu alanın tarihini bilmek, bu alanda çalışanlar için önemli. Öncelikle, bir zamanlar keşfedilip sonra terk edilmiş olan pek çok fikir, teknolojik imkânların iyileşmiş olmasından ötürü artık yaşam şansı bulabilir. Bu okur topluluğunu düşünerek, metin içi dipnotlarda çeşitli kaynaklar belirttim. Genel okur, bu notları görmezden gelse de bir şey kaybetmiş sayılmaz. Zaten ana metin, ilginç filmlerin, sunumların ve temel bilginin bulunabileceği internet sitelerini içeriyor. (Eğer bu sitelerin linkleri bozulmuşsa, okurlar bu sitelere, http://www.archive.org sitesinde "Wayback Machine" üzerinden erişebilir.)

Elinizdeki kitap, arada yalpalasa da aşağı yukarı belli bir zaman sırası takip ediyor. Anlattığım hikâye, kimi aktörleri ve olayları içermiyor olabilir, ama umuyorum ki hikâyem, YZ'nin başlıca fikirlerini, anlaşmazlıklarını, uygulamalarını, sınırlarını makul biçimde temsil ediyordur. Bu işte yer almış şahsiyetler yerine fikirlere ve bunların hayat buluşuna odaklandım. Bana kalırsa, YZ tarihini anlamak için, YZ programlarının gerçekte nasıl çalıştığı en azından genel anlamıyla kavranmalı.

Eğer YZ, makineleri zekâyla donatmak demekse, o halde neyi makine sayacağız? Pek çok insana göre makine oldukça duygusuz bir varlıktır. Bu sözcük, dönen dişliler, püsküren buhar, zangırdayan çelik aksam görüntüsü getiriyor gözümüzün önüne. Fakat bugünlerde bilgisayar, makinenin ne olabileceği hakkında anlayışımızın kapsamını epey genişletti. İşleyen bir bilgisayar sistemi hem donanım hem de yazılım içerir; bizler de yazılımın bizzat "makine" olduğunu düşünürüz çoğunlukla. Örneğin, "satranç oynayan makine" ya da "öğrenen makine" derken, aslında o işleri yapan programlardan bahsediyoruz. Donanım ile yazılım ayrımı artık bir parça bulanık hale geldi, çünkü günümüzde çoğu bilgisayarın kimi programları, donanım devrelerine yerleştiriliyor.

Bu kitabı yazarken faydalandığım tüm bilgi ve imkânlar pek çok insan, kurum ve araştırmalara ödenek sağlayan kuruluşun desteğine dayanıyor. Öncelikle, babam Alfred Nilsson'la (1907-1991) annem Pauline Glerum Nilsson'u (1910-1998) anmalıyım; sonuçta beni dünyaya onlar getirdi. Bana, bayağılıktan ve bahanelerden tiksinmeyi (Walter), özenli davranmayı (Pauline) öğrettiler, beni övüp yüreklendirdiler (ikisi de); bu özellikler, doğru bir karışım teşkil eder. Stanford Üniversitesi, hem sözcüğünün gerçek anlamıyla hem de mecazen *alma mater*'im (Lat. "besleyip büyüten anne") olmuştur. Önce öğrenci sonra akademisyen (şimdi de emekli profesör) olarak üniversitedeki tüm meslektaşlarımdan, özellikle öğrencilerden faydalanıp bir şeyler öğrendim. SRI International (eskiden Stanford Araştırma Enstitüsü denirdi), YZ öğrenmeme ve "yapmama" yardım eden meslektaşlarıma yuva olmuştur. Merhum Charles A. Rosen'i özellikle anmak istiyorum; 1961 yılında, bu enstitüdeki Öğrenen Makineler Ekibine katılmaya beni ikna etmişti. Defense Advanced Research Projects Agency (DARPA), Office of Naval Research (ONR), Air Force Office of Scientific Research (AFOSR), U.S. Geological Survey (USGS), National Science Foundation (NSF), National Aeronautics and Space Administration (NASA) adlı kuruluşlar, son elli yılda parçası olduğum çeşitli araştırmaları desteklemiştir. Hepsine teşekkürü borç bilirim.

Bu kitap için yaptığım araştırmalarda ve kitabın yazımında bana yardım etmiş olan herkes, ayrıca adı bilinmez olan ve adı o kadar da bilinmez olmayan hakemler... bu önsöze hepinizin ismini tek tek koymadığım için özürlerimle birlikte içten teşekkürümü de kabul edin. İsmini saymam gereken onca insan varken, küçük ama önemli katkılarda bulunmuş birilerinin adını zikretmeyi unutmaktan çekiniyorum. Sizler kim olduğunuzu biliyorsunuz. Adını kitapta andığım pek çok kişi arasındasınız zaten. Bununla birlikte, Cambridge University Press'ten Heather Bergman'ı, eski öğrencim Mykel Kochenderfer'i, Darmstadt Teknoloji Üniversitesi'nden Wolfgang Bibel'i anmak istiyorum. Taslağın bütününün ilk halini özenle okudular ve pek çok faydalı öneri getirdiler. (Üstlelik Mykel, dizgi programı LATEX'le ilgili paha biçilemez önerilerde bulundu.)

Ayrıca, kitabı yazmamı kolaylaştırmış olan internet, dünya çapında ağ (www) ve arama motorlarını icat etmiş, geliştirmiş ve günümüzde çekip çeviren insanlara da teşekkür etmek istiyorum. Stanford'ın çeşitli internet sitelerine sağladığı erişim sayesinde, hiç masamdan kalkmaksızın dergi makalelerine, arşivlerine ve kimi metinlere ulaşabildim. (İstediğim kitapları bulmak için kütüphanelere gitmek zorun-

da kaldım oysa. Yayıncılar, lütfen telif hakkı olan kitapların, özellikle de satışları şimdiden düşmüş olanların taranıp internete konulmasına izin verin. 21. yüzyıla siz de katılın!)

Son olarak ve en önemlisi, eşim Grace'e müteşekkirim; beni hep güleryüzle ve sabırla teşvik etti.

YZ'nin kurucularından merhum Allen Newell 1982'de şu cümleyi yazmıştı: "Önünde sonunda, Yapay Zekânın gerçek tarihçeleri yazılacak... hem de bilim tarihçilerinin elinden gelebilecek bir nesnellikle yazılacak bunlar. Fakat bunun vaktinin henüz gelmediği kesin."

Belki o vakit şimdidir.

BİRİNCİ KISIM

TOMURCUKLAR

1
DÜŞLER VE DÜŞ KURANLAR

Yapay Zekâ (YZ) arayışı düşlerle başlar; tıpkı tüm arayışlar gibi. İnsanın yeteneklerini barındıran makineler çok uzun süredir hayal edilegelmiştir: hareket eden otomatlar ve akıl yürüten cihazlar. İnsan benzeri makineler pek çok öyküde betimlenmiş, heykellerde, resimlerde, çizimlerde resmedilmiştir.

Belki bunların birçoğuna aşinasınız, yine de izninizle birkaçını sayayım. Homeros'un *İlyada* eserinde kendi kendine giden ve "üçayaklı" denen sandalyelerden ve onu etrafta gezdirsin diye aksak demirci Tanrı Hephaistos'un yapmış olduğu altın "refakatçiler"den bahsedilir.[1] *Dönüşümler* kitabında Ovidius'un yeniden anlattığı ilkçağ Yunan efsanesinde heykeltraş Pygmalion, güzel bakire Galatea'nın fildişinden heykelini yapar ve Afrodit bu heykele can verir:[2]

> Kız, adamın öpücüklerini hissetti, yanakları kızardı; mahçup gözlerini ışığa çevirdiğinde, hem aşığını hem de gökyüzünü gördü.

İlkçağ Yunan filozofu Aristoteles de (MÖ 384-322) otomasyonu düşlemişti, fakat belli ki bunu imkânsız bir hayal saymış; dolayısıyla insanlara boş vakit kalması için köleliğin şart olduğunu düşünmüştü. *Politika* adlı kitabında şöyle yazar:[3]

> Düşünün, elimizdeki her alet, ya bizim emrimizle ya da ihtiyaç halinde kendi işini icra ediyor; eğer... ozanın [Homeros] "kendi başlarına hareket edip tanrılar meclisine girerler" dediği Hephaistos'un üçayaklıları gibi, dokuma tezgâhının mekiği kendiliğinden gidip gelse ve lirin mızrabı kendi kendine çalsa, o zaman ne usta zanaatkârlar hizmetçiye gereksinim duyardı ne de efendiler kölelere.

1 *The Iliad of Homer*, İngilizceye çeviren: Richmond Lattimore, s. 386. Chicago: The University of Chicago Press, 1951 (karton kapaklı baskısı, 1961).

2 Ovidius, *Metamorphoses*, Kitap X., s. 243-297. 1850 dolaylarındaki İngilizce çevirisinden, http://www.pygmalion.ws/stories/ovid2.htm [Türkçesi: *Dönüşümler*, çev. İsmet Zeki Eyüboğlu, Payel Yayınevi].

3 Aristoteles, *The Politics*, s. 65, İngilizceye çeviren: T. A. Sinclair, Londra: S.C.M. Press, Ltd., 1946 [Türkçesi: *Politika*, çev. Mete Tunçay, Remzi Kitabevi, I:4].

Aristoteles, kendi kendine kumaş dokuyan Jacquard tezgâhını ya da kendi kendine müzik çalan piyanoları görse şaşkınlıktan küçük dilini yutardı herhalde.

Kendi öngörülü düşlerinin peşinden giden Katalan gizemci ve şair Ramon Llull (y. 1235-1316), *Ars Magna* (Yüce Sanat) tabir edilen bir dizi kâğıt disk üretmişti; bunları üretmede amaç, başka şeylerin yanı sıra, Müslümanları Hıristiyan inancına akıl ve mantık yoluyla kazandırmaktı (Resim 1.1). Llull'un disk derlemelerinden birine, Tanrı'nın kimi özellikleri kazınmıştı; adlarını söylemek gerekirse, iyilik, yücelik, sonsuzluk, iktidar, bilgelik, irade, erdem, hakikat ve yücelik. Bu diskler uygun şekilde çevrildiğinde, çeşitli ilahiyat sorularına yanıt bulunabileceğine inanılırdı.[4]

İcatlarıyla (her zamanki gibi) çağının önünde olan Leonardo Da Vinci, 1495 yılı civarında ortaçağ şövalyesi biçiminde insansı robot tasarımları çizmişti (Resim 1.2). Acaba Leonardo ya da çağdaşları bu tasarımı inşa etmeyi denemiş midir kimse bilmiyor. Leonardo'nun şövalyesi, güya dik oturabiliyor, kollarını ve kafasını hareket ettirebiliyor, çenesini açabiliyordu.[5]

Talmud, "golem" adlı yapay varlıklar yaratan kutsal kişilerden bahseder. Bu varlıklar, tıpkı Âdem gibi, çoğunlukla topraktan yaratılırmış. Golemleri hizmetçi gibi kullanan hahamlar hakkında öyküler var. Tıpkı Büyücünün Çırağı misali, golemleri denetim altında tutmak zaman zaman zor olurmuş.

1651'de Thomas Hobbes (1588-1679), toplum sözleşmesini ve ideal devleti ele alan kitabı *Leviathan*'ı yayımladı. Kitabın sunuş bölümünde Hobbes, "yapay hayvan" inşa etmenin mümkün olabileceğini söylüyor gibidir.[6]

> Yaşamı birtakım uzuvların deviniminden ibaret sayıyorsak, bunun içimizdeki asli bir aksamdan kaynaklandığını kabul ediyorsak, tüm otomatların (saatler gibi, yaylar ve çarklar sayesinde kendi kendine hareket eden motorlar) yapay bir yaşamı olduğunu neden söylemeyelim? Kalp yerine yay var demektir, sinirler yerine sicimler vardır, eklemler yerine bedeni hareket ettiren şey çarklardır...

4 Bkz. E. Allison Peers, *Fool of Love: The Life of Ramon Llull*, Londra: S. C. M. Press, Ltd., 1946.

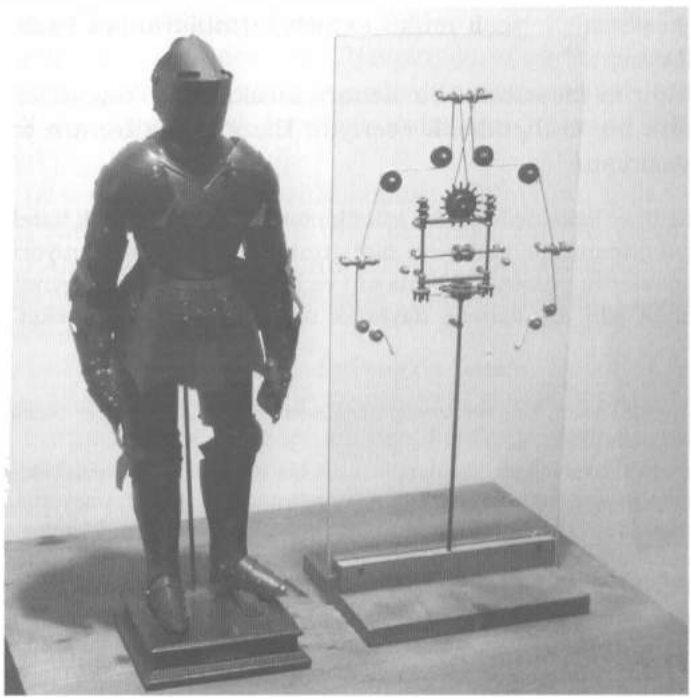
5 Bkz. http://en.wikipedia.org/wiki/Leonardo's_robot.

6 Thomas Hobbes, *The Leviathan*, karton kapaklı baskı, Kessinger Publishing, 2004 [Türkçesi: *Leviathan*, Yapı Kredi Yayınları, çev. Semih Lim].





Resim 1.1. Ramon Llull (solda) ve *Ars Magna*'sı (sağda).



Resim 1.2. Leonardo da Vinci'nin çizimlerine dayanarak robot şövalye modeli.

Belki bundan ötürü bilim tarihçisi George Dyson, Hobbes'a "yapay zekânın atası" der.[7]

Kimi insanlar, kurmaca düzeneklere ek olarak, çarpıcı biçimde canlıymış gibi hareket eden gerçek otomatlar da inşa etmiştir.[8] Bunların en gelişmiş olanı, Fransız mucit ve mühendis Jacques de Vaucanson'un (1709-1782) tasarlayıp inşa ettiği mekanik ördektir. 1738'de Vaucanson, vakvaklayan, kanatlarını çırpan, paytak paytak yürüyen, su içen ve tahıl yiyip öğüten şaheserini sergilemişti.

Bizzat Vaucanson'un belirttiği gibi,[9]

> İkinci makinem yani *otomatım* bir ördektir; bu makinede, yeme, içme ve sindirme işlemlerini yürüten bağırsak mekanizmasını yansıtıyorum: O edimler için gerekli olan tüm aksamın işleyişi, bire bir taklit edilmiştir. Ördek, elinizden tahıl tanesini almak için boynunu uzatır; taneyi yutar, sindirir ve sindirilmiş taneyi her zamanki yoldan dışarı atar.

Ancak, ördeğin "dışkıladığı" malzemenin yutmuş olduğu tahıl tanesinden gelip gelmediği konusunda anlaşmazlık var. Bu kadim otomatla ilgili bir internet sitesi şunu iddia ediyor:[10] "Vaucanson'un ördeğini 1844'te yenileyen sihirbaz Robert-Houdin, 'dışkının önceden hazırlandığını keşfetmişti: yeşil renkli ekmek kırıntılarından hazırlanmış bir tür bulamaç...'"

Sindirim meselesini bir kenara bırakırsak, Vaucanson'un ördeği fevkalade bir mühendislik eseriydi. Bizzat o da bunun farkındaydı. Şöyle yazmıştır:[11]

> Vasıflı ve ilgili kişilerin, bu küçük *otomattaki* onca farklı hareketli aksamı imal etmenin ne kadar zor olduğunu anlayacağına inanıyorum; örneğin bacaklarını yukarı kaldırmasını ve boynunu sağa sola oynatmasını sağlamak gibi. Bu esnada dayanak noktasının ya da hareket merkezinin

7 George B. Dyson, *Darwin Among the Machines: The Evolution of Global Intelligence*, Helix Books, 1997, s. 7.

8 Otomatlarla ve müzik kutularıyla ilgili bir internet sitesi için bkz. http://www.automates-anciens.com/english_version/frames/english_frames.htm.

9 Jacques de Vaucanson'dan: "Otomat mekanizmasının açıklaması ya da çalan Alman flütü görüntüsü: bir hatıratta, Paris Kraliyet Bilimler Akademisi'nden bir beyefendiye sunulduğu gibi... M. Vaucanson... Yapay ördek tasviriyle birlikte..." Fransızca aslından çeviren J. T. Desaguliers, Londra, 1742. Şu internet adresinde: http://e3.uci.edu/clients/bjbecker/NatureandArtifice/week5d.html.

10 http://www.automates-anciens.com/english_version/automatons-music-boxes/vaucanson-automatons-androids.php.

11 Jacques de Vaucanson, *a.g.e.*

değiştiğini görecekler: Ayrıca belli bir hareketli parçanın zaman zaman hareket merkezi olan öğenin, başka bir anda o parçada hareketli olduğunu, ardından o parçanın yine sabit kaldığını da görecekler. Başka bir deyişle, bu makinede şaşılacak sayıda mekanik kombinasyon sezecekler.

Bu makine, bir kez kurulduğunda, tüm o farklı işlemleri artık hiç el değmeden icra eder.

Ördeğin, gerçek ördekmişçesine su içtiğini, suda gagasıyla oynadığını, lık lık sesi çıkardığını söylemeyi unuttum. Kısacası, canlı hayvanın, dikkatle incelediğim tüm hareketlerini bu makineye taklit ettirmeye uğraştım.

Ne yazık ki, elimizde bu ördeğin ancak kopyaları var. Asıl ördek, 1879 yılı civarında Rusya'nın Nijninovgorod şehrinde bir müzede yakıldı. ANAS (ördekgil) adlı modern sürümünün yaptığı işleri şu internet adresinde görebilirsiniz: http://www.automates-anciens.com/video_1/duck_automaton_vaucanson_500.wmv.[12] Ayrıca, mekanik sanatların yaratıcı ismi Frédéric Vidoni'nin 1998'de tasarlayıp inşa ettiği ördek, Grenoble kentindeki Otomatlar Müzesi'nde sergileniyor (Resim 1.3.).

Kurmaca otomatlara gelecek olursak, doğal boyutlarda olan mekanik oyuncak bebek Olympia'dan bahsetmeliyim önce; Jacques Offenbach'ın (1819-1890) *Les Contes d'Hoffmann* (Hoffman'ın Öyküsü) adlı operasının birinci sahnesinde Olympia, şarkı söyleyip dans eder. Şair Hoffmann, operada Olympia'ya aşık olur ama huysuz Coppélius, Olympia'yı kırıp parçalarına ayırınca Hoffmann hüsrana uğrar (ve utanç duyar).

R.E.R. (*Rosssum'un Evrensel Robotları*) adlı oyunu, Çek yazar ve oyun yazarı Karel Čapek (Çapek diye okunur) 1920'de yayımlamıştır (Resim 1.4.). "Robot" sözcüğünü Čapek'e borçluyuz; Çekcede bu sözcük, "zorunlu işgücü" ya da "ağır iş" anlamına gelir. ("Robotnik", köylü ya da serf demektir).

Oyun, 1921 yılının Ocak ayında Prag'da başlar. Robotlar (oyunda baş harfi her zaman büyük yazılır) Rossum'un Evrensel Robotları adlı fabrikanın bulunduğu adada topluca üretilir; üretim esnasında, protoplazmanın yerini alacak bir kimyasal kullanılır. Oyunu betimleyen internet sitesine göre,[13] "Robotlar her şeyi hatırlar ama yeni hiçbir şey düşünmezler. Domin [fabrika müdürü], bunlardan harika üniversite profesörü olur, der... arada bir Robotun teki çalışmayı bırakıp

12 automates-anciens.com internet sitelerini bana anlattığı için Irvine California Üniversitesi'nden Prof. Barbara Becker'e teşekkür ediyorum.

13 http://jerz.setonhill.edu/resources/RUR/index.html.



Resim 1.3 Frédéric Vidoni'nin ANAS makinesi; Vaucanson'un ördeğinden esinlenmiştir. (Fotoğraf, Frédéric Vidoni'nin izniyle).

dişini gıcırdatmaya başlar. Fabrikanın insan idarecileri böyle bir olayı imalat hatası sayar ama Helena [Robotları özgürleştirmek istiyordur] bunu ruhun göstergesi olarak yorumlamayı tercih eder."

Oyunun sonunu anlatacak değilim, ama Čapek'in teknoloji sevdalısı olmadığını söyleyebilirim. Çalışmanın, insan yaşamında hayati bir unsur olduğuna inanıyordu. 1935'te yazdığı bir gazete makalesinde şunu belirtmişti (kendinden üçüncü tekil şahıs olarak bahsetme alışkanlığı vardı): "Yazar, makinelerin, insanın yerini alabileceği ya da yaşam, sevgi, isyan gibi şeylerin makinelerin dişlilerinde doğabileceği düşüncesinin sorumluluğunu kesin dille reddediyor. O, bu karamsar



Resim 1.4. New York yapımı *R.E.R.* oyunundan bir sahne.

görüşü, makinelere aşırı değer biçilmesi ya da yaşama ağır hakaret sayar."[14]

Čapek'in, bu oyunda robot sözcüğünü kullanmaya nasıl karar verdiğinin ilginç bir öyküsü var ve bu öyküyü bizzat Čapek kaleme almıştır. Bu oyun fikri, "aklında henüz tazeyken, ressam kardeşi Josef'in yanına gitmiş; kardeşi, şövalesinin önünde dikilmiş resim yapıyormuş... 'Bu yapay işçilere ne isim vereceğimi bilmiyorum' demiş Čapek. 'Onlara Emekçi diyebilirim ama bu bana fazla kitabi geliyor.' Ressam, ağzıyla fırçasını tutarken, 'o zaman sen de Robot deyiver' diye mırıldanıp resim yapmaya devam etmiş."[15]

Bilim kurgu (ve bilimsel gerçekler) yazarı Isaac Asimov, robotlar hakkında çok sayıda öykü kaleme almıştır. İlk derlemesi olan *Ben Robot*, "pozitronik" robotları anlatan dokuz öyküden oluşur.[16] Asimov'a, robotların her şeyi yıkıp mahvettiği bilim kurgu öykülerinden (örneğin Frankenstein'ın yaratığı gibi) sıkıldığı için, onun robotlarının pozitronik beynine "Üç Robot Yasası" işlenmiştir. Üç yasa şunlardır:

14 "Robotların Yazarı Kendini Savunuyor", başlıklı yazının İngilizce çevirisi için bkz. http://www.depauw.edu/sfs/documents/capek68.htm.

15 Čapek'le ilgili internet sitelerinden biri için bkz. http://Capek.misto.cz/english/robot.html. Ayrıca bkz. http:Capek.misto.cz/english/.

16 Isaac Asimov internet sitesi (http://www.asimovonline.com/) şunu ileri sürüyor: "Bu başlığı Asimov bulmuş değil, bilakis yayıncısı, Eando Binder'ın 1939 tarihli kısa öyküsünün başlığının 'üstüne yatmıştır'."

> Birinci Yasa: Hiçbir robot bir insana zarar veremez ya da eylemsiz kalıp bir insana zarar gelmesine izin vermez.
>
> İkinci Yasa: Robot, İlk Yasayla çelişmediği sürece insanların verdiği buyruklara uymak zorundadır.
>
> Üçüncü Yasa: Robot, Birinci ya da İkinci Yasayla çelişmediği sürece kendi varlığını korumak zorundadır.

Sonradan Asimov, insanlığın çıkarlarını koruma amacıyla "sıfırıncı" yasayı da ekledi:[17]

> Sıfırıncı Yasa: Hiçbir robot, insanlığa zarar veremez ya da eylemsiz kalıp insanlığa zarar gelmesine izin vermez.

Yapay zekâ arayışı, Don Kişotvari olsa da olmasa da, böyle düşlerle başlar; fakat düşleri gerçeğe dönüştürmek, nasıl yol alınacağı konusunda kullanışlı ipuçları gerektirir. Neyse ki, az sonra göreceğimiz üzere, böyle pek çok ipucu vardı.

17 Bu dört yasanın tarihçesi hakkında bilgi için bkz. http://www.asimovonline.com/asimov_FAQ.html#series13

2
İPUÇLARI

Zeki makineler yapmak için nelerin gerekli olabileceği hakkında bol bol ipucu, felsefe, mantık, biyoloji, psikoloji, istatistik ve mühendislik alanlarına dağılmıştır. İnsanlar, zekânın kimi yönlerini otomatikleştirme serüveninde, bu alanlardan gelen ipuçlarından yavaş yavaş artan bir yoğunlukla faydalanmaya koyulmuştu. Öyküme, bu ipuçlarından bazılarını betimleyip yapay zekâ konusundaki ilk atılımlardan bazılarına nasıl ilham verdiklerini anlatarak başlıyorum.

2.1 FELSEFEDEN VE MANTIKTAN GELENLER

Her ne kadar insanlar binlerce yıldır mantık uyarınca akıl yürütüyor olsa da, bu süreci çözümleyip tanımlamaya çalışan ilk kişi, Yunan filozof Aristoteles olmuştur. Aristoteles, *tasım* dediği bir uslamlama çeşidi tanımlar: "Tasım yönteminde, belirli yargılarda bulunulur ve bunlar dışında bir yargıya, o yargıların sonucu olarak ulaşılır.[1]

Aşağıda, belli bir tasım türüne dair ünlü bir örnek görüyoruz:[2]

1. *Tüm insanlar ölümlüdür.* (yargı)
2. *Tüm Yunanlar insandır.* (yargı)
3. *Tüm Yunanlar ölümlüdür.* (sonuç)

Aristoteles'in bu katkısının güzelliği (ve YZ için önemi), tasımın *biçimiyle* alakalıdır. Sırf insanlardan, Yunanlardan ya da ölümlü olmaktan bahsetmek zorunda değiliz. Başka bir şey hakkında da konuşuyor olabilirdik; *insanlar*, *Yunanlar* ve *ölümlü* sözcükleri yerine keyfî simgeler kullanarak tasımı yeniden yazarsak, sonucun ne olacağı apaçık görünür. Bu şekilde yeniden yazarsak şunu elde ederiz:

1 Aristoteles, *Birinci Analitikler*, I. Kitap, MÖ yaklaşık 350 yılında yazılmıştır; İngilizceye çeviren: A. J. Jenkinson, internet eklentisini yayımlayan eBooks@Adelaide, bkz. http://etext.library.adelaide.edu.au/a/aristotle/a8pra/.

2 Ortaçağda mantık araştırmacıları, inceledikleri tasımlara farklı isimler verirdi. Bu tasıma, hatırlatıcı *Barbara* ismini takmışlardı, çünkü tüm önermeler "*all*" (tüm) sözcüğüyle başlıyor ve bunun ilk harfi de "A". "Barbara" sözcüğündeki üç ünlü harf de "a"dır.

1. *Tüm B'ler A'dır.* (yargı)
2. *Tüm C'ler B'dir.* (yargı)
3. *Tüm C'ler A'dır.* (sonuç)

İnsan, A, B, C yerine istediği herhangi bir şeyi koyabilir. Örneğin: *tüm sporcular sağlıklıdır* ve *tüm futbolcular sporcudur*, dolayısıyla *tüm futbolcular sağlıklıdır* vb. (Elbette, "yargılar" doğru değilse, "sonuç" doğru olmayabilir. Ne ekersen onu biçersin!)

Aristotelesçi mantık, uslamlamanın otomatikleştirilmesi yolunda iki ipucu sağlıyor. Birincisi, tasım gibi uslamlama örüntülerini *biçim* ya da *kalıp* olarak göstermek karmaşık değildir. Bu örüntüler, farklı farklı pek çok somut durum yerine geçebilecek genel simgeler kullanabilir. Herhangi bir şey yerine geçebilecekleri için, o simgelerin kendisi önemli değildir.

İkincisi, genel simgelerin yerine belli bir probleme özel simgeler konunca, kişi bir yanıta ulaşmak için sadece "kolu indirse" yeter. Genel simgeler ve benzer geçiş işlemlerinin kullanımı, tüm modern YZ uslamlama programlarının temelidir.

Daha modern zamanlara gelelim; Gottfried Wilhelm Leibniz (1646-1716; Resim 2.1) mantıksal uslamlama hakkında kafa yoran ilk kişilerden biriydi. Alman filozof, matematikçi, mantıkçı Leibniz, birçok başarısının yanında, kalkülüsün (türev ve integral hesabı) mucitlerinden biridir. (Isaac Newton'la bunun hakkında uzun uzun tartışmıştır.) Fakat anlattığımız öykü bakımından daha önemlisi, uslamlamayı mekanikleştirmek istemişti. Leibniz şöyle yazar:[3]

> Kusursuz insanların hesap için köle gibi saatlerini harcaması ne yazık, oysa makineler kullanılıyor olsa, bu iş güvenle başkalarına bırakılabilirdi.

ve

> Çünkü, düzenli üç boyutlu cisimlerin sayısını belirleyen insanlar övgü kazanıyorsa... sahip olduğumuz en mükemmel ve faydalı şey olan insan uslamlamasını matematik yasalarına bağlamak ne kadar daha iyi olacaktır.

Leibniz, tüm insan bilgisinin, hatta felsefe ve metafizik bilgisinin ifade edilebileceği bir dil düşünüp bunu tasarlamaya kalkışmıştır. Bilgiyi

3 Aktaran: Martin Davis, *The Universal Computer: The Road from Leibniz to Turing*, New York: W. W. Norton & Co., 2000. Bu alıntıyı içeren karton kapak baskıdan aktarım için bkz. http://www.wwnorton.com/catalog/fall01/032229EXCERPT.htm.



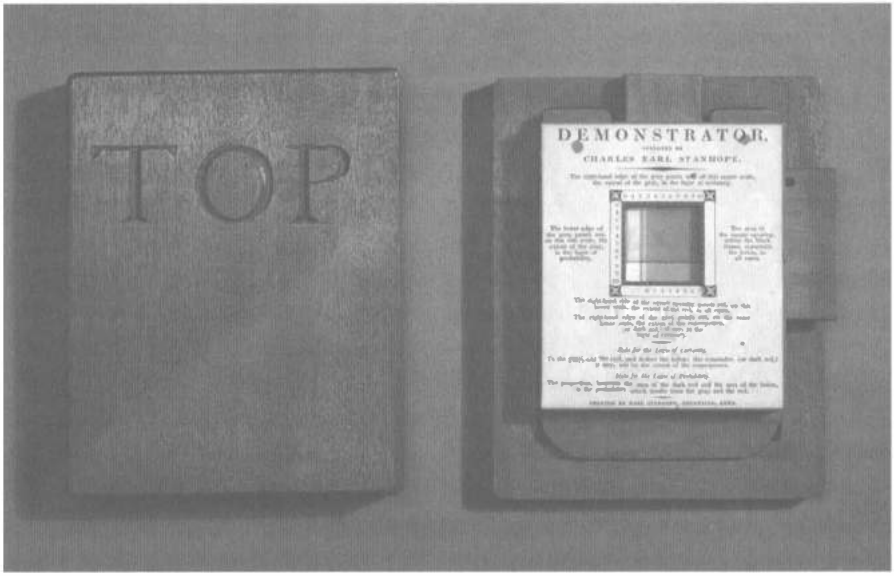
Resim 2.1. Gottfried Leibniz

meydana getiren önermelerin, daha az sayıda temel önermeyle inşa edilebileceğini ileri sürmüştü; tıpkı tüm sözcüklerin alfabedeki harflerle oluşturulması gibi. Leibniz'in bahsettiği *lingua characteristica* yani evrensel dil, bu temel önermelerden meydana gelecek, böylece bir *insan düşünce alfabesi* altyapısı oluşturacaktı.

Bu alfabe, otomatik uslamlamanın temeli olarak hizmet edecekti. Leibniz'in fikri şöyleydi: Eğer alfabedeki birimler sayılarla temsil edilirse, o zaman karmaşık bir önerme, uygun sayıların çarpılması sayesinde temel bileşenlerinden elde edilebilir. Ardından, bu karmaşık önermenin doğru mu yanlış mı olduğunu belirlemek amacıyla ek aritmetik işlemleri gerçekleştirilebilir. Bu sürecin bütünü, *calculus ratiocinator* tarafından başarılacaktı. Böylece filozoflar bir sorun konusunda fikir ayrılığına düştüğünde, "*calculemus*" (hadi hesaplayalım) diyebileceklerdi. Önce o sorunu *lingua characteristica*'da ifade edecekler, ardından *calculus ratiocinator*'da "kolu indirip" sorunu çözeceklerdi.

Bu fikri uygulamada başlıca sorun, temel "alfabenin" bileşenlerini keşfetmekti. Bununla birlikte, Leibniz'in çalışmaları, uslamlamanın nasıl mekanikleştirileceği konusunda önemli ek ipuçları sağlamıştır: Basit simgelerden oluşan bir alfabe icat et, ardından bu simgeleri daha karmaşık anlatımlar oluşturacak şekilde birleştirmenin araçlarını bul.

18. yüzyılın sonuna ve 19. yüzyılın başına doğru Britanyalı bilimci ve siyasetçi Charles Stanhope (Üçüncü Stanhope Baronu), basit mantık ve olasılık problemlerini çözen cihazlar inşa edip bunlarla deneyler yapmıştır (Resim 2.2). Stanhope "kutu"sunun bir çeşidinin yan taraflarında yarıklar bulunur ve bunların içine renkli slaytlar ittirilir. Üst



Resm 2.2. Stanhope Kare Gösterici, 1805. (Fotoğraf, Bilim Müzesi/ SSPL'nin izniyle.)

taraftaki pencereden bakıp, özgül bir problemi temsil etmeye uygun şekilde yerleştirilmiş slaytları görmek mümkündür. Günümüzde, Stanhope kutusunun bir tür analog bilgisayar olduğunu söyleyebiliriz.

Computing Before Computers [Bilgisayarlardan Önce Hesaplama] başlıklı kitap, bu kutunun yaptığı işlemlere örnek veriyor:[4]

> Sayısal bir tasımı çözmek söz konusu olduğunda, örneğin:
>
> *On adet A'dan sekizi B; On A'dan dördü C'dir;*
> *Dolayısıyla, en az iki B, C'dir.*
>
> Stanhope, kırmızı slaytı (B'yi temsil ediyor) pencere (A'yı temsil ediyor) boyunca sekiz birim itiyor, gri slaytıysa (C'yi temsil ediyor) ters yönde dört birim çekiyordu. Slaytların örtüştüğü iki birim, aynı zamanda C de olan B'lerin asgari sayısını gösteriyordu.
>
> ...
>
> Benzer biçimde Gösterici birimi, şöyle bir geleneksel tasımı çözmek amacıyla da kullanılabilirdi:

4 Aktaran: Willam Aspray (ed.), *Computing Before Computers*, Bölüm 3, "Logic Machines", Ames, Iowa: Iowa State Press, 1990, s. 107-8. (Ayrıca şu sitede: http://ed-thelen.org/comp-hist/CBC.html.)

Hiçbir M, A değildir; Tüm B'ler M'dir; Dolayısıyla, hiçbir B, A değildir.

Stanhope'un ağzı bu cihaz konusunda sıkıydı ve neyin peşinde olduğunu kimse bilsin istemiyordu. *Computing Before Computers* kitabında belirtildiği gibi, "Gösterici hakkında açıklamasının bulunduğu *The Science of Reasoning Clearly Explained Upon New Principles* (1800) [Yeni İlkelerle Açıklanan Akıl Yürütme Bilimi] başlıklı kitabı şahsen dağıttığı birkaç arkadaşına ve akrabasına sessiz kalmalarını tavsiye etmişti; 'piç bir taklidin' bu konuda basmaya niyetlendiği kitaptan daha önce piyasaya çıkmasından korkuyordu."

Fakat Stanhope'un ölümünün altmış yıl sonrasına kadar hiçbir yayın çıkmadı. O dönemde, Papaz Robert Harley, Stanhope'un notlarına ve kutularından birine ulaşmış, böylece "Stanhope Göstericisi" ismini verdiği cihaz hakkında bir makale yayımlamıştı.[5]

Llull'un şemalarının ve Leibniz'in umutlarının aksine, Stanhope gerçekten çalışan ilk mantık makinesini inşa etmişti; gerçi bu makinenin ufak tefek sorunları yok değildi. Belki de onun eseri, mantıksal uslamlamanın gerçekten mekanikleştirilebileceği inancını güçlendirmişti.

1854'te İngiliz George Boole (1815-1864; Resim 2.3) *An Investigation of the Laws of Thought on Which Are Founded the Mathematical Theories of Logic and Probabilities* [Mantık ve Olasılığa Dair Matematiksel Kuramların Üstüne Kurulduğu Düşünce Yasalarının Soruşturulması] başlıklı bir kitap yayımladı.[6] Boole'un amacı (birtakım başka amaçlarının yanı sıra), "İnsan zihninin doğasını ve çatısını ilgilendiren kimi olası benzerlikleri... derlemek" idi. Boole, insan uslamlamasının çeşitli mantık ilkelerini gözden geçirmiş, bunları matematiksel olarak ifade etmişti. Örneğin, "Önerme IV" şunu der: "zıtlık ilkesi... herhangi bir varlığın aynı anda bir niteliği hem barındırıp hem barındırmayışının imkânsız olduğunu belirtir..." Ardından Boole, bu ilkeyi cebir denklemi olarak yazmıştır:

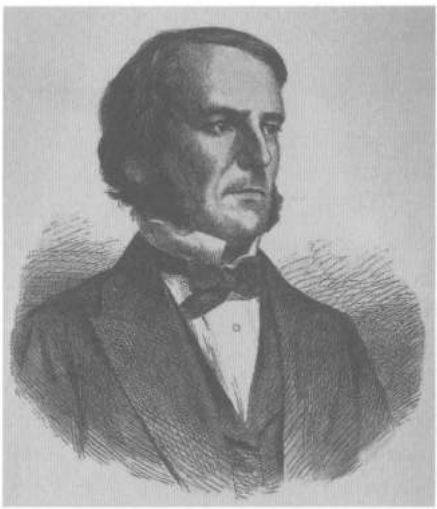
$$x(1 - x) = 0$$

bu denklemde x, herhangi bir nesne sınıfını, $(1 - x)$ ise "zıt ya da ek nesne sınıflarını" temsil eder, 0 ise "var olmayan" bir sınıfı gösterir.

Boole'un çalışmasından filizlenmiş olan Boole cebirinde, 0'ın *yanlışı*, 1'in ise *doğruyu* temsil ettiğini söyleriz. Mantıkta iki temel işlev

5 Robert Haley, "The Stanhope Demonstrator", *Mind*, c. IV, 1879, s. 192-210.

6 George Boole, *An Investigation of the Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities*, Dover Publications, 1854.



Resim 2.3. George Boole

olan VEYA ile VE, Boole cebirinde sırasıyla + ve ×'le temsil edilir. Dolayısıyla, örneğin "*p* ya da *q* ya da ikisi de" ifadesini, $p + q$ diye yazarız. "*p* ve *q*" ifadesini temsil etmek için, $p \times q$ yazarız. Her bir *p* ve *q* doğru ya da yanlış olabilir, o halde $p + q$ ile $p \times q$ ifadelerinin değerini (doğru mu yanlış mı), + ile ×'nın kullanımına dair tanımlardan faydalanarak belirleriz, şöyle ki:

$$1 + 0 = 1,$$
$$1 \times 0 = 0,$$
$$1 + 1 = 1,$$
$$1 \times 1 = 1,$$
$$0 + 0 = 0,$$

ve

$$0 \times 0 = 0$$

Boole cebiri, telefon anahtarı devrelerinin ve bilgisayarların tasarımında önemli rol oynar. Her ne kadar Boole muhtemelen bilgisayarları öngörememiş olsa da, kendi çalışmasının önemini idrak etmişti. George Thomson'a (ileride Lord Kelvin olacaktı) yazdığı 2 Ocak 1851 tarihli mektupta şunu belirtir:[7]

7 Bkz. D. McHale, *George Boole: His Life and Work*, Dublin, 1985. Alıntı şuradan: http://www-groups-dcs.st-and.ac.uk/~history/Mathematicians/Boole.html.

> Bugünlerde, Mantık ve Olasılık kuramı hakkında bir açıklamayı basıma uygun hale getirmeye ciddi ciddi koyuldum; şu anki haliyle bu kuram bana göre, Bilime yaptığım ya da yapacağım en önemli belki de tek katkıdır; bundan sonra benden hatırlanacak tek şeyin bu olmasını isterdim...

Boole'un çalışması göstermiştir ki bazı mantıksal uslamlama türleri, mantıksal önermeleri temsil eden denklemlerin kullanılmasıyla gerçekleştirilebilir; uslamlamanın mekanikleştirilmesi yolunda epey önemli bir ipucudur bu. Yine önermelerin kullanımında ve değerlendirilmesinde eşit oranda hayati öneme sahip ama cebirsel olmayan sisteme, "önermeli kalkülüs" denir (çoğunlukla "önermeli mantık" denir); bu kalkülüs, ileride göreceğimiz üzere, yapay zekâda çok önemli bir rol üstleniyor. [Bazıları Stoacı Yunan filozof Khrysippos'un (MÖ 280-209), önermeli kalkülüsün ilk biçimlerinden birini icat ettiğini ileri sürer.[8]]

Bununla birlikte, Boole mantık sisteminin yetersizliklerinden biri, *p*, *q* gibi önermelerinin "atomik" oluşudur. Yani, önermelerin içeriğini göstermezler. Örneğin, "Jack insandır" önermesini *p* ile, "Jack ölümlüdür" önermesini *q* ile ifade edersek, *p* ya da *q*'da, insan olan Jack'ın ölümlü Jack'le aynı kişi olduğuna işaret eden hiçbir şey yoktur. Bu yüzden, tabiri caizse, içsel unsurlar barındıran "moleküler anlatımlara" gereksinim duyarız.

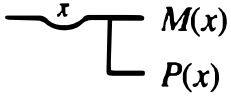
19. yüzyılın sonuna doğru Alman matematikçi, mantıkçı ve filozof Friedrich Ludwig Gottlob Frege (1848-1925), içsel bileşenleriyle birlikte önermelerin bir tür grafik biçiminde kâğıda aktarılabileceği bir sistem icat etti. Bu dile, "kavram yazımı" diye çevirebileceğimiz *Begriffsschrift* adını vermiştir. Örneğin, "Tüm kişiler ölümlüdür" ifadesinin *Begriffsschrift*'te, Resim 2.4'teki çizim gibi gösterilebilir.[9]

Bu çizimde, kişi olduğu, p(x), söylenen x'in, ardından ölümlü olduğu, M(x), iddia edilen x'le açıkça aynı olduğuna dikkat edin. Bugünlerde bu ifadeyi doğrusal biçimde ifade etmek bizim için daha rahat: $(\forall x)\ P(x) \supset M(x)$; konuşma dilinde bu, "tüm x'ler için, eğer x kişiyse, x ölümlüdür" cümlesine denktir.

8 Bkz. örneğin Gerard O'Reagan, *A Brief History of Computing*, Londra: Springer-Verlag, 2008, s. 17.

9 Çevrimiçi Stanford Felsefe Ansiklopedisinde (http://plato.stanford.edu/entries/frege/) kullanılan görsel sürümünü izliyorum; şöyle deniyor: "... Frege'nin formülünü görseli basitleştirmek için değiştirdik. Frege'nin genel ifadelerinde kullanmış olduğu özel yazı tipini (*Gothic*) kullanmıyoruz ya da benimsemiş olduğu özel tertipleri gözetmiyoruz..."

Resim 2.4. *Begriffsschrift*'te "Tüm kişiler ölümlüdür" anlamında.



Frege'nin sistemi, günümüzde "yüklemcil kalkülüs" (*predicate calculus*) dediğimiz sistemin öncülüdür, ki bu yapay zekâda çok önemlidir. Aynı zamanda, günümüzde yapay zekâda kullanılan başka bir gösterim biçimi olan anlamsal ağların öncülü olmuştur. Frege'nin çalışması, uslamlama süreçlerini mekanikleştirme hakkında bundan da fazla ipucu sağlamıştır. Nihayet, uslamlamada kullanılacak bilgiyi ifade eden cümleler açık ve simgesel olarak yazılabiliyordu.

2.2 YAŞAMIN KENDİSİNDEN GELENLER

Özdeyişler kitabının 6:6-8 numaralı cüzünde Kral Süleyman, "Ey tembel kişi, git karıncalara bak, onların yaşamından bilgelik öğren", der. Her ne kadar bu tavsiye, tembelliğe karşı bir uyarı olsa da, bizi yapay nesneler üretip geliştirme konusunda biyolojiden ipuçları aramaya da pekâlâ çağırabilir.

Doğrusu, "hayatın" kimi yönleri, zekâ hakkında önemli ipuçları sunmuştur. Hayvanlarda duyusal bilgiyi eyleme dönüştürmekten sorumlu olan organ, *beyin* olduğu için, beyinleri ve bunun temel bileşeni olan sinir hücrelerini araştıran sinir fizyologları ile sinir anatomicilerinin çalışmalarından kimi sağlam fikirlerin bulunması beklenebilir. Çeşitli yöntemler sayesinde zeki davranışları gerçekleşirken inceleyen psikologların çalışmaları da başka fikirler sağlayabilir. En sonunda, zeki yaşamı üreten unsur evrimsel süreçler olduğu için, bu süreçler de, nasıl ilerleneceği konusunda ipucu sağlayabilir.

2.2.1 Sinir Hücreleri ve Beyin

19. yüzyılın ikinci yarısında ve 20. yüzyılın başında "sinir hücresi doktrini", "sinir hücresi" denen canlı hücrelerin, ara bağlantılarıyla birlikte, beyin işlevi için temel önemi olduğunu belirtmiştir. Bu görüşü ileri sürenlerden biri, İspanyol sinir anatomicisi Santiago Ramón y Cajal'dı (1852-1934). Cajal (Resim 2.5) ve Camillo Golgi, sinir sistemi yapısı üzerine çalışmaları sayesinde 1906'da Fizyoloji-Tıp Nobel Ödülü'nü kazanmışlardır.

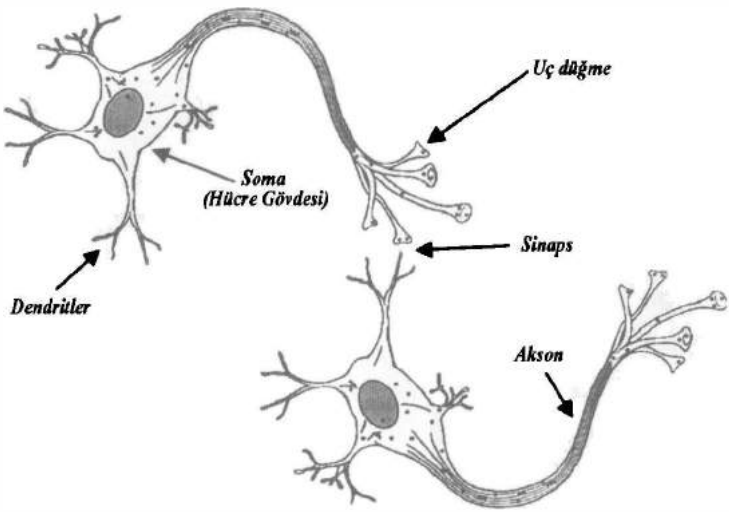


Resim 2.5. Ramón y Cajal

Sinir hücresi canlıdır ve beyinde bunlardan yaklaşık on milyar (10^{10}) tane vardır. Her ne kadar bunların farklı biçimleri olsa da, genelde *soma* ya da *hücre gövdesi* denen merkezî bir parça, *dendrit* denen giriş bağlantısı ve *akson* denen çıkış bağlantısından oluşurlar. Sinir hücresinin aksonunda, *uç düğmeler* denen çıkıntılar, başka sinir hücrelerinin bir ya da daha fazla dendritine iyice yaklaşır. Sinir hücresinin uç düğmesi ile başka bir sinir hücresinin dendriti arasındaki boşluğa *sinaps* denir. Bu boşluğun boyutu yaklaşık 20 nanometredir. Resim 2.6'daki çizimde iki sinir hücresi gösteriliyor.

Elektrokimyasal etki sayesinde sinir hücresi, aksonu boyunca bir dizi atım gönderebilir. Atım, başka bir sinir hücresinin dendritine komşu olan sinapsa ulaştığında, sinapsın karşı tarafındaki öbür sinir hücresinde elektrokimyasal bir etkinliği tetikleyebilir ya da o hücrenin etkinliğini ketleyebilir. Bu ikinci sinir hücresinin kendi başına atım "ateşleyip" göndermesi, hücreye gelen çeşitli dendritlerin sinapslarına ne kadar ve ne çeşit (tetikleyici ya da ketleyici) atımın ulaştığına ve elektrokimyasal etkinlik iletiminde o sinapsların yeterliğine bağlıdır. İnsan beyninde yarım trilyondan fazla sinaps olduğu tahmin ediliyor. Sinir hücresi doktrini, algılama ve düşünme dahil çeşitli beyin etkinliklerinin, sinir hücresi etkinliğiyle doğduğunu iddia eder.

1943'te Amerikalı sinir fizyoloğu Warren McCulloch (1899-1969; Resim 2.7) ve mantıkçı Walter Pitts (1923-1969), sinir hücresinin özünde bir "mantık birimi" olduğunu ileri sürdüler. Ünlü ve önemli maka-



Resim 2.6. İki sinir hücresi. (Kaynak: Science, c. 316, s. 1416, 8 Haziran 2007. İzinle kullanılmıştır.)

lelerinde, basit sinir hücresi modelleri önerip bu modellerden oluşan ağların, olası tüm hesap işlemlerini yapabileceğini göstermişlerdir.[10] McCulloch-Pitts "sinir hücresi", girdileri ve çıktıları (kabaca, sırasıyla dendritlere ve aksonlara karşılık gelirler) olan matematiksel bir soyutlamaydı. Her çıktı, 1 ya da 0 değeri alabilir. (McCulloch-Pitts sinir hücresi gerçek sinir hücresiyle karışmasın diye, McCulloch-Pitts sinir hücresine ve benzerlerine, "sinirsel öğe" diyorum.) Sinirsel öğeler, ağ şeklinde bir sinirsel öğenin çıktısı, başka öğelerin girdisi olacak biçimde bağlanabilir. Bazı sinirsel öğeler tetikleyicidir; bunların çıktısı, bağlantılı oldukları sinirsel öğelerin "sinyal ateşlemesine" katkıda bulunur. Başkalarıysa ketleyicidir; bunların çıktısı, bağlantılı oldukları sinirsel öğelerin "sinyal ateşlemesini" ketlemeye katkıda bulunur. Sinirsel bir öğeyi etkileyen tetikleyici girdilerin toplamı, ketleyici girdilerin toplamından çıkarıldığında, sonuç belirli bir "eşiği" aşıyorsa, sinirsel öğe ateşler ve 1 çıktısını bağlantılı olduğu tüm sinirsel öğelere gönderir.

McCulloch ile Pitts'in önerdiği ağların örnekleri Resim 2.8'de gösteriliyor.

10 Warren S. McCulloch ve Walter Pitts, "A Logical Calculus of Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, c. 5, Chicago: University of Chicago Press, 1943, s. 115-133. ("McCulloch-Pitts sinir hücrelerinin" hesapla ilgili yönlerinin okunabilir bir incelemesi için bkz. Marvin Minsky, *Computation: Finite and Infinite Machines*, Englewood Cliffs, NJ: Prentice Hall, 1967.)



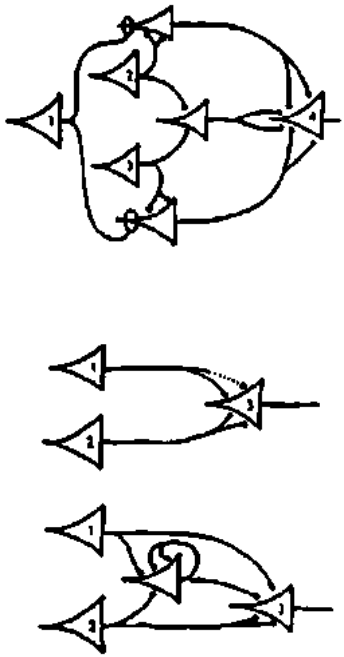
Resim 2.7. Warren McCulloch

Beyindeki sinir hücrelerinin, düşüncenin temel birimi olduğuna Kanadalı sinirfizyoloğu Donald O. Hebb de (1904-1985) inanıyordu. Hebb, önemli bir kitabında şunu söylemişti:[11] "A hücresinin aksonu, B hücresini uyarıp bu hücrenin sinyal göndermesinde defalarca ya da istikrarlı bir şekilde rol alacak denli yakınsa ona, bu hücrelerin birinde ya da ikisinde bir tür büyüme süreci ya da metabolizma değişikliği gerçekleşir, öyle ki B hücresine sinyal gönderen hücrelerden biri olarak A hücresinin verimi artar". İleride, sinirsel "sinaps kuvvetindeki" değişikliklerle ilgili bu Hebb kuralı, canlı hayvanlarla yapılan deneylerde gerçekten gözlemlenecekti. (1965'te sinir fizyoloğu Eric Kandel, deniz yumuşakçası *Aplysia californica*'da basit öğrenme biçimlerinin sinaps değişikliklikleriyle bağlantılı olduğunu gösterdiği deney sonuçlarını yayımlamıştır. Kandel, "sinir sisteminde sinyal iletimi hakkında keşiflerinden ötürü" 2000 yılının Fizyoloji-Tıp Nobel Ödülü'nü kazanan isimlerden biri olmuştur.")

Ayrıca Hebb, birlikte ateşlenen sinir hücresi yığınlarının, *hücre öbekleri* oluşturduğunu öne sürmüştü. Hebb, "birlikte sinyal gönderme" durumunun, beyinde hep olduğunu ve algıyı temsil eden hücre öbeği oluşumuna bunun yol açtığını düşünüyordu. Hebb'e göre "düşünme" işi, hücre öbeği kümelerinin ardışık olarak etkinleştirilmesiydi.[12]

11 Donald O. Hebb, *The Organization of Behavior: A Neuropychological Theory*, New York: John Wiley, Inc., 1949.

12 Hebb hakkında bkz. http://www.cpa.ca/Psynopsis/special.eng.html.



Resim 2.8. McCulloch-Pitts sinirsel öğeleri ağı. (Kaynak: Resim 1, Warren S. McCulloch ve Walter Pitts, "A Logical Calculus of Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, c. 5, 1943, s. 115-133).

2.2.2 *Psikoloji ve Bilişsel Bilim*

Psikoloji, zihinsel süreçleri ve davranışları inceleyen bilim dalıdır. Bu sözcük, Yunanca *psyche* (soluk, tin ya da ruh) ile *logos* (söz) sözcüklerinden türemiştir. İnsan, böylesi bir bilim dalının, zeki nesneler yaratmak isteyenlerin ilgisini çekecek şeyler anlatmasını bekliyor. Bununla birlikte, 19. yüzyılın ikinci yarısına dek psikolojide çoğu kuram, filozofların, yazarların ve insanlığı izleyen başka zeki gözlemcilerin içgörülerine dayanmıştır. (İş insan davranışlarını anlamaya gelince, Shakespeare, Tolstoy ve öbür yazarlar hiç de beceriksiz değildi.)

Çoğu insana göre ciddi bilimsel araştırmalar, Alman Wilhelm Wundt (1832-1920) ve Amerikalı William James'le (1842-1910) başlamıştır.[13] 1875'te birer psikoloji laboratuvarı kurmuşlardı: Wundt Leipzig'de, Ja-

13 İki adamın da hayatlarının ve eserlerinin özeti için, Dr. C. George Boeree'nin "Wilhelm Wundt and William James" başlıklı internet sitesine bkz. http://www.ship.edu/~cgboeree/wundtjames.html.

mes Harvard'da. Pennsylvania Shippensburg Üniversitesi'nde psikoloji tarihi öğreten C. George Boeree'e kulak verelim: "Wundt'un geliştirmiş olduğu yöntem, bir tür deneysel iç gözlemdir: Araştırmacı, nitelik, yoğunluk ya da süre olarak ölçülebilecek basit bir olayı dikkatle gözlemlemeli, ardından o olayın çeşitlemelerine kendi verdiği tepkileri kaydetmeliydi." Günümüzde James çoğunlukla filozof olarak görülse de, 1873'te ve 1874'te yayımlanmış iki ciltlik kitabı *The Principles of Psychology*'yle (Psikolojinin İlkeleri) ünlüdür.

Hem Wundt hem James, salt beynin girdi-çıktı davranışlarını sıralamak yerine beynin nasıl işlediği konusunda bir şeyler söylemeye çalışmışlardır. Psikiyatr Sigmund Freud (1856-1939) ise daha ileri gidip beynin *id*, *ego* ve *süperego* adlı iç bileşenleri olduğunu varsaymış, bunların etkileşime girip davranışları nasıl etkilediğini anlatmıştır. *Psikanaliz* denen, kendisine özgü kılavuzlu içgözlem tarzı sayesinde bu bileşenler hakkında bir şeyler öğrenmenin mümkün olduğunu düşünüyordu.

Psikolojiyi daha bilimsel kılıp öznel içgözlemden koparmaya uğraşan pek çok psikolog, ki en ünlüleri B. F. Skinner'dı (1904-1990; Resim 2.9), salt nesnel olarak ölçülebilen şeylere yoğunlaşmaya koyulmuştu, yani özgül uyartılara verilen özgül tepkilere odaklanmışlardı. *Davranışçılar*, psikolojinin bir zihin bilimi değil, davranış bilimi olması gerektiğini savunuyordu. İnançlar, niyetler, arzular, hedefler gibi içsel zihin hallerini tanımlama fikrini reddetmişlerdi.

Bu gelişme, beynin iç işleyişi hakkında faydalı ipuçlarına erişmek isteyen kişilere, en başta geriye dönük bir adım gibi görünebilir. "Davranışçılık"tan türemiş, istatistik yönelimli kuramları eleştiren Marvin Minsky şunu yazmıştır: "Aslında bir 'anlam' gereksiniminden kaçınmaya niyetlenmiş bu kuramlar, sonunda anlamı açıklama ihtimalinden kaçınmayı başarıyor ancak."[14] Bununla birlikte, Skinner'ın çalışmaları, *pekiştirici* uyartı fikrini doğurmuştur; yani en son davranışı ödüllendirip, gelecekte (benzer şartlar altında) bu davranışın ortaya çıkma ihtimalini güçlendiren uyartılar.

Pekiştirmeli öğrenme, YZ araştırmacılarından rağbet gören bir strateji olmuştur, gerçi bu strateji iç hallere bel bağlar. ABD Ulusal Standartlar Büro'sunda (günümüzde Ulusal Standartlar ve Teknoloji Enstitüsü, NIST) bilgisayar bilimcisi olan Russell Kirsch (y. 1930-), bu stratejiden faydalanan ilk kişilerden biriydi. "Yapay hayvanların", bir

14 M. Minsky (ed.), "Introduction", *Semantic Information Processing*, Cambridge, MA: MIT Press, 1968, s. 2.



Resim 2.9. B. F. Skinner (B. F. Skinner Vakfı'nın izniyle.)

oyundaki doğru hamleleri öğrenmek için pekiştirmeden yararlanabileceğini ileri sürmüştü. 1954'te verdiği bir seminerin notlarında şöyle yazar:[15] "Model hayvan, her uyartıda karşı tarafın hangi hamleyi yaptığına dikkat eder... Sonra, aynı uyartı tekrarlandığında, rakibin daha önce bu uyartı karşısında yapmış olduğu hamleyi tekrarlar. Rakip, herhangi bir uyartının ardından aynı hamleyi ne kadar çok tekrarlarsa, model hayvan bu hamleye o kadar çok 'koşullanmış' olur."

Skinner, pekiştirmeli öğrenmenin, insanlarda sözel davranışları bile açıklayabileceğine inanıyordu. Bu fikirlerini, 1957 tarihli *Verbal Behavior* [Sözel Davranış] kitabında ortaya koydu;[16] laboratuvar çalışmalarını temel alan ve sonuçlara göre şekillenen seçilim ilkelerinin, insanların ne söylediğini, yazdığını, düşündüğünü, hangi jestleri yaptığını açıklamaya kadar varabileceğini ileri sürmüştü.

Skinner'ın dille ilgili fikirlerine karşı çıkan dilbilimci Noam Chomsky (1928- ; Resim 2.10), Skinner'ın kitabının eleştirisinde şunu yazmıştır:[17]

> Bununla birlikte, bu kitap (ve kitabın dayandığı araştırma) özenle incelendiğinde, Skinner'ın inanılmaz iddialarının hiç de haklı olmadığı gö-

15 Russell A. Kirsch, "Experiments with a Computer Learning Routine", Bilgisayar Semineri Notları, 30 Temmuz, 1954. Şu internet adresinde: http://www.nist.gov/msidlibrary/doc/kirsch_1954_artificial.pdf.

16 B. F. Skinner, *Verbal Behavior*, Englewood Cliffs, NJ: Prentice Hall, 1957.

17 Noam Chomsky, "A Review of B. F. Skinner's *Verbal Behavior*", *Readings in the Psychology of Language* içinde, Leon A. Jakobovits ve Murray S. Miron (ed.), Engelwood Cliffs, NJ: Prentice-Hall, 1967. Şu internet adresinde: http://www.chomsky.info/articles/1967-.htm.



Resim 2.10. Noam Chomsky (Foto: Don J. Usner.)

> rülecektir... Pekiştirme kuramcılarının laboratuvarlarda ulaşmış olduğu içgörüler, gerçi oldukça sahici olsa da, bunları karmaşık insan davranışlarına uygulamak kaba ve yüzeysel bir yöntem olmaktan öteye geçemez, ayrıca dille ilgili davranışları sırf bu bağlamda tartışma girişimleri, göze alınması gereken ve temel önem taşıyan etkenlerden yoksundur...

Bu paragrafta Chomsky, keyfî biçimde karmaşık yapı barındıran, daha önce hiç işitilmemiş ve söylenmemiş cümlelerin olası sonsuz çeşitliliği salt deneyimle nasıl üretilir diye soruyor herhalde (gerçekten de bu çeşitliliği yaratabiliyoruz). Skinner'ın hiçe saydığı bu "temel önem taşıyan etkenler", Chomsky'ye göre, doğuştan gelen ama öğrenilmeyen dil becerileridir. "İnsanoğlu, [henüz] bilinmeyen özelliği ve karmaşıklığı olan, veri işleme ve 'hipotez formüle etme' becerisiyle birlikte, bu işi yapmak üzere bir şekilde özel olarak yaratılmıştır", der. Chomsky, tüm insanların "evrensel dilbilgisi" (ya da bunu yaratacak gelişim mekanizmaları) ile doğduğunu iddia eder; bunun da, insanların dil öğrenme ve kullanma becerilerinin çoğunu açıkladığını söyler.[18]

İçsel zihinsel süreçlere ve bunların kısıtlamalarına odaklanmayı sürdüren psikolog George A. Miller (1920-2012), kimi deneycilerin çalışmalarını çözümlemiş, insanların "anlık bellek" kapasitesinin, yaklaşık yedi bilgi "yığını" kadar olduğu sonucuna varmıştır.[19] Bu "sihirli sayı" hakkındaki makalesinin giriş bölümünde Miller, mizahi bir dille

18 Örneğin bkz. N. Chomsky, *Aspects of the Theory of Syntax*, Cambridge: MIT Press, 1965.

19 George A. Miller, "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information", *The Psychological Review*, c. 63, 1956, s. 81-97.

şunu belirtir: "Sorunum şu ki bir tam sayı bana işkence etti durdu. Yedi yıl boyunca bu sayı beni her yerde izledi, en mahrem verilerime sızdı ve en halka açık dergilerimizin sayfalarından bana saldırdı. Bu rakam, çeşit çeşit kılığa girmiş, bazen olağandan az ufak, bazen az iri görünmüş, fakat asla tanınmayacak kadar değişmemiştir. Bu rakamın bana böyle ısrarcı biçimde bulaşması, hiç de rasgele olmasa gerek". Daha da önemlisi, "anlık belleğin süresi, yığın başına düşen bit sayısından neredeyse bağımsız gibi görünüyor", demiştir. Yani, yığının neyi temsil ettiği hiç önemli değil, ister telefon numarasındaki tek bir basamak olsun, ister henüz bahsedilmiş bir insanın adı olsun, isterse de bir şarkının başlığı; belli ki dolaysız belleğimizde bu yığınlardan yedi (artı eksi iki) tane tutabiliyoruz.

Miller'ın "Sihirli Rakam Yedi" konulu makalesi, MIT'de 10-12 Eylül 1956 tarihinde düzenlenmiş Enformasyon Kuramı Sempozyumu'nda sunulmuştu.[20] Orada, Chomsky de önemli bir makale sunmuştu. "Dilin Tanımına Dair Üç Model" başlıklı makalesinde Chomsky, *öbek yapısı dilbilgisi* diye adlandırdığı birtakım sözdizimi kuralları ileri sürmüştür.[21] YZ araştırmaları öncülerinden ikisi (bunları ileride daha çok göreceğiz), o zamanlar Rand Şirketinde bilim görevlisi olan Allen Newell (1927-1992) ile Carnegie Teknoloji Enstitüsü (günümüzde Carnegie Mellon Üniversitesi) profesörlerinden Herbert Simon da (1916-2001) o sempozyumda makale sunmuşlardı; bu makalenin konusu, önermeli mantık kullanarak teorem ispatlayabilen bir bilgisayar programıydı. Örtüşen ilgi alanlarına sahip bilimcileri bir araya toplamış olan bu sempozyumun, zihin araştırmalarına adanmış yeni bir disiplin olan *bilişsel bilimin* doğumuna katkıda bulunduğu düşünülür. Gerçekten de George Miller şöyle yazmıştır:[22]

> Sempozyumdan ayrılırken, deneysel psikolojinin, kuramsal dilbilimin ve bilişsel süreçlerle ilgili bilgisayar benzetimlerinin, büyük bir bütünün parçası olduklarına, gelecekte bunların ortak tasalarının gittikçe rafineleşip eşgüdüm kazanacağına güçlü bir biçimde ikna olmuştum, gerçi bu kanaatim, mantığımdan ziyade içgüdülerime dayanıyordu...

1960'da Miller ve çalışma arkadaşları, TOTE birimi dedikleri (*Test-Operate-Test-Exit* [Sına – İşlet – Sına – Çık]) içsel bir mekanizmanın

20 *IRE Transactions on Information Theory*, c. IT-2, 1956.

21 Makalesinin bir nüshası için bkz. http://www.chomsky.info/articles/195609-.pdf.

22 George A. Miller, "A Very Personal History", MIT Center for Cognitive Science Occasional Paper, no. 1, 1979.

davranışlardan sorumlu olduğunu ileri süren bir kitap kaleme aldılar.[23] Bir etmenin (*agent*) sahip olabileceği her hedefe uyan bir TOTE birimi var, diyorlardı. Birim, kendi algısal becerilerinden faydalanarak, hedefe ulaşılıp ulaşılmadığını sınar. Hedefe ulaşılmışsa, birim dinlenmeye çekilir (çıkar). Hedefe ulaşılmamışsa, bu hedefe erişmeye özgü bir işlem gerçekleştirilir, hedefe ulaşılıp ulaşılmadığı yine sınanır ve hedefe varana dek bu süreç tekrarlanır. Basit bir örnek vermek gerekirse, çekiçle çivi çakmanın TOTE birimine bakalım. Çivi, tam çakılmadığı sürece (hedef), çekiçle buna vurulur (işlem). Nihayet hedefe erişildiğinde, çekiçle vurma kesilir (çıkış). Acaba söz konusu kitap yapay zekâ araştırmacılarına benzer bir çalışmayı esinlemiş midir, söylemek zor. Belli ki bu fikir "tedavülde" kalmıştı, çünkü, ileride göreceğimiz üzere, yaklaşık aynı zamanlarda YZ'de kimi erken çalışmalar çok benzer fikirlerden faydalanmıştır. [SRI'da robot Shakey için (ara düzey) davranış programlarıyla yapmış olduğum çalışmanın ve "teleo-reaktif" programlar dediğim şeylerle ilgili daha sonraki çalışmalarımın, Miller'ın fikirlerinden etkilendiğini söyleyebilirim.]

Bilişsel bilim, içsel zihinsel süreçleri mutlaka sinir fizyolojisine dayandırmaya uğraşmaksızın (en azından başlangıç yılları sırasında), hedefler, bellek, görev listeleri ve stratejiler gibi fikirleri kullanarak bu süreçleri açıklamaya kalkışmıştır.[24] Bilişsel bilim ve yapay zekâ, başlangıçlarından itibaren yakın ilişki içinde olmuştur. Bilişsel bilim, YZ araştırmacılarına ipuçları temin etmiştir; YZ ise, zihnin işleyişini anlama yolunda faydalı olacak yeni icat edilmiş kavramlarla bilişsel bilime yardımcı olmuştur.

2.2.3 *Evrim*

Canlıların evrim geçiriyor oluşu, zeki nesnelerin nasıl inşa edileceği konusunda bize iki ipucu veriyor. Birincisi ve daha iddialısı, bizzat evrim süreçlerinin, yani rasgele yaratım ve seçilimli sağkalım süreçlerinin, düşlediğimiz makineleri inşa etmek amacıyla bilgisayarlarda canlandırılmasıdır. İkincisi, gitgide daha zeki hayvanlar yaratırken evrimin izlemiş olduğu yollardan, daha zeki nesneler yaratmada rehber

23 George A. Miller, E. Galanter ve K. H. Pribram, *Plans and the Structure of Behavior*, New York: Holt, Rinehart & Winston, 1960.

24 Bilişsel bilimin tam bir tarihçesi için bkz. Margaret A. Boden, *Mind As Machine: A History of Cognitive Science*, 1. ve 2. ciltler, Oxford: Clarendon Press, 2006. Daha eski tek ciltlik bir inceleme için bkz. Howard E. Gardner, *The Mind's New Science: A History of the Cognitive Revolution*, New York: Basic Books, 1985.

olarak yararlanılabilir. Basit yönelimleri olan hayvanları taklit ederek başlayıp, o yollardan ilerleyerek daha karmaşık olanlar canlandırılabilir. Bu iki stratejiyi de YZ araştırmacıları şevkle uygulamıştır; ilerleyen bölümlerde bunu göreceğiz. Şimdilik, ilk çabalardan birkaçını anlatsak yeter.

Evrimi bilgisayarda canlandırmaya girişen ilk kişilerden biri, Princeton İleri Araştırmalar Enstitüsü'nde virüs genetikçisi Nils Aall Barricelli (1912-1993) olmuştur. Barricelli'nin 1954 tarihli makalesi, sayıların bir kafes yapısında taşınıp çoğaltıldığı deneyleri betimliyor.[25]

Karmaşık organizmalar üretme meselesinde biyolojik evrimin gösterdiği başarıdan ilham alan kimi araştırmacılar, programların yazılmaktan ziyade nasıl evrilebileceği hakkında kafa yormaya başlamıştı. R. N. Friedberg ile IBM'deki çalışma arkadaşları, yürüttükleri deneylerde,[26] rasgele bir bilgisayar programı kümesinden yola çıkarak, basit bir mantık problemini çözmede daha başarılı olan programlar evriltmeye çalışmışlardı. Friedberg, 1958 tarihli makalesinin özet kısmında şunu yazmıştır: "Makineler, hassas bir biçimde programlanmadıkları görevleri başarmayı öğrenseler, daha faydalı olurlardı... Depolanmış programlarla dolu bilgisayarda programların, bir öğrenme yöntemi sayesinde kademeli olarak iyileştirilebileceği ileri sürülmüştür; bu öğrenme yöntemi, pek çok programı deneyip bellekte, belirli bir yeri işgal eden komutlar arasından, başarılı sonuçla en sık ilişkili olanı seçer". Yani Friedberg, "başarılı" programlardan gelen komutları, sonraki "neslin" programlarına yerleştirmiştir, tıpkı artlarında döl bırakacak kadar başarılı olan bireylerin genlerinin yavrularına aktarılması gibi.

Ne yazık ki, Friedberg'in program evrimleştirme girişimleri çok başarılı olmamıştır. Marvin Minsky'nin işaret ettiği gibi:[27]

> [İlk makalede betimlenmiş olan] makine, aşırı basit kimi sorunları çözmeyi öğrenmiştir. Ancak, problemlerin salt şansa bağlı çözümünden

25 İngilizce çevirisi daha sonra ortaya çıkmıştır: N. A. Barricelli, "Symbiogenetic Evolution Processes Realized by Artificial Methods", *Methodos*, sayı 9, no: 35-36, 1957, s. 143-182. Baricelli'nin deneylerinin bir özeti için bkz. David B. Fogel, "Nils Barricelli-Artificial Life, Coevolution, Self-Adaptation", *IEEE Computational Intelligence Magazine*, c. 1, sayı 1, Şubat 2006, s. 41-45.

26 R. M. Friedberg, "A Learning Machine: Part I", *IBM Journal of Research and Development*, c. 3, sayı 3, 1959, s. 282-287. Makaleleri şu internet sitesinde bulmak mümkün (ücret karşılığında): http//www.research.ibm.com/journal/rd/033/ibmrd0303H.pdf.

27 Marvin L. Minsky, "Steps Toward Artificial Intelligence", *Proceedings of the Institute of Radio Engineers*, c. 49, 1961, s. 8-30. Makale şu internet adresinde: http://web.media.mit.edu/~minsky/papers/steps.html.

1000 kat uzun sürer bu çözüme ulaşmaları...

İkinci makale, bir dizi değişikliği konu ediyor... Bu değişikliklerle ve kimi "ön-hazırlık"larla (makineyi, doğru kulvarda faydalı talimatlarla birlikte çalıştırmak) birlikte bu sistem, salt şanstan ancak bir parça daha kötü hale gelmiştir.

Minsky, Friedberg'in yöntemlerinin zayıf performansını, her yavru makinenin annesinden sadece azıcık farklı oluşuna yorar, oysa faydalı herhangi bir iyileşme, olası makineler "uzay"ında çok daha büyük bir adım gerektiriyordu.

Yapay evrimle ilgili öteki erken çalışmalar daha başarılı olmuştur. Lawrence Fogel (1928-2007) ile çalışma arkadaşları, bir dizide sıradaki öğeyi öngörebilen makineler evriltebilmiştir.[28] Panoramic Research şirketinde Woodrow W. Bledsoe (1921-1995) optimizasyon sorununu ve Berkeley California Üniversitesi'nde Hans J. Bremermann (1926-1969) matematiksel problemleri çözmek amacıyla evrim benzetimlerinden faydalanmışlardır.[29] Ingo Rechenberg (bir YZ araştırmacısına göre), "optimal uçak kanatları ya da roket enjektörü ateşleme haznesi tasarımı gibi karmaşık optimizasyon görevlerini çözecek yapay evrim yönteminin öncüsüdür."[30]

Biyolojik evrimin ilham verdiği ilk belirgin çalışma, John Holland'ın 1960'lardan itibaren "genetik algoritmaları" geliştirmesi olmuştur. Michigan Üniversitesi'nde profesör olan Holland (1929-2015), ikili simge dizileri (0 ve 1'ler) kullanmıştı; biyolojik organizmaların genetik malzemesine atıfta bulunarak bunlara "kromozom" diyordu. (Holland, bu düşünceye ilk kez, 1950'lerin başlarında Michigan matematik kütüphanesinin açık raflarına göz atarken ulaştığını söylüyor.)[31] 0 ve 1'lere

28 Lawrence J. Fogel, A. J. Owens ve J. M. Walsh, *Artificial Intelligence through Simulated Evolution*, New York: Wiley, 1966.

29 Woodrow W. Bledsoe, "The Evolutionary Method in Hill Climbing: Convergence Rates", Teknik Rapor, Panoramic Research, Inc., Palo Alto, CA, 1962.; Hans J. Bremermann, "Optimization through Evolution and Recombination", M. C. Yovits, G. T. Jacobi ve G. D. Goldstein (ed.), *Self-Organizing Systems*, Washington, DC: Spartan Books, 1962, s. 93-106.

30 Jürgen Schmidhuber, "2006: Celebrating 75 Years of AI–History and Outlook: The Next 25 Years", şu kitapta: Max Lungarella ve diğ. (ed.), *50 Years of Artificial Intelligence: Essays Dedicated to the 50th Anniversary of Artificial Intelligence*, Berlin: Springer-Verlag, 2007. Scmiduber, Ingo Rechenberg'den alıntı yapar: "Evolutionsstrategie–Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution", doktora tezi, 1971 (yeniden basan Frommann-Holzboog Verlag, Stuttgart, 1973).

31 Bkz. http://www.aaai.org/AITopics/htm/genalg.html.

kodlanmış bir kromozom, belirli bir soruna çözüm olarak yorumlanabilir. Burada ana fikir, sorun çözmede gitgide daha hünerli olan kromozomlar evriltmekti. Kromozom kümeleri, tek tek kromozomların "mutasyonlara" (bir bileşeni 1'den 0'a değiştirmek ya da tam tersi) uğradığı evrimsel süreçlere maruz bırakılıyordu, ve her evrim safhasında en başarılı olan kromozom çiftleri, yeni bir kromozom oluşturmak üzere birleştiriliyordu. En sonunda bu süreç, sorunu çözen bir kromozom (ya da kromozomlar) barındıran bir nüfus oluşturuyordu.[32]

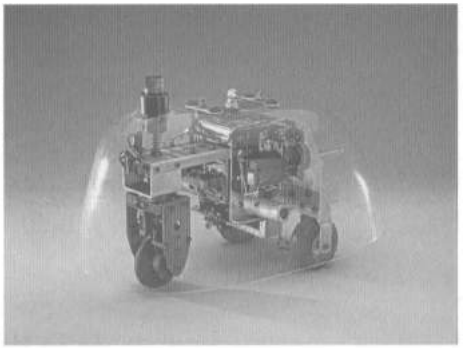
Önünde sonunda araştırmacılar, tüm bu evrimsel yöntemlerin, "eğim çıkış" ya da "tepe tırmanma" denilen çok faydalı matematiksel bir arama stratejisinin bir türü olduğunu fark etmişlerdi. Bu yöntemlerde, artan adımların olabildiğince en dik olanı ele alınarak, belli bir fonksiyonun yerel en büyük noktası aranır. (Yerel asgari değerin arandığı benzer yönteme, "eğim iniş" denir.)

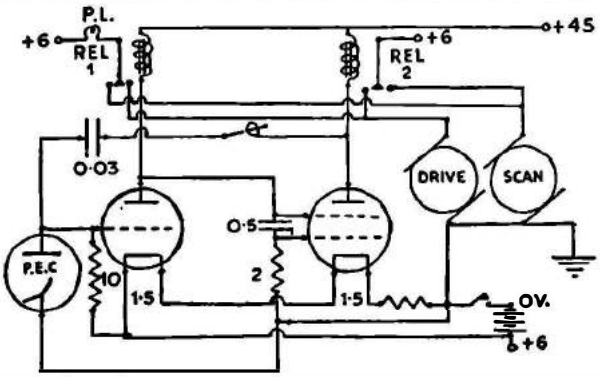
Bazı araştırmacılar, bizzat evrimi kopyalamak yerine, evrim yollarında ilerleyip zeki yaşama ulaşmaya çalışan makineler yapmayı tercih etmiştir. 1940'ların ikinci yarısında ve 1950'lerin ilk yarısında Britanyalı sinirfizyoloğu W. Grey Walter (1910-1977; Missouri Kansas City doğumludur), yaşamın en ilkel varlıklarından bazıları gibi hareket eden birtakım makineler inşa etmişti. Bunlar, tekerlekli araçlardı ve Walter onlara (bakan makine anlamında; bkz. Resim 2.11) *Machina speculatrix* gibi taksonomiye uygun bir isim vermişti.[33] Tosbağaya benzeyen bu makineler, çok basit akım-tüplü devrelerden meydana gelen "beyinlerin" denetimi altındaydı; bu beyinler, ışığa duyarlı sensörlerle çevrelerini duyumsayıp tekerlekli motorlarını denetliyorlardı. Devreler öyle düzenlenebiliyordu ki, makine, benzer bir makinenin üzerine yerleştirilmiş ışık kaynağına ya yaklaşıyor ya da ondan uzaklaşıyordu. Bu makinelerin davranışları maksatlı ve çoğunlukla karmaşık, tahmin edilemez bir görüntü çiziyordu, öyle ki Walter, "bunlar bir dereceye kadar özfarkındalığın kanıtı sayılabilir" demiştir. *Machina speculatrix*, sonraki araştırmacıların geliştireceği, gitgide gelişmiş hale gelecek "eylem yapan makine" çizgisinin başlangıcıydı.

32 John H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor: The University of Michigan Press, 1975. 2. baskı, MIT Press, 1992.

33 W. Grey Walter, "An Imitation of Life", *Scientific American*, Mayıs 1950, s. 42-45. Ayrıca bkz. W. Grey Walter, *The Living Brain*, Londra: Gerald Duckworth & Co. Ltd., 1953.







Resim 2.11. Grey Walter (üstte solda), *Machina speculatrix*'i (üstte sağda), devre diyagramı (altta). (Grey Walter fotoğrafı şu kitaptan alınmıştır: Hans Moravec, *ROBOT*, Bölüm 2: Caution! Robot Vehicle!, s. 18, Oxford: Oxford University Press, 1998; "Tosbağa" fotoğrafı, Ulusal Amerikan Tarihi Müzesi, Smithsonian Enstitüsü'nün izniyle; devre diyagramı şu kitaptan alınmıştır: W. Grey Walter, *The Living Brain*, s. 200, Londra: Gerald Duckworth & Co., Ltd., 1953.)

2.2.4 *Gelişim ve Olgunlaşma*

Belki de, bizzat evrimi yeniden yürütme ya da en ilkel hayvanlardan yola çıkarak yüksek karmaşıklığa giden yolları izleme dışında başka seçenekler vardır. İsviçreli psikolog Jean Piaget, küçük çocukların davranışlarını özenle inceleyerek, bebeklikten ergenliğe girerken düşünme becerilerinin olgunlaşmasında bir dizi evreden geçtiklerini ileri sürmüştür.[34] Acaba bu evreler, zeki nesne tasarımcılarına rehberlik edebilecek bir dizi adım temin edebilir miydi? Bebeğin yapabildikle-

34 B. Inhelder ve J. Piaget, *The Growth of Logical Thinking from Childhood to Adolescence*, New York:Basic Books, 1958. Bu safhaların bir özeti için şu internet adreslerine bkz. http://www.childdevelopmentinfo.com/piaget.sthml ve http://www.ship.edu/~cgboeree/piaget.html.

rini yapan bir makineyle yola çık, ardından bu merdivenin her kademesinde çocukların kazandığı becerileri taklit edebilen makineler tasarla. Bu strateji, evrim benzetimi kullanan "soyoluşsal" (filogenetik) stratejinin aksine, "bireyoluşsal" (ontogeni) olarak tabir edilebilir.

Elbette, bebek zihni taklit edilemeyecek kadar karmaşık, olgunlaşma süreçleriyse takip edilemeyecek kadar zorlu olabilir. Yine de, bu özel ipucu, faydalanılmayı bekliyor.

2.2.5 *Biyonik*

1960'ta bir sempozyumda ABD Hava Kuvvetleri Uzay Dairesi'nden Binbaşı Jack E. Steele, doğadan çıkardığı dersleri teknolojiye uygulayan alanı tanımlamak amacıyla "biyonik" terimini kullanmıştı.[35]

1960'lar boyunca, birkaç biyonik toplantısı ve biyonikle ilintili toplantı düzenlenmiştir. 1963 Biyonik Sempozyumunda Leonard Butsch ve Hans Oestreicher şunu yazmıştı: "Biyonik, milyonlarca yıllık evrim sırasında kendilerini en uygun yaşam şartlarına uyarlamış canlı sistemlerin evriminden faydalanmayı amaçlar. Evrimin en olağanüstü başarılarından biri, canlı sistemlerin bilgi işleme becerisidir; [bu becerinin araştırılması], biyonik araştırmalarının başlıca alanlarından birini teşkil eder."[36]

Günümüzde "biyonik" sözcüğü çoğunlukla protezlerle ve ortezlerle bağdaştırılır, örneğin yapay kulak salyangozu, yapay retina, yapay uzuv gibi. Bununla birlikte, YZ araştırmacıları kendi arayışlarını sürdürdükçe, canlıların, canlı evriminin ve gelişim süreçlerinin araştırılması, zeki nesneler inşa etme yolunda faydalı ipuçları sağlayabilir.

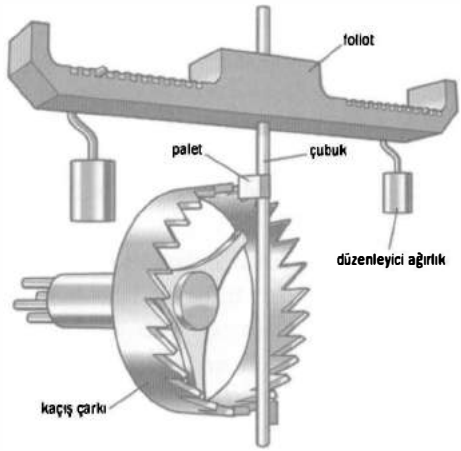
2.3 MÜHENDİSLİKTEN GELENLER

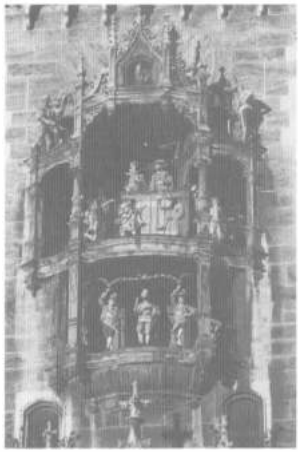
2.3.1 Otomatlar, Duyumsamak ve Geribildirim

Kendi kendine hareket eden, hatta kendi başlarına faydalı işler yapan makineler asırlardır vardır. Belki de ilk örnekleri arasında en yaygını, ağırlıkla çalışan saatlerdir (bkz. Resim 2.12). Bu saatler ilk olarak, ortaçağın ikinci yarısında İtalyan şehirlerinin kulelerinde belirdi. Bu saatin mekanizması, aşağı düşen ağırlığın enerjisini, kademeli

35 *Proceedings of the Bionics Symposium: Living Prototypes – the Key to new Technology*, Teknik Rapor 60-600, Wright Air Development Division, Dayton, Ohio, 1960.

36 *Proceedings of the Third Bionics Symposium*, Aerospace Medical Division, Air Force Systems Command, United States Air Force, Wright-Patterson AFB, Ohio, 1963.



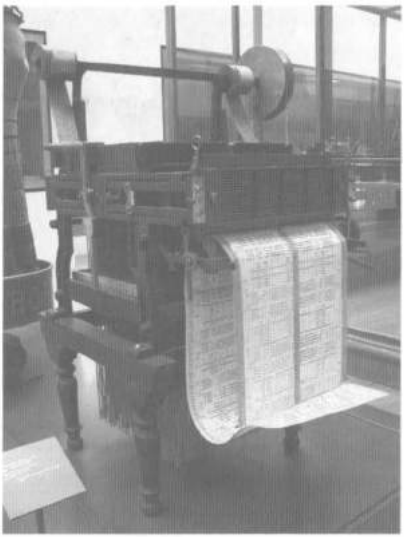


Resim 2.12. Ağırlıklı saat mekanizması (solda) ve Münih Glockenspiel'de otomatlar (sağda).

bir dönme hareketine çevirir, böylece saatin kolları hareket ettirilir. Münih'te Glockenspiel'inkiler misali, otomatların hareketlerini denetlemek amacıyla benzer mekanizmalar geliştirilmiştir.

Üretimde kullanılan ilk otomatik makinelerden biri, Joseph-Marie Jacquard'ın, 1804'te inşa edilmiş dokuma tezgâhıdır (bkz. Resim 2.13). Bu makine, uzun bir dokuma tezgâhı tarihinin mirasçısıydı ve Jacques de Vaucanson'un 1745'te üretmiş olduğu dokuma tezgâhındaki "delikli kart" tasarımını geliştirmişti. (Vaucanson, mekanik ördekler yapmakla yetinmemiştir.) Jacquard dokuma tezgâhının delikli kartları, mekiklerin hareketlerini denetliyor, kumaş tasarımlarının otomatik imalatını mümkün kılıyordu. İcat edilmesinin hemen birkaç sene ardından, Fransa'da dokuma yapan yaklaşık 10.000 Jacquard tezgâhı bulunuyordu. Delikli kâğıt ya da kart kullanma fikri, ileride, 1890 Amerikan nüfus sayımı verilerini sıralamak amacıyla Herman Hollerith tarafından benimsenecek, ayrıca otomatik piyanolarda kullanılacaktı (bunlarda kart yerine delikli rulolar vardı). Kavra-ve-yerleştir tarzıyla çalışan ilk fabrika "robotları", bu fikrin ancak çok az geliştirilmiş bir halinden faydalanıyordu.

Bu ilk makinelere, dışarıdan enerji kaynağı sağlamak yetiyordu (düşen bir ağırlık, sıkışan yay, pedal çeviren insanlar). Bunun dışında hareketleri tamamen otomatikti, insanların kılavuzluk etmesini hiç gerektirmiyorlardı. Fakat önemli bir kısıtlamaları vardı; kendi çevre şartlarını algılamıyorlardı. (Jacquard dokuma tezgâhının "okuduğu" delikli kartlar, çevrenin değil, makinenin parçası sayılır.) Çevre şartla-



Resim 2.13. Jacquard dokuma tezgâhı taklidi.

rını duyumsamak, sonra da duyumsanan şeyin, makinenin yaptığı işi etkilemesine izin vermek, zeki davranış için önemlidir. Örneğin Grey Walter'ın "tosbağaları", çevrelerindeki ışığın varlığını ya da yokluğunu tespit edebilen ışığa duyarlı sensörler bulunduruyor, buna uygun olarak hareket edebiliyordu. Dolayısıyla bunlar, Jacquard dokuma tezgâhından ya da çarklı otomatlardan daha zeki görünürler.

Çevreden duyumsanan şeyin davranışları etkilemesine olanak tanımanın en basit yollarından biri, "geribildirimli denetim"dir. Bu sözcük, makinenin davranışlarının kimi özelliklerini, diyelim ki işlem görme hızını, tekrar makinenin iç kısmına bildirmekten türüyor. Eğer geri bildirilen bu davranış özelliği, yine o özelliği azaltmaya ya da ters çevirmeye yarıyorsa, bu sürece "eksi geribildirim" denir. Öte yandan, eğer davranışın o özelliğini artırmaya ya da kuvvetlendirmeye yarıyorsa, "artı geribildirim" denir. İki geribildirim türü de mühendislikte son derece önemli roller oynar.

Eksi geribildirim teknikleri, mekanik cihazlarda yüzyıllardır kullanılageliyor. MÖ 270'te Yunan mucit ve berber İskenderiyeli Tesibius, suda yüzen bir regülatör icat etmişti; bu cihaz, depoya su akışını denetleyerek, bir su saatini besleyen bu deponun su düzeyini sabit derinlikte tutuyordu.[37] Geribildirim cihazı, bir değneğin ucuna iliştirilmiş mantardan oluşan şamandıralı bir vanaydı. Mantar, de-

37 http://www.mlahanas.de/Greeks/Ctesibius1.htm.

poda suyun üzerinde yüzüyordu. Depodaki su düzeyi yükseldiğinde, mantar da yükseliyor, değneğin, gelen suyu kesmesini sağlıyordu. Su düzeyi azaldığında, mantar da iniyor, değneğin, su akışını açmasına sebep oluyordu. Günümüzde sifonlu klozetlerdeki su düzeyi aşağı yukarı aynı şekilde ayarlanır. MÖ 250'de Bizanslı Philon, yağ kandilinde yağ düzeyini sabit tutmak için benzer bir şamandıralı regülatör kullanmıştı.[38]

İngiliz saatçi John Harrison (1693-1776), kendi yaptığı saatlerde bir tür eksi geribildirimden faydalanırdı. Saatin çevresindeki sıcaklık, denge yayının uzunluğunu, dolayısıyla zaman hassasiyetini etkiler. Harrison, bükülmesi sıcaklığa bağlı olan iki metalli bir şerit (bazen çubuk) kullanmıştı. Bu şerit, denge yayına öyle bir tutturulmuştu ki yayın uzunluğunda dengeleyici değişiklikler yaratıyor, böylece saatin sıcaklıktan bağımsız olmasına olanak tanıyordu. Şerit, sıcaklığı duyumsayıp saatin eskisine göre farklı ve daha hassas çalışmasını sağlar. Günümüzde bunun gibi iki metalli şeritler pek çok kullanım alanı bulmaktadır; bunların en göze çarpanı termostatlardır. (Dava Sobel'in *Longitude: The True Story of a Lone Genius Who Solved the Greatest Scientific Problem of His Time* (1995) [Boylam: Kendi Zamanının En Zorlu Bilim Problemini Çözen Yalnız Dahinin Gerçek Hikâyesi] başlıklı kitabı, Harrison'ın, açık denizde zamanı doğru ölçen ve ödül kazanmış saatini yapma çabalarını naklediyor.)

Belki de geribildirimli denetimin en canlı kullanımı, 1778'de James Watt'ın, buhar makinesinin hızını düzenlemek amacıyla geliştirdiği merkezkaç ağırlıklı mekanik regülatör toplardır (bkz. Resim 2.14). Makinenin hızı arttıkça, regülatör toplar dış çepere doğru açılır, böylece buna bağlı bir mekanizma hava akışını azaltır, makinenin hızı düşer, regülatör toplar içe doğru toplanır, bu da makinenin hızını artırır ve böylece dengeli bir hız tutturulmuş olur.

1940'ların ilk yarısında Norbert Wiener (1894-1964) ve kimi bilimciler, makineler ile hayvanların geri bildirimli denetim sistemlerinin özellikleri arasındaki benzerliklere dikkat çekmişti. Özellikle denetim devrelerinde yanlış uygulanan geribildirimin yol açtığı, denetlenen sistemin sarsak hareketleri, insan hastalardaki marazlı "titreme"lere benziyordu. Arturo Rosenblueth, Norbert Wiener ve Julian Bigelow, "sibernetik" terimini 1943 tarihli makalelerinde uydurmuşlardı. Wiener'in, aynı ismi taşıyan kitabı 1948'de yayımlandı. Bu sözcük, "yönetici" (*governor*) sözcüğüyle ilişkilidir. (Latincede *gubernaculum*, miğfer anla-

38 http://www.asc-cybernetics.org/foundations/timeline.htm.



Resim 2.14. Watt'ın ağırlıklı mekanik regülatörü.

mına gelir, *gubernator* da dümenci. Sibernetik sözcüğünün Latincesi, Yunancada dümencilik sanatı anlamına gelen *kibernetike* sözcüğünden türemiştir.[39])

Günümüzde "siber" öneki, bilgisayarlarla, robotlarla, internetle ve ileri benzetimle ilişkili neredeyse her şeyi tanımlarken kullanılıyor. Örneğin yazar William Gibson, "siberuzay" terimini 1984 tarihli bilimkurgu romanı *Neuromancer*'da uydurmuştur. Bununla birlikte teknik olarak sibernetik, geribildirimle ve denetimle ilişkili etkinlikleri tanımlamayı sürdürüyor.[40]

İngiliz psikiyatr W. Ross Ashby (1903-1972; Resim 2.15), "öte-denge" ve "içdenge" araştırmaları sayesinde sibernetik alanına katkıda bulunmuştur. Ashby'ye göre öte-denge, bir sistemin çeşit çeşit çevresel şartta dengeli hale ulaşma kapasitesidir. Bu fikri örnekle açıklamak amacıyla, "homeostat" dediği elektromekanik bir cihaz yapmıştır. Cihaz, sabitlenmiş dört mıknatıstan meydana gelir; mıknatısların konumu, geribildirim mekanizmaları sayesinde birbirlerine bağımlıdır. Herhangi bir mıknatısın konumu bozulduğunda, öbür mıknatısların etkisi ve o mıknatısın kendi kendine etkisi, tüm mıknatısların denge durumuna dönmesiyle sonuçlanır. Ashby bu cihazı, tanınmış kita-

39 Şu internet sitesinden: http://www.nickgreen.pwp.blueyonder.co.uk/control.htm.

40 Sibernetik tarihçesi için bkz. Amerikan Sibernetik Cemiyeti internet adresi, http://www.asc-cybernetics.org/foundations/history.htm.



Resim 2.15. Paris'te bir toplantıda W. Ross Ashby, Warren McCulloch, Grey Walter ve Norbert Wiener. (P. de Latil, *Thinking by Machine* içinde, 1956.)

bı *Design For a Brain*'in (1952) Sekizinci Bölümünde betimlemiştir. Onun fikirleri, birtakım YZ araştırmacılarını etkilemiştir. İleride anlatacağım "teleo-reaktif programlar"ımın çıkış noktası kısmen bu iç-denge fikriydi.

Sibernetikle ve biyonikle gevşek bağı olan birtakım başka fikirler, "öz-örgütlemeli sistemler" üzerine yapılan araştırmalardan doğmuştur. Atom ve molekül birleşimleri dahil, basit aksamın örgütsüz pek çok birleşimi, enerji dolu "itiş kakışa" tepki olarak dengeli hale girer; bu denge halinde aksam, daha karmaşık yığınlar olarak düzenlenir. Sibernetik ve sistemler kuramına adanmış bir internet sözlüğü, hoş bir örnek sunuyor: "Ataçlardan meydana gelen zincir, birinin zahmet edip ataçlarını birbirine tutturarak zincir oluşturduğunu akla getirir. Ataçların tabiatında zincir oluşturmak yoktur. Fakat, sürüyle ataç alıp, onları hafifçe açarak hepsini kokteyl karıştırıcıda sallarsanız, sonunda ataçların uzun ya da kısa zincirler oluşturacak şekilde düzenlendiğini görürsünüz. Bu zincirler, elle dizilen zincir kadar şık olmaz olmasına ama yine de zincirdir bunlar."[41]

41 Bkz. http://pespmc1.vub.ac.be/ASC/SELF-ORGANI.html.

"Öz-örgütlemeli" terimini kullanıma ilk olarak 1947'de Ashby'nin sunduğu anlaşılıyor.[42] Ashby, öz-örgütlenmenin, organizmanın çevreye ya da deneyimlere tepki olarak oluşan bir niteliği olmadığını, organizma ile çevrenin *birlikte ele alınan* bir niteliği olduğunu vurgulamıştı. Her ne kadar öz-örgütlenme, yaşamın nasıl ortaya çıktığıyla ilgili fikirlerde önem taşıyormuş gibi görünse de, zeki makineler inşa etme yolunda ipucu temin edip etmediği pek açık değildir.

2.3.2 İstatistik ve Olasılık

Neredeyse tüm uslamlama ve karar verme süreçleri bir belirsizlik ortamında gerçekleştiği için, belirsizlikle başa çıkmak, zekânın otomatikleştirilmesinde önemli rol oynar. Belirsizliğin ve "şans yasalarının" nicel değerini belirleme girişimleri, istatistiği ve olasılık kuramını doğurmuştur. Bayes kuralının, olasılık kuramında, en azından yapay zekâ için en önemli sonuçlardan biri olduğu anlaşılmıştır; bu kuralı hemen şimdi bir örnek bağlamında açıklayacağım. Söz konusu kural, adını İngiliz Rahip Muhterem Thomas Bayes'ten (1702-1761) almıştır.[43]

Bayes kuralının önemli uygulamalarından biri, sinyal tespitidir. Diyelim ki bir radyo alıcısı, gece yarısından sonra (rasgele) iki tondan biriyle, ya A ya da B tonuyla yayın yapan bir kanala ayarlı ve belli bir gece hangi tonla yayın yapıldığını bilmek istiyoruz. Herhangi bir gün, o gece hangi tonla yayın yapılacağını önceden biliyor değiliz, fakat diyelim ki olasılıklarından haberdarız. (Örneğin, iki tonun da olasılığı eşit olabilir.) Alıcıya gelen sinyali dinleyerek, hangi tonla yayın yapıldığını anlayabilir miyiz? Bir bakalım; sırf dinlemek bu meseleyi bütünüyle çözmez, çünkü radyo vericisi uzakta ve rasgele gürültü, tonu kısmen perdeleyebilir. Bununla birlikte, perdeleyici gürültünün doğasına bağlı olarak, o geceki tonun A (ya da B) olma olasılığını çoğunlukla hesaplayabiliriz. O halde sinyale y, gerçek tona ise x (ya A ya B olabilir) diyelim. Gelen sinyalin bu tona dair içerdiği bulgu göz önüne alındığında, $x=A$ olasılığı, $p(x=A \mid y)$ diye yazılır ve "sinyal y

42 W. Ross Ashby, "Principles of the Self-Organizing Dynamic System", *Journal of General Psychology*, c. 37, 1947, s. 125-128. Ayrıca şu internet adresine bkz. http://en.wikipedia.org/wiki/Self-organization.

43 Bayes, bu kuralın bir şeklini içerdiği söylenen bir deneme yazmıştır. İleride Marquis de Laplace (1749-1829), Bayes'in yapmış olduğu işi genelleyecekti (ondan bağımsız biçimde bu işi başardığını söyleyenler var). Bayes'in denemesinin (ölümünden sonra Richard Price tarafından yazılmış) bir şeklini için bkz. http://www.stat.ucla.edu/history/essay.pdf.

olduğunda x'in A olma olasılığı" olarak okunur. Aynı bulgu söz konusu olduğunda, $x=B$ olasılığı, $p(x=B\,|\,y)$'dir.

Makul bir "karar kuralı", eğer $p(x=A\,|\,y)$, $p(x=B\,|\,y)$'den büyükse, A tonu lehine karar verir. Aksi takdirde B tonunun lehine karar verir. (İki kararın da olası hatalarının "bedel"lerindeki farklılıkları hesaba katan dolambaçsız bir kural ayarlaması var.) Bu kuralı uygulamadaki sorun, bu iki olasılığın kolaylıkla hesaplanamayışıdır; işte burada Bayes kuralı devreye giriyor. Bayes kuralı, daha kolay tahmin edilen ya da normalde elde edilebilen başka olasılıklar bağlamında bu olasılıkları hesaplamamıza olanak tanır. Özelde Bayes kuralı şudur:

$$p(x\,|\,y) = p(y\,|\,x)p(x)/p(y)$$

Bayes kuralından faydalanarak, karar kuralımızı şöyle yazabiliriz:

> Eğer $p(y\,|\,x=A)p(x=A)/p(y)$, $p(y\,|\,x=B)p(x=B)/p(y)$'den büyükse, A tonu lehine karar ver. Aksi takdirde B tonu lehine karar ver.

$p(y)$ iki ifadede de bulunduğu, dolayısıyla hangisinin büyük olacağını etkilemediği için, denklemi şöyle sadeleştirebiliriz:

> Eğer $p(y\,|\,x=A)p(x=A)$, $p(y\,|\,x=B)p(x=B)$'den büyükse, A tonu lehine karar ver. Aksi takdirde B tonu lehine karar ver.

Bu tonların *a priori* olasılıklarını, yani $p(x=A)$ ile $p(x=B)$'yi bildiğimizi varsayıyoruz, dolayısıyla bize, $x=A$ ve $x=B$ için $p(y\,|\,x)$'i hesaplamak kalıyor. Bu ifadeye, x varken y'nin olabilirliği denir. Bu iki olasılık, $p(x=A)$ ile $p(x=B)$ eşit olduğunda (yani iki ton da *a priori* eşit oranda mümkün olduğunda), hangisinin olabilirliğinin daha yüksek olduğuna karar verebiliriz. Belirsizlik durumunda alınan pek çok karar, bu "azami-olabilirlik" yönteminden yararlanır. Bu olabilirlikler için yapılan hesaplama, alınan sinyal y'yi nasıl temsil ettiğimize ve araya giren gürültünün istatistiğine bağlıdır.

Benim örneğimde y, radyo sinyalidir, yani zamanla değişkenlik gösteren voltajdır. Bilişim bakımından, bu değişken voltaj, aldığı değerlerden uygun biçimde seçilmiş bir dizi örneklemle temsil edilebilir; eşit aralıklıklarla dağılmış zaman noktalarından alınan örneklemlerle, örneğin $y(t_1)$, $y(t_2)$,... $y(t_i)$,..., $y(t_N)$ gibi. Gürültü, bu noktaların normalde gürültü olmadan alacağı değerleri değiştirdiğinde, bunların dizisinin olasılığı (ton A olduğunda ve ton B olduğunda), gürültünün bilinen istatistiki özellikleri kullanılarak hesaplanabilir. Burada ayrıntılara dalacak değilim, ancak şunu diyebilirim ki, pek çok parazit istatistiği için bu hesaplamalar oldukça kolaydır.

20. yüzyılda Karl Pearson (1857-1936), Sör Ronald A. Fisher (1890-1962), Abraham Wald (1902-1950) ve Jerzey Neyman (1894-1981) gibi bilimciler ve istatistikçiler, parametreleri tahmin etme ve karar verme süreçlerinde istatistik ve olasılık yöntemlerinin kullanımına önemli katkılarda bulunanlar arasındaydı. Bunların çalışmaları, Bayes kuralının ilk mühendislik uygulamaları için bir temel döşemiştir, örneğin tıpkı az önce gösterdiğim, gürültünün sinyalleri perdelediği durumlarda en az iki elektrik sinyalinden hangisinin var olduğuna karar verme örneğindeki gibi. Bu uygulamalarla tanışmam, Amerikan mühendisler David Van Meter ile David Middleton'ın birlikte yazdıkları makale sayesinde olmuştur, ki 1955'te acemi bir yüksek lisans öğrencisiyken okumuştum onu.[44] Yapay zekâya gelince, Bayes kuralının bu kullanımları, hem konuşma seslerini hem de görsel imgeleri algılamanın nasıl mekanikleştireceği konusunda ipuçları sağlamıştır. Algı çalışmalarının ötesinde, Bayes kuralı, yapay zekâyla ilgili başka modern çalışmaların merkezinde yer alır.

2.3.3 Bilgisayar

A. İlk Bilişim Cihazları

Leibniz'in, Boole'un, Frege'ninkiler gibi önermeler, ileride yapay zekâ "yazılımı" olacak şeye bir temel sağlamaya yönelik ilk girişimler sayılabilir. Fakat uslamlama ve zeki davranışın tüm öteki yönleri, yazılımın ötesinde *fiziksel* bir motor da gerektirir. İnsanlarda ve hayvanlarda bu motor beyindir. Grey Walter ile Ross Ashby'nin basit cihazları, elbette fikirlerinin fiziksel olarak ortaya çıkmış halleriydi. Üstelik ileride göreceğimiz üzere, sinir hücresi benzeri birimlerden oluşan ilk ağlar, fiziksel biçim kapsamında kavranmıştır. Bununla birlikte, mantıktan, sinir fizyolojisinden, bilişsel bilimden gelen çoğu ipucunda yerleşik fikirleri keşfetmek için daha güçlü motorlara gereksinim duyulacaktı. McCulloch, Wiener, Walter, Ashby ve başkaları, zekânın makineleştirilmesi hakkında fikir yürütürken, çok güçlü ve gerekli bir makine hayat buldu: çok amaçlı sayısal bilgisayar. Bu makine, kendi başına tüm bu fikirlere ve daha fazlasına gereken motoru temin etmiştir. Zekânın otomasyonu yolunda şu ana dek en baskın donanım budur.

44 David Van Meter ve David Middleton, "Modern Statistical Approaches to Reception in Communication Theory", Bilgi Kuramı Sempozyumu, *IRE Transaction on Information Theory*, PGIT-4, Eylül 1954, s. 119-145.



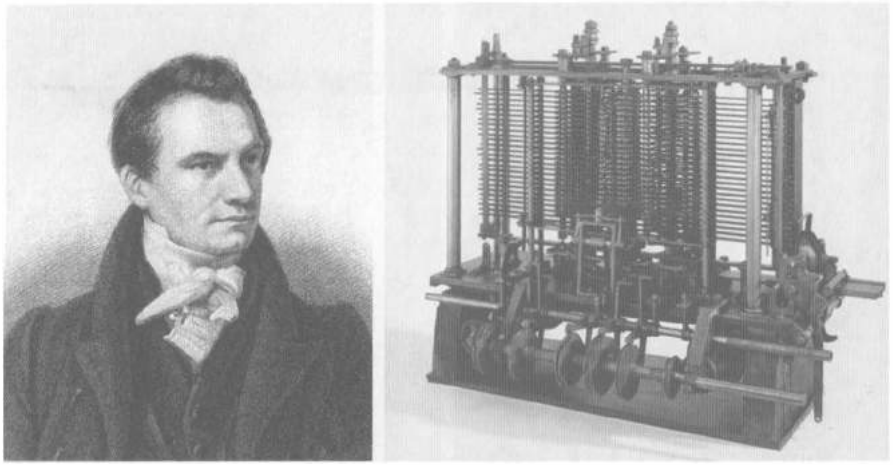
Resim 2.16. Wilhelm Schickard (solda) ve Blaise Pascal (sağda).

Hesap yapan makine inşasının uzun bir tarihçesi var. William Aspray'ın hazırlamış olduğu mükemmel kitap *Computing Before Computers* [Bilgisayarlardan Önce Hesaplama], hesaplamanın eski günlerini anlatır.[45] İlk makineler, aritmetik işlemleri yapabiliyor ama programlanamıyordu. Wilhelm Schickard (1592-1635; Resim 2.16), bu makinelerin ilklerinden birini 1623'te yapmıştı. Bu makinenin, gökbilim cetvellerindeki hesaplamalarda kullanma amacıyla altı basamaklı sayıları toplayıp çıkarabildiği söylenir. Makine, bir basamaktan ötekine "elde"leri saklayabiliyordu.

1642'de Blaise Pascal (1623-1662; Resim 2.16), kendisine ait yaklaşık elli hesaplama makinesinden ilkini yarattı. Bir basamaktan ötekine eldeleri tutabilen bir toplama makinesiydi bu. "Cihaz, bir yazı masasına ya da küçük bir yemek masasına sığacak kadar küçük bir kutuda tutuluyordu. Kutunun üst yüzeyi... pek çok dişli çarktan meydana geliyordu; bunların üstünde yer alan bir dizi küçük pencere sonuçları gösteriyordu. Çıkan sonuca bir sayı, örneğin 3 eklemek için, üç rakamıyla işaretlenmiş konumdaki dişli çarka küçük bir iğne sokup, iğne sabit noktaya gelene dek çarkı çevirmek gerekiyordu..."[46]

45 William Aspray (ed.), *Computing Before Computers*, Ames, Iowa: Iowa State University Press, 1990. Bkz. http://ed-thelen.org/comp-hist/CBC.html.

46 *A.g.e.*, Birinci Bölüm.



Resim 2.17. Charles Babbage (solda) ve Çözümlemeci Makinesinin bir modeli (sağda).

Pascal'ın makinelerinden esinlenen Gottfried Leibniz, 1674'te "Basamak Hesaplayıcısı" denen bir mekanik çarpma cihazı yapmıştı. Bu cihaz, toplayabiliyor, çıkarabiliyor, (toplama işlemini tekrarlama sayesinde) çarpabiliyordu. "Bir rakamı beşle çarpmak için, mili sadece beş kez çevirmek yetiyordu."[47]

Sonraki yüzyıllarda başka hesap makineleri de yapıldı. Kendi gününe göre anlaşılması fazlasıyla güç olan özellikle ilginç bir makine, 1822'de İngiliz matematikçi ve mucit Charles Babbage (1791-1871) tarafından tasarlanmıştır (bkz. Resim 2.17). "Farklılık Makinesi" denen bu cihazın görevi, sonlu farklılık yönteminden faydalanarak matematik cetvelleri (örneğin deniz seyrüseferinde kullanılan cetveller gibi) hesaplamaktı. Babbage'ın 2 Numaralı Farklılık Makinesi aslında (Babbage'ın tasarımları ve 19. yüzyılın mekanik imkânları kullanılarak) 1991'de yapılmıştır ve günümüzde Londra Bilim Müzesi'nde sergilenmektedir. Müze, cihazın başka bir kopyasını, Microsoft'un eski Teknoloji Şefi Nathan Myhrvold için inşa ettirmiştir. (Makinenin tasviri ve bununla ilgili bir film, Bilgisayar Tarihi Müzesi'nin internet sitesinde bulunabilir: http://www.computerhistory.org/babbage/)

Fakat toplama makineleri ancak toplama ve çıkarma işlemleri yapıyordu (bu işlemlerin tekrar edilmesiyle çarpma ve bölme işlemlerini de). Bunlar elbette önemli işlemlerdir ama ihtiyaç duyulan işlemler sadece bunlar değildi. 1834 ile 1837 arasında Babbage, "Çözümlemeci

47 *A.g.e.*

Makine" denen bir makinenin tasarımı üzerinde çalışmıştı; bu tasarım, genel hesaplama için gerekli olan çoğu fikri kapsıyordu. Ara sonuçları "değirmen" denilen bir birimde saklıyor, ayrıca programlanabiliyordu. Bununla birlikte, buharla çalışan, etkileşimli pirinç çarklardan ve kamlardan oluşan donanımını hayata geçirme aşamasında ödenek bulma sorunlarıyla karşılaşılmış, dolayısıyla asla inşa edilmemiştir.

Lord Byron'ın kızı Ada Lovelace (1815-1852), Çözümlemeci Makine için program tasarlama aşamasındaki iddia edilen rolünden ötürü" dünyanın ilk programcısı" sayılır. Ancak *Computing Before Computers* kitabında şu iddia ileri sürülüyor:[48]

> Romantik cazibesi olan bu imge, temelsizdir. Lovelace'in notlarında [Babbage'ın İtalya'da, Torino şehrinde vermiş olduğu seminerin açıklaması için yaptığı çeviride] atıfta bulunulan programların biri hariç hepsi, üç ila yedi sene önce Babbage tarafından hazırlanmıştı. O tek istisnaysa, Babbage tarafından Lovelace için hazırlanmıştı, gerçi Lovelace bu program içinde "hata" tespit etmiştir. Ada Lovelace'in, Çözümlemeci Makine için program hazırladığını gösteren hiçbir bulgu olmadığı gibi, Babbage'la mektuplaşmaları bunu yapacak bilgisinin olmadığını gösterir.

Çözümlemeci Makine ve bununla ilgili bir öykünücü (*emulator*) ve programlar hakkında daha fazla bilgi için bkz. http://www.fourmilab.ch/babbage/.

Kullanışlı bilgisayarlar, pirinç cihazlardan ziyade elektrikli cihazların icadını beklemek zorundaydı. 1940'ların başında ilk bilgisayarlar elektromekanik röleler kullanmıştı. Kısa süre sonra vakum tüpler (Britanya'daki adıyla, termoiyonik vanalar) daha hızlı ve güvenilir hesaplamalara olanak tanıdıkları için kullanılır oldu. Günümüzde bilgisayarlar, silikon plakalara dizilmiş milyarlarca minik transistör kullanıyor. Günün birinde bunların yerini ne alacak kim bilir?

B. Hesaplama Kuramı

İnsanlar gerçekten bilgisayar inşa etmeye bile başlamadan önce, 1930'larda kimi mantıkçılar ve matematikçiler, nelerin hesaplanabileceği konusuna kafa yormuşlardı. Alonzo Church, hesaplanabilen bir fonksiyonlar sınıfı belirlemişti; bunlara "özçağrılı" (*recursive*) fonksiyonlar demiştir.[49] İngiliz mantıkçı ve matematikçi Alan Turing

48 *A.g.e.*, İkinci Bölüm.

49 Alonzo Church, "An Unsolvable Problem of Elementary Number Theory", *American Journal of Mathematics*, c. 58, 1936, s. 345-363.



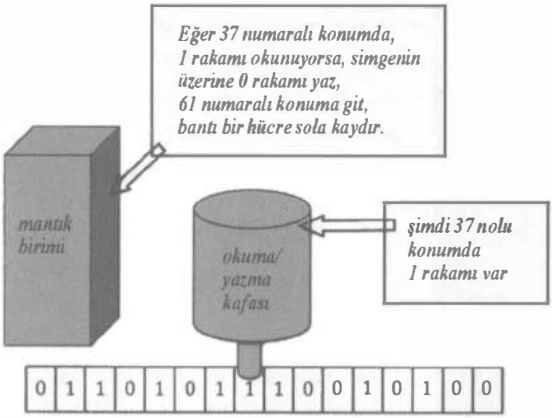
Resim 2.18. Alan Turing (Alan Mathison Turing, Elliott & Fry © Ulusal Portre Galerisi, Londra)

(1912-1954; Resim 2.18), günümüzde buna eşdeğer bir sınıf olduğu düşünülen bir şey önermişti; "mantıksal bilişim makinesi" (*logical computing machine;* LCM) dediği, günümüzdeyse "Turing makinesi" denen hayalî bir makineyle bu fonksiyonların hesaplanabileceğini söylemişti[50] (bkz. Resim 2.19). Bu iki düşüncenin eşdeğer olduğu iddiasına, "Church-Turing Tezi" denir. (Bu iddia kanıtlanmış değil, ama mantıkçılar bu iddiayı kuvvetle destekliyor ve karşıt bir örnek şimdiye dek bulunmadı.)[51]

Turing makinesi, anlaması oldukça basit, varsayımsal bir hesaplama makinesidir. Sadece birkaç aksamdan meydana gelir. Hücrelere bölünmüş sonsuz uzaklıkta bir bant (makinenin bir hayalden öteye geçmeyip inşa edilmeyişinin sebeplerinden biri budur) ve bir bant sürücüsü vardır. Her hücreye 1 ya da 0 rakamı basılmıştır. Makinenin, bantın tek hücresi üzerine konumlanmış bir okuma-yazma kafası bulunur. Okuma işlevi, bant üzerindekini okur. Okunan şeye ve mantık makinesinin durumuna göre, kendi durumunu değiştirmeye, yazma işlevine buyruk verip okunan hücreye ya 1 ya 0 yazdırtmaya (muhtemelen hücrede yazılı olan şeyi silip üzerine yazarak), bantı bir hücre sağa ya da sola kaydırmaya (her seferinde yeni hücre okunur ve bu

50 Alan M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem", *Proceedings of the London Mathematical Society,* Dizi 2, c. 42, 1936-1937, s. 230-265.

51 Turing hakkında daha fazla bilgi için, Turing biyograficisi Adrew Hodges'in hazırladığı şu internet sitelerine bkz. http://www.turing.org.uk/turing/.

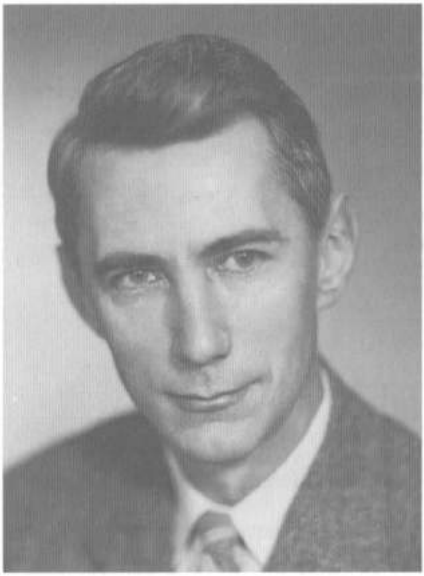


Resim 2.19. Turing makinesi

böyle sürer) ya da işlemi bütünüyle bitirmeye karar veren bir mantık birimi de vardır. Girdi (hesaplanacak "problem"), banta en başta işlenir. (Her böyle girdi, 1'ler ve 0'lar olarak yazılabilir.) Makine, işlemi durdurduğunda, eğer durdurursa, çıktı (girdi problemine verilen 1'ler ve 0'lar olarak kodlanmış "yanıt") bantın üzerine basılmış olur.

Turing, kendi makinesi için belli bir mantık biriminin (makinenin edimlerine karar veren aksam) daima belirlenebileceğini ve bu birim sayesinde, hesaplanabilir herhangi bir fonksiyonun hesaplanabileceğini ispatlamıştır. Daha önemlisi, bizzat bantın üstüne, belirli bir sorunu çözmek üzere uzmanlaşmış herhangi bir mantık birimi için talimatların işlenebileceğini, ardından *tüm* sorunlar için çok amaçlı bir mantık birimi kullanılabileceğini de göstermiştir. Özel amaçlı mantık birimi için yapılan kodlama işlemi, makine "programı" olarak düşünülebilir; bu mantık birimiyse, çözülecek sorunun tasviriyle birlikte bant üstüne depolanabilir (dolayısıyla, bizzat makinenin işleyişi sayesinde değişime açıktır!). Turing'in sözcükleriyle söylemek gerekirse: "Bu tür tek bir, özel makinenin, her türlü işi yapabileceği gösterilebilir. Aslında, herhangi başka bir makinenin modeli olarak çalıştırılabilir. Bu özel makineye, evrensel makine denebilir."[52]

52 Alıntı, Alan M. Turing'den: "Lecture to the London Mathematical Society", King's College, Cambridge, s. 112, yayımladığı yer: Alan M. Turing, *ACE Report of 1946 and Other Papers* (derleyenler B. E. Carpanter ve R. W. Doran, Cambridge, MA: MIT Press, 1986) ve *The Collected Works of A. M. Turing* kitabının üçüncü cildi (D. C. Ince (ed.), Amsterdam: North-Holland 1992.)



Resim 2.20. Claude Shannon.

C. Sayısal Bilgisayarlar

Turing'den bir parça bağımsız olarak mühendisler, programların kapsadığı komutları yerine getiren, programlardan ve mantık devrelerinden meydana gelen gerçek hesaplama makinelerini nasıl inşa edeceklerini düşünmeye başlamışlardı. Bilgisayarların mantık devreleri tasarlamada bazı kilit fikirleri, Amerikan matematikçi ve mucit Claude Shannon (1916-2001; Resim 2.20) geliştirmiştir.[53] Shannon, 1937'de Yale Üniversitesi'ne sunduğu yüksek lisans tezinde,[54] telefon anahtarlarını basitleştirmek amacıyla Boole cebirinin ve ikili aritmetiğin kullanabileceğini göstermişti. Aynı zamanda bu anahtar devrelerin (rölelerin, vakum tüplerin, vesairenin birleşimleriyle gerçekleştirilebilirler), Boole mantığındaki işlemlerin uygulanmasında kullanılabileceğini göstermiştir; dolayısıyla bilgisayar tasarımında bunların önemini açıklamış oldu.

Bir bilgisayar programını, bilgisayarın belleğinde, verilerle birlikte saklama fikrini ilk kimin düşündüğünü belirlemek zor. Programı saklamak, programda kolayca değişiklikler yapılmasına olanak tanır, ama daha önemlisi, programın saklandığı yerde uygun bellek parçalarını

53 Biyografik bir eskiz için bkz. http://www.research.att.com/njas/doc/shannonbio.html.

54 Howard Gardner, *The Mind's New Science* başlıklı kitabında bu tezi, "muhtemelen yüzyılın en önemli, ayrıca en ünlü yüksek lisans tezi", olarak nitelemiştir.

değiştirerek, programın kendi kendini değiştirmesine olanak sağlar. Bu fikri ilk düşünmüş olabilecek kişiler arasında, Alman mühendis Konrad Zuse (1910-1995) ve Amerikan bilgisayar öncüleri J. Presper Eckert (1919-1995) ile John W. Mauchly (1907-1980) var. (Elbette Turing, programa denk gelen bir kodu, evrensel Turing makinesinin bantı üzerinde saklamayı önermişti.)

Konrad Zuse'nin katkılarının ilginç bir tarihçesi için şu adresteki internet sitelerine bakılabilir: http://irb.cs.tu-berlin.de/~zuse/Konrad_Zuse/en/index.html. Buradaki internet sitelerinden biri şundan bahsediyor: "Konrad Zuse'nin Z3'ü hiç tartışmasız dünyanın programla denetlenen (serbestçe programlanabilen) tam işlevsel ilk bilgisayarıydı... Z3, 12 Mayıs 1941 tarihinde Berlin'de bilimcilerden oluşan bir izleyici kitlesine sunulmuştur". Z3'te, vakum tüpler yerine 2.400 adet elektromekanik röle kullanılmıştı. Asıl Z3, 21 Aralık 1943'te müttefiklerin hava saldırısı sırasında imha olmuştur.[55] 1960'ların başlarında bunun bir kopyası yeniden inşa edildi ve günümüzde Münih'teki Alman Müze'sinde sergileniyor. Ayrıca Zuse'nin, Plankalkül denilen ilk programlama dilini yaratmış olduğu da söylenir.

Amerikan matematikçi John von Neumann (1903-1957), program saklayan ilk bilgisayarlardan EDVAC hakkında bir "taslak rapor" yazmıştı.[56] Belki de bu rapordan ötürü, günümüzde bu gibi bilgisayarların "von Neumann mimarisi" kullandığını söylüyoruz. İdeal von Neumann mimarisi, (göreve özgü) saklanmış programı, herhangi bir programın komutlarını (sıralı olarak) icra eden (çok amaçlı) donanım devrelerinden ayırır. (Genelde programa "yazılım" deriz ki bilgisayarın "donanım" kısmından bunu ayırt edebilelim. Ancak günümüzün çoğu bilgisayarında bu ayrım bulanıklaşmıştır, zira çoğu bilgisayarda programlardan bazıları doğrudan devrelerin içine yerleştirilir.)

Program saklayan başka bilgisayarlar 1940'larda Almanya'da, Büyük Britanya'da, ABD'de tasarlanıp inşa edilmiştir. Bunlar iri, cüsseli makinelerdi. Büyük Britanya'da ve ABD'de çoğunlukla askerî amaçlar için kullanılmışlardır. Resim 2.21 bu tür bir makineyi gösteriyor.

Bilgisayarlara "makine" diyoruz ama günümüzde bilgisayarlar, hiç hareketli aksam olmaksızın tamamen elektrikle çalışacak şekilde imal edilebiliyor. Dahası, hesaplama (*computing*) makinesi dediği-

55 Hava saldırısı için çeşitli kaynaklar farklı tarihler vermiştir, ama Zuse'nin oğlu Horst Zuse'nin elindeki bir mektup, 1943 tarihini veriyor (Horst Zuse'yle iletişim kurmuş olan Wolfgang Biebel'in bana 10 Şubat 2009'da göndermiş olduğu e-postaya göre).

56 Bu raporun bir nüshası ve tanıtıcı bir yorum için bkz. http://qss.stanford.edu/~godfrey/.



Resim 2.21. Cambridge Üniversitesi EDSAC Bilgisayarı (y. 1949). (Telif hakkı, Bilgisayar Laboratuvarı, Cambridge Üniversitesi.)

mizde, genelde bilgisayar ve bunun yürüttüğü programın birleşimini kastediyoruz. Bazen, sırf programa bile makine dediğimiz oluyor. (Bu kullanıma örnek vermek gerekirse, ileride bahsedeceğim "dama oynayan makine" aslında dama oynayan bir programdır.)

Program yüklenmiş sayısal bilgisayarların büyük önemi, herhangi bir amaç için kullanılabilmelerinde yatar; yani, herhangi bir bilişim amacı için elbette. Modern sayısal bilgisayar, pratikte, bu tür bir evrensel makinedir. "Pratikte" nitelemesi gerekli, çünkü Turing'in sonsuz bantının aksine modern bilgisayarların bile sonsuz saklama kapasitesi yok. Bununla birlikte, saklama alanları olağanüstü büyüktür, dolayısıyla pratikte evrenseldirler.

D. "Düşünen" Bilgisayarlar

İlk bilgisayarlar inşa edildikten sonra, Turing, eğer bunlar uygulama bakımından evrenselse, her şeyi yapabiliyor olmaları gerekir mantığını yürütmüştü. 1948'de şöyle yazmıştır: "Evrensel makinenin önemi açık. Farklı işler için sonsuz sayıda farklı makine gerekmiyor. Tek ma-

kine tüm işlere yetecektir. Farklı işler için farklı makineler üretmeyi ele alan mühendislik sorununun yerine, evrensel makineyi bu işleri yapmak üzere "programlama" gibi masa başı bir iş geçmiştir."[57]

Turing'in, bilgisayarların yapabileceğini düşündüğü şeyler arasında, insan zekâsını taklit etmek de vardı. Turing'in yaşam öyküsünü kaleme alan yazarlardan biri olan Andrew Hodges, şunu iddia ediyor: "Hesaplanabilir şeylerin alanının, açık yazılmış komut dizilerinden çok daha fazlasını kapsayacağına inanıyordu; öyle ki bu ölçeğin, ne kadar yaratıcı ya da özgün olursa olsun insan beyninin yaptığı her şeyi kapsayacak kadar büyük olabileceğini düşünüyordu. Yeterli karmaşıklık barındıran makinelerin, açıkça programlanma gerektirmeyen insan davranışlarını evriltme kapasitesi olacaktır, diyordu."[58]

İnsan tarzı zekâyı *bütünüyle* mekanikleştirme olasılığını ele alan ilk modern makaleyi 1950'de Turing yayımlamıştır.[59] Bu makale, birkaç sebepten ötürü ünlüdür. Öncelikle, Turing, "Makineler düşünebilir mi?" sorusunun fazlasıyla muğlak olduğunu düşünüyordu. Bunun yerine, makine zekâsı meselesinin, "Turing testi" denecek şey tarafından çözüme kavuşturulmasını önermişti.

Her ne kadar bu test birkaç şekilde ifade edilmiş olsa da (çoğunlukla basitleştirilmiştir), bizzat Turing bunu şöyle betimlemişti:

> Bu sorunun [Makineler düşünebilir mi?] yeni biçimi, "taklit oyunu" diyebileceğimiz bir oyun bağlamında tanımlanabilir. Oyun, üç kişiyle oynanır: bir erkek (A), bir kadın (B) ve herhangi bir cinsiyetten bir sorgucu (C). Sorgucu, öteki ikisinden ayrı bir odada durur. Sorgucu için oyunun amacı, öteki iki kişiden hangisinin erkek, hangisinin kadın olduğunu belirlemektir. Onları X ve Y isimleriyle tanır ve oyunun sonunda ya "X, A'dır ve Y, B'dir" ya da "X, B'dir ve Y, A'dır", der. Sorgucu, A'ya ve B'ye şu tür sorular sorabilir:
>
> C: Lütfen X bana saçının uzunluğunu söylebilir mi?
>
> Diyelim ki X aslında A, o zaman bu soruya A yanıt vermek zorunda. Oyunda A'nın hedefi, C'nin yanlış tespitte bulunmasını sağlamaya çalışmaktır. Dolayısıyla yanıtı şöyle olabilir:
>
> "Saçlarım kısa kesimli, en uzun tel yaklaşık 20 santimetre."

57 Alan M. Turing, "Intelligent Machinery", National Physical Laboratory Report, 1948. Yeniden baskısı: B. Meltzer ve D. Michie (ed.), *Machine Intelligence* 5, Edinburgh: Edinburgh University Press, 1969. Bu raporun bir kopyası için bkz. http://www.AlanTuring.net/intelligent machinery.

58 Andrew Hodges, *Turing*, Londra: Phoenix, 1997.

59 Alan M. Turing, "Computing Machinery and Intelligence", *Mind*, c. LIX, no: 236, Ekim 1950, s. 433-460. (Bkz. http://www.abelard.org/turpap/turpap.htm.)

Ses tonu sorgucuya yardımcı olmasın diye, yanıtlar kâğıda yazılmalı ya da daha da iyisi daktilo edilmeli. En uygun düzenleme, iki oda arasında iletişimi sağlayan bir teleyazıcı olurdu. Buna seçenek olarak, soru ve yanıt bir aracı çizerinden iletilebilir. Oyunda üçüncü oyuncunun (B) hedefi, sorgucuya yardım etmektir. B için en iyi strateji muhtemelen doğru yanıtlar vermektir. Yanıtlarına, "Ben kadınım, o adama kulak asma!" gibi eklemeler bile yapabilir, fakat bu çaba nafiledir, çünkü erkek de benzer sözler söyleyebilir.

O halde şu soruyu soruyoruz: "Bu oyunda A'nın yerine bir makine geçerse ne olur?" Bu durumda sorgucu, oyun kadınla erkek arasında oynanırken sık sık başına geldiği gibi yine yanlış karar verir mi? Bu sorular, bizim özgün "Makineler düşünebilir mi?" sorumuzun yerini alır.

Turing testinin sadeleştirilmiş biçimlerinden pek çoğu, "öteki odada", sorgucuyu kandırıp kendisinin insan olduğuna inandırmaya çalışan bir bilgisayar bulunmasını içerir. Bu hedefe ulaşılmasını teşvik etmek amacıyla ödül konmuştur; aslında yıllık dağıtılır bu ödüller. Ödülü duyuran internet sitesinin sözcükleriyle söylemek gerekirse: "1990'da Hugh Loebner, Cambridge Davranış Araştırmaları Merkezi'yle anlaşıp Turing testine hayat verecek bir yarışmanın maliyetini üstlendi. Dr. Loebner, tepkileri bir insanınkinden ayırt edilemeyecek ilk bilgisayara verilecek 100.000 dolarlık bir Büyük Ödül ve bir Altın Madalya sözü verdi. Böyle bir bilgisayarın 'düşündüğü' söylenebilir. Her sene, insana en yakın bilgisayara yıllık 2000 dolarlık ödül ve bronz madalya vaat ediliyor. Yıllık yarışmanın kazananı, mutlak anlamda ne kadar iyi olduğuna bakılmaksızın, o sene içindeki tüm katılımcılar arasından en iyisi olacaktır."[60]

Ne var ki, sadeleştirilmiş biçimlerinin, hatta özgün testin kimi yetersizliklerine işaret edenler de çıkmıştır.[61] Aslında, Joseph Weizenbaum'un zeki olmadığı aşikâr sohbet programı ELIZA'nın (1966) bile insanları kandırıp insan olduğunu düşündürttüğü de söylenmiştir.[62]

Örneğin, aşağıda ELIZA'yla yapılmış bir sohbetin kısmi dökümü var:

60 Bkz. "Home Page of the Loebner Prize in Artificial Intelligence", şurada:http://www.loebner.net/Prizef/loebner-prize.html.

61 Tartışma için Wikipedia makalesine bkz. http://en.wikipedia.org/wiki/Turing.test.

62 Joseph Weizenbaum, "ELIZA – A Computer Program for the Study of Natural Language Communication between Man and Machine", *Communication of the ACM*, c. 9, no: 1, Ocak 1966, s. 35-36. Bkz. http://i5.nyu.edu/~mm64/x52.9265/january1966.html.

ELIZA: Nasıl gidiyor. Lütfen sorununu söyle.
> Bilgisayarlarla çok vakit geçiriyorum.
ELIZA: Neden bilgisayarlardan bahsettin?
> Sen bir bilgisayarsın ya.
ELIZA: Bilgisayar olduğuma inanmak seni hoşnut ediyor mu?
> Evet.
ELIZA: Oldukça olumlu görünüyorsun.
> Sanırım.
ELIZA: Ne zamandır?
> 31 yıldır.
ELIZA: Seni tam anladığımı sanmıyorum.
> Sıraya gir.
ELIZA: Lütfen devam et.
> Pek de mantıklı konuşmuyorsun.
ELIZA: Benden değil senden bahsediyoruz.

Sohbetin ilk kısmı makul görünüyor ama orta kısımda ELIZA batağa saplanıyor, çünkü program, "sanırım" sözünün ardından "mutluyum" gibi bir sözcük gelmesini bekliyordu. (İnternette birkaç ELIZA benzetimi bulunuyor. Özgün ELIZA programını aslına sadık biçimde tekrarladığı iddia edilen program için şu internet adresine bkz.: http://www.chayden.net/eliza/Eliza.html. Siz de deneyin!)

Turing'in makalesinin ikinci önemli özelliği, zeki bilgisayarlara ulaşma olasılığı aleyhine ileri sürülebilecek savları ele alışıydı. Turing'in bahsetmiş olduğu savlardan alıntı yapacağım:

(1) Dinî İtiraz: Düşünmek, ölümsüz insan ruhunun işlevidir. Tanrı, her insana ölümsüz bir ruh bahşetmiştir, ama hiçbir hayvana ya da makineye ruh vermemiştir. Dolayısıyla hayvanlar ve makineler düşünemez.

(2) "Kafayı Kuma Gömme" İtirazı: "Düşünen makinelerin yaratacağı sonuçlar, fazlasıyla ürkütücü olacaktır. Umalım ve inanalım ki öyle olmasın."

(3) Matematiksel İtiraz: Ayrık durumlu makinelerin güçlerinin kısıtlı olduğunu göstermek üzere kullanılabilecek, matematiksel mantıkla elde edilmiş pek çok sonuç var.

(4) Bilinçlilikten Doğan İtiraz: Bu sav, Profesör Jefferson'un 1949'daki Lister Nutku'nda gayet güzel ifade edilmiştir; oradan alıntı yapıyorum:

> Bir makine, simgeleri gelişigüzel yerleştirerek değil ama düşüncelerinden ya da hissettiği duygulardan ötürü sone yazmadığı ya da konçerto bestelemediği müddetçe, makinenin beyne denk olduğunu kabul edemeyiz; yani yazmakla kalmamalı, bunu kendisinin yazdığının da farkında olmalı. Hiçbir mekanizma, başarılarından ötürü haz, vanaları eridiğinde keder duyamaz, övgü aldığında içi ısınmaz, hatalarıyla sefil duruma düşmez, cinsellikten büyülenmez, istediğini alamayınca öfkelenmez ya da kederlenmez (yapay bir sinyalden bahsetmiyorum, bu kolay bir tertibat olurdu).

(5) Çeşitli Yetersizliklerden Doğan Savlar: Bu savlar şu biçimi alıyor: "Bahsettiğiniz her işi yapan makineler inşa edebileceğinizi kabul edebilirim ama işini yapacak bir makine asla inşa edemeyeceksiniz."

(6) Leydi Lovelace'in İtirazı: Babbage'ın Çözümlemeci Makinesi'yle ilgili en ayrıntılı bilgimiz Leydi Lovelace'in hatıratına dayanıyor. Bu hatıratta Lovelace şunu söyler: "Çözümlemeci Makinenin, herhangi bir şey icat ediyormuş iddiası yok. *Nasıl yapılacağını bildiğimiz her şeyi, buyruk verdiğimizde* yapabilir" (vurgular ona ait).

(7) Sinir Sisteminde Süreklilik Savı: Sinir sistemi kesinlikle ayrık durumlu bir makine değildir. Sinir hücresine etki eden sinirsel itkinin boyutuyla ilgili küçük bir hata, dışarı gönderilecek itkinin boyutunda büyük bir farklılık yaratabilir. Bu durumda, ayrık durumlu bir sistemin, sinir sistemi davranışlarını taklit edememesi beklenebilir.

(8) Davranışın Rasgele Oluşundan Kaynaklanan Sav: İnsanın akla gelen her şartta ne yapacağını tanımlamayı iddia eden bir kurallar takımı oluşturmak mümkün değil.

(9) Duyu-Ötesi Algıdan Doğan Sav.

Turing bu makalesinde tüm bu noktalarla (bence) hoş bir biçimde başa çıkıyor, bir tek son itiraz belki istisna olabilir (duyu ötesi algının makul olduğunu düşündüğü için besbelli). Karşıt savlarını görmek için Turing'in 1950 tarihli makalesini okumayı size bırakıyorum.

Turing'in 1950 tarihli makalesinin üçüncü önemli özelliği, insan düzeyinde düşünsel beceriler taşıyan programları üretmeye nasıl başlayacağımız konusundaki önerisidir. Makalenin sonuna doğru şunu söyler: "Yetişkin zihnini taklit eden program yaratmaya çalışmak yerine, çocuk zihnini taklit eden program yaratmak neden olmasın? Bu



Resim 2.22. Herbert Siemon (oturan) ve Allen Newell (ayakta duran). (Carnegie Mellon Üniversitesi Arşivinin izniyle.)

program uygun bir eğitimden geçirildiğinde yetişkin beyni elde edilebilir". Bu öneri, zeki makineler geliştirme yolunda bireyoluşsal strateji kullanmak hakkında daha önce belirtmiş olduğumuz fikrin gerçek kaynağıdır.

Allen Newell ile Herb Simon (bkz. Resim 2.22), sayısal bilgisayarın evrenselliğinin, doğru yazılıma sahip olduğu takdirde, zekâyı tüm dışavurumlarıyla mekanikleştirmek için kullanılabileceği anlamına geldiğine pekâlâ inanan kişiler arasındaydılar. 1975 ACM Turing Ödülü konuşmalarında[63] betimlemiş oldukları "Fiziksel Simge Sistemi Hipotezi" adlı hipoteze şüphesiz çok önceden inanmaya başlamışlardı. Bu hipotez, "fiziksel bir simge, zeki eylem için gerekli ve yeterli imkânlara sahiptir", der. Dolayısıyla, hipoteze göre, uygun şekilde programlanmış sayısal bilgisayarların zeki eylemler yapma gücü vardır. Tersi olarak şu da söylenebilir: İnsanlar zekâ eseri eylemler yapma güçleri olduğu

63 AllenNewell ve Herbert A. Simon, "Computer Science as Empirical Inquiry: Symbols and Search", *Communications of the ACM*, sayı 19, no: 3, Mart 1976, s. 113-126.

için, bu hipoteze göre birer fiziksel simge sistemidir. Bunlar, hâlâ tartışılan çok güçlü iddialardır.

Hem hayalî Turing makinesi, hem de gerçek sayısal bilgisayar, Newell ile Simon'un kastettiği anlamda simge sistemleridir. Fakat, üzerine 0'lar ve 1'ler basılı bir bant kullanan Turing makinesi nasıl olur da "simge sistemi" sayılır? Şöyle düşünelim: banta basılı 0'lar ve 1'ler, bunlarla ilişkili rakamların yerine geçmiş simgeler olarak düşünülebilir. "A" ve "M" gibi başka simgeler, örneğin 0'lar ve 1'ler gibi ilkel simge dizileri olarak kodlanabilir. Sözcüklerse harf dizileri olarak kazılabilir vs. Genelde sayısal bilgisayarı, 0'lar ve 1'ler işleyen bir makine sayışımız, aynı zamanda daha karmaşık simgeleri de işlediğini düşünmekten bizi alıkoymasın. Unutmayalım ki, hepimiz bilgisayarı "sözcük işlemek" ve e-posta göndermek amacıyla kullanıyoruz.

Newell ile Simon, hipotezlerinin aslında yanlış olabileceğini kabul etmişlerdir: "Akıllı davranışı oluşturmak, herhangi bir sistemin bunu zoraki sergileyebileceği kadar kolay bir şey değil. Gerçekten de, ya felsefi ya da bilimsel zemine dayanan çözümlemeleri sonucu bu hipotezin yanlış olduğu sonucuna ulaşmış insanlar var. Bilimsel açıdan, insan bu hipotezi ancak doğal dünyayla ilgili deneysel bulguları öne çıkararak çürütebilir ya da savunabilir". Ardından şu sonuca ulaşırlar:

> Simge sistemi hipotezi, insanın simgesel davranışının, fiziksel simge sistemi özellikleri barındırdığı için ortaya çıktığını ima ediyor. Bu yüzden, insan davranışını modelleme çabalarının sonuçları, bu hipotez için önemli birer bulgu olmuştur ve yapay zekâ araştırmaları, bilgi işleme psikoloji denen alanın araştırmalarıyla işbirliği içinde seyretmektedir.

Her ne kadar bu hipotez, 1976 tarihli makalede yayımlanana dek resmen betimlenmiş olmasa da, 1950'lerde Turing ve başka araştırmacıların inandıkları görüşte kesinlikle örtük biçimde vardı. Allen Newell'in vefatının ardından, Herb Simon şu cümleyi kaleme almıştır: "Ta en baştan itibaren araştırmalara, fiziksel simge sistemi hipotezi benzeri bir şey gömülmüştü."[64]

Andığımız ipuçlarından esinlenen ve genel amacı sayısal bilgisayarla donanmış araştırmacılar, 1950'li yıllar boyunca, zekâyı mekanikleştirmeye açılan çeşitli yolları keşfetmeye koyuldu. Simge sistemi hipotezine sıkı sıkıya inanan kimileri, insanların becerebildiği bazı düşünsel görevleri yaptırtmak amacıyla bilgisayar programlamaya

64 National Academy of Sciences, *Biographical Memoirs*, c. 71, 1997. Bkz. http://www.nap.edu/catalog.php?record_id=5737.

başladı. Yaklaşık aynı dönemde başka araştırmacılar, salt simge işlemeye bel bağlamayan yaklaşımları keşfetmeye girişti. Bunlar ilhamlarını genelde, McCulloch ile Pitts'in sinir hücresi benzeri birimlerden oluşan ağlar üzerindeki çalışmalarından ve karar vermekle ilgili istatistiksel yaklaşımlardan almıştır. Simge işleme yöntemleri ile "beyin tarzı" ve "simgesel olmayan" diye adlandırılan yöntemler arasındaki ayrım bugün de sürüyor.

İKİNCİ KISIM

İLK KEŞİFLER: 1950'LER VE 1960'LAR

Eğer makineler zeki olacaksa, en azından, insanların becerebildiği, düşünceyle ilintili işleri yapabilmek zorunda. Bu durumda, yapay zekâ arayışında ilk adımlar, zekâ gerektiren bazı işler saptama ve makinelere bunların nasıl yaptırılacağını belirlemek olmuştur. Bulmaca çözmek, satranç ve dama benzeri oyunlar oynamak, teoremler ispatlamak, basit sorulara yanıt vermek, görsel imgeleri sınıflandırmak gibi işler, 1950'lerin ve 1960'ların birinci yarısı boyunca ilk öncülerin boğuştuğu problemlerin arasında yer almıştır. Her ne kadar bunların çoğu, zaman zaman "oyuncak" denen laboratuvar tarzı problemler olsa da, ticari önem taşıyan kimi gerçek dünya problemleriyle de boğuşulmuştur, örneğin banka çeklerinin stilize manyetik karakterlerini otomatik olarak okumak ve bir dilden ötekine çeviri gibi. (Bildiğim kadarıyla, "oyuncak problem" deyimini ilk kullanan, Seymour Papert'tı. 1967'de Georgia eyaletinin Athens şehrinde katıldığım bir YZ çalıştayında, yapay zekâda *tau,* yani "oyuncak" problemler ile *rho,* yani gerçek-dünya problemleri arasında, ayrıca *theta,* yani "kuram" sorunları arasında ayrım yapmıştı. Bu ayrım günümüzde de gayet iyi iş görüyor.)

Kitabın bu kısmında, zeki makine inşa etmeye yönelik ilk gerçek çabalardan bazılarını betimliyorum. Bunlardan kimileri konferanslarda ve sempozyumlarda anlatılmış ya da bildirilmiştir; böylece bu toplantılar, YZ'nin doğumunda önemli kilometre taşları olmuştur. Aynı zamanda, bu ilk YZ programlarının temel işleyişini açıklamak için elimden geleni yapacağım. Bu dönemin görece büyük başarıları, izleyen yapay zekâ araştırmaları için sağlam bir taban oluşturmaya yardımcı olmuştur.

Bazı araştırmacılar, kullandıkları yöntemlere merak sarmışlardı (hatta onlara tutulmuşlardı diyebiliriz); seçtikleri bu teknikleri, bunları gerektirdiği düşünülen görevlere uygulamak yerine, kendilerini o tekniklerin gücünü ve genel geçerliğini geliştirmeye adamışlardı. Dahası, kimi araştırmacılar, makinelere problem çözdürtmeye uğraşırken, insan beyninin nasıl problem çözdüğünü açıklamaya da ilgi

gösterdiği için, geliştirilen yöntemler çoğunlukla, insanın zihinsel süreçleri hakkında kuramlara katkı olarak önerilmiştir. Dolayısıyla, bilişsel psikoloji araştırmaları ile yapay zekâ araştırmaları çoğu kere iç içe geçmiştir.

3
TOPLANTILAR

Eylül 1948'de California eyaletinin Pasadena kentindeki California Teknoloji Enstitüsü'nde (Caltech) sinir sisteminin davranışları nasıl düzenlediği ve beynin bilgisayarla nasıl karşılaştırılabileceği hakkında disiplinler arası bir konferans düzenledi. Bu toplatıya, Davranışta Beyinsel Mekanizmalar konulu Hixon Sempozyumu adı verilmişti. Bu konferansa, birkaç aydın katılıp makale sunmuştu; bunlar arasında Warren McCulloch, John von Neumann ve önde gelen psikologlardan Karl Lashley (1890-1958) vardı. Lashley'in yaptığı konuşma kimilerine göre sempozyumun en önemli konuşmasıydı. Lashley, beyin işlevi hakkında durağan bir görüş benimsediği için davranışçılığa yüklenmiş, insanın tasarlama ve dil yeteneklerini açıklama yolunda psikologların dinamik ve hiyerarşik yapıları gözden geçirmeye başlaması gerektiğini ileri sürmüştü. Lashley'in konuşması, ileride bilişsel bilim adını alacak dalın temellerini döşemişti.[1]

Yapay zekânın, tam teşekküllü bir araştırma sahası olarak ortaya çıkışı, üç önemli toplantıya tesadüf etmişti (ve YZ bunlar sayesinde doğmuştur): biri 1955'te, biri 1956'da, öbürü de 1958'de. 1955'te "Öğrenen Makineler Oturumu", Los Angeles'ta 1955 Batı Ortak Bilgisayar Konferansıyla bir arada düzenlenmişti. 1956'da "Yapay Zekâ Yaz Araştırma Projesi", Dartmouth Koleji'nde toplanmıştı. 1958'de ise, "Düşünce Süreçlerinin Mekanikleşmesi" konulu sempozyuma, Birleşik Krallık'ta Ulusal Fizik Laboratuvarı hamilik etmişti.

3.1 ÖĞRENEN MAKİNELER OTURUMU

1955'te Los Angeles'ta dört önemli makale sunuldu. Willis Ware, oturum başkanı olarak kaleme aldığı tanıtım yazısında şöyle yazıyordu:

1 Bu sempozyumun tutanakları için bkz. L. A. Jeffries (ed.), *Cerebral Mechanisms in Behavior: The Hixon Symposium*, New York: Wiley, 1951. Lashley'in değindiği noktaları içeren mükemmel bir inceleme için bkz. Howard Gardner, *The Mind's New Science: A History of the Cognitive Revolution*, İkinci Bölüm, New York: Basic Books, 1985.

> Bu makaleler, gelecekte, öğrenen makinelerin çok amaçlı sayısal bilişim cihazı örüntüsüyle inşa edilmesi gerektiğini değil, sayısal bilişim sistemi, bu modellerin davranışlarını irdeleyecek kullanışlı ve epey esnek bir araç sunduğunu söylüyor... Bu makaleler, özellikle bu model tekniği için sayısal bilişim makinelerinden faydalanmaya niyetlenen geleceğin makine inşacılarına gelişim doğrultuları öneriyor. İşlem hızı, katbekat artmak zorunda; pek çok paralel kipte eşzamanlı gerçekleşecek işlemlerin kuvvetle altı çiziliyor; rasgele erişimli belleğin boyutu, birkaç kat yukarı sıçramalı; yeni girdi-çıktı ekipman tipleri gerekli. Bu tür ilerlemelerle ve bu makalelerde anlatılan tekniklerle birlikte, beynin ve sinir sisteminin hatırı sayılır kısmını taklit edecek sistemlerin görece yakın gelecekte inşa edilebileceğine dair kayda değer bir umut var.

Neyse ki, Ware'nin "gelişim doğrultuları" listesindeki maddeler konusunda fevkalade gelişmeler kaydettik. İşlem hızı katbekat arttı, pek çok YZ sisteminde paralel işlemler kullanılıyor, rasgele erişilen belleğin boyutu birkaç kat yukarı sıçradı, üstelik yeni girdi-çıktı ekipman tipleri artık var. Belki, daha başka gelişmeler de gerekli olacak.

Oturumun ilk makalesini, MIT Lincoln Laboratuvarından Wesley Clark ile Belmont Farley yazmışlardı ve bu makale, sinir hücresi benzeri öğelerden oluşan ağlarla yapılmış birtakım örüntü tanıma deneylerini betimliyordu.[2] Sinir hücresi öbeklerinin, ara bağlantılarının kuvvetini ayarlayarak öğrenebileceğini ve uyum gösterebileceğini söyleyen Hebb'in önermesiyle şevke gelen deneyciler, kendi ağları içinde bağlantıların kuvvetini ayarlamak amacıyla, çoğunlukla bilgisayarda benzetimini yaptıkları çeşitli düzenekler denemişlerdi. Kimileri sadece, bu ağların ne yapabildiğini görmek istemiş, Clark ve Farley gibi başkaları, örüntü tanıma misali özgül uygulamalarla ilgilenmişti. Ancak, aşırı basitleştirmeden şikâyet eden sinir fizyologlarının korktuğu başlarına gelmişti, zira bu ağlara, *sinir ağları* denir olmuştu. Clark ve Farley şu sonuca vardılar: "Kaba ama faydalı genelleme özelliklerini, betimlenen ağ tipinin rasgele düzenlenmiş olanları bile taşır."[3]

2 W. A. Clark ve B. G. Farley, "Generalization of Pattern Recognition in a Self-Organizing System", *Proceedings of the 1955 Western Joint Computer Conference*, Institute of Radio Engineers, New York, s. 86-91. Clark ile Farley'in deneyleri, daha önce şu makalede bildirmiş oldukları çalışmaları sürdürmüştür: B. G. Farley ve W. A. Clark, "Simulation of Self-Organizing Systems by Digital Computer, *IRE Transactions on Information Theory*, c. 4, 1954, s. 76-84. (1962'de Clark, ilk kişisel bilgisayar LINC'i inşa etti.)

3 Alan Wilkes ve Nicholas Wade, ilk sinir ağını icat etmenin itibarını İskoçyalı psikolog Alexander Bain'e (1818-1903) vermişlerdir; Bain bunu 1873 tarihli şu kitabında betimlemiştir: *Mind and Body: The Theories of Their Relation*". (Bkz. Alan L. Wilkes



Resim 3.1. Oliver Selfridge (Fotoğraf, Oliver Selfridge'in izniyle.)

Bir tanesini Gerald P. Dinneen'in (1924-2012), diğerini Oliver Selfridge'in (1926-2008; Resim 3.1) yazdığı sıradaki iki makale (her iki yazar da MIT Lincoln Laboratuvarı mensubuydu) örüntü tanımayla ilgili farklı bir yaklaşım sunuyordu. Dinneen'in makalesi,[4] görüntü işlemede kullanılabilecek bilişim tekniklerini anlatıyordu. Görüntüler bilgisayara, görüntüdeki çeşitli gri tonlara denk gelen yoğunluk değerlerinin dikdörtgen bir dizisi olarak sunuluyordu. Dinneen, rasgele gürültüyü giderecek, hatları kalınlaştıracak, kenar çizgilerini bulacak süzgeç yöntemleri kullanmanın öncülüğünü yapmıştır. Makalesine şu satırlarla başlar:

> Son aylarda, laboratuvarda mesai bitiminde ve öğle yemeği aralarında düzenlediğimiz bir dizi toplantıda, bu alandaki sorunlara kafa yorduk. Aşağı yukarı oy birliğiyle hepimizin görüşü, işi uygulamaya dökme, gerçek hayattan bir sorun seçip bunun peşinden koşma *ihtiyacının* var olduğuydu.

ve Nicholas J. Wade, "Bain on Neural Networks", *Brain and Cognition*, c. 33, 1997, s. 295-305.)

4 Gerald P. Dinneen, "Programming Pattern Recognition", *Proceedings of the 1955 Western Joint Computer Conference,* Institute of Radio Engineers, New York, 1955, s. 94-100.

Selfridge'in makalesi,[5] adeta Dinneen'in makalesinin tamamlayıcısıydı. "Temizlenmiş" görüntüler üzerine çalışan (örneğin, Dinneen'in programının ürettiği gibi) Selfridge, bu görüntülerdeki kimi "özellikleri" belirleyip görüntüleri bu özelliklere göre sınıflandıran teknikler betimlemişti. Örneğin, ya kare ya da üçgen olduğu bilinen bir görüntünün köşeleri belirlenir, sonra da kaç köşe olduğu sayılır ki resim kare mi üçgen mi anlaşılsın. Selfridge, "en sonunda başka özellik türlerini tanıyabilmeyi umuyoruz, örneğin eğri, tekil noktaların bitişmesi (yani, göreli duruşları ve mesafeleri), vesaire gibi", demişti.

Selfridge ile Dinneen'in öncülüğünü yaptıkları yöntemler, ileride makinelere "görme" yetisi kazandırmaya çalışacak çoğu çalışmanın zeminini teşkil edecekti. Onların çalışması, kullandıkları bilgisayar düşünüldüğünde daha da olağanüstü görünür; çünkü Lincoln Laboratuvarı "Bellek Testi Bilgisayarı" günümüzde olsa son derece ilkel kabul edilirdi. [Bellek Testi Bilgisayarı (BTB), Jay Forrester'ın geliştirmiş olduğu ferrit çekirdekli rasgele erişimli bellek modülleri kullanan ilk bilgisayardı. 1953'te Digital Equipment Şirketinde (DEC) Ken Olsen tarafından tasarlanıp inşa edilmişti. BTB, sinir ağlarının işleyişini taklit eden ilk bilgisayardı; Clark ile Farley'in tasarladıkları sinir ağlarının benzetimini yapmıştı.]

Oturumda bir sonraki makale,[6] satranç oynayan bilgisayar programlamakla ilgiliydi. O zamanlar Santa Monica'daki Rand Şirketinde araştırmacı olan Allen Newell yazmıştı bunu. Carnegie Mellon Üniversitesi'nden çalışma arkadaşı Herb Simon'ın kaleme aldığı Newell biyografisi sayesinde, Newell'ı bu konuya iten şeyi ve bu sorunla nasıl ilgilenmeye başladığını biliyoruz:[7]

> Eylül 1954'te Allen, RAND'ta bir seminere katılmıştı; orada Lincoln Laboratuvarı'ndan Oliver Selfridge, harfleri ve başka örüntüleri tanımayı öğrenmiş bir bilgisayar programının işleyişini tasvir etmişti. Allen, Selfridge'in görece ilkel fakat işleyen sistemini tanımlamasını dinlerken, kendi "dönüşüm deneyimi" olarak atıfta bulunacağı şeyi yaşadı. "Şim-

5 Oliver Selfridge, "Pattern Recognition and Modern Computers", *Proceedings of the 1955 Western Joint Computer Conference,* Institute of Radio Engineers, New York, 1955, s. 91-93.

6 Allen Newell, "The Chess Machine: An Example of Dealing with a Complex Task by Adaptation", *Proceedings of the 1955 Western Joint Computer Conference,* Institute of Radio Engineers, New York, 1955, s. 101-108. (Aynı zamanda RAND Teknik Raporu P-620 olarak yayımlanmıştır.)

7 National Academy of Sciences, *Biographical Memoirs*, c. 71, 1997. Şu internet sitesinde: http://www.nap.edu/catalog.php?record_id=5737.

> diye kadar yapılmış her şeyden uzak ara daha karmaşık olan, kendini çevreye uyarlayabilen zeki sistemlerin inşa edilebileceğini" aniden Allen anlamıştı. Allen'ın bilgisayarlar (simgesel becerileri dahil), sezgisel yöntemler, örgütlenmelerde bilgi işleme, sibernetik gibi konulardaki var olan bilgisine ve satranç programlarıyla ilgili önermelere, şimdi de karmaşık süreçlerle çalışan bilgisayar benzetiminin işe yaradığının somut şekilde gösterilmesi eklenmişti. Tam o anda kendini, insanın öğrenme ve düşünme yetilerini, benzetimlerini yaparak anlamaya adadı.

Simon, Newell'in satrançla ilgili makalesini de özetler:

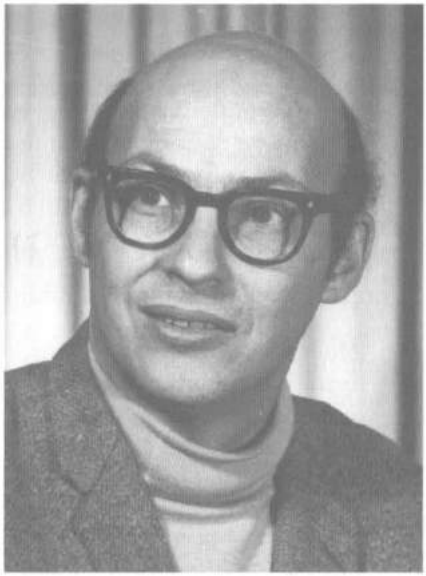
> [Makale] insan tarzında satranç oynayacak bir bilgisayar programının hayalî tasarımını belirtiyordu; bu tasarım, hedeflerle ilgili birtakım kavramlar, aramayı sonlandıracak istek düzeyleri, "yeterince iyi" hamlelerle tatmin olma özelliği, çokboyutlu değerlendirme fonksiyonları, hedefleri hayata geçirmek adına tali hedefler yaratma yetisi ve en iyi arama özelliğini de içeriyordu. Satranç tahtasına dair bilgi, yüklemsel biçimsel dizgeye benzeyen bir dilde simgelerle ifade edilecekti. Bu tasarım asla hayata geçirilmedi ama buradaki fikirlerden 1958'de NSS [Newell, Shaw, Simon] satranç programında faydalanılacaktı.[8]

Newell, satrancın ötesine geçmeye niyetlendiğinin ipuçlarını vermişti. Makalesinde şöyle yazar: "O halde bu çabanın hedefi, günümüz bilgisayarlarını iyi satranç oynamaları için programlamaktır. Bu uğraş, anlaşılması son derece güç problemlerle boğuşmak için gerekli olan bilgisayar, mekanizma ve program türleri hakkında daha fazla şey anlamayı sağlayacak araçtır". Newell'in önermiş olduğu teknikler, ileride Simon'la birlikte Fiziksel Simge Sistemi Hipotezi diyecekleri şey için kanıt yaratmaya yönelik ilk girişimleri sayılabilir.

Oturumun sunucusu Walter Pitts, oturumu şunu söyleyerek kapatmıştır: "Ancak, Sayın Farley, Clark, Selfridge ve Dinneen sinir sistemini taklit ederlerken, Sayın Newell, geleneksel olarak zihin denilen, nihai sebepler hiyerarşisini taklit etmeyi tercih ediyor. Sonunda hepsi hiç şüphesiz aynı kapıya çıkacak..." "Aynı kapıya çıkacak" olan bu iki yaklaşım, yani sinirleri modelleme ile simge işleme, beyinde olup biten şeyler hakkında farklı betimleme düzeyleri olarak kabul edilmeli. Farklı zihinsel görüngüleri betimlemek için farklı betimleme düzeyleri

8 Allen Newell, J. C. Shaw ve Herbert A. Simon, "Chess-Playing Program and the Problem of Complexity", *IBM Journal of Research and Development*, c. 2, 1958, s. 320-355. Makale şu internet adresinde bulunabilir: http://domino.watson.ibm.com/tchjr/journalindex.nsf/0/237cfeded3be103585256bfa00683d4d?OpenDocument.





Resim 3.2 John McCarthy (solda) ve Marvin Minsky (sağda). (McCarthy fotoğrafı, John McCarthy'nin izniyle. Minsky'nin fotoğrafı, MIT Müzesinin izniyle.)

uygundur. Kitabın ilerleyen bölümlerinde betimleme düzeyleri hakkında daha fazla şey söylüyorum.

3.2 DARTMOUTH YAZ PROJESİ

1954'te John McCarthy (1927-2011; Resim 3.2) New Hampshire'ın Hanover kentindeki Dartmouth Koleji'ne Matematik Bölümü'nde Yardımcı Doçent olarak katılmıştı. McCarthy, ileride yapay zekâ diye nitelenecek alana gitgide artan bir ilgi besliyordu. "Bu ilgiyi, matematik yüksek lisansına başladığım Caltech'te Eylül 1948'de düzenlenmiş olan, Davranışlarda Beyin Mekanizmaları konulu Hixon Sempozyumuna katılmam tetiklemişti", der.[9] Dartmouth'dayken, Nathaniel Rochester (1919-2001) onu, 1955 yazını Rochester'ın New York eyaletindeki Poughskeepsie IBM Enformasyon Araştırmaları Merkezi'nde geçirmeye davet etti. Rochester, IBM 701 bilgisayarının tasarımcısıydı, ayrıca sinir ağlarıyla ilgili araştırmalara katılmıştı.[10]

9 2006 Darmouth anma toplantısında John McCarthy'nin gayriresmi yorumlarından.

10 Nathan Rochester ve diğ., "Tests on a Cell Assembly Theory of the Action of the Brain Using a Large Digital Computer", *IRE Transaction of Information Theory*, cilt

IBM'de geçirdikleri o yaz mevsiminde McCarthy ile Rochester, Claude Shannon'ı ve o zamanlar Harvard'ta matematik ve nörolojide genç araştırmacı olan Marvin Minsky'yi (1927-2016; Resim 3.2), bir sonraki yaz Darmouth'da düzenlenecek bir çalıştay teklifi hazırlamada kendilerine katılmaya ikna ettiler. Daha önce bahsetmiş olduğum Shannon, Bell Telephone Laboratuvarları'nda çalışan bir matematikçiydi ve geçiş kuramı ile istatistiksel enformasyon kuramı üzerine çalışmalarıyla zaten ünlenmişti. McCarthy, bu teklifin yazılışında ve "Yapay Zekâ Yaz Araştırma Projesi" diye anılacak etkinliği örgütlemede başı çekmiştir. Teklif, Ağustos 1955'te Rockefeller Vakfı'na sunuldu.

Teklifin kimi kısımlarına göz atalım:[11]

> New Hampshire, Hanover'deki Dartmouth Koleji'nde 1956 yazı boyunca gerçekleştirilecek, 2 aylık ve 10 kişilik bir yapay zekâ araştırması öneriyoruz. Bu araştırma, öğrenmenin ya da zekânın herhangi başka bir yönünün ilkesel olarak bir makine tarafından taklit edilecek şekilde tam olarak betimlenebileceği varsayımı temelinde ilerleyecek. Makinelere doğal dil kullandırtmanın, soyutlamalar ve kavramlar oluşturtmanın, şimdilik insanlara özgü sorun türlerini çözdürtmenin, kendilerini geliştirtmenin yollarını bulmaya çalışacağız. Eğer özenle seçilecek bir bilim ekibi bütün yaz boyunca birlikte çalışırsa bu sorunlardan en az biri ya da daha çoğunda önemli ilerlemeler kaydedilebileceğini düşünüyoruz.
>
> ...
>
> Mevcut amaç uyarınca, yapay zekâ sorunu, insan öyle davrandığında zeki olarak nitelendirilecek davranışları makinelere yaptırtmak olarak tanımlanacak.

Rockefeller Vakfı, 1956 yazında altı hafta sürmüş olan bu etkinliğin maliyetini üstlenmiştir. Fakat, bir yaz "araştırması"ndan ziyade inişli çıkışlı altı haftalık bir çalıştay olmuştur bu. McCarthy, Minsky, Rochester ve Shannon'a ek olarak o yaz çalıştaya katılanlar arasında, IBM şirketi mühendislerden, zaten dama oynayan bir program yazmış Arthur Samuel (1901-1990), Oliver Selfridge, tümevarımı otomatikleştirmekle ilgilenen MIT'li Ray Solomonoff, Allen Newell ve Herbert Simon da vardı. Newell ve Simon (başka bir Rand bilimcisi Cliff Shaw'la birlikte), simgesel mantıkta teoremler ispatlayan bir program yarat-

IT-2, 1956, s. 80-93.

11 Bkz. http://www-formal.stanford.edu/jmc/history/darmouth/darmouth.html. Bu teklifin kimi kısımlarının tekrar basımı için bkz. John McCarty, Marvin L. Minsky, Nathaniel Rochester ve Claude E. Shannon, "A Proposal for the Darmouth Summer Research Project on Artificial Intelligence", *AI Magazine*, c. 27, sayı: 4, Kış 2006, s. 12.

mışlardı. Çalıştaya katılan başka bir IBM bilimcisi de, satranç oynayan bir program üzerinde çalışan Alex Bernstein'dı.

McCarthy, "yapay zekâ" terimini kullanmasıyla ilgili birkaç sebep ileri sürmüştü. İlk sebep, Dartmouth çalıştayı için önerilen konuyu, McCarthy ile Shannon'un birlikte hazırlamış oldukları ve talep edilmiş makaleleri derleyen *Automata Studies* [Otomat Araştırmaları] başlıklı önceki kitabın konusundan ayırt etmekti; söz konusu kitap (ki McCarthy'yi epey hayal kırıklığına uğratmıştır), gizemli ve görece dar kapsamlı bir matematik konusu olan "otomatlar kuramı"yla ilgileniyordu büyük oranda. McCarthy'ye göre ikinci sebep, "'sibernetik'le ilgili görülmekten uzak durmaktı." Şöyle diyordu: "Sibernetiğin, analog geribildirime yoğunlaşmış olması, gözüme hatalı bir yaklaşım gibi görünmüştü. Ayrıca, Norbert Wiener'i ustam olarak kabul etmekten ya da onunla tartışmaya girmekten kaçınmak istemiştim."[12]

Yapay zekâ isminin etrafında dönen bir anlaşmazlık vardı (hâlâ da vardır). Pamela McCorduck'un yazmış olduğu, yapay zekânın ilk günlerine dair mükemmel tarihçeye göre, Arthur Samuel şu yorumda bulunmuştu: "Yapay sözcüğü, bu işte bir düzmece olduğu hissini veriyor; ya da sanki bu iş tamamen yapaymış, hiçbir gerçekliği yokmuş düşüncesini uyandırıyor."[13] McCorduck şunu da söyler: "Bu ifadeyi ne Newell ne Simon beğenmişti; dolayısıyla sonraki yıllarda kendi çalışmalarını, karmaşık bilgi işleme olarak adlandırdılar." Fakat bu yeni alanda çalışmaya katılan çoğu insan (bendeniz dahil) "yapay zekâ" adını kullanmıştır ve artık günümüzde alanın ismi budur. (Newell ileride bu isimle uzlaşacaktı. Bu alanın içeriği hakkında yorum yaparken şunu demiştir: "O halde *yapay zekâ* adını bağrımıza basalım. Güzel bir isim bu. Tüm bilimsel alanların ismi gibi, büyüyecek ve bu alanın tam da kastettiği anlama kavuşacak.")[14]

Bu çalıştaya katılan insanların yaklaşımları ve güdüleri farklı farklıydı. Rochester bu konferansa, sinir hücresi benzeri öğelerden oluşan ağlarla ilgili geçmişiyle gelmişti. Newell ve Simon, simge işleme yaklaşımının peşinden koşuyorlardı (doğrusu, vücut bulmasına yardımcı olmuşlardı). Shannon'ın kafa yormak istediği konu başlıkları arasında (teklife göre), "bilgi kuramı kavramlarını bilişim makineleri-

12 Şu internet sitesinden: http://www-formal.stanford.edu/jmc/reviews/bloomfield/bloomfield.html.

13 Pamela McCorduck, *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*, San Francisco: W. H. Freeman and Co., 1979, s. 97.

14 Bkz. Allen Newell, "The First AAAI President's Message", *AI Magazine*, c. 26, no: 4, Kış 2005, s. 24-29.

ne ve beyin modellerine uygulamak" vardı. (Ancak, çalıştaydan sonra Shannon'ın yapay zekâya ilgisi kaybolmuştur.)

McCarthy, "varsayım ve kendine atıfta bulunma gerektiren problemlerde bilgisayarın kullanmak üzere programlanabileceği yapay bir dil" oluşturmakla ilgilendiğini yazmıştı: "Bu dil, belirli bir konudaki kısa İngilizce ifadelerin kısa karşılıklarını içermesi, dolayısıyla kısa ya da varsayımsal savların belirtilebilmesi anlamında İngilizceye benzemeli. Bu özellikleri taşıyan bir dil tanımlamayı umuyorum..." Her ne kadar McCarthy daha sonra, bu konuyla ilgili fikirlerinin henüz konferansta sunulamayacak kadar "eksik tanımlı" olduğunu söylemiş olsa da, bilgiyi temsil etmekte ve uslamlamada bir mantık dilini ve bu dilin çıkarım mekanizmalarını kullanmaya yönelik özgül önerilerde bulunması uzun sürmemiştir.

Her ne kadar doktora tezi[15] ve kimi sonraki çalışmalar sinir ağlarına yoğunlaşmış olsa da, Dartmouth çalıştayı döneminde Minsky yön değiştirmeye başlamıştı. Artık, "bulunduğu ortamın soyut bir modelini kendi içinde kurmaya çalışan" bir makine tasarlamak istediğini yazmıştı: "Bu bilgisayara bir sorun verildiğinde, önce çevrenin içsel soyut modeli kapsamında birtakım çözümler keşfeder, sonra da dışsal deneyler yapmaya kalkışır". Bu çalıştayda Minsky, ileride "Steps Toward Artificial Intelligence" [Yapay Zekâya Doğru] başlığıyla çığır açan bir makale olarak yayımlanacak taslak üzerinde çalışmayı sürdürmüştü.[16]

1956 tarihli bu buluşmanın en önemli teknik katkılarından biri, Newell ile Simon'un, simgesel mantıkta teorem ispatlamaya yarayan "Mantık Kuramcısı [Logic Theorist] (LT)" adlı programlarını sunmalarıydı. LT, "simge yapılarını" işlemenin ve Newell ile Simon'un "sezgisel" (*heuristics*) dedikleri yöntemlerin kullanımının, problemleri zekâ üzerinden çözme yolunda temel gelişmeler olduklarının somut ispatıdır. Bu fikirlerden bazılarını gelecek bölümde daha ayrıntılı betimliyorum.

Newell ve Simon, LT'yle ilgili fikirler üzerinde aylardır çalışıyorlardı ve 1955'in sonuna doğru, bu fikirlere, çalışır bir programda vücut kazandırabileceklerine kanaat getirmişlerdi. 1956 başlarında Carnegie'de Herb Simon'dan ders almış olan Edward Feigenbaum'un (1936-) aktar-

15 M. L. Minsky, *Theory of Neural-Analog Reinforcement Systems and Its Application to the Brain-Model Problem*, doktora tezi, Princeton Üniversitesi, 1954.

16 Marvin L. Minsky, "Steps Toward Artificial Intelligence", *Proceedings of the IRE*, c. 49, sayı: 1, Ocak 1961, s. 8-30. Ayrıca; Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw Hill, 1963. (Bkz. http://web.media.mit.edu/~minsky/papers/steps.html.)

dığına göre: "Noel tatilinin hemen ardından (Ocak 1956), Herb Simon sınıfa gelip, 'Noel'de Allen Newell'la birlikte düşünen bir makine icat ettik' dedi."[17] Kısa süre sonra LT olarak programlanacak alet, Simon'un bahsettiği "düşünen makine"ydi. Şüphesiz Simon makineyi böyle isimlendirmişti, çünkü bu makinenin, problem çözerken insanların kullandığı yöntemlerden bazılarını kullandığını düşünüyordu. İleride Simon şöyle yazacaktı: "15 Aralık Perşembe... ilk kanıtı elle benzetimlemeyi başardım... 15 Aralık 1955 tarihini daima, sezgisel yöntemlerle problem çözen bilgisayarın doğum günü olarak anmışımdır."[18] Simon'un otobiyografisi *Models of My Life*'a [Hayatımın Modelleri] göre[19] LT, programın durum değişkenlerini içeren sayaçlar olarak üstüne yazdığı ya da kaldırıp gösterdiği not kartları, hesap birimi olarak da kendi çocuklarını kullandığı bir el benzetimi olarak doğmuştu.[20]

Dartmouth'da konuşulan başka bir konu başlığıysa, geometri teoremlerini ispatlamaydı. (Belki bazı okurlar, lisede geometri ispatlarıyla boğuşmalarını anımsayacaktır.) Minsky, geometri teoremlerini ispatlayan bir programa zaten kafa yoruyordu. McCorduck, Minsky'nin şunu söylediğini aktarır:[21]

> Kendi gelişimimde belki de önemli olay ve Newell-Shaw-Simon çalışmasını belki de rasgele kabul edişimin açıklaması, [bir] geometri makinesi için sezgisel arama yöntemi karalayıp, sonra da aşağı yukarı bir saat içinde kâğıt üstünde bunun elle benzetimini yapışımdır. Elimin altında, ikizkenar üçgen teoreminin yeni ispatı hayat bulmuştu, katılımcılar için yeni ve zarif bir ispattı bu; sonradan bu kanıtın bilindiğini öğrendik...

Temmuz 2006'da Dartmouth'da, ilk konferansın ellinci yılını anan başka bir konferans tertiplendi (bkz. Resim 3.3). Yapay zekânın kurucularından bazıları ve önde gelen birtakım YZ araştırmacıları bu konferansa katılmış, 1956'dan bu yana kaydedilen ilerlemeleri gözden geçirmişlerdir. McCarthy eski günleri hatırlarken şöyle konuşmuştu: "1956 Dartmouth çalıştayının beklentilerimi karşılamayışının başlıca nedeni, YZ'nin düşündüğümüzden zor oluşudur". Yine de 1956 çalıştayı, yapay zekâda ciddi çalışmaların resmî başlangıcı ve Minsky, McCarthy, Ne-

17 Pamela McCorduck, *a.g.e.*, s. 116.

18 Herbert A. Simon, *Models of My Life*, Cambridge, MA: MIT Press, 1996. Alıntı http://www.post-gazette.com/pg/06002/631149.stm adlı siteden yapılmıştır.

19 *A.g.e.*

20 http://www.post-gazette.com/downloads/20060102simon_notes.pdf, Simon'un LTF kanıtı benzetimiyle ilgili karalamalarını içeriyor.

21 Pamela Mcorduck, *a.g.e.*, s. 106.



Resim 3.3 Temmuz 2006'da düzenlenen Dartmouth ellinci yıl toplantısında YZ'nin kurucularından bazıları. Soldan sağa: Trenchard More, John McCarthy, Marvin Minsky, Oliver Selfridge, Ray Solomonoff. (Fotoğraf, fotoğrafçı Joe Mehling'in ve Dartmouth Koleji Yapay Zekâ Konferansı Gelecek Elli Yıl'ın izniyle.)

well ve Simon yapay zekânın "babaları" kabul edilir. Bu etkinliğe adanmış ve Dartmouth'da Baker Kütüphanesi'ne yerleştirilmiş bir plaket, yapay zekânın bir bilim disiplini olarak doğuşunu anıyor.

3.3 DÜŞÜNCE SÜREÇLERİNİN MEKANİKLEŞMESİ

Kasım 1958'de, İngiltere Middlesex Teddington'daki Ulusal Fizik Laboratuvarı'nda "Düşünce Süreçlerinin Mekanikleşmesi" başlıklı bir sempozyum düzenlendi. Konferans raporunun önsözüne göre, sempozyumun amacı, "yapay düşünce, karakter ve örüntü tanıma, öğrenme, otomatik dil çeviri, biyoloji, otomatik programlama, endüstriyel planlama, yazı işlerinin mekanikleşmesini araştıran bilimcileri bir araya toplamaktı."

Bu sempozyumda makale sunmuş kişiler arasında, bu hikâyede adını çoktan anmış olduğum pek çok insan vardı. Bunlara Minsky (o zaman Lincoln Laboratuvarı kadrosundaydı ve MIT Matematik Bölümü'nde yardımcı doçent olma yolundaydı), McCarthy (o zaman-

lar MIT İletişim Bilimleri'nde yardımcı doçentti), Ashby, Selfridge ve McCulloch dahildi. (Bilgisayar programlama dili FORTRAN'ın yaratıcılarından John Backus ile "otomatik programlama"nın öncülerinden Grace Murray Hopper da bu sempozyumda makale sunmuşlardır.)

Bu konferansın bildiri kitabı,[22] yapay zekâ tarihinde oldukça etkili olmuş bazı makaleleri içerir. Bunlar arasından, Minsky, McCarthy ve Selfridge'in makalelerinden bahsedeceğim.

Minsky'nin "Some Methods of Artificial Intelligence and Heuristic Programming" [Yapay Zekâ ve Sezgisel Programlama Yöntemleri] başlıklı makalesi, Dartmouth çalıştayından hemen önce çalışmaya başladığı konunun en son haliydi. Bu makale, sezgisel programlamada kullanılmış (ve kullanılabilecek) çeşitli yöntemleri betimliyordu. Ayrıca örüntü tanıma, öğrenme ve tasarlama yöntemlerini de kapsıyordu. Metnin son hali, "Steps Toward Artificial Intelligence" başlığıyla yayımlanacak, ve alana yeni giren kişilerin (bendeniz dahil) muhakkak okuması gereken makalelerden biri olacaktı.

McCarthy'nin, YZ için yapay dil geliştirme umudundan zaten bahsetmiştim. Konferansta sunmuş olduğu "Programs with Common Sense" [Sağduyulu Programlar] başlıklı makaleyi şu şekilde özetlemiştir:

> Bu makale, uygun bir örgün dilde (muhtemelen yüklemcil biçimsel dizgenin [*predicate calculus*] parçası olarak) etkili yaygın yargıları işleyen programları anlatacak. Temel program, bir önerme listesinden doğrudan sonuçlar çıkaracaktır. Bu sonuçlar ya bildirimsel cümleler ya da emir cümleleri olacak. Emir cümlesi geldiğinde, program gerekli eylemi yapacaktır.

McCarthy, bu makalesinde, "tavsiye alıcı" dediği YZ programının gereksinim duyduğu olguların, "birinci dereceden mantık" adlı ve matematik (bilgisayar dostu) dilindeki ifadelerle temsil edilebileceğini söylemişti. Örneğin, "masamda oturuyorum" ve "masam evde" olguları, `yer (ben, masa)` ve `yer (masa, ev)` ifadeleriyle temsil edilebilir. Nasıl konum değiştirileceğiyle (örneğin yürüyerek ya da arabayla) ilgili bilgiyle birlikte bu kadarı, önerilen (ama henüz programlanmamış) tavsiye alıcı tarafından, örneğin havaalanında olmak gibi bir hedefe nasıl ulaşılacağını belirlemek üzere kullanılabilir. Dolayısıyla tavsiye alıcının uslamlama süreci, arabaya yürümek ve arabayı havaalanına sürmeyi içeren zorunlu mantıksal ifadeler üretecektir.

22 D. V. Blake ve A. M. Uttley (ed.), *Proceedings of the Symposium on Mechanization of Thought Processes*, c. 1 ve 2, Londra: Her Majesty's Stationary Office, 1959.

Olguları mantık dilinde göstermenin bazı faydaları bulunur. Daha sonra McCarthy'nin belirteceği üzere:[23]

> Bilgiyi bildirimsel cümleler halinde ifade etmek, bu bilgiyi bilgisayar programının kesitleri ya da tablolar olarak ifade etmekten çok daha birimseldir. Bu durumda cümleler, özgül programların faydalı olabileceğinden daha geniş bağlamlarda geçerlilik kazanır. Olguları sağlayan birim, alıcının nasıl işlev göreceğini ya da alıcının bunu kullanıp kullanmayacağını anlamak zorunda değildir. Aynı olgu, pek çok farklı amaç için kullanılabilir çünkü olgu derlemelerinin farklı mantıksal sonuçları olabilir.

Daha sonra McCarthy, ek bir raporla bu fikirlerin kapsamını genişletmişti.[24] İleride bahsedeceğim gibi, McCarthy'nin tavsiye-alıcı hakkındaki kimi önerileri, sonradan Stanfordlu lisansüstü öğrencisi C. Cordell Green tarafından hayata geçirilecekti.

Oliver Selfridge'in 1955 yılındaki örüntü tanıma çalışmasından zaten bahsetmiştim. 1958'deki Teddington Sempozyumu'nda Selfridge, yeni bir örüntü tanıma modeli hakkında makale sunmuştu (muhtemelen başka bilişsel görevlerin de altından kalkabiliyordu bu model).[25] Buna, tüm cinlerin, zebanilerin toplandığı yer anlamına gelen "Pandemonium" adını vermişti. Selfridge'in modeli özellikle ilginçtir, çünkü "cin" diye adlandırdığı bileşenleri, alt düzey sinir hücresi türü işlevler ya da üst düzey (simge işleme çeşidinden) bilişsel işlevler gerçekleştirmeyi temsil edebilir. Dolayısıyla Pandemonium, hiyerarşik örgütlenmiş bir dizi simge işlemcisi olarak sinir ağı biçimine kavuşabilir; bu işlemcilerin hepsi paralel işliyordur ya da bu biçimlerin belli bir kombinasyonuyla çalışıyordur. Eğer bir biçimler kombinasyonu söz konusuysa, bu model, o iki farklı YZ yaklaşımını birleştirmek için kışkırtıcı bir öneridir.

Selfridge, makalesinin giriş kısmında, paralel çalışan hesaplamaların önemini vurgular:

23 John McCarthy, "Artificial Intelligence, Logic and Formalizing Common Sense", *Philosophical Logic and Artificial Intelligence*, Richmond Thomason (ed.), Dordrecht: Kluwer Academic, 1989.

24 J. McCarthy, "Situations, Actions and Causal Laws, Stanford Artificial Intelligence Project", Memo 2, 1963. Bu iki parça birlikte şurada basılmıştır: M. Minsky (ed.), *Semantic Information Processing*, s. 410-417, Cambridge, MA: MIT Press, 1968. İlişkili konular şurada irdelenmiştir: J. McCarthy ve Patrick Hayes, "Some Philosophical Ideas From the Standpoint of Artificial Intelligence", MI-4, 1969.

25 Oliver G. Selfridge, "Pandemonium: A Paradigm for Learning", *Proceedings of the Symposium on Mechanisation of Thought Processes* içinde, D. V. Blake ve A. M. Uttley (ed.), Londra: Her Majesty's Stationary Office, 1959, s. 511-529.

> Modelimizin ardındaki temel motif, paralel işlem düşüncesidir. Bu, iki temele dayanıyor: Birincisi, verileri paralel biçimde ele almak çoğunlukla daha kolaydır, üstelik verileri bu şekilde ele almak daha "doğal" olan biçimdir; ikincisi, tüm parçaları dolaysız ve karmaşık biçimde etkileşime giren bir makineden ziyade yarı bağımsız bölmelerin bir birleşimi üzerinde değişiklikler yapmak daha kolaydır.

Selfridge, Pandemonium'un nasıl öğrenebileceği konusunda bazı önerilerde bulunmuştu. Bunlardan bazılarını betimlemeye değer, çünkü bunlar öğrenen makinelerle ilgili çalışmaların öncülüydü. Fakat önce, Pandemonium'un yapısı hakkında daha fazlasını anlatmalıyım.

Pandemonium'un yapısı, bir tür şirket örgütlenmesi çizelgesine benzer. Dipte emekçiler bulunur; Selfridge bunlara "veri cinleri" diyor. Bunlar, basılı bir harfin ya da rakamın görüntüsü gibi girdi verilerine "bakan" bilişim süreçleridir. Her cin, görüntüde özel bir öğe arar; belki bir yatay çizgi, başkası, diyelim ki dikey çizgiler, bir başkası çember eğrileri arar vs. Her cin, bulduklarını, örgütlenmede daha üst düzeydeki cinlere "haykırır". (Bu üst düzey cinlerini, orta kademe müdürler gibi düşünün.) Cin haykırışının gürlüğü, cinin, aradığı şeyi gördüğünden ne ölçüde emin olduğuna bağlıdır. Elbette Selfridge, "aramak" ve "haykırmak" gibi terimler kullanırken mecazen konuşuyor. Bilgisayarları, görüntüde belirli özellikleri "aramak" üzere programlamanın pek de zor olmadığını söylememiz yeterli olacaktır. (Selfridge, daha önce bahsetmiş olduğum 1955 tarihli makalesinde bunun nasıl yapıldığını zaten göstermişti.) "Haykırış" ise, aslında hesaplama süreci sonucu oluşan çıktının şiddetidir.

Cinlerden oluşan her üst düzey, veri cinlerinden gelen haykırışların belirli birleşimlerini dinleme konusunda uzmanlaşır. Örneğin, bu düzeydeki cinlerden biri, 3, 11 ve 22 numaralı veri cinlerinden gelen haykırışları dinlemeye ayarlanmış olabilir. Özellikle bu cinlerin gürültülü biçimde haykırdıklarını gördüğünde, o da, hiyerarşide bir üst katta bulunan kendi cinlerine haykırır vs.

Örgütlenmenin en üst düzeyinin hemen altında, Selfridge'in "bilişsel cinler" dediği şey vardır. Öteki düzeylerdeki cinler gibi bunlar da aşağı düzey cinlerinin haykırışlarının belirli birleşimlerini dinleyip en üst kattaki "karar cini"ne, yani büyük patrona kendi haykırışlarını gönderir. "Elemanlarından" gelen haykırışlara bağlı olarak, karar cini, görüntüdeki şeklin kimliği hakkındaki kararını bildirir; o şekil belki "A" harfidir, belki "R" ya da her neyse.

Asıl cin tasarımı, Pandemonium'un yapması gereken işe bağlıdır. Fakat Selfridge, her cinin görevini belirtmeksizin bile, Pandemo-

nium'un üstlendiği işi daha iyi yapmayı nasıl öğrenebileceğine dair çok ilginç önerilerde bulunmuştur. Önerilerinden biri, her cine, haykırışlarını iletmeye yarayan "megafon"a benzer bir şey tedarik etmeyi kapsıyor. Megafonun çıkardığı sesin gürlüğü ayarlanabilir. (Selfridge'in Pandemonium'u, burada betimlediğim sürümden biraz daha karmaşıktır. Onun sürümünde her cin, üstteki farklı cinlerin her biriyle iletişim kurmak için farklı kanallar kullanır. Her kanaldan yukarı giden haykırışın gürlüğü, öğrenme mekanizması tarafından tek tek ayarlanır.) Ancak, cinlerin kendi ses gürlüklerini ayarlamasına izin yoktur. Tüm ses düzeyleri, bütün yapının performansını iyileştirmeyi amaçlayan dışsal bir öğrenme süreci aracılığıyla belirlenir. Ses düzeylerinin en başta gelişigüzel ya da tasarımcının keyfine göre belirlendiğini düşünün. Ardından cihaz, deneme verileriyle sınanır ve performans puanı kaydedilir. Diyelim ki %81 puan almış olsun. Sonra, ses düzeylerinde olası her türlü şekilde ayarlamalar yapılır ta ki belirli bir ayar kümesi, performansı diyelim en fazla %83'e çıkarana dek. Bu durumda, o ayarlar hayata geçirilir ve puan artık geliştirilemeyene kadar işlem defalarca tekrarlanır (olabildiğince ek verilerle).

(Bu örgütlenmede pek çok megafon olabileceği için, her türlü ayarı denemek ve bunların tek tek puanlarını belirlemek pratikte imkânsız gibi görünebilir. Bu süreç gerçekten de çok zaman alabilir, ama bilgisayarlar hızlıdır; günümüzde daha da hızlılar. Kitabın ilerleyen kısımlarında, Pandemonium'a benzer şekilde örgütlenmiş sinir ağlarında en uygun ayarlamaları, deneyle bulmak yerine nasıl hesaplanabileceğini göstereceğim.)

Eğer söz konusu puanı bir arazinin tepe noktası, ayarları da arazi üzerindeki hareketler sayarsak, bu süreç, daima en dik yükselti doğrultusunda ilerleyerek tepe tırmanmaya benzetilebilir. Eğim artış (ya da bazen dendiği gibi tepe tırmanma yöntemleri), matematikte iyi bilinir. Selfridge, bunun kullanımının doğurabileceği tehlikeler hakkında şunu söylemiştir:

> Mekanizma ya da organizmanın küçük değişikliklerinin yükseklik ya da puan üstünde gözle görülebilir iyileşmelere neden olacağı biçimde tepelerde dolaşabilmesini sağlamak, böyle öğrenmenin sorunlarından biri olarak görülebilir. İşin en zorlu kısmının, kişinin puanını artırmanın bir yolunu bulmak olduğu birtakım öğrenme süreçlerini tasvir edebiliriz, örneğin tek tekerlekli bisiklet sürmeyi öğrenmede, bisikletin üzerinde bir saniye durmayı öğrenmek, o saniyeyi bir dakikaya çıkarmaktan uzun sürer; ayrıca başka öğrenme süreçleri var ki bunlarda azıcık öğrenmek kolayken, çok iyi öğrenmek epey zordur, örneğin satranç oynamak gibi.

> Aynı zamanda, başlıca zirvenin, tek başına duran bir yükselti yerine yayla olduğu pek çok durum da söz konusudur.

Selfridge, Pandemonium'da öğrenme için başka bir yöntem daha betimlemişti. Bu yöntem, bir kurumda doğru düzgün randıman vermeyen müdürlerin yerlerine yenilerinin atanmasına benzetilebilir. Selfridge'e kulak verelim:

> Cinlerin bir araya geldiği bu çatının doğuşunda, faydalı olacağını düşündüğümüz çok sayıda ast cini biraz keyfî şekilde toplamıştık... ama seçtiğimiz ast cinlerin becerikli olduğu konusunda bir güvencemiz yok. Ast cin seçimi, denemeden geçirilecek yeni ast cinler yaratıp verimsiz olanları, yani puanı artırmaya yaramayanları ayıklar.

Cin seçme süreci, ses gürlüğünü ayarlayan öğrenme mekanizmasının bir süre puanı iyileştiremeyişinin ardından devreye girer. Sonra, her cinin "değeri", Selfridge'e göre, haykırışlarının öğrenilmiş ses düzeyini temel alan bir yöntem kullanılarak belirlenir. Ses düzeyleri yüksek cinlerin, nihai puan üzerindeki etkileri büyüktür, dolayısıyla bunların değerinin yüksek olduğu düşünülebilir. Öncelikle, ses düzeyi düşük cinlerin hepsi elenir. (Bu adım, puana pek zarar vermez.) Ardından, cinlerden bazıları gelişigüzel "mutasyonlar" geçirir ve hizmete geri alınır. Sonrasında, değerli cinlerin kimi çiftleri seçilip, Selfridge'in dediği gibi, "eşleştirilerek" yavru cinler elde edilir. Eşleşme için Selfridge'in betimlemiş olduğu yöntemin incelikleri burada bizi ilgilendirmiyor, ancak bu sürecin ruhu, anne babalarının faydalı özelliklerini taşıdığı umulan yavrular üretmektir. Ardından bu yavrular da kullanıma açılır. Artık, hayatta kalıp "evrilmiş" cinlerin ses düzeylerini ayarlama süreci baştan başlayabilir, böylece yeni oluşumun puanı daha da iyileştirilebilir görülür.

4
ÖRÜNTÜ TANIMA

Dartmouth Yaz Projesine katılanların çoğu, insan düşüncesinin yüksek düzeylerini taklit etmekle ilgileniyordu. Bunların çalışmaları, insanların nasıl problem çözdüğüne dair içgözlemlerden belli bir oranda faydalanmıştı. Ancak, zihinsel yetilerimizin pek çoğu, içgözlem gücümüzün ötesindedir. Konuşma seslerini nasıl tanıdığımızı, eğri büğrü yazıları nasıl okuduğumuzu, fincanı tabaktan nasıl ayırt ettiğimizi ya da yüzleri nasıl tanıdığımızı bilmiyoruz. Bu işleri, üzerinde düşünmeksizin otomatik olarak beceriyoruz. İçgözlemden gelen ipuçlarından yoksun olan ilk araştırmacılar, kimi algısal becerilerimizi makineleştirmekle ilgilenmiş ve çalışmalarını, nasıl ilerleyeceklerine dair sezgisel fikirlere, basit sinir hücresi modellerinden oluşan ağlara ve istatistik tekniklerine dayandırmışlardı. İleride araştırmacılar, hayvanların görme yetisiyle ilgili sinir fizyolojisi araştırmalarından gelen ek içgörüler elde edecekti.

Kitabın bu bölümünde, 1950'ler ve 1960'larda "örüntü tanıma" konusunda yürütülmüş çalışmaları betimliyorum. Bu ifade, bir girdi görüntüsünü, bir konuşma kesitini, elektronik bir sinyali ya da herhangi bir veri kümesini çözümleyip bunu belli kategorilerden birine yerleştirme sürecini anlatır. Örneğin yazı karakteri tanıma söz konusu olunca bu kategoriler yaklaşık bir düzine abecesayısal (*alphanumeric*) yazı karakterine denk gelir.

Bu dönemde gerçekleştirilmiş örüntü tanıma çalışmalarının çoğu, basılı sayfalar ve fotoğraflar gibi iki boyutlu malzemeleri ele almıştır. Görüntüleri tarayıp bunları sayı dizilerine (ileride "piksel" denecekti bunlara) çevirmek o günlerde zaten mümkündü; ardından bu diziler, Dinneen ve Selfridge'inkiler benzeri bilgisayar programlarınca işlenebilirdi. Russell Kirsch ve Ulusal Standartlar Bürosu'ndaki (günümüzde Ulusal Standartlar ve Teknoloji Enstitüsü) çalışma arkadaşları, imge işleme konusunda ilk öncüler arasındaydı. 1957'de Kirsch, üç aylık oğlu Walden'ın fotoğrafını taramak amacıyla ilk silindir tarayıcıyı yapıp kullandı. Taranmış ilk fotoğraf olduğu söylenen bu fotoğrafın bir kenarı 176 piksel olarak ölçülmüştür (Resim 4.1).[1]

1 Bkz. http://www.nist.gov/public_affairs/techbeat/tb2007_0524.htm.



Resim 4.1. Taranmış ilk fotoğraflardan biri. (Fotoğraf, NIST'in izniyle kullanılmıştır.)

Kirsch ve meslektaşları, bu tarayıcıyı kullanarak, SEAC (Standards Eastern Automatic Computer) adlı bilgisayarlarında çalıştırdıkları resim işleme programlarıyla deneyler yapmıştır.[2]

4.1 YAZI KARAKTERİ TANIMA

Görsel imge algısıyla ilgili ilk çabalar, belgelerdeki abecesayısal yazı karakterlerini tanımaya yoğunlaşmıştı. Bu alan, "optik karakter tanıma" olarak bilinir. Ocak 1962'de Washington DC'de düzenlenmiş bir sempozyum, bu konudaki ilerlemeleri bildirmeyi amaçlamıştı.[3] Özetlemek gerekirse, o zamanlar evrak üzerindeki (daktiloyla yazılmış ya da basılı) sabit yazı tipli karakterleri makul başarıyla tanıyan cihazlar vardı. Belki de işin ne halde olduğunu, sempozyum katılımcılarından biri en güzel şekilde ifade etmiştir. Rabinow Mühendislikten J. Rabinow şunu söylemişti: "Şirketimizde bizler, basılı herhangi bir şeyi okuyabileceğimizi düşünüyoruz, hatta kimi elyazılarını bile okuyabiliriz. Tek sorun, 'bu işe ne kadar papel harcayabilirsiniz?'"[4]

2 Russell A. Kirsch ve diğ., "Experiments in Processing Pictorial Information with a Digital Computer", *Proceedings of the Eastern Joint Computer Conference*, Institute of Radio Engineering and Association for Computing Machinery, Aralık 1957, s. 221-229.

3 Konferans raporu şurada yayımlanmıştır: George L. Fischer Jr. ve diğ., *Optical Character Recognition*, Washington, DC: Spartan Books, 1962.

4 J. Rabinow, "Developments in Character Recognition Machines at Rabinow Engineering Company", George L. Fischer Jr. ve diğ., *Optical Character Recognition* içinde, s. 27.



Resim 4.2. MICR yazı karakteri kümesi.

1950'lerde göze çarpan bir uygulama, SRI International (o zamanlar Stanford Araştırma Enstitüsü deniyordu) araştırmacılarının geliştirmiş olduğu manyetik mürekkep karakter tanıma sistemiydi (MICR); bu sistemin görevi, çeklerin alt tarafındaki stilize manyetik mürekkepli karakterleri okumaktı (bkz. Resim 4.2). MICR, SRI'nın, çek işlemlerini ve çek hesabı idaresi ile kayıtlarını otomatikleştiren ERMA (Electronic Recording Method of Accounting; Elektronik Muhasebe Kayıt Yöntemi) adlı sisteminin parçasıydı.

Bir SRI internet sitesine göre: "Nisan 1956'da Bank of America, üretim modelleri imal etmek üzere General Electric şirketinin seçildiğini duyurdu... 1959'da General Electric, Amerika Bankası'na ilk 32 ERMA hesap sistemini teslim etti. ERMA, 1970'e dek Banka'nın muhasebe bilgisayarı ve çek idaresi sistemi olarak hizmet etmiştir."[5]

O zamanlar tanıma yöntemlerinin çoğu, bir yazı karakterini (sayfada ayırt edilip 0 ve 1'ler dizisine dönüştürdükten sonra), bu karakterin "şablon" denen prototip sürümleriyle (bunlar da bilgisayarda dizi olarak saklanıyordu) eşleştirmeye dayanıyordu. Eğer bir yazı karakteri, "A" şablonuna uymuşsa, örneğin öteki şablonlara kıyasla buna daha iyi eşleniyorsa, o girdinin "A" yazı karakteri olduğu söylenir. Eğer girdi karakterleri standart bir eğimle sunulmamışsa, şablonla aynı yazı tipinde değilse ya da kusurları varsa, tanıma işleminin hata payı artar.

Selfridge ile Dinneen'in 1955 tarihli makaleleri (yukarıda bundan bahsetmiştim), şablon eşleştirmenin ötesine geçen kimi fikirler ileri sürmüştü. Oliver Selfridge ve Ulrich Neisser imzalı makale (1960), bu çalışmayı daha da ileri taşımıştı.[6] Bu makale önemlidir, çünkü elyazısı karakteri tanımada görüntü işlemeyi, özellikleri belirlemeyi

5 http://www.sri.com/about/timeline/erma-micr.html.

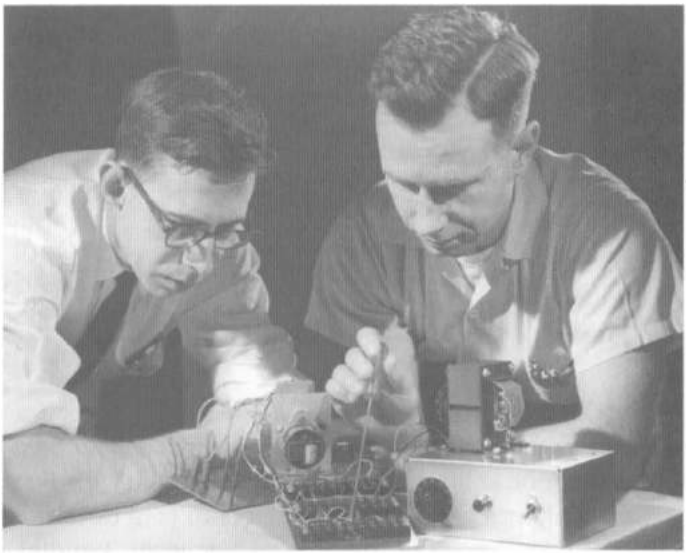
6 Oliver G. Selfridge ve Ulrich Neisser, "Pattern Recognition by Machine", *Scientific American*, c. 203, s. 60-68, 1960. (Yeniden baskısı: Edward A. Feigebaum ve Julian Feldman (ed.), *Computers and Thoughts*, New York:McGraw Hill, 1963, s. 237vd.)

ve öğrenilmiş olasılık değerlerini kullanmaya yönelik başarılı, erken girişimlerden biriydi. Harfler taranıp 32 × 32'lik bir "retinada" yani 0 ve 1'lerden oluşan bir dizide temsil ediliyordu. Ardından, gelişigüzel parazitleri kaldırmak, boşlukları doldurmak, hatları kalınlaştırmak ve kenar çizgilerini vurgulamak için çeşitli rafineleştirme yöntemleriyle işleniyordu (bu yöntemler, 1955 tarihli Dinneen makalesiyle bağlantılı olarak bahsettiğim yöntemlere benzer). Böylece "temizlenmiş" görüntüler, belli "özellikler" var mı yok mu diye inceleniyordu (bu özellikler, 1955 tarihli Dinneen makalesiyle bağlantılı olarak bahsettiğim özelliklere benziyor). Toplamda 28 özellikten faydalanılmıştı; örneğin, görüntüyü kesen azami yatay çizgi sayısı, farklı kenar çizgilerinin göreli uzunlukları, acaba görüntünün "güneye bakan bir içbükeyliği" var mı gibi özelliklerden yararlanılmıştı.

Selfridge'in Pandemonium sistemini anımsarsak, "cinlerin" yürüttüğü özellik tespit etme süreçleri aklımıza gelir. Hiyerarşide, özellik tanıyan cinlerin bir düzey üstünde "tanıma cinleri" bulunur; her harf için bir tane. (Sistemin, Lincoln Laboratuvarı'ndan Worthie Doyle tarafından sınanmış sürümü, elle yazılan on farklı harfi tanımak üzere tasarlanmıştı: A, E, I, L, M, N, O, R, S, T.) Her tanıma cini, özellik tespit eden cinlerin her birinden girdi alır; fakat öncelikle, her tanıma cinine gelen girdiler, ilgili özelliğin, karar verme sürecine katkısının önemini hesaba katan bir ağırlık değeriyle çarpılır. Örneğin, eğer 17 numaralı özellik, girdi harfinin "A" yazı karakterinde olduğuna karar verme meselesinde 22 numaralı özellikten daha önemliyse, o halde 17 numaralı özellikten "A" tanıyıcısına gelen girdi, 22 numaralı özellikten gelen girdiden ağır basar. Her tanıma cini, ağırlık verdiği girdilerin toplamını hesapladıktan sonra, "karar cini", en yüksek toplama ulaşan yazı karakteri lehine karar verir.

Söz konusu ağırlıkların değerleri, 330 "eğitici" görüntünün çözümlendiği bir öğrenme süreci tarafından belirlenir. Alıştırma kümesinde her farklı harf için, her özelliğin kaç kere bulunduğu, çizelge haline getirilir. Bu istatistiksel veriler, her harf için tespit edilen belirli bir özelliğin olasılıklarını tahmin etmek amacıyla kullanılır. Bu olasılık tahminleri, tanıyıcı cinlerin topladığı özelliklerin ağırlığını tartmak için kullanılır.

Eğitim sürecinden sonra, sistem, henüz görmemiş olduğu elyazısı harflerle sınanır. Selfridge ile Neisser'a göre, "bu program, insan okurlardan yüzde on düşük bir randıman veriyor; kesinlikle saygın bir başarı."



Resim 4.3. Frank Rosenblatt (solda), A-birimi prototipi üzerinde çalışıyor (Charles Wrightman'la birlikte). (Cornell Üniversitesi Kütüphanesi, Nadir Eserler ve El Yazmaları Dairesinin izniyle.)

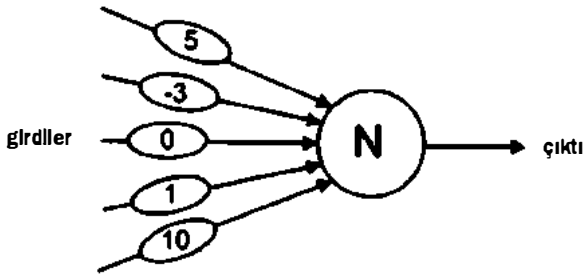
4.2 SİNİR AĞLARI

4.2.1 Perceptron

New York Buffalo'daki Cornell Havacılık Laboratuvarı psikologlarından Frank Rosenblatt (1928-1969; Resim 4.3) 1957'de PARA (Perceiving and Recognizing Automoton; Algılayan ve Tanıyan Otomatlar) adlı proje kapsamında sinir ağları üzerine çalışmaya başladı. Rosenblatt, McCullooch ile Pitts'in ve Hebb'in eski çalışmaları etkilenmişti ve *perceptron* (algılayıcı) dediği bu ağlara, insanın öğrenme, biliş ve bellek yetilerinin olası modeli olarak ilgi duyuyordu.[7]

1960'ların ilk yarısında Ithaca Cornell Üniversite'sinde profesör olarak çalışmalarını sürdüren Rosenblatt, farklı perceptron türleriyle deneyler yaptı. Clark ile Farley'in ve sinir ağları alanındaki başka öncülerin çalışmalarından daha çok onun çalışması, YZ'deki simge işleme yöntemlerine, yani sinir ağlarının başlıca alternatiflerinden birini başlatmaktan sorumluydu.

7 İlk kaynakçalardan biri: Frank Rosenblatt, "The Perception: A Probabilistic Model for Information Storage and Organization in the Brain", *Pyschological Review*, c. 65, 1958, s. 386 vd.

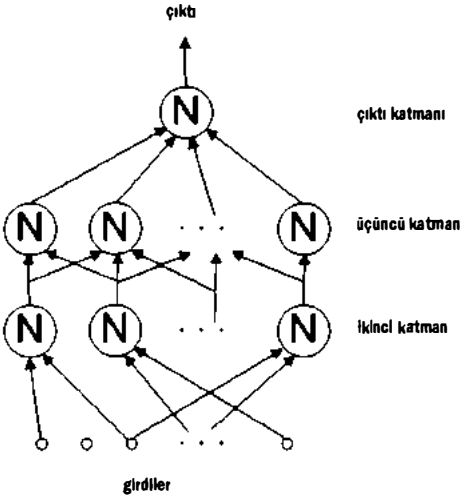


Resim 4.4. Rosenblatt'ın sinirsel öğesi, ağırlıklarıyla birlikte.

Rosenblatt'ın perceptronları, McCulloch-Pitts tarzı sinirsel öğelerden meydana geliyordu, tıpkı Resim 4.4'te gösterilen çizimdeki gibi. Her öğede girdiler (resimde sol taraftan geliyor), "ağırlıklar" (girdi çizgilerinin üzerindeki yuvarlaklarla gösteriliyor) ve tek çıktı (sağ tarafa uzanan ok) vardı. Girdilerin değeri ya 0 ya da 1'di ve her girdi, kendisiyle ilişkili ağırlık değeriyle çarpılıyordu. Sinirsel öğe, ağırlık verilmiş bu değerlerin toplamını hesaplıyordu. Örneğin, Resim 4.4'te sinirsel öğeye gelen tüm girdiler 1'se, toplamları 13 olur. Eğer bu toplam, sinirsel öğeyle bağlantılı bir "eşik değeri"nden, örneğin 7'den büyükse, o zaman sinirsel öğenin çıktısı 1 olur, ki bu örnekte öyle; yoksa çıktı 0 olacaktır.

Perceptron, bu sinirsel öğelerin ağından meydana gelir; bu ağda bir öğenin çıktıları, başka öğelerin girdisi olur. (Bu düzen, Selfridge'in Pandemonium'una benziyor, çünkü orada orta kademe cinlere alt düzey cinlerin "haykırışları" ulaşır. Sinirsel öğelere gelen girdi hatlarının ağırlıkları, Pandemonium'un şiddeti artan ya da şiddeti azalan "ses denetimine" benzetilebilir.) Resim 4.5'te bir perceptron örneği gösteriliyor. [Rosenblatt, kendi perceptron şemalarını yatay biçimde çizerdi (elektrik mühendisi tarzında); girdileri sola, çıktıları sağa yerleştirirdi. Burada, hiyerarşi çizimlerinde genellikle bilgisayar bilimcilerinin tercih ettiği dikey tarzı kullanıyorum: en alçak düzey dipte, en yüksek düzey üsttedir. Çizimi sadeleştirmek için, ağırlıkları belirten yuvarlaklar eklenmemiştir.] Her ne kadar sergilenen perceptron, sadece tek çıktı birimiyle ancak iki farklı çıktı (0 ve 1) verebiliyor olsa da, birkaç çıktı biriminden oluşan düzende çoklu çıktı (1 ve 0'lar kümesi) elde edilebilir.

Resim 4.5'in alt tarafında gösterilen girdi katmanı, siyah-beyaz görüntünün "pikselleri" denen hücrelere denk gelen 1 ve 0'ların dikdörtgen biçimli bir düzenidir. Rosenblatt'ın ilgilendiği uygulamalardan biri, tıpkı Selfridge gibi, yazı karakteri tanımaydı.



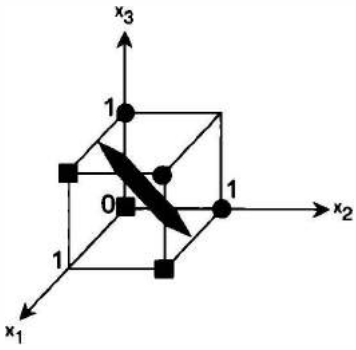
Resim 4.5. Perceptron

Perceptron ağlarındaki sinirsel öğelerin, arzulanan çıktıları yaratmak üzere nasıl "eğitilebileceğini" göstermek amacıyla birtakım basit cebir ve geometri örneklerinden yararlanacağım. Örneğin, girdi değerleri x_1, x_2 ve x_3 üstlendiği ağırlık değerleri w_1, w_2 ve w_3 olan tek bir sinirsel öğeyi gözümüzün önüne getirelim. Bu öğenin hesapladığı toplam, tamı tamına eşik değerine, yani t'ye eşit olduğunda, elimizde şu denklem var demektir:

$$w_1x_1 + w_2x_2 + w_3x_3 = t$$

Cebirde, böyle bir denkleme "doğrusal denklem" denir. Bu denklem, doğrusal bir sınırı, yani üç boyutlu uzayda bir düzlemi tanımlar. Söz konusu düzlem, sinirsel öğenin çıktı olarak 1 değerini almasını sağlayan girdi değerlerini, 0 değerini almasını sağlayanlardan ayırır. Resim 4.6'da, tipik bir düzlemsel hattı gösteriyorum.

Sinirsel öğeye gelen girdiler, üç boyutlu bu uzayda birer nokta (yani vektör) olarak betimlenebilir. Bu noktanın koordinatları x_1, x_2 ve x_3'tür ve her biri ya 1 ya da 0 değeri alabilir. Resim, bu tür altı nokta gösteriyor; bunlardan üçü (küçük çemberler) sinirsel öğenin çıktısını 1 yaparken, üçü de (küçük kareler) çıktıyı 0 yapıyor. Eşiğin değerini değiştirmek, düzlemin, kendine paralel olan bir doğrultuda yanlara hareket etmesine yol açar. Ağırlıkların değerini değiştirmek, düzlemin kendi ekseni etrafında dönmesine sebep olur. Dolayısıyla, ağırlık değerleri değiştirilerek, düzlemin bir tarafında kalan noktaların öbür tarafa geçmesi sağlanabilir. Sistemin "eğitilmesi", bu gibi değişiklik-



Resim 4.6. Üç boyutlu uzayda sınır düzlemi.

ler sayesinde gerçekleştirilir. Biraz sonra eğitim yöntemleri hakkında daha fazla şey anlatacağım.

Üçten fazla boyutta (genelde söz konusu olan durum budur), doğrusal sınıra "hiperdüzlem" denir. Gerçi yüksek boyutlu uzaylarda olup bitenleri göz önüne getirmek kolay değil ama matematikçiler bu uzaydaki girdi noktalarından, ağırlıkların ve eşiklerin değerlerindeki değişikliklere bağlı olarak hiperdüzlemlerin dönüşlerinden ve hareketlerinden bahseder.

Rosenblatt, birkaç perceptron türü tanımlamıştı. Çizimde gösterilen perceptrona "seri-bağlanmış, dört katmanlı perceptron" diyordu. (Rosenblatt, girdileri ilk katman sayıyordu.) "Seri-bağlanmış" denmesinin sebebi, her sinirsel öğenin çıktısının, sonraki katmandaki sinirsel öğeleri beslemesidir. Daha yeni bir terminolojide, "seri-bağlanmış" yerine "ileri besleme" denir. Bunun aksine "çapraz bağlanmış" perceptronda, bir katmandaki sinirsel öğelerin çıktıları aynı katmandaki sinirsel öğelerin girdisi olabilir. "Geri-bağlanmış" perceptronda ise, bir katmandaki sinirsel öğelerin çıktıları, alt katmanlardaki sinirsel öğelerin girdisi olabilir.

Rosenblatt, perceptronlarını, beyin içinde bağlantı kuran parçaların modeli sayıyordu. Bu sebeple, çıktı katmanı hariç tüm katmanlardaki sinirsel öğelere "bağdaşım birimleri" ("A-birimleri"; *association units*) demiştir, çünkü bu birimlerin, beyindeki sinir hücresi ağlarının bağdaşımlarını modellemesini istiyordu.

Rosenblatt'ın araştırmasında özellikle ilgi çeken şey, "alfa-perceptron" dediği şeydir. Bu yapı, üç katmanlı ileri beslemeli ağdan meydana gelir; bir girdi katmanı, bir bağdaşım katmanı ve en az bir çıktı birimi bulunur. Rosenblatt'ın deneylerinin çoğunda girdi değerleri ya 0 ya da 1'dir; bu, "retina" dediği şeyin üzerinde sunulan görsel imgenin siyah

ya da beyaz piksellerine denk gelir. Her A-birimi, piksellerin rasgele seçilmiş bir altkümesinden girdi alır (bunlar, ağırlık değerleriyle çarpılmaz) ve çıktısını, bir dizi ayarlanabilir ağırlık aracılığıyla, en son çıktı birimlerine gönderir; bu birimlerin çiftli değerleri, girdi görüntü sınıfı için kod olarak yorumlanabilir.

Alfa-perceptronundaki çıktı birimlerini ayarlamak amacıyla çeşitli "eğitim yöntemleri" denenmiştir. Bunların en başarılılarından birinde (örüntü tanıma bakımından), çıktı birimlerine giren ağırlıklar, bu birimler ancak bir girdiyi sınıflandırırken hata yaptığında ayarlanabilir. Ayarlar öyle yapılır ki, çıktıyı, tam da o girdi için doğru sınıflandırmayı yapmaya zorlar. Kısa süre içinde standart haline gelen bu tekniğe, "hata düzeltme yöntemi" denir. Rosenblatt, abecesayısal yazı karakteri gibi görsel girdileri ya da konuşma sesleri gibi akustik girdileri sınıflandırma konusunda perceptonları eğittiği pek çok deneyde bu yöntemi başarıyla kullanmıştır. Rosenblatt'la çalışan Cornell matematikçilerinden profesör H. David Block, bu yöntemin, öyle bir hiperdüzlem var olduğunda, eğitim girdileri kümesini kusursuz ayıran bir hiperdüzlem bulmasının kesin olduğunu ispatlamıştı.[8] (SRI matematikçisi Albert B. Novikoff gibi başka matematikçiler, daha şık kanıtlar geliştirmiştir.[9] Bu kanıtın bir türünü, *Learning Machines* [Öğrenen Makineler] adlı kitabımda sunuyorum.[10])

Her ne kadar bu sistemin uygulanabilirliği ve tasarımıyla ilgili kimi çalışmalar bilgisayar benzetimleri kullanılarak yapılmış olsa da, Rosenblatt, kendi perceptronlarının donanım gerçekleşmesini inşa etmeyi seçmişti. (İlk bilgisayarlarda benzetim programları yavaş çalışıyordu; bu durum, özel-amaçlı perceptron donanımını inşa etmeye gösterilen ilgiyi açıklar.) MARK I, Cornell Havacılık Laboratuvarı'nda Deniz Araştırmaları Dairesi Bilgi Sistemleri Şubesi ile Rome Havacılık Geliştirme Merkezi'nin hamiliğinde inşa edilmiş bir alfa-perceptrondu. Halka açık gösterimi ilk kez 23 Haziran 1963'te gerçekleştirildi. MARK I, ağırlıklar için ses ayar düğmeleri (elektrik mühendislerinin

8 H. David Block, "The Perception: A Model for Brain Functioning", *Reviews of Modern Physics*, c. 34, no: 1, Ocak 1962, s. 123-135.

9 Albert J. Novikoff, "On Convergence Proofs for Perceptions", *Proceedings of the Symposium on Mathematical Theory of Automata* içinde, Brooklyn, NY: Polytechnic Press of Polytechnic Inst. Of Brooklyn, 1963, s. 615-622.

10 Nils J. Nilsson, *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, New York: McGraw-Hill Book Co., 1965; yeniden yayımlanışı: *The Mathematical Foundations of Learning Machines*, San Francisco: Morgan Kaufmann Publishers, 1990.

"gerilimöçer" dediği şey) kullanıyordu. Bunların üzerine, ağırlık değerlerini artırmaya ya da azaltmaya yarayan ayarları yapsınlar diye küçük motorlar iliştirilmişti.

1959'da Frank Rosenblatt, perceptron çalışmasını New York Buffalo'daki Cornell Havacılık Laboratuvarı'ndan, psikoloji profesörü olduğu Cornell Üniversitesi'ne taşıdı. Block ve birkaç öğrenciyle birlikte Rosenblatt, perceptronlar hakkında deneysel ve kuramsal çalışmalarını sürdürdü. *Principles of Neurodynamics* [Sinir Dinamiği İlkeleri] başlıklı kitabı, kuramsal fikirlerinin ve deney sonuçlarının ayrıntılı bir incelemesini sunar.[11] Rosenblatt'ın Tobermory isimli son sistemi, konuşma tanıma cihazı olarak inşa edilmişti.[12] [Saki'nin (H. H. Munro) yazdığı bir dizi kısa öyküden oluşan *The Chronicles of Clovis* kitabında, Tobermory, konuşmayı öğrenen kedinin adıdır.] George Nagy, Carl Kessler ve R. D. Joseph dahil kimi doktora öğrencileri, Cornell'de Rosenblatt'ın gözetimi altında perceptron projeleri tamamlamıştır.

Rosenblatt, Cornell'de geçirdiği son yıllarda, yassı solucanların ve birtakım başka hayvanların kimyasal bellek aktarımını araştırmaya kaydı; bu konu, perceptron çalışmalarından hayli uzaktı. Ne acı ki Rosenblatt, 1969'da Chesapeake körfezinde uğradığı tekne kazasında vefat etti.

Rosenblatt'ın alfa-perceptronuyla yaklaşık aynı zamanda, New Mexico Albuquerque'daki Sandia Laboratuvarları'nın iki matematikçisi, Woodrow W. (Woody) Bledsoe (1921-1995) ile Iben Browning (1918-1991), rasgele görüntü girdileri kullanan bir yazı karakteri tanıma sistemine ilişkin araştırmalarını sürdürüyorlardı. Deney yaptıkları sistem, 10×15'lik ışıkgözü (fotosel) mozaiği üzerinde abecesayısal görüntüler yansıtıyor ve rasgele seçilmiş 75 ışıkgözü çiftinin durumunu örnekliyordu. Bu fikrin, daha büyük piksel gruplarını, örneğin *N* sayıda pikseli içerecek denli kapsamının genişletilebileceğine işaret ederek, bu yönteme "*N*-li" yöntem adını verdiler. Bu örnek toplama işleminin sonuçlarını, girdi harflerinin sınıfını belirlemek amacıyla kullanıyorlardı.[13]

11 Frank Rosenblatt, *Principles of Neurodynamics*, Washington, DC: Spartan Books, 1962.

12 Frank Rosenblatt, "A Description of the Tobermory Perceptron", *Collected Technical Papers*, c. 2, Cognitive Systems Research Program, Cornell Üniversitesi, 1963.

13 Woodrow W. Bledsoe ve Iben Browning, "Pattern Recognition and Reading by Machine", *Proceedings of the Eastern Joint Computer Conference*, New York: Association for Computing Machinery, 1959, s. 225-232.

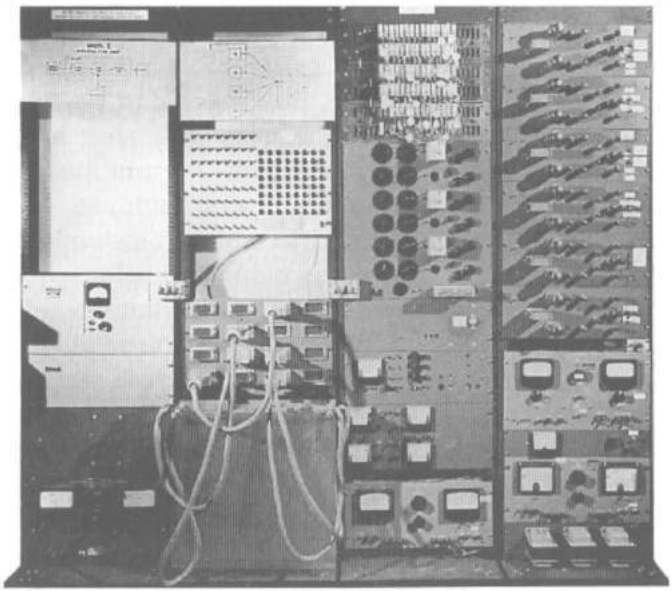
4.2.2 ADALINE ve MADALINE

Başını Stanford Elektrik Mühendisliği Bölümü'nden Profesör Bernard Widrow'un (1929-) çektiği bir ekip, Rosenblatt'dan bağımsız olarak, 1950'lerin ikinci yarısında ve 1960'ların başında sinir ağı sistemleri üzerinde çalışmıştı. Widrow, MIT'de kontrol kuramı alanında doktorasını tamamladıktan sonra Stanford'a yeni katılmıştı. "Uyarlanımsal denetim" dediği denetim biçimi için sinir ağı sistemlerini kullanmak istiyordu. Widrow'un inşa etmiş olduğu cihazlardan birinin adı "ADALINE"dir (*adaptive linear network*, yani uyarlanımsal-doğrusal ağ anlamında). Ayarlanabilir ağırlıkları açılıp kapanır nitelikli (dolayısıyla ayar yapmaya uygun) direnç elemanı devreleriyle yürürlüğe giren, tek bir sinirsel öğeydi bu. Widrow ve öğrencilerinden Marcian E. "Ted" Hoff. Jr. (ileride İntel'de ilk mikroişlemciyi icat edecekti), "memistor" dedikleri ayarlanabilir ağırlığı icat ettiler. Memistor, bakır bir plakayla kaplanabilen ya da bu tabakanın kaldırılabildiği grafit bir çubuktan meydana geliyordu; böylece çubuğun elektrik direnci değişiyordu. Widrow ve Hoff, ADALINE sinirsel öğesi için bir eğitim yöntemi geliştirmişlerdi; bu yönteme, Widrow-Hall en küçük-ortalamalı-kareler uyarlama algoritması denir. Widrow'un deneysel çalışmalarının büyük kısmı, IBM1620 bilgisayarında çalıştırılan benzetimlerle gerçekleştirilmiştir. En karmaşık ağ tasarımlarının adı "MADALINE" (çoklu ADALINE anlamında) idi. Buna yönelik eğitim yöntemini, Stanfordlu doktora öğrencisi William Ridgway geliştirmiştir.[14]

4.2.3 SRI'daki MINOS Sistemleri

Örüntü tanıma uygulamalarında Rosenblatt'ın perceptronlarla yakaladığı başarı, onun sonuçlarını tekrarlayıp kapsamını genişletmek isteyenlerin başını çektiği bir araştırma furyası doğurdu. 1960'lar boyunca belki de en göze çarpan sinir ağı temelli örüntü tanıma çalışması, California Menlo Park'ta Stanford Araştırma Enstitüsü'nde gerçekleştirilmişti. Burada Charles Rosen (1917-2002), mikroskopik vakum tüplerini, katı hal malzemelerin üzerine kaynaklamaya çalışan bir laboratuvarın başındaydı. Rosen, bu tür tüpler içeren devrelerin, Frank Rosenblatt'ın keşfetmiş olduğu kimi eğitim yöntemlerini kullanarak birtakım faydalı görevleri yerine getirecek şekilde birbirine "bağlana-

14 William C. Ridgway, "An Adaptive Logic System with Generalizing Properties", Stanford Elektronik Laboratuvarı Teknik Raporu 1556-1, Stanford Üniversitesi, Stanford, CA, 1962.

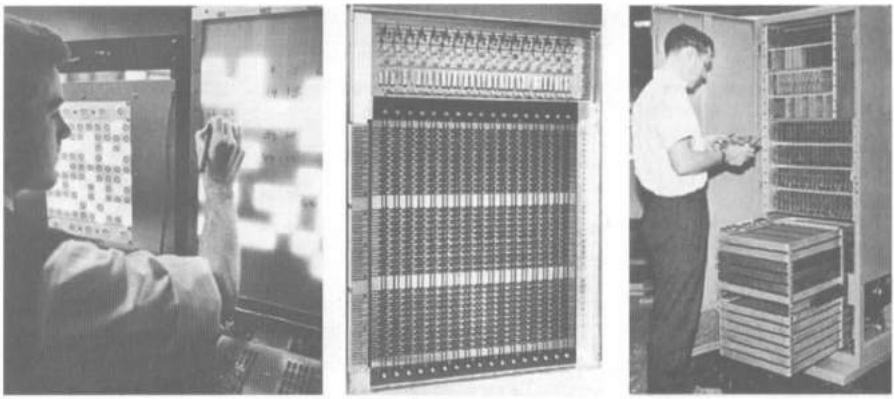


Resim 4.7. MINOS. Girdi anahtarlarına ve donanımın soldan ikinci rafında duran bunların işaret ışıklarına dikkat edin. Manyetik ağırlıklar, üçüncü rafın üst kısmındadır. (Fotoğraf, SRI International'ın izniyle kullanılmıştır.)

bileceği" fikrini ileri sürmüştü. SRI, keşifçi bir sinir ağının tasarımına yardımcı olması amacıyla Rosenblatt'ı danışman olarak işe aldı.

1960'da SRI'da iş için mülakata girdiğimde, Rosen'ın laboratuvarında Alfred E. (Ted) Brain (1923-2004) önderliğinde bir ekip, MINOS adlı küçük bir sinir ağını henüz tamamlamıştı (Resim 4.7). (Yunan mitolojisinde Minos, Girit kralıdır ve Zeus ile Europa'nın oğludur. Ölümünden sonra, yeraltı âleminin üç yargıcından biri olmuştur.) Brain, sinir ağlarıyla ilgili bilgisayar benzetimlerinin fiili uygulamalarda fazlasıyla yavaş olduğunu düşünüyordu, dolayısıyla programlamak yerine inşa etmeye karar vermişti. (Aynı dönemde Stanford'da sinir ağlarının benzetimini yapmaya uğraşan Widrow ekibince kullanılan IBM 1620 bilgisayarının, 21 saniyelik temel makine devri ve azami 60.000 "hane"li rasgele erişimli belleği vardı.) Ayarlanabilir ağırlıklar bakımından MINOS, Brain'in tasarlamış olduğu manyetik cihazlardan faydalanıyordu. Rosenblatt SRI'yla irtibatını koparmamıştı, çünkü motor itmeli gerilimölçerler yerine bu manyetik cihazları kullanmakla ilgileniyordu.

Sinir ağlarının olanakları konusunda Rosen'in şevki ve iyimserliği, SRI'ya katılmaya beni ikna etmişti. Temmuz 1961'de SRI'ya gel-



Resim 4.8. MINOS II: Operatörün görüntü kartı (solda), tek başına bir ağırlık çerçevesi (ortada) ve mantık devreleriyle birlikte ağırlık çerçeveleri (sağda). (Fotoğraflar, SRI International'ın izniyle kullanılmıştır.)

memle birlikte, okuyayım diye elime Rosenblatt'ın kitabının bir taslağı tutuşturuldu. Brain'in ekibi MINOS II isimli büyük sinir ağının yapımı üzerinde çalışmaya yeni başlıyordu; bu sistem, daha küçük olan MINOS'un devamı niteliğindeydi. (Resim 4.8)

MINOS sistemleriyle ilgili çalışmaları, 1958 ile 1967 arasında öncelikle ABD Kara Kuvvetleri Muhabere Sınıfı desteklemişti. MINOS çalışmalarının amacı, "askerî gereksinimler uyarınca grafik veri işlemeye yönelik pratik uygulamalara ilişkin tekniklerin ve ekipman özelliklerinin araştırılıp deneylerle incelenmesi"ydi. Projenin başlıca hedefi, askerî haritalar üzerindeki simgelerin otomatik tanınmasıydı. Havadan çekilmiş fotoğraflarda tank gibi askerî araçların tanınması ve elyazısı harflerin tanınması misali başka uygulamalar da denenmişti.[15]

MINOS II işleminin ilk safhasında girdi imgesi, 10×10'luk plastik mercek düzeneğiyle 100 kez çoğaltılır. Ardından bu özdeş görüntüler tek tek, optik öznitelik tespit etme maskesinden geçirilir; ve maskeden geçen ışık ışıkgözüyle sezilip bir eşik değeriyle kıyaslanır. Elde edilen sonuç, 100 adet ikili (açık-kapalı) değerden oluşan bir kümedir. Bu değerler, her biri 100 değişken manyetik ağırlık taşıyan 63 sinirsel öğeden oluşan (Rosenblatt'ın terminolojisiyle "A-birimleri") bir

15 MINOS II'nin bir betimlemesi için bkz. Alfred E. Brain, George Fosen, David Hall ve Charles Rosen, "A Large, Self-Contained Learning Machine", *Proceedings of the Western Electronic Show and Convention*, 1963. Bu makale, bir SRI proje önerisinin C Eki olarak yeniden basılmıştır. Bkz. http://www.ai.sri.com/pubs/files/rosen65-esu65-1tech.pdf.

kümeye gelen girdilerdir. Bu sinirsel öğelerden elde edilen 63 adet ikili çıktı, özgün girdi imgesinin kategorisine dair 64 karardan birine dönüştürülür. (63 boyutlu uzayda eşit mesafeli 64 "nokta" tanımlamış ve sinir ağını öyle eğitmiştik ki her imge girdisi, ötekilere kıyasla kendi prototip noktasına yakın duran bir nokta yaratıyordu. Bu prototip noktalarından her biri, 63 boyutun 64 adet "azami-uzunluk-kaydırma-yazmacı dizisi"nden biridir.)[16]

1960'lar boyunca, o zamanlar Öğrenen Makine Ekibi denen SRI sinir ağı ekibi, farklı farklı pek çok ağ düzeneği ve eğitim yöntemi keşfetmişti. Bilgisayarlar hem yaygınlık kazanıp hem de güçlendikçe, (çeşitli bilgisayar merkezlerinde) Burroughs 220 ile 5000 ve IBM 709 ile 7090 bilgisayarlarında benzetimlerden faydalanmaya başlamıştık. 1960'ların ortasında, kendimize özel bilgisayarımız olmuştu artık: SDS 910. (Scientific Data Systems'de geliştirilmiş olan SDS 910, silikon transistörler kullanan ilk bilgisayardı.) Bu bilgisayarı, sinir ağı donanımımızın (artık 1.024 önişleme yapan mercek dizisi kullanıyordu) en son sürümüyle birlikte kullanıyorduk; bu birleşime MINOS III diyorduk.

MINOS III sistemiyle elde edilmiş en başarılı sonuçlardan biri, FORTRAN kodlama kâğıtları üzerindeki elyazısı karakterlerin otomatik tanınmasıydı. (1960'larda bilgisayar programları genelde elle yazılır, sonra delikli kart operatörleri bunları delikli kartlara dönüştürürdü.) Bu çalışmanın önderliğini John Munson (1939-1972; Resim 4.9), Peter Hart (1941- ; Resim 4.9) ve Richard Duda (1936- ; Resim 4.9) üstlenmişlerdi. MINOS III sisteminin sinir ağı kısmı, her yazı karakteri için belli bir güvenirlik payıyla birlikte olası bir sınıflandırma sıralaması yaratıyordu. Örneğin, bir dizi yazı karakterinde karşılaşılan ilk karakter, sinir ağı tarafından 90 güvenirlikle "D" olarak ve 10 güvenirlikle "O" olarak tanınabilir. Fakat her yazı karakteri için en güvenilir kararı kabul etmek, FORTRAN dilinde geçerli bir komutla sonuçlanmayabilir; bu durum, kararlardan en az birinin hatalı olduğunu gösteriyor (kodlama kâğıdına komutları yazan kişinin geçerli komutlar yazmış olduğunu varsayıyoruz.) Geçerli bir komut yaratmak için karakterlerden bazılarında en güvenilir ikinci ya da üçüncü tercihleri kabul etmek gerekebilir.

16 Kaydırmalı yazmaç (*shift-register*) kodlarıyla ve başka kodlarla ilgili bahis için bkz. W. Peterson, *Error-Correcting Codes*, New York: John Wiley & Sons, 1961. Bizim tekniğimiz şurada bildiriliyor: A. E. Brain ve N. J. Nilsson, "Graphical Data Processing Research Study and Experimental Investigation", Üç Aylık İlerleme Raporu, no: 8, SRI Raporu, Haziran 1962, s. 11. Bkz. http://www.ai.sri.com/pubs/files/1329.pdf.



Resim 4.9. John Munson (solda), Peter Hart (ortada) ve Richard Duda (sağda). (Fotoğraflar, Faith Munson'un, Peter Hart'ın, Richard Duda'nın izinleriyle.)

Bütün bir karakter dizisinin toplam güvenirliği, dizideki karakterlerin tek tek güvenirliği toplanarak hesaplanır. Ardından gerekli olan şey, her karakter için yapılmış farklı tercihlerinden türeyen olası dizilerin her biri için bu toplam güvenilirlik rakamlarını sıralamanın bir yoluydu. Tüm olası dizilerin bu sıralaması arasından, sistem, en güvenilir *geçerli* komutu seçer.

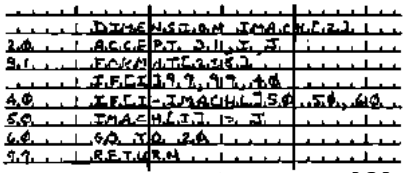
Ancak, Richard Duda'nın yazdığı gibi, "En güvenilir 1., 2., 3. karakter dizisini bulma sorunu... hiç de öyle basit bir iş değildir." Sıralamayı etkin şekilde hesaplamada kilit adım, *dinamik programlama* denen bir yöntem kullanmaktı.[17] (İlerideki bir bölümde, dinamik programlamanın konuşma tanıma sistemlerinde de kullanıldığını göreceğiz.)

Özgün kaynağın ve son çıktı için bir örnek Resim 4.10'da gösteriliyor.

Sistemin sinir ağı kısmı eğitildikten sonra, sistemin bütünü (en güvenilir geçerli komuta karar vermiştir), sistemin eğitiminde kullanılmamış büyük bir örnek kümesinde %98'i aşan tanıma doğruluğuna ulaşabilmişti. Elle yazılmış harfleri bu doğruluk düzeyiyle tanımak, 1960'larda önemli bir başarıydı.[18]

17 SRI'dan Robert E. Larsen, bu yöntemin kullanımını önermiştir. Çevrimiçi ansiklopedi Wikipedia, dinamik programlamayı gayet anlaşılır biçimde anlatıyor. Bkz. http://en.wikipedia.org/wiki/Dynamic_programming.

18 Sistemin bütününün teknik ayrıntıları, iki makalede betimlenmiştir: John Munson, "Experiments in the Recognition of Hand-Printed Text: Part I – Character Recognition" ve Richard O. Duda ve Peter E. Hart, "Experiments in the Recognition of Hand-Printed Text: Part II – Context Analysis", *AFIPS Conference Proceedings*,



```
      DIMENSION IMACM[2]
20    ACCEPT 31,I,J
31    FORMAT[215]
      IF[I]79,99,40
40    IF[I-IMACHL]50,50,60
50    IMACH[I]=J
60    GO TO 20
99    RETURN
```

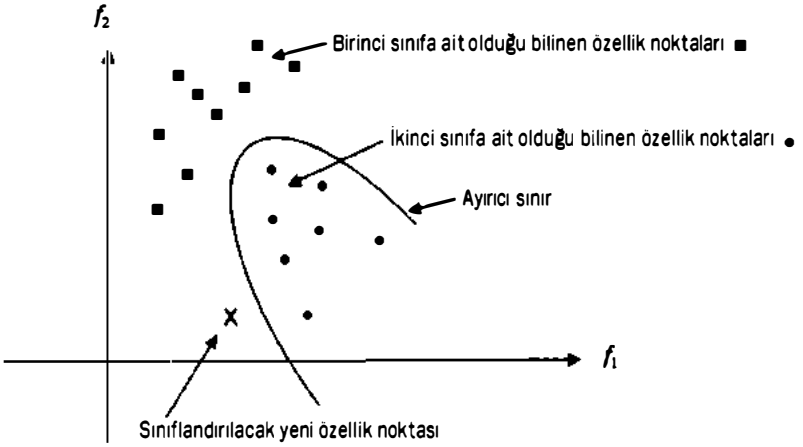
Resim 4.10. FORTRAN karakterlerinin tanınması. Girdi üstte, çıktı (yalnızca iki hatayla) altta. (Çizim, SRI International'ın izniyle kullanılmıştır.)

İlgi alanını sinir ağlarının ötesine taşıyan Öğrenen Makineler Ekibi, sonradan SRI Yapay Zekâ Merkezi'ne dönüşecekti; günümüzde bu merkez başta gelen bir YZ araştırma kurumu olmayı sürdürüyor.

4.3 İSTATİSTİKİ YÖNTEMLER

1950 ve 1960'larda, istatistiksel yöntemlerinin örüntü tanımada çok sayıda uygulaması vardı. Bu yöntemlerden birçoğu, kimi sinir ağı tekniklerine yakından benzer. Hatırlarsanız, gürültülü bir radyo sinyalinde iki tondan hangisinin var olduğuna nasıl karar verildiğini açıklamıştım. Örüntü tanıma için benzer bir teknik kullanılabiliyordu. Görüntüleri (ya da başka algısal girdileri) sınıflandırmak bakımdan, girdiyi, ayırt edici "özellikler" listesiyle göstermek olağan bir yaklaşımdı, örneğin Selfridge ile çalışma arkadaşlarının kullanmış olduğu liste gibi. Örneğin, abecesayısal karakter tanımada, önce sınıflandırılacak karakterin görüntüsünde belli özellikler bulunur. Çoğunlukla bu özelliklere sayısal değerler verilir, örneğin yazı karakterini kesen farklı açılara sahip çizgilerin sayısı ya da o karakteri bütünüyle çevreleyen en küçük çemberin çevre uzunluğu gibi. Uygun öznitelikleri seçmek çoğunlukla bilimden çok sanat kapsamına girer, fakat iyi performans bakımından çok önemlidir.

(Güz 1968'de düzenlenmiş Ortak Bilgisayar Konferansından), c. 33, Washington, DC: Thompson Book Co., 1968, s. 1125-1149. İlave bilgi için: SRI YZ Merkezi Teknik Raporları, http://www.ai.sri.com/pubs/files/1343.pdf ve http://www.ai.sri.com/pubs/files/1344.pdf.



Resim 4.11. Özellik noktalarının ve ayırıcı sınırın iki boyutlu uzayı.

İstatistik yönelimli bu örüntü tanıma yöntemlerini betimlemek için bir parça temel matematik gösterimine gereksinim duyarız. Yazı karakterinden çekip çıkarılan öznitelikler listesi $\{f_1, f_2, \ldots f_i, \ldots f_N\}$ olsun. Bu listeyi, kalın $\mathbf{X}$ simgesiyle kısaltacağım. $\mathbf{X}$'in betimlediği yazı karakterinin ait olabileceği k sayıda sınıf olsun: $C_1, C_2, \ldots C_i, \ldots C_N$. Bayes kuralını daha önce anlatıldığı şekilde kullanırsak, karar kuralı şöyle olur:

> $p(C_i)$, C_i kategorisinin *a priori* olabilirliğiyken ve belirli bir C_i için $p(\mathbf{X}|C_i)$, $\mathbf{X}$'in olabilirliğiyken, $p(\mathbf{X}|C_i)p(C_i)$'si en yüksek olan sınıfı seç. Bu olabilirlikler, büyük bir yazı karakteri örnek kümesinden toplanacak istatistiksel verilerden çıkarsanabilir.

Daha önce bahsetmiş olduğum gibi, örüntü tanıma alanındaki araştırmacılar, karar işlemini sık sık geometri bağlamında betimler. İmge örnek kümesinden elde edilen özellik değerlerinin, çok boyutlu bir uzayda nokta olarak gösterilebileceğini hayal ederler. Diyelim ki elimizde, bilinen iki veri kategorisi için birkaç örnek küme var, o zaman bu örnek kümelerini, uzamda oraya buraya saçılmış noktalar olarak temsil edebiliriz. Yazı karakteri tanıma işleminde, noktaların bu dağılımı, salt yazı karakteri imgesinin gürültülü oluşundan değil, ayrıca aynı sınıfa giren yazı karakterlerinin birbirinden farklı çizilmesinden ötürü de oluşabilir. Resim 4.11'de, f_1 ve f_2 özelliklerini taşıyan iki boyutlu bir örnek gösteriyorum. Her sınıftaki noktaların dağılımından yola çıkarak, olabilirlik hesaplamaları için gerekli olasılıklara dair

tahminler hesaplayabiliriz. Ardından, bu olabilirlikleri ve önceki olasılıkları kullanarak karar verebiliriz.

Bu olabilirlik ve önceki olasılıklardan yola çıkılarak hesaplanmış ve uzamı iki bölgeye ayıran sınırı bu resimde gösteriyorum. Bölgelerden birinde, 1 numaralı sınıf; öteki bölgede, 2 numaralı sınıf lehine karar veririz. Aynı zamanda, sınıflandırılması gereken yeni bir özellik noktası olan **X**'i de gösteriyorum. Bu örnekte **X**'in sınıra göre konumu, **X**'i 1 numaralı sınıfın üyesi olarak sınıflandırmayı getirir.

Özellik noktalarını sınıflandırmanın başka yöntemleri de var. "En yakın komşu yöntemi" ilginç örneklerden biridir. 1951'de E. Fix ile J. L. Hodges tarafından icat edilmiş bu yöntemde,[19] yeni bir özellik noktası, buna en yakın özellik noktasının ait olduğu sınıfa atanır. Resim 4.11'de yeni **X** noktası, en yakın komşu yöntemi kullanıldığında, 2 numaralı sınıfın üyesi olarak sınıflandırılır.

En yakın komşu yönteminin geliştirilmiş bir sürümü, yeni noktayı, k en yakın noktaların çoğunluğuyla aynı sınıfa atar. Böylesi bir karar kuralı inandırıcı görünür (her sınıf için çok ama çok örnek noktası olduğu durumlarda), çünkü C_i sınıfının, C_j sınıfının örnek noktalarına kıyasla bilinmeyen **X** noktasına yakın duran daha fazla sayıda örnek noktası oluşu, $p(\mathbf{X}|C_i)p(C_i)$'in $p(\mathbf{X}|C_j)p(C_j)$'den büyük olduğuna kanıttır. Bu genel gözlemin kapsamını genişleten Thomas Cover ve Peter Hart, en yakın komşu yöntemlerinin performansını titizlikle çözümlemişlerdir.[20]

Herhangi bir örüntü tanıma tekniğinin, hatta sinir ağları ya da en yakın komşu yönteminin, çok boyutlu bir özellikler uzamında ayırıcı sınırlar tanımladığı düşünülebilir. "Potansiyel fonksiyonlar" kullanarak sınır tesis etmenin başka bir yöntemini, 1960'larda Rus bilimci M. A. Aizerman, E. M. Braverman ve L. I. Rozonoer önermişti.[21]

19 E. Fix ve J. L. Hodges Jr., "Discriminatory Analysis, Nonparametric Discrimination", USAF School of Aviation Medicine, Randolph Field, Texas, Proje 21-49-004, Rapor 4, Contract AF41 (128)-31, Şubat 1951. Ayrıca bkz. B. V. Dasarathy (ed.), *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, Los Alamitos, CA: IEEE Computer Society Press; Fix ile Hodges'in yayımlanmamış 1951 tarihli çalışmalarının yeniden basımı.

20 Thomas M. Cover ve Peter E. Hart, "Nearest Neighbor Pattern Classification", *IEEE Transactions on Information Theory*, Ocak 1967, s. 21-27. Bkz. http://ieeexplore.ieee.org/iel5/18/22633/01053964.pdf.

21 Bkz. M. A. Aizerman, E. M. Braverman ve L. I. Rozonoer. "Theoritical Foundations of the Potential Function Method in Pattern Recogniton Learning", *Automation and Remote Control*, c. 25, 1964, s. 917-936 ve A. G. Arkadev ve E. M. Braverman, *Computers and Pattern Recognition*, (Rusçadan İngilizceye çevirenler W. Turski ve J.

Örüntü tanıma alanında istatistiksel yöntemlerin kullanımı hakkında ilk önemli kitaplardan bazılarını George Sebestyen,[22] bendeniz[23] ve Richard Duda ile Peter Hart[24] yazmıştır. Benim kitabım aynı zamanda, istatistiksel yöntemler ile sinir ağlarını temel alan yöntemler arasındaki ilişkilerden bazılarını betimliyor. George Nagy (eskiden Frank Rosenblatt'ın lisansüstü öğrencilerinden biriydi), 1960'ların ikinci yarısına ait örüntü tanıma teknolojisini hoş bir biçimde özetliyor.[25]

4.4 HAVADAN KEŞİF SEFERLERİNDE ÖRÜNTÜ TANIMA UYGULAMALARI

Sinir ağları ve örüntü tanımaya yönelik istatistiksel yöntemler, 1950'lerin ikinci yarısında ve 1960'ların başında pek çok havacılık ve hava elektroniği şirketinin ilgisini çekmiştir. Bu şirketlerin, ABD Savunma Bakanlığı'yla sözleşmelerinden kaynaklanan büyük araştırma-geliştirme bütçeleri vardı. Bu şirketlerin pek çoğu, özellikle havadan keşif sorunuyla, yani havadan çekilmiş fotoğraflarda "hedeflerin" konumunu belirleyip kimliğini saptamayla ilgileniyordu. Buna ve bununla ilişkili sorunlara adanmış kapsamlı araştırma programlarına sahip şirketler arasında, Ford Motor'un Aeronutronic Dairesi,[26] Douglas Uçak Şirketi (o zamanlar bilindiği adıyla), General Dynamics, Lockheed Füze ve Uzay Dairesi ve Philco Şirketi vardı. (Philco'yu 1961'in ikinci yarısında Ford devralacaktı.)

Bu araştırmalara örnek olarak, Philco'daki kimi çalışmalardan bahsedeceğim. Burada Laveen N. Kanal (1931-), Neil C. Randall (1930) ve Thomas Harley (1929-), istatistiksel örüntü tanıma yöntemlerinin hem kuramı hem de uygulamaları üzerinde çalışıyordu. Geliştirmiş

D. Cowan), Washington, DC: Thompson Book Co., Inc., 1967.

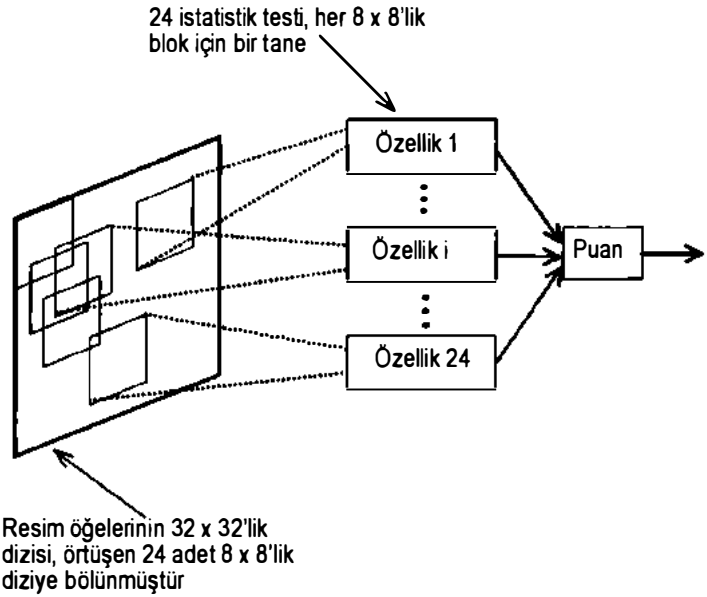
22 George S. Sebestyen, *Decision-Making Processes in Pattern Recognition*, Indiniapolis, IN: Macmillan Publishing Co., Inc., 1967.

23 Nils J. Nilsson, *a.g.e.*

24 Richard O. Duda ve Peter E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973; güncellenmiş versiyonu: Richard O. Duda, Peter Hart ve David G. Stork, *Pattern Classification*, 2. baskı, New York: John Wiley & Sons, 2000.

25 George Nagy, "State of the Art in Pattern Recognition", *Proceedigs of the IEEE*, c. 56, no: 5, Mayıs 1968, s. 836-857.

26 Örneğin bkz. Joseph K. Hawkins ve C. J. Munsey, "An Adaptive System with Direct Optical Input", *Proceedings of the IEEE*, c. 55, no: 6, Haziran 1967, s. 1084-1085. IEEE üyeleri için şu internet sitesine bakınız: http://ieeexplore.ieee.org/iel5/5/31078/01446273.pdf?tp=&arnumber=1446273&isnumber=31078.



Resim 4.12. Philco tank tanıma sistemi. (Uyarlandığı yer: Laveen N. Kanal ve Neal C. Randall, "Target Detection in Aerial Photography", makale 8.3, *Proceedings of the 1964 Western Electronics Show and Convention (WESCON)*, Los Angeles, CA, Institute of Radio Engineers (günümüzde IEEE), 25-28 Ağustos 1964.)

oldukları sistemler, havadan çekilmiş fotoğraflarda tank gibi askerî bakımdan ilginç hedefleri arıyordu. Bunların sistemlerinden birinin şematik çizimi Resim 4.12'de gösterilmiştir.[27]

Philco'nun aygıtı, New York Fort Drum'daki ABD Kara Kuvvetleri tank manevraları sırasında U2 keşif uçağının çekmiş olduğu filmlerin negatiflerini taramıştı. Muhtemelen bir M-48 tankını (standart konum ve boyutunda) içeren, taranmış fotoğrafın küçük bir kısmı, önce kenar çizgilerini vurgulamak üzere işleniyor, sonuçta hedef tespit sistemine 1 ve 0'lar dizisi olarak sunuluyordu. İlk sistemleri 22×12'lik bir dizi kullanırken, daha sonraki sistemler, Resim 4.12'de gösterildiği gibi 32×32'lik diziler kullanmıştır. Ardından dizi, birbirleriyle örtüşen 8×8'lik "özellik blokları" halinde ayrılmıştı. Her özellik

27 Laveen N. Kanal ve Neal C. Randall, "Target Detection in Aerial Photography", makale 8.3, *Proceedings of the 1964 Western Electronics Show and Convention (WESCON)*, Los Angeles, CA, Institute of Radio Engineers (günümüzde IEEE), 25-28 Ağustos 1964. (Bu konferansta, örüntü tanımayla ilgi başka makaleler de sunulmuş ve konferans raporuna konmuştur.)

bloğundaki verilere, o bloğun temsil ettiği küçük resim parseli tankın bir kısmını içeriyor mu içermiyor mu karar verilebilsin diye bir istatistiksel test uygulanmıştı.

İstatistiksel testler, tank içeren elli görüntü ile tank içermeyen elli arazi fotoğrafından oluşan "eğitim örnek kümesi"ni temel alıyordu. 8×8'lik her özellik bloğu için, bu örnek kümelerinden istatistiksel parametreler derlenmişti, böylece tanklı örnekleri tanksız örneklerden en iyi şekilde ayıracak altmış-dört boyutlu uzamda (doğrusal) bir sınır belirleniyordu.

Bu sınırlar kullanılarak, bu sistem, tank içeren elli farklı görüntüyle ve tank içermeyen elli görüntüyle test edildi. Her test görüntüsü için, "tankın var olduğuna" karar veren özellik bloklarının sayısının çetelesi tutuldu, böylece nihai bir sayısal "puan" elde edildi (örneğin 24 bloktan 21'inde tank olduğuna karar verilmesi gibi). Ardından bu puan, görüntüde tankın olup olmadığını belirlemek için kullanılabilirdi.

Bu testleri uygulayanlar, "istatistiksel sınıflandırma yönteminin deneylerdeki performansı tüm beklentileri aştı" demiştir. Test örneklerinin neredeyse yarısı mükemmel puanlar almıştı (yani tüm 24 özellik bloğu, tankın varlığı ile yokluğu arasında doğru ayrım yapabilmiştir). Dahası, tank içeren tüm test örneklerinin puanı 11'e eşit ya da büyüktü; tank içermeyen test örneklerinin puanı ise 7'ye eşit ya da küçüktü.

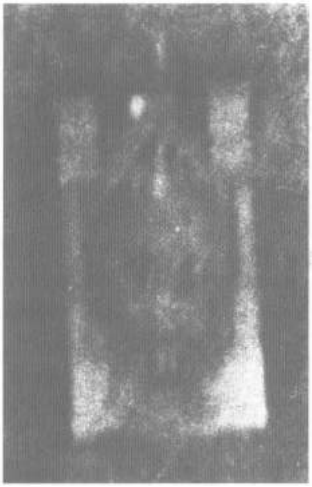
Philco'da ilk tank tespit sistemlerinden biri analog devrelerle kurulmuştu, yani bilgisayarda programlanmamıştı. Bu sistemin proje başkanı Thomas Harley'in ileride diyeceği gibi:[28]

> Bu çalışmanın gerçekleştirildiği dönemin teknoloji bağlamını aklımızdan çıkarmayalım. Hayata geçirdiğimiz sistemin kendi bünyesinde bilişim becerileri yoktu. Ayırt edici doğrusal fonksiyondaki ağırlıklar, kaydırma yazmacı öğelerindeki (ikili) voltaj kaynağından gelen akımı denetleyen direnç birimleriydi. O akımlar birbirleriyle toplanıyordu ve her özellik, bu akımların toplamının belli bir eşik değerini aşmasına göre tanınıyor ya da tanınmıyordu. O ikili özellik kararları, bilgisayarda değil yine analog bir elektrik devresinde toplanıyordu, ve yine toplamın belli bir eşik değerini aşıp aşmamasına göre karar [tank var ya da yok] alınıyordu.

Başka bir sistemde istatistiksel sınıflandırma, Philco 2000 bilgisayarında çalışan MULTINORM adlı bir program tarafından gerçekleşmişti.[29] Başka deneylerde Philco, sonuç puanı hesaplamada belirli özellik

28 Thomas Harley, e-posta üzerinden kişisel iletişim, 15 Temmuz 2007.

29 Laveen N. Kanal ve Neal C. Randall, *a.g.e.*



Resim 4.13. Tipik tank görüntüsü. (Fotoğraf, Thomas Harley'in izniyle.)

bloklarına öteki testlere kıyasla daha fazla ağırlık veren ek istatistiksel testler kullanmıştı. Kanal, özellik bloklarının çıktısına ağırlık vermekle ilgli bu deneyler hakkında bana şunu söylemişti: "destek vektör makinesi (DVM) sınıflandırma fikrinin öncüsüydü bu... ilk katman, tankın varlığı ile yokluğu arasındaki sınıra yakın duran eğitim örneklerini saptamak amacıyla kullanılıyordu."[30] (Önemli DVM yöntemini daha sonraki bir bölümde betimliyorum.)

Elbette bu sistemlerin işi görece basitti. Tüm tanklar standart konumdaydı ve fotoğrafta zaten yalıtılmışlardı. (Fakat sistemin yaratıcıları, görüntüde herhangi bir konumda ya da açıda görünen tanklarla başa çıkmak üzere sistemin nasıl uyarlanabileceğinden bahsederler.) Resim 4.13'teki fotoğraf, tipik bir tank görüntüsü gösteriyor. (Tankın olmadığı görüntüler buna benzer, ama onlarda tank yok.)

Bu sistemi ilginç buluşumun sebebi, salt performansı değil, aynı zamanda katmanlı bir sistem oluşudur (bu bakımdan Pandemonium'a ve alfa-perceptrona benzer), ayrıca özgün görüntünün, örtüşen alt-görüntülere bölündüğü ve her birinin ayrı ayrı işlemden geçirildiği sistemlere örnektir. İleride bahsedeceğim üzere, örtüşen alt-görüntüler, kimi hesaplamalı neokorteks modellerinde belirgin rol oynar.

Ne yazık ki, bu çalışmanın ayrıntılarını sunan Philco raporlarına ulaşmak kolay değil.[31] Dahası, Philco ve bu çalışmayla meşgul olmuş

30 Laveen Kanal, e-posta üzerinden kişisel iletişim, 13 Temmuz 2007.

31 Laveen N. Kanal, "Statistical Methods for Pattern Classification", Philco Raporu, 1963; aslı şurada: T. Harley ve diğ., "Semi-Automatic Imagery Screening Research

kimi ekipler ortadan kaybolmuştur. Tom Harley, Philco raporları ve bizzat Philco hakkında bana şunu yazmıştı:[32]

> 1960'larda Philco'daki örüntü tanıma çalışmalarının büyük kısmına SB [Savunma Bakanlığı] hamilik etmişti; dolayısıyla bu raporlar kamuya açık değildi. O zamandan bu yana şirketin kendisi buharlaştı. 1961'de Philco'yu Ford Motor satın aldı ve 1966'ya gelindiğinde, Laveen [Kanal] ile benim çalıştığımız Philco araştırma laboratuvarlarını kaldırmışlardı. Ford, küçük örüntü tanıma ekibimizi Newport Beach, California'ya [Ford Aeronutronic Dairesinin yeri; ileride bunların da örüntü tanıma ekibi kapatılacaktı] taşımaya çalıştı ama bizler gitmemeye karar verdiğimizde bizi İletişim Dairesine naklettiler ve örüntü tanıma projelerimizi kapatmamızı söylediler. Sonunda Laveen ayrılıp Maryland Üniversitesi'ne geçti ve 1975'te beni Ford Aerospace Western Development Labs'e (WDL) naklettiler; orada istihbarat teşkilatı için büyük sistemler üzerinde çalıştım. İleriki yıllarda, eskiden Philco olarak anılan şube Loral'a satıldı ve bunun da büyük kısmı daha sonra Lockheed Martin'e satıldı. Lockheed'ten 2001'de emekli oldum.

YZ problemlerini çözmede sinir ağlarını ve istatistik tekniklerini içeren yaklaşımlar, teorem kanıtlamakla, oyun oynamakla ve problem çözmekle ilgilenenlerin "simge işleyen" çalışmalarının aksine "simgesel olmayan" yaklaşımlar olarak nitelenmiştir. Simgesel olmayan bu yaklaşımlar çoğunlukla örüntü tanıma, konuşma işleme ve bilgisayarla görme (*computer vision*) alanlarında uygulama imkânı bulmuştur. Özellikle bu konu başlıklarına adanmış çalıştaylar ve konferanslar 1960'larda tertiplenmeye başlandı. IEEE Bilgisayar Cemiyeti'nin bir altekibi (Veri Algılama ve Dönüştürme Komitesi'nin Örüntü Tanıma Altkomitesi), ilk "Örüntü Tanıma Çalıştayı"nı düzenlemişti; bu çalıştay, Ekim 1966'da Porto Riko'da toplanmıştır.[33] İkinci bir çalıştay (ona ben de katılmıştım) Ağustos 1968'de Hollanda'nın Delft şehrinde düzenlendi. 1966'da bu altekip, IEEE Bilgisayar Derneği Örüntü Çözümleme ve Makine Zekâsı (PAMI) Teknik Komitesi adını alıp konferanslar ve çalıştaylar düzenlemeyi sürdürdü.[34]

Study and Experimental Investigation", Philco Raporları VO43-2 ve VO-43-3, c. I, Kısım 6, Ek H, ABD Kara Kuvvetleri Elektronik AR-GE Dairesi için hazırlanmıştır.

32 Thomas Harley, e-posta üzerinden kişisel iletişim, 11 Temmuz 2007.

33 Laveen N. Kanal (ed.), *Pattern Recognition, Proceedings of the IEEE Workshop on Pattern Recognition*, Porto Riko'nun Dorado kentinde düzenlenmiş, Washington DC: Thompson Book Co., 1968.

34 Bkz. http://tab.computer.org/pamitc/.

Bu esnada, 1950'lerin ikinci yarısı ve 1960'ların ilk yarısı boyunca simge işleme konusuyla ilgilenenler genelde MIT, Carnegie Mellon Üniversitesi, IBM ve Stanford Üniversitesi'nde çalışmıştı. Şimdi de bu insanların yapmış olduğu kimi işleri anlatacağım.

5
İLK SEZGİSEL PROGRAMLAR

5.1 MANTIK KURAMCISI VE SEZGİSEL ARAMA

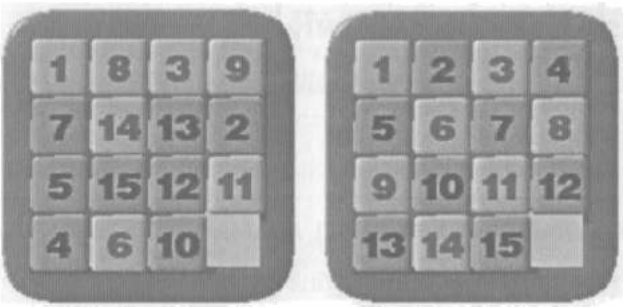
Dartmouth çalıştayının hemen öncesinde Newell, Shaw ve Simon, RAND şirketindeki JOHNNIAC (John von Neumann'ın şerefine verilmişti bu isim) adlı bilgisayarda bir LT sürümünü programlamışlardı. İleride yazılacak makaleler,[1] Russell ile Whitehead'in klasik eserleri *Principia Mathematica*'nın[2] [Matematik İlkeleri] birinci cildinde kanıtlamış oldukları kimi teoremlerin simgesel mantıkla nasıl ispatlandığını betimleyecekti. LT, Russell ile Whitehead'in önerme mantığının beş aksiyomunun kanıtlanacak teoreme uyan bir yapı üretilene dek dönüşümlerini gerçekleştirerek çalışıyordu; bunlar, bilgisayar için "simge yapıları" tarafından temsil ediliyordu. Farklı farklı pek çok dönüşüm mümkün olduğu için, teoremi kanıtlama yolunda uygun olanları bulmak, bilgisayar bilimlerinde "arama süreci" denilen işlemi içerir.

LT'nin ve başka simgesel YZ programlarının nasıl işlediğini betimlemek için, öncelikle "simge yapısıyla" ve bunları "dönüştürmekle" kastedilen şeyi açıklamam şart. Bilgisayarda simgeler, listeler halinde birleştirilebilir, örneğin (A, 7, Q) gibi. Simgeler ile simge listeleri, en basit simge yapısı türüdür. ((B, 3) (A, 7, Q)) gibi simge listelerinin listelerinden ve simge listelerinin listelerinin listelerinden, vesaire oluşan yapılar daha karmaşıktır. Bu tür liste listeleri oldukça karmaşık olabileceği için, bunlara "yapı" denir. Simge yapılarını başka simge yapılarına dönüştüren bilgisayar programları yazılabilir. Örneğin, uygun program sayesinde, "(yedi ve beşin toplamı)" yapısı, "(7+5)" yapısına ve bu da "12" simgesine dönüştürülebilir.

Simge yapılarını dönüştürmek ve sorun çözücü, uygun dönüşüm dizisini aramak, Newell ile Simon'ın zekâyı mekanikleştirme fikirle-

1 A. Newell ve H. A. Simon, "The Logic Theory Machine: A Complex Information Processing System", *Proceedings IRE Transactions on Information Theory*, c. IT-2, s. 61-79, Eylül, 1956 ve A. Newell, J. C. Shaw ve H. A. Simon, "Empirical Explorations of the Logic Theory Machine: A Case Study in Heuristics", *Proceedings of the 1957 Western Joint Computer Conference*, Institute of Radio Engineers, 1957, s. 218-230.

2 Alfred North Whitehead ve Bertrand Russell, *Principia Mathematica*, c. 1, Cambridge: Cambridge University Press, 1910.



Resim 5.1. On beşli bulmaca probleminin başlangıç (solda) ve hedef (sağda) dizilişleri.

rinin tam göbeğinde bulunuyordu. İleride bir makalede (itibarlı Turing Ödülü'nü almaları üzerine sundukları makalede), süreci şöyle özetlemişlerdir:[3]

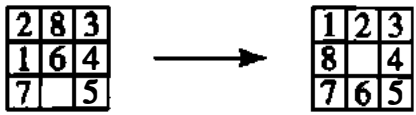
> Sorunların çözümleri, simge yapıları olarak temsil edilmiştir. Fiziksel bir simge sistemi, sorun çözmedeki zekâsını, arama süreciyle gösterir; yani, bir çözüm yapısı oluşturana dek simge yapıları yaratıp bunları kademe kademe değiştirir.
>
> ...
>
> Bir sorunu ifade etmek: (1) bir simge yapısı sınıfı için test belirlemek (sorunun çözümleri) ve (2) simge yapıları üretmek (olası çözümler) demektir. Sorun çözmek ise, (1)'in testini başarıyla veren (2)'yi kullanarak bir yapı üretmek demektir.

MK'nin, teorem kanıtlamak için simge yapılarını ve dönüşümlerini bizzat nasıl kullandığını ayrıntılarıyla anlamak, azıcık matematik ve mantık altyapısı gerektirir. YZ'nin gözde "oyuncak problemlerinden" birinin yardımıyla bu süreci açıklamak daha kolay: "on beşli bulmaca" (bkz. Resim 5.1). On beşli bulmaca, kaydırmalı blok bulmaca tiplerinden biridir. Çözülmesi istenen sorun, başlangıçtaki kare dizilişini, karelerin komşu boş yuvaya ardı ardına itilmesiyle, "hedef" dizilişe dönüştürmektir.

Bulmacanın daha basit bir versiyonundan faydalanacağım; 4 × 4 diziliş yerine sekiz kareli 3 × 3 diziliş. (YZ araştırmacıları, bulmacanın 5 × 5 ve 6 × 6 gibi daha büyük hallerini çözmeyi hedefleyen programlarla deneyler yapmıştır.)

Diyelim ki kareleri Resim 5.2'de sol tarafta gösterilen dizilişten sağdaki dizilişe aktarmak istiyoruz.

3 Allen Newell ve Herbert A. Simon, "Computer Science as Empirical Inquiry: Symbols and Research", *Communications of the ACM*, c. 19, no: 3, Mart 1976, s. 113-126.



Resim 5.2. Sekizli bulmaca.

Newell ile Simon'un yaklaşımı uyarınca, önce karelerin konumunu bilgisayar için simge yapılarıyla temsil etmeliyiz ki bilgisayar bunları işleyebilsin. Başlangıç konumunu aşağıdaki gibi sunacağım; bu, üç altlisteden oluşan bir liste:

((2, 8, 3) (1, 6, 4) (7, B, 5)).

İlk altliste, yani (2, 8, 3), bulmaca dizilişinin ilk sırasındaki rakamları gösteriyor vb. B, üçüncü sıranın ortasındaki boş hücreyi simgeliyor.

Aynı tarzla, hedef diziliş şu şekilde temsil edilir:

((1, 2, 3) (8, B, 4) (7, 6, 5)).

Ardından, bilgisayarın, bulmacada izin verilen hamlelere denk gelecek biçimde kurduğumuz yapıları nasıl dönüştürebileceğimizi göstermek zorundayız. Unutmayalım ki bir kare hareket ettirildiğinde, boş hücreyle yer değiştirir, yani boş hücre de hareket eder. Boş hücre, ya sıra içinde yer değiştirir ya da sırasını değiştirir.

Boş hücrenin bu hareketleriyle ilişkili olarak, bir kare kendi sırası içinde yer değiştirdiğinde, B, kendi listesinde aynı sayıda sola (eğer yer varsa) ya da sağa (eğer yer varsa) kayar. Bilgisayarlar, bu iki dönüşümden birini kolaylıkla başarabilir. Boş hücre aşağı ya da yukarı hareket ettiğinde, B, listede denk gelen rakamla sola (eğer yer varsa) ya da sağa (eğer yer varsa) kayar. Bu dönüşümler bir bilgisayar programıyla kolayca yapılabilir.

Newell ve Simon yaklaşımından faydalanarak sekizli bulmacanın başlangıç dizilişini temsil eden ve hedefe ulaşılana dek geçerli dönüşümleri uygulayan simge yapısıyla yola çıkarız. Başlangıç simge yapısının üç dönüşümü var. Bu dönüşümler, şu yapıları üretir:

((2, 8, 3) (1, 6, 4) (B, 7, 5)),
((2, 8, 3) (1, 6, 4) (7, 5, B)),

ve

((2, 8, 3) (1, B, 4) (7, 6, 5)).

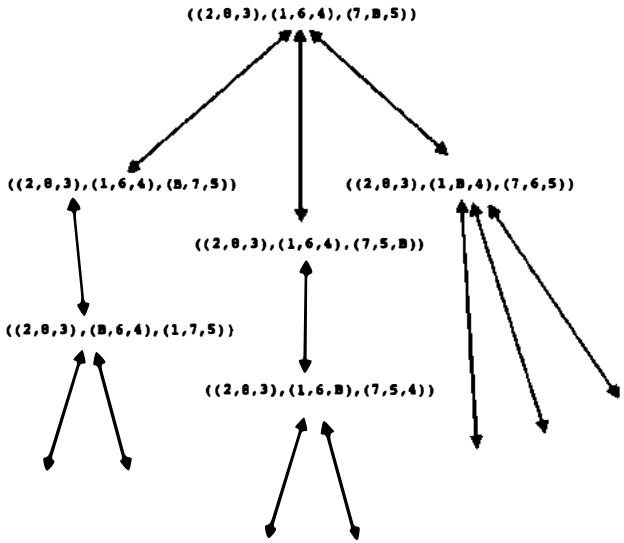
Bunların hiçbiri hedef dizilişi temsil etmiyor, dolayısıyla bunların her birine dönüşüm uygulamayı sürdürürüz ve böylece hedefi temsil eden bir yapıya ulaşılır. Bizler (ve bilgisayar), Resim 5.3'te gösterilene benzer şekilde ağaç benzeri yapı halinde düzenlenmiş dönüşümlerin izleyebiliriz. (Dönüşümleri temsil eden çizgilerin iki ucundaki oklar, her dönüşümün tersine çevrilebilir olduğunu gösteriyor.)

Sekizli bulmacanın bu hali görece basittir, dolayısıyla hedefe gidilirken çok fazla dönüşüm denenmez. Gerçi tipik olarak (özellikle de bulmacanın daha büyük versiyonlarında), bilgisayar tüm olası dönüşümlerle dolup taşar; öyle ki, asla bir hedef ifadesi yaratamaz. İleride dönüşümlerin "birleşimsel patlaması" denecek şeyi kısıtlamak maksadıyla Newell ile Simon, sadece çözüme açılan yolda olması muhtemel dönüşümleri yaratmak için "sezgisel yöntem"den (*heuristics*) faydalanma fikrini ileri sürdüler.

LT'yle ilgili makalelerinden birinde şöyle yazmışlardır: "Bir sorunu *belki* çözecek ama bunun için güvence vermeyen sürece, o problemin sezgisel yöntemi denir". LT, kanıt arayışında her doğrultuyu körlemesine zorlamak yerine, sezgisel yöntemin ya da "sezgisel aramanın" kılavuzluk ettiği bir arama sürecinden faydalanıyordu. LT örneğinin gösterdiği gibi, sezgisel aramanın başarılı olacağının garantisi yoktur; fakat başarılı olduğunda (ki sık sık başarılı olur), normalde verimsiz olacak arama çabalarını elemiş olur.

Sekizli bulmaca sorununa çözüm arayışı, simge yapısı ağacının "yapraklarına" dönüşümler uygulayıp böylece uzamalarını sağlayarak bu ağacı büyütmeyi içerir. Ağacın büyümesini sınırlamak amacıyla, yalnızca çözüm yolunda olduğu düşünülen yapraklara dönüşüm uygularken sezgisel yöntem kullanmalıyız. Bu tür bir sezgi, hedef diziliş bakımından yanlış yerde olan en az sayıda kareyi barındıran yaprağı seçmektir. Kare kaydırma problemi adamakıllı incelendiği için, faydalı olduğu ispatlanmış pek çok sezgisel yöntem vardır; henüz bahsettiğim o basit, hedef konum dışındaki karelerin sayısı yaklaşımından daha iyileri vardır.

Çözülecek problemle ilgili sezgisel yöntem kullanmak, yapay zekâda başlıca konu haline gelmiş, "sezgisel programlama" alanını doğurmuştur. Belki de sezgisel arama fikri Dartmouth çalıştayı zamanında zaten "tedavüldeydi". Claude Shannon'un eski çalışmalarından birinde örtük halde bulunur. Hevesli bir satranç oyuncusu olan Shannon, 1950 yılının Mart ayında, satranç oynayan bilgisayar programlamayla ilgili



Resim 5.3. Arama ağacı.

fikirler ileri sürdüğü bir makale yayımladı.[4] Bu makalesinde Shannon, "A tipi" ve "B tipi" stratejiler olarak belirlediği ayrımdan bahseder. A tipi stratejiler, olası her hamle bileşimini incelerken, B tipi stratejiler en verimli olduğu düşünülen oyun gidişatına odaklanmak amacıyla uzmanlaşmış satranç bilgisinden faydalanır. B tipi stratejiler, ileride Newell ile Simon'ın sezgisel yöntem diyeceği yönteme bel bağlamıştı. Dahası Minky şunu söylemiştir: "Sezgisel yöntem fikrini zaten aşikâr ve doğal görüyordum, dolayısıyla Mantık Kuramcısı bana etkileyici gelmedi."[5]

Problemin kurulma şeklinin ve "gösteriminin", çözüm için kilit önem taşıdığı, YZ alanında oldukça erken fark edilmiştir. Gösterimin, sorun çözümünü etkileme şeklinin örneklerinden biri John McCarthy'den geliyor ve buna "eksik dama tahtası" problemi denir.[6]

4 Claude E. Shannon, "Programming a Computer for Playing Chess", *Philosophical Magazine*, seri 7, c. 41, no: 314, Mart 1950. Bkz. http://www.pi.infn.it/~carosi/chess/shannon.txt. (Bu makale ilk kez, Mart 1950'de New York'ta Ulusal Radyo Mühendisleri Kongresi Enstitüsü'nde sunulmuştur.)

5 Pamela McCorduck, *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*, s. 106, San Francisco: W. H. Freeman and Co., 1979.

6 John McCarthy, "A Tough Nut for Theorem Provers", Stanford Artificial Intelligence Project Memo no: 16, 17 Temmuz 1964. Şu internet sitesinde: http://www-formal.

İşte problem şöyle: "Dama tahtasının zıt çapraz köşelerindeki kareler çıkarılıyor diyelim. Kalan kareleri domino taşlarıyla kaplamak mümkün müdür?" (Domino taşı, iki komşu kareyi kaplayan dikdörtgen bir parçadır.) Çözüm aramanın saf yollarından biri, domino taşlarını dama tahtasına olası her şekilde yerleştirmek olacaktır. Fakat, dama tahtasının bir renkten 32 ve öbür renkten 32 karesi olduğu ve zıt köşelerdeki karelerin renginin aynı olduğu bilgisi göz önüne alındığında, eksik dama tahtasında bir renkten 30, öbür renkten 32 kare olduğu fark edilir. Domino taşı, zıt renkli iki kareyi kapladığı için, geride kalan kareleri domino taşıyla kapatmanın hiç yolu yoktur. McCarthy, acaba insanlar bu bulmacayı "yaratıcı" biçimlerde tanımlayıp, mantıksal çıkarıma dayalı yöntemler kullanan bilgisayarların bulmacayı çözmesini sağlayabilir mi meselesiyle ilgileniyordu.

Farklı gösterimlerin etkilerini araştırmak amacıyla kullanılmış başka bir klasik örnek ise, "misyoner ve yamyam" problemidir: Üç yamyam ve üç misyoner nehrin öte yanına geçmek zorunda. Kayık sadece iki kişi alabiliyor. Nehrin iki yakasının birinde yamyam sayısı misyonerlerin sayısını aşarsa, o yakadaki misyonerler telef olur. Her misyoner ve her yamyam kayıkta kürek çekebiliyor. Altı kişi de nehrin karşı yakasına güvenle nasıl geçebilir? Çoğu insan, bu bulmacayı bir arama sorunu olarak formüle etmekte zorlanmaz ve çözüm görece kolaydır; fakat sezgilere görece aykırı bir adım gerektirir. Bilgisayar bilimcisi ve YZ araştırmacısı Saul Amarel (1928-2002), bu bulmacayı ve misyoner ile yamyam sayısının değişkenlik gösterdiği çeşitli karmaşık hallerini çözümleyen, epey atıfta bulunulmuş bir makale yazmıştır.[7] (Karmaşık halleri o kadar kolay görünmez.) Bir gösterimden ötekine geçen Amarel, en sonunda problemin genel hali için, çözümü neredeyse hiç arama süreci gerektirmeyen bir gösterim geliştirmiştir. YZ araştırmacıları, problemleri en iyi şekilde nasıl temsil edeceklerini ve en önemlisi, kendi gösterimlerini oluşturan YZ sistemlerini nasıl elde edeceklerini hâlâ araştırıyorlar.

5.2 GEOMETRİ TEOREMLERİNİ KANITLAMAK

Nathan Rochester, Marvin Minsky'le, Minsky'nin geometri teoremlerini ispatlayabilecek bir bilgisayar programı hakkındaki fikirleriyle ilgili

stanford.edu/jmc/thoughnut.pdf.

7 Saul Amarel, "On Representations of Problems of Reasoning About Actions", *Machine Intelligence 3* içinde, Donald Michie (ed.), Edinburgh: Edinburgh University Press, 1968, s. 131-171.

yaptığı tartışmalardan heyecanlanmış olarak Dartmouth çalıştayından IBM'e döndü. Bu fikirleri, IBM'in yeni çalışanlarından Herb Gelernter'e (1929-2015) anlattı. Gelernter, kısa süre içinde, geometri teoremi kanıtlayan makine geliştirmek amacıyla bir araştırma projesi başlattı. Haziran 1959'da Paris'teki bir konferansta,[8] programının ilk sürümüyle ilgili bir makale sundu; bu makalede şunu kabul ediyordu:

> bizzat bu araştırma projesi, 1956'da düzenlenmiş olan Yapay Zekâ konulu Dartmouth Yaz Araştırma Projesinin bir sonucudur; bu proje sırasında M. L. Minsky, geometri teoremi kanıtlayan makinenin olası faydasına işaret etmişti.

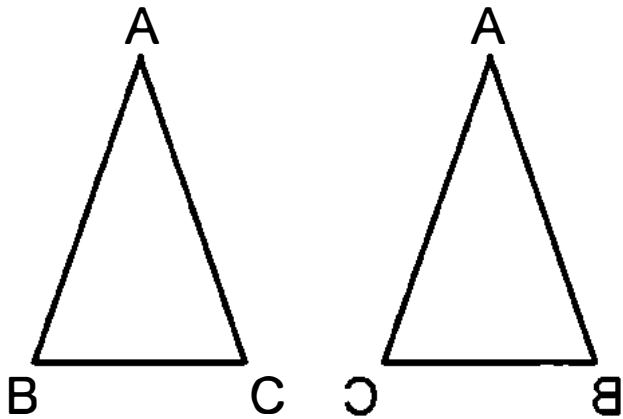
Gelernter'in programı, iki önemli fikre yaslanıyordu. Bir tanesi, althedeflerin (zaman zaman "geriye dönük uslamlama" ya da "böl ve yönet" denir) açıkça kullanımıydı, öbürüyse gereksiz arama yollarını tıkayan bir diyagramdan faydalanmaktı.

Lisede öğretilen geometri teoremi kanıtlama stratejisi, birtakım yan geometrik olgular bulmayı içerir, zira bunlar doğruysa, teoremin de doğruluğu hemen görülür. Örneğin, iki açının eşit olduğunu kanıtlamak için, bu açıların, iki "eş" üçgenin karşılık gelen açıları olduğunu göstermek yeterlidir. (Eğer bir üçgen, başka bir üçgene tamı tamına uyacağı şekilde dönüştürülebiliyor, ekseni etrafında çevrilebiliyor, hatta tersyüz edilebiliyorsa, o iki üçgen eştir.) O halde özgün problem, iki üçgenin eş olduğunu gösterme problemine dönüştürülebilir. İki üçgenin eş olduğunu göstermenin yollardan biri (başka yolların yanı sıra), iki üçgenin karşılık gelen kenarlarının ve kapalı açılarının aynı büyüklükte olduğunu göstermektir. İspatlanması gereken şey, teoremin önermelerinden biri olduğunda, bu geriye dönük uslamlama süreci sona erer.

Geometriye aşina okurlar, Resim 5.4'te gösterilen açıklayıcı örneği anlayacaktır. Bu resmin sol tarafında, AB kenarının AC kenarına eşit olduğu bir ABC üçgeni görüyoruz ve ABC açısının ACB açısına eşit olduğunu kanıtlamamız isteniyor. Sağ taraftaki üçgen, ACB üçgeninin ters çevrilmiş halidir.

İşte kanıt: Eğer ABC üçgeninin, ACB üçgenine eş olduğunu kanıtlayabilirsek, teorem ispatlanmış olur, çünkü bu iki açı, iki üçgen-

8 Herbert Gelernter, "Realization of a Geometry-Theorem Proving Machine", *Proceedings of the International Conference on Information Processing*", s. 273-282, Paris: UNESCO House, Münih: R. Oldenbourg ve Londra: Butterworths, 1960. Ayrıca; Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw Hill, 1963, s. 134-152.



Resim 5.4. İki kenarı eşit bir üçgen (solda) ve ters çevrilmiş hali (sağda).

de birbirinin muadili açılardır. Eğer, (ABC üçgeninde) AB kenarının AC kenarına (AƆƐ üçgeninde), AC kenarının (ABC üçgeninde) CB kenarına (AƆƐ üçgeninde) ve A açısının (ABC üçgeninde) A açısına (AƆƐ üçgeninde) eşit olduğunu belirleyebilirsek, bu iki üçgenin eş olduğu ispatlanır. Ancak, önermelere göre AB kenarı AC kenarına eşittir ve bu uzunluklar, terse çevrilmiş üçgende değişmiyor. Benzer biçimde, A açısı da terse çevrilmiş üçgendeki haline eşittir; işte kanıtımız bu.

Gelernter'in programını anlatmayı sürdüreceğim ama yeri gelmişken biraz tarihten bahsedelim. Yukarıda kanıtlamış olduğumuz geometri teoremi ünlüdür; Öklid'in *Elementler* kitabının birinci cildindeki beşinci önermedir bu. Öklid'in bu önermeye ilişkin sunduğu kanıt, işin acemileri için zor bir problem olduğundan, *pons asinorum* yani "aptallar köprüsü" olarak bilinir. Elinizdeki kitapta anlattığımız ispat, Öklid'in ispatından basittir ve bir şekli İskendireyeli Pappus (y. MÖ 290-350) tarafından verilmiştir.

Minsky'nin, geometri teoremleri ispatlayan programa dair "elle yaptığı benzetim", ki Dartmouth'da bundan bahsetmişti, tam da bu kanıtı bulmuştur (faydalı olduğunu sandığım, üçgeni ters çevirme adımını göz ardı edersek). Minsky şöyle yazar:[9]

> 1956'da, geometri teoremleri ispatlayan programlar için benzetimi elle yapılmış bir program hakkında iki bilgilendirme notu yazmıştım. İlk notta, bu yöntem, eğer bir üçgenin iki kenarı eşitse, üçüncü kenarla yaptıkları açılar da eşittir teoreminin basit bir kanıtını gösteriyordu. Bunu,

9 Bkz. http://www.math.niu.edu/~rusin/known-math/99/minsky.

> ABC üçgeninin, "kenar-açı-kenar"dan ötürü CBA üçgenine eş oluşunu vurgulayarak yapmıştı. İlginç olan nokta, bu yöntemin, kısa bir aramadan sonra bulunmuş olmasıydı; nasılsa deneyecek çok seçenek yoktu. Bu programın sıradan bir insanın yaptığı şeyi yapamayacak kadar aptal olduğunu söyleyebilirsiniz, yani "Bunların ikisi de aynı üçgen. Bunlara iki farklı isim vermekten bir hayır gelmez" diye düşünemez program. (Programda, Öklidvari teoremleri kanıtlamaya yönelik bir sezgisel yöntemler derlemesi var ve bir tanesi şöyle: "Eğer iki üçgenin eşit olduğunu ispatlamak istiyorsan, eş üçgenlerin karşılık gelen parçaları olduklarını göster bunların". Ayrıca, eşliği göstermek için de birkaç yöntemi vardı. Bu ilk benzetimin içerdikleri aşağı yukarı bu kadardı.) Ama o bilgilendirme notunu hiçbir yerde bulamıyorum.

Minsky'nin dediği gibi, bilgisayar için çok kolay bir problemdir bu. Gelernter'in programı, çok daha zorlu teoremleri kanıtlamıştır ve bu iş için diyagram kullanması yaşamsal önem taşıyordu. Program, gerçekten diyagram çizip buna bakıyor değildi. Bunun yerine, Gelernter'in anlattığı gibi:

> [Programa], o teoremde diyagram, belirtilen noktaların olası koordinatlarının bir listesi olarak veriliyordu. Bu nokta listesine, bölütlerle uç uca eklenen noktaların belirtildiği başka bir liste eşlik ediyordu. Koordinatlar ise, şekillerdeki olası en büyük genelliği yansıtacak şekilde seçiliyordu.

O halde, örneğin, iki açının eşit olduğunu kanıtlamakla ilgili problemde belirtilen noktalar, ABC üçgeninin köşeleridir, yani A, B, C noktalarıdır. Bu noktaların her biri için koordinatlar seçilir ve bu koordinatların, hiçbir özel ad verilmemiş niteliği karşılamaması için özen gösterilir.

Gelernter'in programı, althedefler belirleyerek çalışıyordu ve althedefler, yukarıda verdiğim örnekteki althedeflere benziyordu. Ardından, doğrudan önermeler sayesinde belirlenen althedeflerde, bu uçların bir zincirini arıyordu. Bununla birlikte, üzerinde çalışılacak herhangi bir althedef, program tarafından seçilmeden önce, acaba diyagramda ifade ediliyor mu diye sınanıyordu. Eğer ifade ediliyorsa, belki ispatlanabilir ve dolayısıyla kanıtın olası rotalarından biri sayılabilir; fakat, diyagramda ifade edilemiyorsa, olasılıkla geçerli olamaz. Dolayısıyla, artık göz önüne almaya gerek kalmaz ve böylece arama ağacı "budanmış" olup sonuçsuz uğraşlardan kurtulunur. İlerideki YZ çalışmaları, bu nevi "anlamsal" bilgiden de faydalanacaktı.

Geometri programlarında kullanılan stratejiler ile problem çözerken yararlandığımız stratejiler arasında benzerlikler görebiliriz. Geriye dönük çalışmak bizde yaygındır; zorlu bir problemi altproblemlere dö-

nüştürürüz, sonra bunları da alt-altproblemlere dönüştürürüz, ta ki problemler çok çok kolaylaşana dek. Bir altproblemin pek çok parçası olduğunda, hepsini çözmemiz gerektiğini biliriz. Önümüze sürülen bir altproblemin çözümü imkânsız olduğunda bunu fark edip o problemi reddederiz. Şimdi betimleyeceğim program, yaratıcılarının özellikle insanlara ait problem çözme stratejileri olarak düşündüğü şeyi temel alıyordu.

5.3 GENEL PROBLEM ÇÖZÜCÜ

Gelernter'in programını sunduğu 1959 Paris Konferansında, Allen Newell, J. C. Shaw ve Herb Simon, problem çözümünü mekanikleştirmekle ilgili kendi makalelerini sunmuşlardı.[10] "Genel Problem Çözücü" (GPÇ) dedikleri programları, insanların problem çözme yöntemleri hakkındaki düşüncelerinin vücut bulmuş haliydi. Aslında, bizzat programın, insanda problem çözme davranışının kuramı olduğunu ileri sürmüşlerdi. Newell ve Simon, zeki makineler inşa etmek kadar insanların zekâ eseri davranışlarını da açıklamakla ilgileniyorlardı (belki ikincisiyle daha çok ilgileniyorlardı). Şöyle yazmışlardır: "İnsanların yaptığı işleri makinelerle *başarmak* ile insanların bu işleri başarmak amacıyla yararlandığı süreçleri *taklit etmek* arasına dikkatli bir sınır çizilmesi gerektiği sık sık ileri sürülür... GPÇ (Genel Problem Çözücü) bu iki yaklaşımı, ikisinin de yararına olacak şekilde en iyi harmanlıyor."[11]

GPÇ, bu araştırmacıların, simge yapılarını yönlendirmeyi temel alan (insanların da bunu yaptığına inanıyorlardı) Mantık Kuramcısıyla ilgili eski çalışmalarından tomurcuklanmıştı. Fakat GPÇ'de, simgeleri idare etme stratejilerinin yanı sıra, önemli bir ek mekanizma da vardı. Tıpkı Gelernter'in geometri programı gibi, GPÇ, problemleri altproblemlere dönüştürüyordu. GPÇ'nin yeniliği, çözülmesi gereken problem (simge yapısı olarak sunulur) ile zaten bilinen ya da verili şey

10 Allen Newell, J. C. Shaw ve Herbert A. Simon, "Report on a General Problem-Solving Program", *Proceedings of the International Conference on Information Processing*, Paris: UNESCO House, Münih: R. Oldenbourg ve Londra: Butterworth, 1960, s. 256-264.

11 İnsanın sorun çözme davranışının kuramı ve açıklaması olarak GPS hakkında daha fazlası için bkz. Allen Newell ve Herbert Simon, "GPS, a Program That Simulates Human Thought", *Lernende Automaten* içinde, H. Billings (ed.), s. 109-124, Münih: R. Oldenbourg KG, 1961. Yeniden baskısı: *Computers and Thought*, s. 279-293.

(bu da simge olarak sunulur) arasında bir "fark" hesaplamaktı. Ardından program, baştaki simge yapısına, simgeleri idare eden bir "operatör" (bu farkla ilgili olduğu bilinir) uygulayarak bu farkı azaltmaya çalışırdı. Newell ile Simon, bu stratejiye, "araç-amaç çözümlemesi" demiştir. (Eldeki ayar ile istenen ayar arasındaki farkı sürekli küçültmeye çalışan geribildirimli denetim sistemlerine benzediğine dikkat edin.) Söz konusu çözümleme, operatörün uygulanma koşulunun, yani altproblemin, yerine getirildiğini göstermek zorundaydı. Sonra program, bu altprogram üzerinde çalışacak kendi kopyasını devreye sokup farklılık arar ve böyle sürer.

Örneğin, diyelim ki hedef, Sammy'nin evde olduğu bilinirken, okulda olmasını sağlamak olsun.[12] GPÇ (Genel Problem Çözücü), bir "fark" hesaplar, yani Sammy yanlış yerdedir ve bu farkı azaltacak bir operatör bulur: Sammy'yi okula arabayla götürmek. Sammy'yi arabayla okula götürmek, arabanın doğru düzgün çalışıyor olmasını gerektirir. Problemi biraz ilginçleştirmek adına, diyelim ki arabanın aküsü bitmiş olsun, dolayısıyla GPÇ araba-sürme operatörünü uygulayamıyor, çünkü operatörün sağlam bir aküye ihtiyacı var. Sağlam akü bulmak, GPÇ'nin kendi kopyalarından birini uygulayabileceği bir altproblemdir. GPÇ'nin bu "aşağı" kopyası, bir fark hesaplar, yani sağlam bir akü ihtiyacını belirler ve bir tamirci çağırıp yeni akü takmak için bir operatör bulur. Tamirci çağırmak için telefon numarası gerekir (diyelim ki elimizde var), dolayısıyla GPÇ tamirciyi-ara operatörünü uygular ve tamirci gelip yeni aküyü takar. GPÇ'nin aşağı kopyası, kendi problemini başarıyla çözmüş olur, böylece üst GPÇ artık işine geri dönebilir; arabayı-sür koşulu, yani çalışır durumda bir akünün olması koşulu yerine getirilmiştir. Dolayısıyla GPÇ bu operatörü uygular, Sammy okula gider, asıl problem çözülmüş olur. (Bu örnek, GPÇ'nin genel işleyişini göz önüne seriyor. Koşulları vesaireyle birlikte gerçek simge yapıları, farklılıklar ve operatörler kullanan gerçek bir GPÇ anlatmak çok zahmetli olurdu, ama bundan daha açıklayıcı olmazdı.)

GPÇ, kendisinin yeni bir kopyasını yaratarak altproblemler üzerinde çalıştığında, bilgisayar biliminde (ve matematikte) "özçağrı" (*recursion*) denen çok önemli bir fikirden yararlanıyor. Bilgisayar programcılarının karmaşık programları hiyerarşik olarak düzenlediği görüşüne aşina olabilirsiniz. Yani, ana programlar altprogramları harekete geçirir, bunlar da alt-altprogramları harekete geçirir vs. Ana program, bir

12 Şu internet adresinden örnek veriyorum: http://www.math.grinnell.edu/~stone/events/scheme-workshop/gps.html.

altprogram "çağırdığında", altprogram yapması gereken işi tamamlayana dek (bir olasılıkla ana programa veri döndürene dek) kendini askıya alır, ardından da çalışmaya geri döner. YZ'de (ve başka uygulamalarda da), ana programın kendi kopyalarından birini çağırması yaygındır; yeni kopyanın daha basit problem üzerinde çalışması sağlanır ki sonsuz tekrarlardan ve "döngü"lerden kaçınılsın. Bir programın kendini çağırmasına "özçağrı" denir.

Peki insanlar, düşünürken altprogramlardan ve özçağrıdan faydalanıyor mu? Büyük olasılıkla, fakat o süreç bir alt düzey süreçler zincirini başlattığında, üst düzey düşünce sürecinin yaptığı işe nasıl geri döneceğini anımsama becerileri kesinlikle kısıtlıdır. GPÇ'nin, insan düşüncesinin bu kısıtlamasını taklit etmeye kalkıştığına inanmıyorum.

Newell ve Simon, GPÇ'nin yararlandığı yöntemlerin, çeşit çeşit problemi çözmek için kullanılabileceğine, "genel" teriminin bundan kaynaklandığına inanıyorlardı. GPÇ'yi belirli bir probleme uygulamak için, o probleme özgü bir "fark tablosu" tanımlanmalıdır. Bu tablo, doğabilecek her türlü farklılığı sıralar ve bunları, operatörlerle eşleştirir, ki operatörler, o probleme ilişkin farklılıkları azaltsın. Aslında GPÇ, farklı farklı pek çok mantık problemine ve bulmacaya uygulanmıştır,[13] ayrıca hem yapay zekâ hem bilişsel bilim alanlarında başka çalışmalara ilham kaynağı olmuştur. Fakat bizzat problem çözme programı ve insanda problem çözme davranışı kuramı olarak GPÇ'nin ömrü kısa sürmüştür ve ancak çeşitli mirasçıları sayesinde yaşamını devam ettiriyor (bunları daha sonra anlatacağız).

Sezgisel arama yöntemleri, 1960'larda geliştirilmiş birçok YZ programında kullanılmıştır. Örneğin, yine Minsky'nin doktora öğrencilerinden James Slagle, simge yapıları olarak sunulduğu takdirde kalkülüs problemleri çözebilen SAINT adlı bir sistem programlamıştı. Bu sistem, MIT birinci sınıf sene sonu kalkülüs sınavlarından alınmış 54 problemin 52'sini çözmüştü.[14] Sezgisel yöntemler, masaüstü oyunları oynayan programlarda da epey kullanılmıştır; şimdi bu konuya değiniyorum.

13 Bkz. George Ernst ve Allen Newell, *GPS: A Case Study in Generality and Problem Solving*, New York: Academic Press, 1969.

14 James R. Slagle, "A Heuristic Program That Solves Symbolic Integration Problems in Freshman Calculus", doktora tezi, MIT, Mayıs 1961. SAINT hakkında bir makale için bkz. James R. Slagle, "A Heuristic Program That Solves Symbolic Integration Problems in Freshman Calculus", *Journal of the ACM*, c. 10, no: 4, Ekim 1963, s. 507-520.

5.4 OYUN OYNAYAN PROGRAMLAR

Shannon ile Newell'ın, Shaw'ın ve Simon'un satranç oynayan programlarla ilgili erken dönem çalışmalarından zaten bahsetmiştim. Mükemmel satranç oynamak zekâ gerektiren bir iştir. Aslında, Newell, Shaw ve Simon şunu yazmıştır: "Başarılı bir satranç makinesi tasarlandığında, insanın düşünsel uğraşının çekirdeğine ulaşmış olacağız."[15]

Satranç oynayan makineler hakkında kafa yormanın geçmişi Babbage'a dek uzanır. Dünya şampiyonu bir satranç programının (bundan ileride bahsediyorum) tasarımına yardım etmiş olan IBM araştırmacısı Murray Campbell'a göre, Babbage'ın 1854 tarihli kitabı *The Life of a Philosopher* [Bir Filozofun Yaşamı], bir bilgisayarın satranç oynamak için programlamanın ilk belgelenmiş tartışmasını içerir.[16] Alman tasarımcı ve Z1 ile Z3 bilgisayarlarının mimarı Konrad Zuse, Plankalkül adlı kendi programlama dilini, 1940'ların ilk yarısında satranç oynayan bir program tasarlarken kullanmıştı.

1946'da Turing, bilgisayarların "zekâ"yı, satranç oyunu paradigması üzerinden göstermesi fikrinden bahsetmişti.[17] 1948'de Turing ve lisanstan meslektaşı D. G. Champernowne, bir satranç programı yazmaya başladılar. 1952'de, programı çalıştıracak kadar güçlü bir bilgisayardan yoksun olan Turing, tek hamlesi yarım saat süren bilgisayarı benzetimlediği bir satranç maçı yaptı. (Bu maç kaydedilmiştir. Şu internet adresinde görebilirsiniz: http://www.chessgames.com/perl/chessgame?gid=1356927.) Program, Turing'in meslektaşı Alick Glennie'yle yaptığı maçı kaybetmişti; fakat programın, Champernowne'un eşini yendiği söylenir.[18]

15 Allen Newell, J. Shaw ve Herbert Simon, "Chess-Playing Programs and the Problem of Complexity", *IBM Journal of Research and Development*, c. 2, Ekim 1958, s. 320-355.

16 David G. Stork (ed.), *Hal's Legacy: 2001's Computer as Dream and Reality*, Beşinci Bölüm, Cambridge, MA.: MIT Press, 1996. Şu internet sitesine bakınız. http://mitpress.mit.edu/e-books/Hal/chap5/five3.html.

17 Andrew Hodges, "Alan Turing and the Turing Test", *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer* içinde, Robert Epstein, Garry Roberts ve Grace Beber (ed.), Dordrecht, Hollanda: Kluwer, 2009. Bkz. A. M. Turing, "Proposed Electronic Calculator", National Physical Laboratory Raporu, 1946, *A. M. Turing's ACE Report of 1946 and Other Papers* içinde, B. E. Carpenter ve R. W. Doran (ed.), Cambridge, MA: MIT Press, 1986.

18 http://en.wikipedia.org/wiki/Alan_Turing.

Bu ilk programlardan sonra, satranç programlarıyla ilgili çalışmalar sonraki on yıllar boyunca dura kalka sürdürüldü. John McCarthy'ye göre, Rus YZ araştırmacısı Aleksander Kronrod, "Satranç, YZ'nin Drosophila'sıdır [meyve sineği]" demiştir; yani daha açık uçlu düşünsel görevlere kıyasla YZ'ye araştırma için daha faydalı bir laboratuvar deneği olarak hizmet ettiğini kastediyordu. Minsky'nin dediği gibi, "oyunların ve matematik problemlerinin seçilmesinin nedeni, açık ve basit olmaları değil, en küçük başlangıç yapılarından bize en büyük karmaşıklığı sunmalıdır; insan, programlama alanına görece küçük bir girişten sonra aşılması gerçekten zor olan durumlarla karşı karşıya kalabilir."[19] Satranç, YZ'nin önüne çok zorlu sorunlar çıkarır ve yetkin satranç programlarının boy göstermesi 1960'ların ortalarını buldu. Sonraki bölümde bu programlardan bahsetmeyi sürdüreceğim.

Bununla birlikte, daha etkileyici erken bir başarı, basit dama oyunuyla elde edilmişti (Britanya İngilizcesinde *draughts*[20] diye bilinir). Arthur Samuel (Resim 5.5), Elektrik Mühendisi Profesörü olduğu Illinois Üniversitesi'nde 1940'ların ikinci yarısında, dama oynayan bilgisayar programlama hakkında kafa yormaya başlamıştı. Samuel 1949'da IBM'in Pouhkeepsie Laboratuvarı'na katılmış ve çalışır ilk dama programını 1952'de IBM'in 701 adlı bilgisayarında tamamlamıştı. Bu program, 1954'te IBM 704 için kaydedilmişti. John McCarthy'ye göre,[21] "IBM'in kurucusu ve başkanı Thomas J. Watson Sr., [Samuel'in programının] sergilenmesi, IBM hisse fiyatlarını 15 puan yükseltecektir yorumunu yapmıştı. Öyle de olmuştur."

[Görülüyor ki Samuel, dama programı yazan ilk kişi değildi. *Encyclopedia Brittanica, Online*'a göre, "Başarılı ilk YZ programı 1951'de, ileride Oxford Üniversitesi Programlama Araştırma ekibinin yöneticisi olacak Christopher Strachey tarafından yazılmıştı. Strachey'in dama (*draughts*) programı, İngiltere Manchester Üniversitesi'nde Ferranti Mark I bilgisayarında çalıştırılmıştır. 1952 yazında bu program, makul bir hızla tam bir dama oyunu oynayabiliyordu."][22]

19 Marvin Minsky (ed.), "Introduction", *Semantic Information Processing*, Cambridge, MA.: MIT Press, 1968, s. 12.

20 Burada belirtilen 8×8 tahtada 12 taşla oynanan, bizde İngiliz daması olarak bilinen dama oyunudur —*ed. notu.*

21 İnternet üzerinden geçmişe dönük bakış için: http://www-db.stanford.edu/pub/voy/museum/samuel.html.

22 Bkz. Christopher Strachey, "Logical or Non-mathematical Programmes", *Proceedings of the 1952 ACM National Meeting (Toronto)*, 1952, s. 46-49.



Resim 5.5 Arthur Samuel. (Foto: Samuel'in kızı Donna Hussain'in izniyle.)

Dama oynayan bilgisayar programlarken Samuel'in başlıca amacı, bilgisayarların nasıl öğrendiğini keşfetmekti. Programlamanın "zaman alan ve masraflı yöntem[lerinin]" farkında olan Samuel, "Deneyimlerden bir şeyler öğrenen bilgisayarlar programlamak, gelecekte, bu ayrıntılı programlama çabasına olan gereksiniminin çoğunu ortadan kaldıracaktır" demişti.[23] Samuel'in çabaları, yapay zekânın çok önemli bir parçası olacak şeye, yani "öğrenen makinelere" yönelik ilk çabaların arasındaydı. Samuel, öğrenme yetisi içeren ilk programını 1955'te tamamladı ve 24 Şubat 1956'da televizyonda sergiledi.

Samuel'in yararlandığı öğrenme yöntemlerini betimlemeden önce, onun programının hamleleri nasıl seçtiğini genel olarak anlatacağım. Bu teknik, daha önce betimlemiş olduğum sekizli bulmacada hamlelerin seçilme şekline oldukça benzer; fakat bu programda, rakibin de hamle seçtiği koşulu göz önüne alınmalı. Yine, tahta üzerindeki konumları temsil eden simgesel ifadelerden bir ağaç inşa edilir. İlk dizilişten itibaren, program, olası tüm hamleleri gözden geçirir (ilk hamleyi programın yaptığı varsayımıyla yola çıkarak). Sonuçta, başlangıç dizilişinden sürgün veren olası tüm tahta dizilişleri elde edilir. Ardından, tüm bu dizilişlerden yola çıkılarak, rakibin tüm olası hamleleri gözden geçirilir; bu da daha fazla dallanma yaratır vs.

23 Arthur L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers", *IBM Journal of Research and Development*, c. 3, no: 3, 1959, s. 210-229. Yeniden baskısı: Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw Hill, 1963, s. 71.

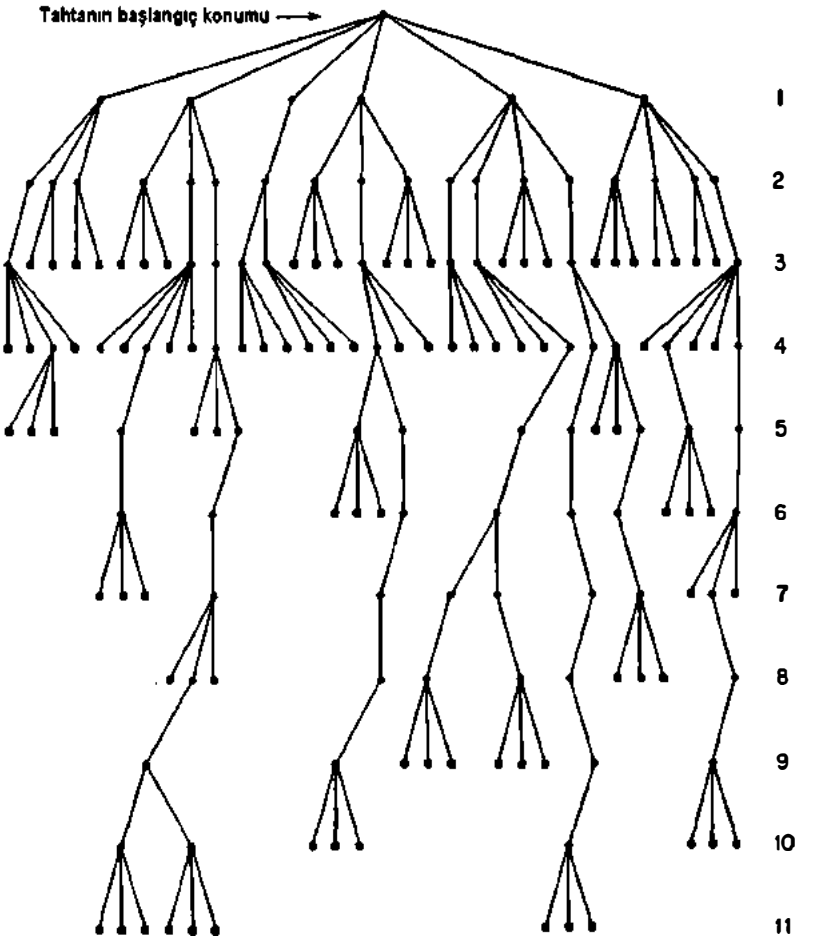
Eğer oyunun bütünü için böyle bir ağaç inşa edilebilirse, bu ağacın incelenmesi sayesinde, kazanan hamle hesaplanabilir. Ne yazık ki, olası yaklaşık 5×10^{20} dama konumunun var olduğu hesaplanmıştır. Oyun oynayan bilgisayar programlamanın önde gelen uzmanlarından Jonathan Schaeffer, yaklaşık 10^{14} konumu incelemeye epey zaman ayırarak damayı "çözmeyi" başarmıştı (iki oyuncu da ideal oyunu oynadığında berabere kalacaklarını ispatlayarak). Bana şunu yazmıştır: "Bu, arama uzamının, ihtiyaç duyduğumuz şeyi bulma olasılığımızın en kuvvetli olduğu kısımlarına odaklanmayı amaçlayan pek çok iyileştirmenin sonucudur."[24] Schaeffer'in çalışmasını ileride daha ayrıntılı anlatacağım.

Öyleyse Samuel'in programı, ağacın ancak bir parçasını inşa edebilirdi; yani, ancak ilerideki birkaç hamleyi düşünebilirdi. Çeşitli dallarından ne kadar ileriye baktığı, bizi burada ilgilendirmesi gerekmeyen kimi etkenlere bağlıydı. (Yakında taş alma imkânı var mı yok mu gibi konularla ilgilidir.) Üç hamle ilerisini görmek sıkça rastlanan durumdu, gerçi kimi dallar, (seyrek olarak) on hamleye ulaşan bir derinliğe kadar incelenebilirdi. Resim 5.6'da gösterilen, Samuel'in makalesinden bir diyagram, genel fikri göz önüne seriyor. Samuel, "gerçek dallanmalar çok daha kalabalıktır" demiştir.

O halde program, böyle eksik bir ağaçtan nasıl hamle seçiyor? Oyun oynayan tüm programlar bu sorunla karşı karşıya kalır ve hepsi, ağacın uçlarındaki, yani "yaprak"larındaki (programın oluşturduğu eksik ağacın yapraklarındaki) konumlar için bir puan hesaplayıp, şu anki konumdan kaynaklanan hamleler sonucu oluşan konumlara bu puanı geri "taşır". Önce, puanın nasıl hesaplandığını, sonra puanın nasıl geri taşındığını, daha sonra da Samuel'in, performansını artırmak amacıyla öğrenme yöntemlerinden nasıl faydalandığını betimleyeceğim.

Samuel'in programı önce, ağacın yapraklarındaki konumlara verilecek puanları, programın bakış açısıyla "yararlılıklarına" göre belirliyordu. Puanlara katkı veren özellikler arasında, göreli taş üstünlüğü (dama, sıradan taşlardan değerlidir), programının taşlarının toplam "hareketliliği" (hareket etme serbestliği) ve oyun tahtasında merkezin denetim altında tutulması vardı. (Programın, bu tür 38 özelliği tanımlıydı; fakat herhangi bir anda bunların sadece 16'sını kullanıyordu.) Her özelliğin getirdiği puanlar (o özelliğin göreli önemini yansıtan) bir "ağırlık" ile çarpılıyordu ve bir konum için toplam bir puan oluşturmak için sonuçlar toplanıyordu.

24 E-posta, 14 Şubat 2009.



Resim 5.6. Açıklayıcı bir dama oyunu ağacı (Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw Hill, 1963, s. 74)

Ağacın ucundaki konumların hemen üzerindeki bir konumla başlayarak, ki eğer bu hamle sırasının bilgisayarda olduğu bir konumsa, programın en yüksek puanı olan konuma hareket etmek isteyeceğini varsayabiliriz, böylece en yüksek puan, "hemen üzerindeki" o konuma geri taşınabilir. Bununla birlikte, eğer hamle sırası rakipteyse, rakibin en düşük puanlı konuma hareket etmek isteyeceğini varsayabiliriz. Bu durumda, en düşük puan, hemen üzerindeki o konuma taşınır. "Yüksek-alçak" arasında gidip gelme stratejisi ağacın en tepesine kadar sürdürülür ve buna "En küçük en büyük" stratejisi denir.

[Bu stratejinin "alfa-beta" yöntemi adlı basit bir farklı sürümü daha önce taşınmış puanlardan yola çıkarak belirli dalların incelenmesinin (haklı olarak) gerekmediğini çıkarsamak amacıyla kullanılır; böylece başka dalların daha derinlemesine incelenmesine olanak tanınır. Bu önemli geliştirmeyi ilk kimin düşündüğü hakkında çeşitli görüşler var. McCarthy de, Newell da, Simon da payeyi kendilerine biçer. Samuel, bu yöntemden faydalandığını ama hakkında yazmayı gerektirmeyecek kadar açık olduğunu söylemişti bana.]

Eğer o anki konumda hamle yapma sırası bilgisayardaysa ve puanlar o konumun hemen altındaki konumlara zaten geri taşınmışsa, program hamlesini en yüksek puanlı konuma yapacaktır. Ardından oyun, rakibin hamle yapmasıyla sürer, böylece ağacın büyüdüğü, puanın hesaplanıp taşındığı yeni bir evreye girilmiş olur, ta ki taraflardan biri kazanana ya da kaybedene dek.

Samuel'in programındaki öğrenme yöntemlerinden biri, puanlama sisteminin kullandığı ağırlık değerlerinin ayarını yapar. (Pandemonium'da ve sinir ağlarında ağırlık değeri ayarlamalarının, o sistemlerin öğrenme usulü olduğunu anımsayın.) Ağırlıklar öyle ayarlanır ki tahta üzerindeki bir konumun puanı (ağırlık verilmiş özellik puanlarının toplanarak hesaplanmış haliyle), arama süreci sonrasında ulaşılmış puanın değerine yakın bir değere taşınır. Örneğin, eğer bir başlangıç konumunun puanı 22 olarak hesaplanmışsa (ayarlamadan önceki ağırlıklarla), ve arama sürecinden sonra o konumun ulaşılmış puanı 30'sa, o zaman başlangıç konumunun puanını hesaplamak üzere kullanılan ağırlıklar, öyle bir biçimde ayarlanır ki yeni puan (ağırlıkların ayarlanmış değeri kullanılarak) 30'a yakınlaştırılır, örneğin 27 yapılır. (Bu teknik, ileride Richard Sutton'ın dile getireceği, "zamansal-fark öğrenmesi" adlı çok önemli bir öğrenme yönteminin habercisiydi.) Burada ana fikir, oyundaki gibi ileriyi görmeye dayanan, ulaşılmış puanın, asıl puandan daha iyi bir tahmin kabul edilmesidir. Bu yüzden tahmin süreci öyle iyileştirilmiştir ki "öngörülen" puanla daha tutarlı değerler üretir.

Ayrıca Samuel, "ezbere öğrenme" denen başka bir yöntemden de faydalanmıştır; buna göre program, tahtada çeşitli konumları ve gerçek oyun sırasında bunların ulaşılmış puanlarını saklar. Ardından, arama sürecinin sonunda, eğer varılan yaprak konumu, saklanmış konumlardan biriyle aynıysa, bunun puanı çoktan biliniyor demektir (dolayısıyla, ağırlıklar ve özellikler kullanılarak hesaplanmasına gerek kalmaz). Önceki bir arama sürecini temel alan bu bilinen puan, hesaplanan puana göre, konum için daha sağlam bir gösterge olacaktır muhtemelen.

Samuel'in programı ayrıca "kayıtlı oyunların" kullanımından da faydalanıyordu; bunlar, usta dama oyuncularının oyunlarının kayıtlarıdır. Samuel'in çalışmasını yorumlayan John McCarthy, şöyle yazmıştır: "Dama oyuncuları, iyi hamlelerin kötü hamlelerden ayırt edildiği pek çok açıklama kitabına sahiptir. Samuel'in öğrenen programı, hamle seçerken ölçütlerini belirlemek için *Lee's Guide to Checkers*[25] [Lee'nin Dama Kılavuzu] kitabından faydalanıyordu, böylece program, dama oyuncularının sık sık başvurduğu sağlam hamleleri yeğleyecekti."

Samuel'in programı çok iyi dama oynuyordu ve 1962 yazında, Connecticutlı kör dama ustası Robert Nealey'i yenmişti. (Bay Nealey ile Samuel'in programı arasındaki oyunu şu internet adresinden izleyebilirsiniz: http://www.fierz.ch/samuel.htm.) Ancak, Jonathan Schaefer ile Robert Lake'e göre: "1965'te program Walter Hellman'a ve Derek Oldburry'ye (o zamanlar Dünya Şampiyonluğu unvanı için karşılaşıyorlardı) karşı dörder maç yapmış, sekiz maçı da kaybetmiştir.[26]

25 John W. Dawson, *Lee's Guide to the Game of Draughts*, Gözden geçirilmiş baskısı, Londra: E. Marlborough, 1947.

26 Jonathan Schaeffer ve Robert Lake, "Solving the Game of Checkers", *Games of no: Chance*, MSRI Publications, c. 29, 1996, s. 119-133. (Şu internet adresinde: http://www.msri.org/communications/books/Book29/files/schaeffer.pdf.)

6
ANLAMSAL GÖSTERİMLER

Şimdiye dek betimlemiş olduğum bilgisayar programları, görece basit simge yapıları üzerinden dönüşümler gerçekleştiriyordu; bu kadarı, o programların boğuştuğu matematik problemleri, bulmacalar ve oyunlar için yeterliydi. Bu programlarda başlıca çaba, çözüm arayışında bu yapıların dönüşüm sayısını kısıtlamak amacıyla probleme özgü sezgisel yöntemler (örneğin, dama konumlarının değerini hesaplarken kullanılacak özellikler gibi) belirleyip kullanmaktı. Minsky'nin belirttiği gibi, "1962 öncesi dönemin en merkezî fikri, *deneme yanılma yöntemiyle yapılan aramanın* genişliğini denetim altına alacak sezgisel araçlar keşfetmekti."[1] 1960'ların ilk yarısında, bir kısmı MIT'de Minsky'nin denetiminde gerçekleştirilmiş birtakım doktora araştırma programları, çeşitli düşünsel görevleri gerçekleştiren programları daha karmaşık simge yapılarıyla donatmaya başladı. Bu yapılara, problem konuları hakkındaki bilgi içeriklerinin ayrıntılı ve zengin oluşundan ötürü, *anlamsal gösterimler* denmiştir.[2] Minsky'nin yazmış olduğu gibi: "Her programın iş gördüğü küçük alan kapsamında, [bu programların] performansı, birtakım insan etkinlikleriyle kıyaslandığında çok kötü değildir... Fakat bu özel deneylerin başardıklarından çok daha önemlisi, yaptıklarını başarırken kullandıkları yöntemlerdir, *zira bu yöntemlerin her biri, daha önce sınanmamış fikirlerin ilk denemesiydi.*"[3] Şimdi de, bu tür projelerin kimi örneklerini ve kullandıkları yeni yöntemleri anlatacağım.

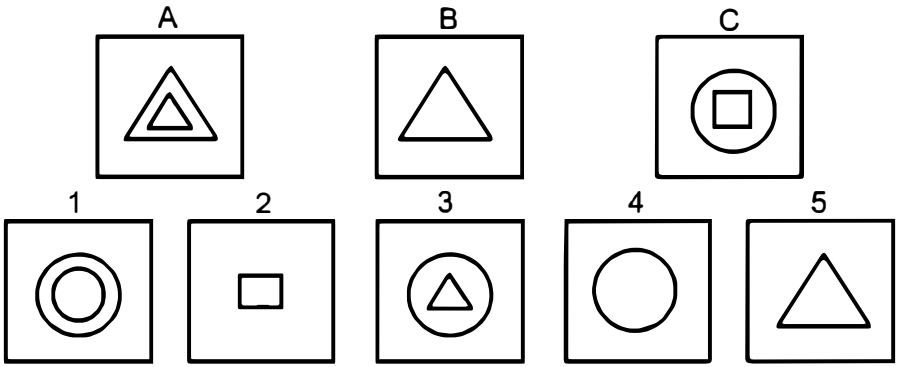
6.1 GEOMETRİK BENZERLİK PROBLEMLERİNİ ÇÖZMEK

Thomas G. Evans (1934-), kimi iyi bilinen geometrik benzerlik testlerinde başarılı olan bir sistem programlamıştı. Anlaşılıyor ki o güne dek, John McCarthy'nin yeni programlama dili LISP'le yazılmış en

1 Marvin Minsky (ed.), "Introduction", *Semantic Information Processing*, Cambridge, MA: MIT Press, 1968, s. 9.

2 Gelernter'in geometri programının kullanmış olduğu diyagramın, anlamsal gösterimin erken bir kullanımı olduğu ileri sürülebilir.

3 Marvin Minsky, *a.g.e.*, s. 1.



Resim 6.1. Bir benzerlik problemi.

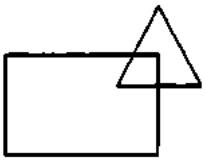
büyük programdı bu (ileride tekrar bu konuya döneceğim). Evans'ın, bu çalışmayı sunan 1963 tarihli tezini temel almış bir makaleden aktarıyorum:[4]

> Makinenin, "geometrik benzerlik" nitelikli zekâ testi sorularına getirdiği çözümlere göz atacağız. Bu problem sınıfının her üyesi, etiketlenmiş çizimler kümesinden meydana gelir. Yapılması istenen görev, şu soruyla tanımlanabilir: "Resim A, Resim B'yle eşleşiyorsa, Resim C aşağıdakilerden hangisiyle eşleşir?" Örnekte [Resim 6.1'de], yanıt olarak [4 numarayı] seçerek çoğu insanın, betimlemek üzere olduğumuz programla aynı fikirde olacağı güvenle söylenebilir.

Şunu da eklemiştir: "Bu tür problemlerin çözümlerinin genelde yüksek zekâ gerektirdiği düşünülür ve gerçekte, üniversite başvurularında ve başka amaçlar için faydalanılan kimi genel zekâ testlerinde zekânın mihenk taşı olarak bu tür problemlerden yararlanılır". Yani, bir kez daha, YZ araştırması insan zekâsı gerektiren görevleri mekanikleştirmeye yoğunlaşmıştı.

Evans'ın programı, sunulan diyagramları öncelikle bileşenlerden nasıl oluştuğu ortaya çıkacak biçimde dönüştürüyordu. Bunlara, "eklemli" gösterimler diyordu. Birkaç ayrışma olasılığı arasından "bağlamına" göre biri seçiliyordu. (Bu seçim, programın kullandığı sezgisel yöntemlere örnektir.)

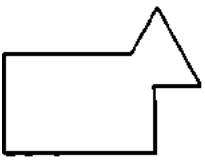
4 Thomas G. Evans, "A Program for the Solution of a Class of Geometric-Analogy Intelligence-Test Questions", *Semantic Information Processing* içinde, Marvin L. Minsky, s. 271.



Örneğin, yukarıdaki diyagram

 ve  olarak, ya da

ya da

 ve  olarak

ayrışabilirdi . Fakat, eğer benzerlik problemi (bağlamın parçası olan) başka bir diyagram içeriyorsa, o zaman



ilk ayrışma seçilecektir.

Evans, diyagramları ve bunların bileşenlerini karmaşık simge yapıları olarak gösteriyordu; bu bileşenler, öğeleri, başka bileşenlere kıyasla hangi bileşenlerin içeride ya da dışarıda (yukarıda ya da aşağıda) olduğunu vesaire gösteren listelerin ya da liste listelerinin görece ayrıntılı birleşimlerinden oluşuyordu. Bu gibi ayrıntılara kafa yormamıza gerek yok ama bunlar sayesinde Evans, programının, bir diyagramın başka bir diyagrama nasıl dönüştürülebileceğini göstermek amacıyla kullandığı "kuralları" belirlemiştir. Bu program, belli bir problemdeki A Şeklini B Şekline dönüştüren kural birleşimlerini çıkarsayabiliyordu. Ardından bu dönüşüm kuralını C Şekline uyguluyordu. Eğer çoktan seçmeli yanıtlardan birine ulaşmışsa, yanıt olarak bunu veriyordu. Aksi takdirde program, dönüşümü yanıtlardan birine ulaşabilecek kadar "gevşetirdi" ve programın yanıtı bu olurdu.

Evans, sonuçlarını şu şekilde özetlemiştir:

> Kendimize sadece [programın şimdiye dek gerçekleşmiş kısımlarını] kullanacak kadar olanak tanırsak, tipik bir ACE testindeki 30 geometrik benzerlik sorusundan [programın] en az 15'ini, belki 20 kadarını başarıyla çözeceğini tahmin edebiliriz.

Bu başarı düzeyinin, ortalama lise öğrencisinin başarısını aştığını belirtir.

6.2 BİLGİ SAKLAMAK VE SORULARI YANITLAMAK

1960'ların ilk yarısında yine Minsky'nin doktora öğrencilerinden Bertram Raphael (1936-), "yapay anlama" sorununa odaklanmıştı. Raphael, tezinde bunu şöyle açıklıyor:[5]

> Zekice konuşabilen, yani anlatılan şeyi anımsayabilen, sorulara yanıt veren ve insan gözlemcilerin makul sayacağı tepkiler veren bilgisayarın "anlayabildiği" kabul edilmeli.

Raphael, bilgisayara birtakım şeyler anlatmak ve yanıtları bu anlatılanlardan çıkarsanabilecek sorular sorabilmek istiyordu. (Anlatma ve sorma işi, klavyeyle cümleler ve sorular yazarak yapılacaktı.) Raphael'in bilgisayara anlatmak istediği şeylere bazı örnekler şöyle:

> Her oğlan bir kişidir.
> Parmak, elin bir parçasıdır.
> Her kişide iki el vardır.
> John, oğlandır.
> Her elde beş parmak vardır.

Raphael, verilen bu bilgilerle birlikte, sisteminin, "John'un kaç parmağı vardır?" sorusunun yanıtı çıkarsamasını istiyordu.

Raphael, sisteminin insanlarla iletişim kurmasını arzuladığı için, girdi ve çıktı lisanının "doğal İngilizceye makul ölçüde yakın" olmasından yanaydı. Şunu fark etmişti: "Genel bir anlamsal bilgi edinme sistemi kurmadan önce, doğal dil girdisini faydalı bir biçime dönüştürmek şeklindeki dilbilim sorunu çözülmeli". Bu "dilbilim sorunu" oldukça zorludur ve 1960'lardan bu yana epey ilerleme kaydedilmiş olsa da henüz "çözülmüş" değildir. Raphael, "İngilizce benzeri anlaşılır bir girdi ve çıktı dili kullanırken, [doğal dili kullanmak şeklindeki genel sorunun]

5 Bertram Raphael, "SIR: Semantic Information Retrieveal", Marvin Minsky, *a.g.e.* içinde, s. 33-145. (Bu metin, 1964 tarihli tezinin bir kısmını içermektedir.)

etrafından dolaşmak" amacıyla çeşitli "araçlar" (onları bu sözcükle nitelemiştir ve bunlar, buradaki bahsimizle alakalı değil) kullanmıştı.

Raphael'in saldırdığı ana iş, bilgisayarın belleğindeki olguları, bunlarla ilişkili çıkarımlar yapabileceği şekilde düzenlemekti. Raphael'in belirttiği gibi: "'Anlama' yetisinde en önemli önkoşul, saklanmış bilgiye uygun bir iç gösterim ya da modeldir. Bu model öyle yapılanmış olmalı ki soruya yanıt vermekle ilgili bilgiye kolaylıkla erişilebilmeli."[6]

Rafael, sistemine SIR (Semantic Information Retrieval'ın [Anlamsal Bilgi Erişimi] kısaltması) adını vermişti; bu sistemi LISP dilinde programlamıştı. "Anlam" sözcüğünü kullanmasının nedeni, SIR'ın cümleleri anlamlarına bağlı olarak modellemesiydi. SIR'ın işleyebildiği cümleler, birtakım "öğeler"i (örneğin John, oğlan, el, parmak vs.) ve bu öğeler arasındaki ilişkileri ("küme-üye", "parça-bütün", "sahiplik", "yukarıda", "aşağıda" ve başka uzamsal ilişkiler gibi) içeriyordu. Dolayısıyla bu model, öğeleri ve bunların arasındaki ilişkilerin gösteriminin yöntemlerini kapsıyordu.

John ve oğlan gibi birimler, LISP bilgisayar dilinde JOHN ve BOY sözcükleriyle gösterilir. (Elbette bilgisayar, JOHN sözcüğünün gerçek kişi John'la ne ilişkisi olduğunu bilemez. Raphael, John'u bilgisayarda X13F27'yle de gösterebilirdi, yeter ki bu gösterimi tutarlı olarak John yerine kullansın. Bilgisayarda JOHN sözcüğünü kullanmak, programcının hatırlamasına yardımcı olur, yoksa bilgisayarın hatırlamasına değil!) John'un oğlan olduğu olgusu temsil edilirken, SIR, (SUPER-SET JOHN BOY) bilgisayar ifadesiyle JOHN ifadesi arasında bağlantı kurar, ayrıca (SUPER-SET JOHN BOY) bilgisayar ifadesiyle BOY ifadesi arasında da bağlantı kurar. Dolayısıyla, eğer SIR'a oğlanın adı sorulursa, kendi modelinde BOY'a atıfta bulunarak "JOHN" yanıtı verir, çünkü SUB-SET [altküme] bağlantısına bakıp JOHN sözcüğünü çekip çıkarmıştır. (Ana fikri anlatabilmek için gösterimleri bir parça sadeleştirdim; SIR'ın asıl gösterimleri biraz daha karmaşıktır.)

SIR, düzinelerce farklı öğe ve bunlar arasındaki ilişkilerle başa çıkabiliyordu. Yeni bilgi aldığı her seferinde, ihtiyaca göre yeni öğeler ve bağlantılar ekliyordu. Aynı zamanda, mantıksal çıkarımlar ve basit aritmetik işlemler yapacak birtakım mekanizmaları da vardı. Tam da modelin yapısı, yaptığı çıkarımların pek çoğunu kolaylaştırıyordu, çünkü, Raphael'in tezinden bahsederken Minsky'nin de söylediği gibi, "dolaysız belirtim-bağlantıları... verili bilginin dolaysız mantıksal so-

6 Marvin Minsky, *a.g.e.*, s. 35.

nuçlarını neredeyse fiziksel olarak birbirine zincirliyor."[7]

Aynı zamanda SIR, uslamlamada "istisna kuralını" kullanan ilk YZ sistemiydi. Bu ilkeyi, Raphael'in tezinden yapacağımız alıntıyla gayet güzel açıklayabiliriz:

> Bir kümenin "tüm öğeleri" hakkındaki genel bilginin, ancak o öğelerle ilgili özgül bilginin yokluğunda belirli öğelere uygulanabileceği düşünülür. Dolayısıyla, "balina her zaman suda yaşayan bir memeli" olsa da "memeliler kara hayvanıdır" bilgisini öğrenmek çelişkili değildir. Programda bu fikir, bireyin ait olduğu kümelerin betimlemelerine bakmadan *önce* arzulanan bilginin, söz konusu bireyin özellik listesine [yani bağlantılarına] atıfta bulunulmasıyla gerçekleşir.
>
> Aristocu mantığın, istisna kabul etmeyen ilkerinden bu kopuşun gerekçesi, belirli olguların temel bilginin önüne geçişinin insanların işlem yapma yöntemiymiş gibi görünmesidir, üstelik bilgisayarın insanlarla olabildiğince doğal bir iletişim kurmasını istiyorum.
>
> Şu anki program, "balina suda yaşayan bir memeli olsa da memeliler kural olarak karada yaşar" gibi olgularla karşılaştığında insanlara sık sık hissettiği o rahatsızlığı deneyimlemiyor.

İleride YZ araştırmacıları istisna ilkesi üzerinde çok daha ayrıntılı çalışacak, böylece bu ilke, daha sonra göreceğimiz üzere, olağan uslamlama ve tekdüze olmayan mantık denilen sistemleri doğuracaktı.

6.3 ANLAMSAL AĞLAR

SIR'ın gösterim şemasını ağ bağlamında düşünmek eğitici olacaktır. Birimler (JOHN ve BOY gibi), ağın "düğüm noktalarıdır" ve ilişki bağlarıysa (SUB-SET gibi), düğüm noktaları arasındaki bağlantılarıdır. SIR, yapay zekâ alanında önemli bir gösterim fikri olacak şeyin, yani *anlamsal ağların* ilk sürümlerinden biriydi. Fakat ilk sürüm değildi. Anlamsal ağlar üzerine epey kalem oynatmış John Sowa'nın iddiasına göre, "bilinen en eski anlamsal ağ, MS 3. yüzyılda Yunan filozof Porphyrios tarafından, Aristoteles'in kategorileri hakkındaki incelemesinde çizilmiştir."[8] 1961'de Cambridge Dil Araştırmaları Birimi'nin

7 Marvin Minsky, *a.g.e.*, s. 17.

8 John F. Sowa'nın makalesinden; bkz. http://www.jfsofa.com/pubs/semnet.htm. (Bu, aslında Stuart C. Shapiro'nun hazırlamış olduğu *Encyclopedia of Artificial Intelligence* (Wiley, 1987, 2. baskı, 1992) için yazılan makalenin gözden geçirilmiş ve genişletilmiş baskısıdır.)

Müdürü Margaret Masterman (1910-1986), kavramların hiyerarşiye göre dizildiği bir çeviri sisteminde anlamsal ağ kullanmıştı.[9]

Carnegie Teknoloji Enstitüsü'nde Herb Simon'un öğrencilerinden M. Ross Quillian, Neill ve Simon'la birlikte, insanda zihinsel süreçlerin, özellikle de bellek örgütlenmesinin hesaplamalı modellerine ilgi duyuyordu. Geliştirmiş olduğu bellek modeli, İngilizce sözcüklere karşılık gelen düğüm noktalarıyla dolu anlamsal ağlardan oluşuyordu. Düğüm noktaları, "bağdaşım bağlantıları" dediği şeylerle birbirlerine bağlıydı. Quillian'ın sözcükleriyle söylemek gerekirse: "Bellek modelinde, bir kavram inşa etmek üzere kullanılan birimler, başka kavramları belirten hatırlatıcı düğüm noktalarınca temsil edilir, oysa kavramın biçimsel anlamı, o hatırlatıcı düğüm noktalarını birbirine bağlayan belirli bir ara bağlantı yapısı tarafından temsil edilir."

Quillian şunu da yazar: "Bu araştırmanın ana sorusu şu olmuştur: Kişinin belleği içinde anlamsal bilginin nasıl örgütlendiğine dair makul bir görüşü ne oluşturuyor? Başka bir deyişle: Saklanacak sözcüklerin 'anlamlarına' elveren, böylece bu anlamları insana benzer biçimde kullanmayı mümkün kılan gösterim biçimi hangisidir?"[10]

Quillian'ın ağ biçiminin, anlamı nasıl temsil ettiğini, verdiği örneklerden birini kullanarak açıklayabilirim. Bitki anlamındaki PLANT sözcüğünün farklı anlamlarını gözünüzün önüne getirin.[11] Bu anlamlardan biri, BİTKİ düğüm noktasının CANLI, YAPRAK, GIDA, HAVA, SU ve TOPRAK gibi başka düğüm noktalarına bağlanmasıyla verilir; bu bağlantılar, bitkinin (sözcüğün bu anlamı uyarınca) canlı olup yapraklarının bulunduğunu ve gıdasını havadan, sudan, topraktan aldığını gösteriyor. Fabrika anlamındaki PLANT ise, İNSANLAR, SÜREÇ, SANAYİ gibi başka düğüm noktalarına bağlanır; bu bağlantılar ise, fabrikanın (sözcüğün bu anlamı uyarınca), üretimde kullanıldığını ve insanlardan faydalanan bir düzenek oluşunu gösterir.

Quillian'a göre bir terimin anlamı, ağdaki yeri ve öteki terimlerle arasındaki bağlantı üzerinden gösterilir. Aynı fikirden sözlüklerde de

9 Margaret Masterman, "Semantic Message Detection for Machine Translation, Using an Interlingua", *Proceedings of the 1961 International Conference on Machine Translation of Languages and Applied Language Analysis* içinde, Londra: Her Majesty's Stationery Office, 1962, s. 438-474.

10 M. Ross Quillian, "Semantic Memory", doktora tezi, Carnegie Teknoloji Enstitü (günümüzde Carnegie Mellon Üniversitesi), Ekim 1966. (Bu çalışma aynı zamanda AFCRL-66-189 Raporu olarak yazılmıştır ve kısmen şu kitapta basılmıştır: M. Minsky (ed.), *Semantic Information Processing*, Cambridge, MA: MIT Press, 1968, s. 216-270.)

11 Hem "bitki" hem "tesis" anlamına gelir —*çev. notu.*

faydalanılır; sözlükte bir sözcüğün anlamı, bu sözcüğün başka sözcüklerle ilişkisinden bahsedilerek verilir. O sözcüklerin anlamları da daha başka sözcüklerle ilişkileri üzerinden tanımlanır. O halde sözlükleri, sözcüklerle bağlantılı sözcüklerden oluşan büyük birer anlamsal ağ sayabiliriz.

Bu görüşten yararlanıldığında, bir kavramın *tam* anlamı oldukça kapsamlı olabilir. Quillian'ın belirttiği gibi:

> Diyelim ki birinden, "makine" kavramı hakkında bildiği her şeyi ifade etmesi isteniyor... Bu bilgi, makineler hakkında daha "gerekli" olgularla başlayacaktır, örneğin genelde insan yapımı oluşları, hareketli aksamlarının bulunması, vesaire gibi; ardından gitgide daha az kapsayıcı olgulara "inilecektir", örneğin daktiloların makine oluşu gibi ve sonunda makineler hakkında iyice uzak bilgilere ulaşılacaktır, örneğin daktilonun, şaryosu başa çekildiği her seferinde uçup gitmesin diye sabit bir tıpasının oluşu gibi. Tüm bu bilgilerin, öznenin "makine" kavramının faydalı bir parçası olarak görülebileceğini öne sürüyoruz.

Peki Quillan'ın bahsettiği ağ, hangi bakımdan insan bellek örgütlenişinin modelidir? Quillan, bu ağ üzerinden modelini kurduğu insan belleğinin iki yetisini irdelemişti. Bir tanesi, iki farklı sözcüğü karşılaştırıp zıtlıklarını belirlemektir. Quillian, bu işin, "yayılan etkinlik" diye nitelenen bir süreç tarafından başarıldığını ileri sürmüştü. Kavramsal olarak, iki sözcüğe karşılık gelen düğüm noktalarından yola çıkılır, bunlardan dışarı doğru uzanan bağlantılar yavaşça kat edilir, üstelik yol boyunca düğüm noktaları "etkinleştirilir". Bu süreç, iki etkinleştirme "dalgası" kesişene kadar sürer, böylece baştaki iki düğüm noktası arasında bir "yol" oluşur. Quillian, iki sözcük arasındaki yolun toplam uzaklığının, benzerliklerinin bir ölçüsü olarak kullanılabileceğini ileri sürer. Bu yoldan, iki sözcüğü kıyaslayan bir açıklama oluşturmak üzere faydalanılabilir. (Quillian'ın programı, bu açıklamayı basit İngilizceyle ifade edecek mekanizmalar barındırıyordu.)

Quillian'ın örneklerinden birini kullanmak için, diyelim ki "ağlamak" ve "avutmak" sözcüklerini karşılaştırmak istiyoruz. Yayılan etkinlikler, "üzüntülü" sözcüğünde kesişir ve konuşma dilindeki açıklama şunun gibi bir şey olur: "Ağlamak, üzüntülü bir ses çıkarmak demektir ve avutmak, bir şeyin üzüntüsünü azaltmaktır."

Quillian ayrıca, bu ağdan, aynı sözcüğün olası iki farklı kullanımını "ayırt etmek" üzere nasıl faydalanılabileceğine de ilgi gösteriyordu. Örneğin şu cümleye göz atalım: "*Strike*'tan [grev, vurma, çarpma, isabetli atış] sonra, başkan ona yol verdi". Ağ, *strike* sözcüğünün fark-

lı anlamlarını kodlayabilir. Bu anlamlardan biri, grevi, öbürü beyzbolu, öbürü de askerî bir hava saldırısını içerebilir. Peki cümlede bu anlamlardan hangisi kastediliyor? Sanırız ki, "başkan" sözcüğünden başlayan etkinleşme sonunda, beyzbolla ya da askeriyeyle ilgili kavramlara varmadan önce, grevle ilgili kavramlara ulaşacaktır. Dolayısıyla, cümlede "başkan" sözcüğü geçtiğine göre, "daha yakın" olduğu için "grev" anlamı yeğlenecektir. Bunun aksine, "*strike*'ın ardından, hakem onu dışarı gönderdi" cümlesi için farklı bir sonuca varılacaktır.

Quillian'ın modeli, önceden belirlenmiş bir üstsınıf ve altsınıf hiyerarşisi barındırmayışı bakımından, daha sonraki anlamsal ağlardan farklıydı. Quillian'ın belirttiği gibi: "Her sözcük, *bununla birlikte bir arama süreci başladığında*, kendi ayrı hiyerarşisinin atasıdır. Benzer biçimde her sözcük, arama süreci bu kavramlarla başladığında, pek çok başka sözcük kavramının hiyerarşileri içinde (bunların bir maddesi olarak) çeşitli yerlerde bulunur.

7
DOĞAL DİL İŞLEME

İster tektip yazı karakteriyle ister elyazısıyla yazılmış olsunlar, tek tek abecesayısal karakterlerin örüntüsünü tanımanın ardında, İngilizce gibi "doğal" bir dilin sözcüklerini, cümlelerini, daha büyük metinlerini oluşturan harf dizilerini anlama sorunu yatar. İngilizce gibi dilleri, bilgisayarların kullandığı dillerden ayırt etmek amacıyla ilkine, "doğal diller" denir. Yapay zekâda, doğal dil girdisini anlamak çoğunlukla, ya bunu bir tür bellek modeline dönüştürmek (SIR sisteminde Raphael'in ya da anlamsal ağlarla Quillian'ın kullandığı modeller gibi) ya da girdiye uygun birtakım eylemleri çağrıştırmak demektir.

Doğal diller yazılı oldukları kadar sözlüdür. Üstelik, konuşma sesleri, kâğıt üzerindeki yazı karakterleri kadar iyi ayrılmış olmadığı için, konuşmayı anlamak ek zorluklar doğurur; bunları ileriki bir bölümde anlatıyorum.

Doğal dil girdisini anlamanın tersi, hem yazılı hem sözlü olarak doğal dil çıktısı oluşturmaktır. Bir dilden başka dile çeviri yapmak, hem anlamayı hem üretmeyi (*generation*) içerir. Bir sohbeti sürdürmek de öyle. Tüm bu sorunlar (anlamak, üretmek, çeviri yapmak, sohbet etmek), "doğal dil işleme" (bazen DDİ olarak kısaltılır) genel başlığı altına giriyor.

7.1 DİLBİLİMDE DÜZEYLER

Dilbilimciler ve dilleri araştıranlar, dilin çözümlendiği birkaç düzey olduğunu bilirler. Bu düzeyler bir tür hiyerarşi oluşturur; bu hiyerarşi, dilin en temel bileşenlerini (sesler, heceler) ele almakla başlayıp yukarı çıkarak cümle dizileriyle uğraşan düzeylere ulaşır. Eğer konuşma ele alınacaksa, sesbilgisi (dil sesleri [*phonetics*]) ve sesbilim (seslerin sözcük olarak örgütlenişi [*phonology*]) düzeyleri söz konusudur. Hem konuşma hem de metin için, biçimbilim (*morphology*), sözcük bütünlerinin ufak parçalardan nasıl oluşturulduğunu inceler. Örneğin, "yürümek", "yürü" artı "mek" ekinden oluşur.

Ardından, cümle yapısını ve dilbilgisini ele alan sözdizim (*syntax*) gelir. Sözdizim, belirli bir dildeki sözcük zincirlerinin dilbilgisine uy-

gunluk kurallarını belirlemeye uğraşır. Örneğin, "John topa vurdu" sözcük zinciri dilbilgisine uyar ama "vurdu topa John" devriktir. Bir cümleyi anlamak bakımından, önem sırasında sözcüklerin sözlük anlamları, sözdiziminden önce gelir. Örneğin, "John, teleskopla adamı gördü" cümlesinin, sözdizim yapısına bağlı olarak iki farklı anlamı vardır ("teleskopla" sözcüğünün, teleskopla yan yana duran "adamı" sözcüğünü ya da "gördü" fiilini nitelemesine göre).

Ancak, salt dilbilgisine uygunluk, anlamı belirlemek yolunda yetersiz kalır. Örneğin, "Renksiz yeşil fikirler, hunharca uyuyordu" cümlesi, dilbilgisi açısından doğru olabilir ama anlamsızdır. Anlambilim (*semantics*) düzeyi, mantıksal çözümlemeleri devreye sokarak bir cümlenin anlamını (ya da anlamsızlığını) belirlemeye yardımcı olur. Örneğin, anlamsal çözümleme uyarınca bir "fikir" aynı anda hem "renksiz" hem "yeşil" olamaz.

Ardından edimbilim (*pragmatics*) düzeyi gelir; bu düzeyde, anlamı açıkça belirlemek üzere bağlam dikkate alınır. Örneğin, İngilizcede *John went to the bank* cümlesi, ticaretle ilgili bir cümlede ayrı, nehir balıkçılığıyla ilgili bir cümlede ayrı anlama gelir.[1] Edimbilim, özgül durumlar bağlamında anlamları ele alır.

Bu düzeylerden bir tanesi, yani sözdizim, eskiden beri birtakım çalışmaların konusu olmuştur ve DDİ'nin (doğal dil işleme) önemli bir yüzü olmayı sürdürmektedir. 1957'de Amerikalı dilbilimci Noam Chomsky, *Syntactic Structure* [Sözdizim Yapısı] başlığıyla çığır açıcı bir kitap yayımladı; bu kitapta, bir dilde "geçerli" cümleler kurmak üzere kullanılabilecek bir dizi dilbilgisi kuralı öne sürer.[2] Aynı kurallardan, bir sözcük zincirinin dilde geçerli cümle oluşturup oluşturmadığını çözümlemek amacıyla da yararlanılabilir. Chomsky'nin "öbek yapısı dilbilgisi" (ÖYD) [*phrase-structure grammar*] dediği kavramı kullanan bu çözümlemenin nasıl yapıldığını göstereceğim.[3] Bu işlem, ta ilkokulda cümleleri "öğelerine ayrıştırmamıza" çok benziyor.

Dilbilgileri, dizilerdeki sözcükleri, isim, sıfat, yüklem gibi sözdizimi kategorilerine ait simgelerle yer değiştiren kuralları belirleyerek

1 İngilizcede "bank" sözcüğünü hem banka hem de ırmak kıyısı anlamına geldiği için —*çev. notu.*

2 Noam Chomsky, *Syntactic Structures*, 's-Gravenhage: Mouton & Co., 1957.

3 TYD'nin temel yapısı, bilgisayar bilimcisi John Backus tarafından, ALGOL programlama dilinin sözdizimini betimlemek amacıyla bağımsız olarak icat edilmiştir. Bkz. John Backus, "The Syntax and Semantics of the Proposed International Algebraic Language of the Zürich ACM-GAMM Conference", *Proceedings on the International Conference on Information Processing*, UNESCO, 1959, s. 125-132.

tanımlanır. Aynı zamanda dilbilgileri, bu sözdizimi simgesi zincirlerinin yerine başka simgeleri geçirecek kurallar da barındırır. Bu fikirleri açıklamak için, Chomsky'nin örneklerinden uyarlanmış basit bir dilbilgisi kullanacağım. Bu dilbilgisinde yalnızca üç sözdizimi kategorisi var: niteleyici, isim, yüklem. Bu üçü, *the man hit the ball* (adam topa vurdu) gibi cümleleri çözümlemek için yeterli.

Bu açıklayıcı dilbilgisinde kurallardan biri, "o" ya da "bir" sözcüklerinin yerine "NİT" (niteleyici) simgesini geçirebileceğimizi söylüyor. Dilbilimciler bu kuralı aşağıdaki gibi yazar:

the | a →NİT

(| simgesi, çevresindeki *iki* sözcüğünün yerine de okun sağındaki sözdizimi simgesinin geçebileceğine işaret ediyor.)

Aynı formatta yazılmış başka kurallar da var:

man | ball | john → İ
("man", "ball" ve "john" sözcüklerinin yerine, isim anlamında "İ" simgesi geçebilir.)

hit | took | threw → Y
("hit", "took" ve "threw" sözcüklerinin yerine, yüklem anlamında "Y" simgesi geçebilir.)

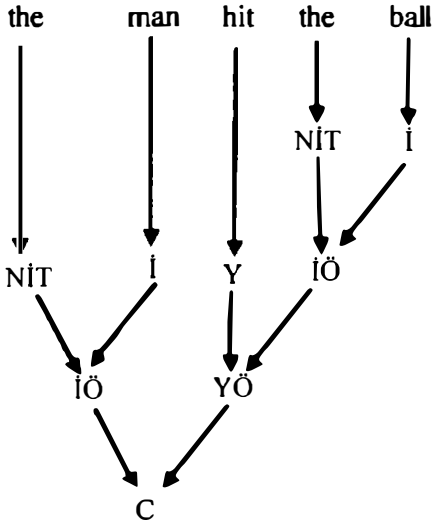
NİT İ → İÖ
("NİT" ve "İ" simge zincirlerinin yerini, isim öbeği anlamında "İÖ" simgesi alabilir.)

Y İÖ → YÖ
("Y" ve "İÖ" simge zincirlerinin yerini, yüklem öbeği anlamında "YÖ" simgesi alabilir.)

İÖ YÖ → C
("İÖ" ve "YÖ" simge zincirlerinin yerini, cümle anlamında "C" simgesi alabilir.)

"C", "NİT", "İÖ" gibi simgelere, dilbilgisinin tanımladığı, "uç olmayan" (*nonterminal*) simgeler denir; oysa "ball", "john", "threw" gibi dağarcık sözcükleri, dilin "uç" (*terminal*) simgeleridir.

Bu kuralları, *the man hit the ball* zincirine uygulayıp bunu "C"ye dönüştürebiliriz. Bu şekilde "C"ye dönüştürülebilen her sözcük zincirinin dilbilgisine uygun olduğu söylenir; bu çok basit dilbilgisinin



Resim 7.1. Cümleyi çözümleyen ağaç.

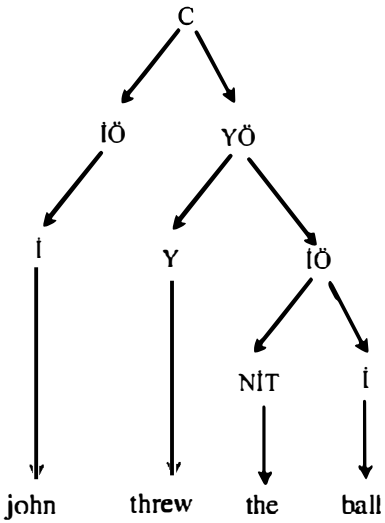
tanımladığı geçerli bir cümledir o. Kural uygulamalarını göstermenin yollarından biri, *ayrıştırma ağacıdır* (Resim 7.1).[4]

Bu örnek, sözdizimi çözümlemesi hakkında başlıca fikirleri anlatırken, küçük bir sözdizimi kategorisi ve yerdeğiştirme kuralları kümesini temel alır. Dilbilgisini biraz daha gerçekçi kılmak amacıyla sıfatlar, zarflar, edatlar vesaire için simgeler ve yerdeğiştirme kurallarını içermesi gerekiyor. Üstelik, çok daha fazla sözcük içermeli elbette.

Eğer tüm kuralları, okun sağ tarafında sadece tek bir "uç olmayan" simge barındırıyorsa, o dilbilgisine *bağlamdan bağımsız dilbilgisi* (BBD) [*context-free grammars*] denir. Böyle denmesinin nedeni, kurallar ters sırayla kullanıldığında (dilbilgisine uygun cümleleri çözümlemekten çok kurmak amacıyla), uç olmayan simgenin yerdeğiştirme şeklinin, başka simgelerin varlığına bel bağlamayışıdır. Dolayısıyla ÖYD'ler bağlamdan bağımsızdır.

Resim 7.2'deki diyagram, anlattığımız bu basit dilbilgisi kurallarıyla nasıl cümle kurulacağını gösteriyor. Bu örnekte, "C" simgesiyle yani cümle yerine geçen simgeyle başlanıp, *John threw the ball* cümlesi kuruluyor.

4 George Boeree'ye göre (bkz. http://www.ship.edu/~cgboeree/wundtjames.html), Wilhelm Wundt, "dilbilim metinlerinden hepimizin aşina olduğu sözdizimi ağacı diyagramını icat etmiştir."



Resim 7.2. Cümle kurmaya yarayan ayrıştırma ağacı.

Bu basit dilbilgisinin, geçerli ya da kabul edilebilir olduğunu ileri sürebileceğimiz tüm cümleleri oluşturması kesinlikle mümkün değil. Aynı zamanda, kabul etmek istemeyeceğimiz cümleler de yaratır, örneğin *the john threw the ball* gibi. Chomsky'nin kitabı çok daha karmaşık dilbilgileri sunuyor ve sonraki çalışmalar daha da gelişmiş olanlarını üretmiştir. 1960'ların ilk yarısında bilgisayar programlarına kodlanmış kimi dilbilgileri, İngilizce metinleri çözümleyebiliyordu.[5] Önümüzdeki bölümlerde farklı dilbilgilerinden bahsetmeyi sürdürüyorum ve bunlardan bazıları BBD'den daha karmaşık olacak. Bununla birlikte, en karmaşık dilbilgisi bile, dilbilgisine uygun saydığımız ve uygun saymadığımız cümleleri açıkça ayırt edemez. İlerideki bir bölümde, bu zorluğa ve bununla başa çıkmanın bir yoluna değiniyorum.

Bir cümlenin dilbilgisi tarafından çözümlenme şekli, bu cümlenin anlamını belirleyebilir, dolayısıyla doğal dil işlemenin önemli bir parçası, cümleler için kabul edilebilir ayrıştırma ağaçları bulacak dilbilgisi kuralları kullanmayı içerir. Ayrıştırma ağacı bulmak, bir arama süreci gerektirir; ya "C"den itibaren, hedef cümleyle eşleştirme amacıyla dilbilgisi kuralları kullanarak uç olmayan simgelerin yerini başka simgelerin alışının farklı yollarını ya da "C" simgesi yaratmak

5 Bu dönem boyunca yapılan çalışmalarla ilgili bir etüt için bkz. Daniel Bobrow, "Syntactic Analysis of English by Computer: A Survey", *Proceedings of the 1963 Fall Joint Computer Conference*, c. 24, Baltimore: Spartan Books, 1963, s. 365-387.

amacıyla, hedef cümledeki sözcüklerin yerini uç olmayan simgelerin alışının farklı yollarını aramak söz konusudur. Bu arama türlerinden ilkine "yukarıdan-aşağı" arama ("C"den cümleye); ikincisine "aşağıdan-yukarı" arama (cümleden "C"ye) denir.

Belirli bir dilbilgisinde sık sık (belki de çoğunlukla), cümlelerin, her biri farklı anlama sahip birden fazla ayrıştırma ağacı olabilir. Örneğin, *the man hit the ball in the park* cümlesinin, *in the park* sözcüğünün *hit* yüklemiyle kurulan yüklem öbeğinin parçası olduğu bir ayrıştırma ağacı ve yine *in the park* sözcüğünün *ball* ismiyle kurulan isim öbeğinin parçası olduğu başka bir ayrıştırma ağacı vardır. Dahası, daha önce bahsettiğim gibi, cümlelerin kimi çözümlemeleri anlamsız olabilir. Örneğin, yukarıda bahsettiğim basit dilbilgisine göre, *the ball threw the man* cümlesi kurallara uygun ama büyük olasılıkla anlamsız bir cümledir. Hangi ayrıştırma ağacının uygun olduğuna karar vermek, anlam hakkında karar verme sürecinin parçasıdır, ayrıca anlambilim (hatta olasılıkla edimbilim) düzeyinin işidir bu. 1950'lerin ikinci yarısından itibaren ve 1960'ların büyük kısmı boyunca ve ötesinde sözdizimi çözümlemesi, anlambilimden daha gelişmişti.

Sözdizimsel çözümleme çoğunlukla, girdi cümlesinin bir tür iyi tanımlanmış "anlam gösterim dilinde"ki bir ifadeye ya da girdi cümlesine uygun şekilde tepki veren bir programa dönüştürülmesini yönlendirmek amacıyla ayrıştırma ağacı kullanmayı içerir. Örneğin, *the man threw the ball* cümlesi, şöyle bir mantık ifadesine çevrilebilir:

$$(\exists x, y, z)[\text{Geçmiş}(z) \wedge \text{MAN}(x, z) \wedge \text{Ball}(y, z) \wedge \text{Olay}(z) \wedge \text{Throws}(x, y, z)],$$

bu ifade şöyle yorumlanabilir: "öyle **x**, y ve z vardır ki z geçmişte olmuş bir olay (*event*), **x** bu olayı yaşamış kişi, y de o olaydaki toptur (*ball*) ve o olayda **x**, y'yi fırlatmıştır (*throws*)."

Anlamsal çözümleme aynı zamanda, *the man threw the ball* cümlesini, bir şekilde, geçmişte top fırlatmış bir adamın benzetimini yapan bir programa dönüştürür.

7.2 YAPAY ÇEVİRİ

Bilgisayarları sıradan sayısal hesaplamalardan fazlası için kullanmaya yönelik ilk girişimlerden bazıları, dilden dile otomatik cümle çevirmeyi içeriyordu. Bilgisayar belleğinde kelime sözlükleri saklanabiliyordu (ya bantlarda ya da delikli kartlarda) ve bu sözlükler, yabancı sözcüklerin İngilizce karşılıklarını bulmak için kullanılabiliyordu.

Her yabancı sözcüğün cümledeki uygun karşılığını seçerek, biraz da sözdizimi çözümlemesiyle birlikte, yabancı dildeki (örneğin Rusça) bir cümleyi İngilizceye çevirmenin mümkün olduğu düşünülüyordu.

New York Times, Ulusal Standartar Bürosu'nda (günümüzde Ulusal Standartlar ve Teknoloji Enstitüsü deniyor) Harry D. Huskey'nin yönettiği bir ekibin geliştirmiş olduğu yeni bilgisayarı[6] sayfalarına taşıyıp, 31 Mayıs 1946 tarihli sayısında şunu bildirmişti:[7]

> Karmaşık matematik problemlerini çözmekle kalmayıp yabancı dilden çeviri bile yapacak yeni bir "elektrik beyinli" hesaplama makinesi tipi, California Üniversitesi Sayısal Çözümleme Enstitüsü'ndeki ABD Standartlar Bürosu Laboratuvarı'nda yapım aşamasında. Bu makinenin, çeviri alanındaki tam kapsamına karar verilmiş olmasa da, üzerinde çalışan bilimcilere göre bu bilgisayar, Webster Collegiate Sözlüğü'nün 60.000 sözcüğünü ve her sözcüğün üç dilde karşılığını kapsayabilecek.

Makinenin çeviriyi nasıl yapacağını açıklayan *Times* muhabiri şöyle yazıyor:

> Makineye çevirmesi için yabancı bir sözcük, bant ya da kart üzerinde elektro-matematiksel simge biçiminde sunulduğunda, makine bu sözcüğü "belleğinde" tarayacak ve eğer o simgenin bir kaydını bulursa, önceden belirlenmiş karşılığı, yani İngilizce sözcüğü otomatik bir şekilde çıktı olarak verecek.
> ...
> Elbette bu, sözdiziminden yoksun kelimesi kelimesine çeviriye denk geliyor; fakat tasarımcalara göre yine de son derece değerli olacaktır, çünkü yabancı teknik makaleleri bilimcilerin çevirmesi gibi işlerde, sözcük dağarcığı, sözdiziminden çok daha büyük bir sorundur.

Bu makine gerçekte herhangi bir çeviri yapmış değildi; çeviri yapma fikri, Huskey'nin gözünde canlandırdığı bir olasılıktı sadece. Fakat bilimci olmayanlar bile bu işin zorluklarını gözünde canlandırabilir. Ertesi gün *New York Times*'da çıkmış bir köşe yazısı sorunu gayet güzel ortaya koyuyordu:

> Her çevirinin doğruluğu hakkında bizim bile kuşkularımız oluyor. Makine, Fransızca "pont" sözcüğünü "köprü" mü yoksa "güverte" mi diye çe-

6 The Standards Western Automatic Computer (ileride SWAC olarak kısaltılacaktı)

7 John Hutchins, "From First Conception to First Demonstration: The Nascent Years of Machine Translation, 1947-1954. A chronology", *Machine Translation*, c. 12, no: 3, 1997, s. 195-252. (Minik eklemelerle düzeltilmiş 2005 versiyonu: http://www.hutchinsweb.me.uk/MTJ-1997-corr.pdf.)

> vireceğine nasıl karar verecek, ya da Almancada "operation" sözcüğünün cerrahi işlem anlamına geldiğini nasıl bilecek? Makinenin yapıp yapabileceği, sözlükte sözcüklere bakıp bunların İngilizce karşılıklarını bulma işini kolaylaştırmaktır, böylece çevirmen, uygun cümleleri kurabilir ve sözcüklere bağlamlarına uygun düşen anlamı verebilir.

Matematikçi ve bilimsel proje yöneticisi Warren Weaver, 1947'de Norbert Wiener'e yazdığı bir mektupta, doğal insan dilleri arasında belge çevirme amacıyla sayısal bilgisayar kullanma olasılığından bahsetmişti. Weiner, bu olasılığın gerçekleşeceğinden şüphe ediyordu. Weaver'e yanıtında Wiener şunu yazmıştır: "Dürüst olmak gerekirse, farklı dillere ait sözcüklerin arasındaki sınırın, matematiğe benzeyen herhangi bir çeviri şeması oluşturma konusunda umutlarımızı kıracak kadar belirsiz, duygusal ve uluslararası çağrışımların yine umutlarımızı kıracak kadar kapsamlı olmasından korkuyorum". Bununla birlikte Temmuz 1949'da Weaver, fikirlerini "Çeviri" başlıklı bir bildiriyle ifade edip kimi meslektaşlarına göndermişti.

Weaver, bildirisine şu sözlerle başlar:

> Dillerin çeşitliliğinin, dünya halkları arasındaki kültürel değiştokuşu zorlaştırdığı ve uluslararası uzlaşıya caydırıcı bir etkisi olduğundan uzun uzadıya bahsetmeye gerek yok. Bu gerçeğin geçerli ve önemli olduğunu varsayan bu bildiri, dünya çapındaki çeviri sorununun çözümüne, yüksek kapasitesi, esnekliği, hızı olan elektronik bilgisayarların kullanımı sayesinde katkıda bulunma olasılığıyla ilişkili yorumlar ve öneriler içeriyor.

Söz konusu bildirinin basılmış olduğu kitabı[8] hazırlayanlara göre, "Weaver bu bildiriyi çeşitli alanlardaki 200 tanıdığına gönderdiğinde bu insanların çoğu, bilgisayar teknikleriyle dil çevirisinin mümkün olabileceğini söyleyen bir öneriyi gerçekten ilk kez görmüş oldu". Weaver'in belgesinin genelde, yapay çeviri (YÇ olarak kısaltılır) alanını başlattığı söylenir.[9]

Haziran 1952'de MIT'de, o zamanlar MIT Elektronik Araştırma Laboratuvarı'nın mantıkçılarından olan Yehoshua Bar-Hillel (1915-

8 W. N. Locke ve A. D. Booth (ed.), *Machine Translation of Languages: Fourteen Essays*, Cambridge, MA.: MIT Press, 1955, s. 15-23.

9 MT'nin tarihi için bkz. W. John Hutchins, "Machine Translation: A Brief History", *Concise History of the Language Sciences: From the Sumerians to the Cognitivists* içinde, E. F. K. Koerner ve R. E. Asher (ed.), Oxford: Permagon Press, 1995, s. 431-445. (Ayrıca bkz. http://www.hutchinsweb.me.uk/ConcHistoryLangSci-1995.pdf.) Hutchins, kendi yayınlarına bir internet sitesi ayırmıştır: http://www.hutchinsweb.me.uk/.

1975), salt yapay çeviriyi ele alan ilk konferansı düzenlemişti.[10] Bar-Hillel, en başta iyimser olsa da, daha sonraları tam otomatik çevirinin imkânsız olduğu sonucuna varacaktı.

Ocak 1954'te, New York City 57. Cadde, Madison Bulvarı adresindeki IBM Dünya Merkezi'nde Rusça metinlerin İngilizceye çevirisiyle ilgili bir gösterim yapıldı. Dar bir sözcük dağarcığından ve kısıtlı bir dilbilgisinden faydalanan bu gösterim, IBM ile Georgetown Üniversitesi'nin işbirliğinin sonucuydu. Projeye, IBM Uygulamalı Bilimler Dairesinin müdürü Cuthbert Hurd ile Georgetown'dan Léon Dorset başkanlık ediyordu. IBM'in 8 Ocak 1954 tarihli basın açıklamasına göre:[11]

> Rusça bugün İngilizceye ilk kez bir elektronik "beyin" tarafından çevrildi.
>
> Georgetown Üniversitesi Diller ve Dilbilim Enstitüsü'nün dilbilimcileri, siyaset, matematik, kimya, metalurji, iletişim ve askeriyeyle ilgili kısa Rusça ifadeleri IBM'in meşhur 701 bilgisayarına gönderdi. Bu devasa bilgisayar, o cümleleri birkaç saniye içinde rahat okunur bir İngilizceye çevirdi.
>
> Sovyetlerin dilinden tek sözcük anlamayan bir kız Rusça mesajları IBM kartlarına işledi. "Beyin", İngilizce çevirileri saniyede iki buçuk satır gibi başdöndürücü bir hızla otomatik yazıcısında bastı.
>
> Örneğin kız, karta, "Mi pyeryadem mislyi posryedstrom ryechyi", ifadesini işlediğinde, 701 şu yanıtı verdi: "Düşünceleri konuşma aracılığıyla naklediyoruz."
>
> "Vyelyichyina ugla opryedyelyayetsya otnoshyenyiyem dlyini dugi k radyiusu" ifadesini "beyin" şöyle çevirdi: "Açının büyüklüğü, yay uzunluğunun yarıçapa oranıyla belirlenir."

Her ne kadar bu gösterim, büyük heyecan doğurup çeviri araştırmalarına ayrılan ödenekleri artırmış olsa da, bu alandaki sonraki çalışmalar hayal kırıcı olmuştur.[12] Araştırmacılar arasında dolanan 1959 tarihli bir raporda YÇ çalışmalarını değerlendiren Bar-Hillel, tam oto-

10 Bu konferansla ilgili raporlar için bkz. E. Reifler, "The First Conference on Mechanical Translation", *Mechanical Translation*, c. 1, no: 2, 1954, s. 23-32 ve A. C. Reynolds, "The Conference on Mechanical Translation Held at MIT, Haziran 17-20, 1952", *Mechanical Translation*, c. 1, no: 3, 1954, s. 47-55.

11 http://www-03.ibm.com/ibm/history/exhibits/701/701_translator.html.

12 IBM-Georgetown çalışmasının özeti için bkz. W. John Hutchins, "The Georgetown-IBM Experiment Demonstrated in January 1954", *Proceedings of Machine Translation: From Real Users to Research* içinde, Robert E. Frederking ve Kathryn B. Taylor (ed.), 6th Conference of the Association for Machine Translation in the Americas, AMTA-2004, Washington DC, USA, 28 Eylül-2 Ekim, 2004, Berlin: Springer, 2004, s. 102-114. Çevrimiçi versiyonu için: http://www.hutchinsweb.me.uk/ATMA-2004.pdf.

matik ve yüksek nitelikli çevirinin (buna kısaca İgilizce *Fully Automatic High-Quality Translation*'ın başharflerinden FAHQT diyordu) "ne yakın gelecekte ne de hiçbir zaman" uygulanamayacağına ikna olmuştu. Genişletilmiş raporu, yaygınlıkla okunan 1960 tarihli bir makalede basılmıştır.[13]

Bar-Hillel'i bu olumsuz sonuca götüren etkenlerden biri, bilgisayarlara yüksek nitelikli çeviri için gereksinim duyacakları "dünya bilgisini "sağlamanın anlaşılır zorluğuydu. Bu sorunu şu örnekle açıklamıştı:

> Küçük John, oyuncak kutusunu arıyordu. Sonunda kutuyu buldu. Kutu, oyun parkının (*pen*) içindeydi. John çok mutlu olmuştu.

"The box was in the pen" cümlesini nasıl çevirmeli? Bar-Hillel, İngilizcede "pen" sözcüğünün yalnızca iki anlamı olsa bile (kalem ve portatif bebek oyun parkı), yalnızca o anlamları bilen bir bilgisayarın hangi anlamın kastedildiğine karar vermesinin mümkün olmadığını ileri sürmüştü. Çeviri bilgisayarı, sözcük dağarcağı ve sözdizimi bilgisine ek olarak, "kalem anlamındaki *pen*'in, oyuncak kutusunun ve oyun parkı anlamındaki *pen*'in göreli boyutlarını" bilmeliydi. Bar-Hillel, böyle bir bilgiye elektronik bilgisayarın ulaşamayacağını ileri sürmüştü. Bilgisayarı bu tür ansiklopedi bilgileriyle doldurmanın, "bütünüyle saçma ve üzerinde konuşulmayı hak etmeyen" bir düşünce olduğunu söylemişti.

Sonraki araştırmacıların da kabul ettiği gibi, Bar-Hillel, epey yetkin doğal dil işleme sistemlerinin (doğrusu, genelde oldukça yetkin YZ sistemlerinin), ansiklopedi bilgisine gereksinim duyacağını iddia ederken haklıydı. Bununla birlikte, çoğu YZ araştırmacısı, bilgisayarlara gerekli ansiklopedi bilgisini vermenin boş iş olduğu konusunda ona katılmaz. Bar-Hillel, yapay zekâya kötümser bakmasıyla tanınıyordu. (1958 Teddington Konferansında John McCarthy'nin "Sağduyulu Programlar" makalesi hakkında yorum yapan Bar-Hillel, "Dr. McCarthy'nin makalesi, kurulmasını kısa süre önce Dr. I. J. Good'un teklif ettiği Yarı Pişmiş Fikirler Dergisine yaraşır", demişti.)[14]

Nisan 1964'te Ulusal Bilimler Akademisi, Automatic Language Processing Advisory Committee'yi (ALPAC; Otomatik Dil İşleme Danışma Komitesi) kurdu ve komite başkanlığına Bell Laboratuvarları'ndan John R. Pierce (1910-2002) getirildi; komitenin amacı, "yabancı dille-

13 Yehoshua Bar-Hillel, "The Present Status of Automatic Translation of Languages", *Advances in Computers*, c. 1, no: 1, 1960, s. 91-163.

14 D. V. Blake ve A. M. Uttley (ed.), *Proceedings of the Symposium on Mechanisation of Thought Processes*, Londra: Her Majesty's Stationary Office, 1959, s. 85.

rin yapay çevirisi alanındaki araştırma geliştirme çabaları hakkında Savunma Bakanlığı'na, Merkezî İstihbarat Teşkilatı'na (CIA) ve Ulusal Bilim Vakfı'na (NSF) danışmanlık yapmaktı". Komite raporunu 1965'te yayımladı ve ileri sürdüğü kimi görüşlerin yanı sıra, "işe yarar bir yapay çeviri için acil ya da tahmin edilebilir bir beklenti olmadığı" sonucuna da vardı.[15] Temel dilbilime ve çeviriye "yardımcı" olacak öğelere destek verilmesini tavsiye etmiş, ama tam otomatik çeviriye artık destek olunmamasını önermişlerdi. Bu rapor, yapay çeviriye ayrılan kapsamlı ödeneklerin küçülmesine yol açmıştır. Bununla birlikte, ilerideki bölümlerde göreceğimiz üzere, yapay çeviri ayakta kalıp sonunda palazlanmıştır.

Association for Machine Translation and Computational Linguistics (AMTCL; Yapay Çeviri ve Hesaplamalı Dilbilim Derneği), ilk toplantısını 1962'de düzenlenmiştir. 1968'de adını Association for Computational Linguistics (ACL) (Hesaplamalı Dilbilim Derneği) olarak değiştirdi ve doğal dil ile bilişimi ilgilendiren sorunlar üzerinde çalışanlar için uluslararası bir bilim ve meslek derneği haline geldi. Bu dernek, üç ayda bir çıkan *Computational Linguistics* dergisini yayımlar, ayrıca konferanslar ve çalıştaylar düzenler.[16]

7.3 SORU YANITLAMA

Yapay çeviri çalışmalarına ek olarak araştırmacılar, İngilizce gibi doğal bir dildeki cümlelerin bilgisayarlarla iletişim kurmak üzere nasıl kullanılabileceğini incelemeye koyulmuştu. Weizenbaum'un ELIZA programını hatırlarsınız; bu program, ne söylendiğini hiç "anlamasa" bile insanlarla sohbet edebiliyordu. Ayrıca, Raphael'in, kendisine verilen bilgiyi belleğinde uygun bir gösterimle saklayıp soruları yanıtlayabilen SIR sisteminden zaten bahsetmiştim.

Bu dönemde gerçekleştirilmiş doğal dil işleme çalışmalarını size bir parça tattırmak amacıyla başka projelerden de bahsedeceğim. BASEBALL adlı bir program (Newell, Shaw ve Simon tarafından geliştirilmiş özel bir liste işleme program dili olan IPL-V'le yazılmıştı; bundan ileride bahsedeceğim), Carnegie Teknoloji Enstitüsü profesörlerinden

15 John R. Pierce ve diğ., *Language and Machines: Computers in Translation and Linguistics*, ALPAC Report, National Academy of Sciences Publication 416, National Research Council, Washington, DC, 1966.

16 Bkz. http://www.aclweb.org/.

Bert Green'in yönetiminde Lincoln Laboratuvarı'nda geliştirilmişti.[17] Bu program, tek yıl içinde Amerikan liginde oynanmış beyzbol maçlarıyla ilgili veri tabanından faydalanarak beyzbolla ilgili basit İngilizce soruları yanıtlayabiliyordu. Örneğin, "Red Sox takımı 7 Temmuzda nerede oynadı?" gibi soruları cevaplayabilirdi. Sorular öncelikle basitçe sorulmalı ve programın dağarcığındaki sözcüklerle sınırlı olmalıydı. Programın yaratıcısına kulak verelim:[18]

> Sorular tek cümleyi aşmamalı; çoklu cümlecik yapılarının yasak oluşu, sözdizim çözümlemesini epey basitleştiriyor. *Ve, veya, değil mi ki* gibi mantıksal bağlaçlar ve *en fazla, en yüksek* gibi ilinti imleyen çatılar yasaktır. Son olarak, "Red Sox hiç altı maç *üst üste* kazanmış mıdır?" gibi sıralı olguları içeren sorular da yasaktır.

Bu program, soruları, hem özel amaçlı sözdizim çözümlemelerini hem de anlamsal çözümlemeleri kullanarak, "belirtim listesi" denen özel bir biçime dönüştürerek işler. Ardından bu liste, soruya yanıt vermek üzere programın veri bankasına ulaşmak amacıyla kullanılır. Örneğin "Red Sox takımı 7 Temmuzda nerede oynadı?" sorusu önce bu listeye çevrilir:

Yer = ?
Takım = Red Sox
Ay = Temmuz
Gün = 7

Programın yaratıcıları, "koydukları kısıtlamaların, programın ileriki sürümlerinde kaldırılacak geçici önlemler olduğunu" ileri sürmüşlerdi. Bildiğim kadarıyla, programın başka sürümü hiç olmamıştır. (Yazdığım bu YZ tarihçesi ilerledikçe göreceğimiz gibi, "geçici" kısıtlamaları kaldırmanın çok zor olduğunun anlaşıldığı kimi vakalar yaşanmıştır.)

Başka bir doğal dil programı SAD SAM, 1962-1963'te Carnegie Teknoloji Enstitüsü'nde Robert Lindsay tarafından IPL-V diliyle yazılmıştı.[19] Bu program, akraba ilişkileriyle ilgili İngilizce cümleleri çö-

17 Bert F. Green Jr., Alice K. Wolf, Carol Chomsky ve Kenneth Laughery, "BASEBALL: An Automatic Question Answerer", s. 219-224, *Proceedings of the Western Joint Computer Conference*, Mayıs 1961. Yeniden basıldığı yer: Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, s. 207-216, New York: McGraw-Hill, 1963 ve B. Grosz, K. Spark Jones, B. Lynn Webber (ed.), *Readings in Natural Language Processing*, Morgan Kaufman, Los Altos, CA, 1986.

18 *A.g.e.*

19 Bkz. Robert K. Lindsay, "Inferential Memory as the Basis of Machines Which Understand Natural Language", *Computers and Thought* içinde, Edward A. Feigen-

zümleyip bu ilişkileri aile ağacına yerleştirebiliyordu. Ardından bu ağacı kullanarak, akrabalık ilişkileriyle ilgili İngilizce soruları yanıtlayabiliyordu.

Örneğin, eğer SAD SAM, "Joe ve Jane, Tom'un evlatlarıdır" cümlesini aldığında, Tom'un baba ve Joe ile Jane'in çocuklar olduğu belirli bir "aile birimi" için ağaç benzeri bir liste yapısı oluşturur. Ardından, "Mary, Jane'in annesidir" cümlesini aldığında, bu yapıya Mary'yi, Tom'un eşi olarak ekler. Böylece, "Joe'nun annesi kimdir?" sorusunu yanıtlayabilecek hale gelir.

SAD SAM, Sentence Appraiser and Diagrammer and Semantic Analyzing Machine'in (Cümle Değerlendiricisi ve Diyagramcısı ve Anlamsal Çözümleme Makinesi) baş harflerinden oluşan bir addır. SAD kısmı, girdi cümlelerini çözümleyip bunları SAM'e aktarıyordu; bu da aile ağaçları kurmak ve sorulara yanıt bulmak için gerekli olan anlamsal bilgiyi çekip çıkarıyordu. Bu program, Temel İngilizce olmak kaydıyla çeşit çeşit cümleleri kabul edebiliyordu; burada bahsedilen Temel İngilizce, Charles K. Ogden'ın tanımlamış olduğu, yaklaşık 850 sözcükten oluşan bir sözcük dağarcığı ve dilbilgisi sistemidir.[20]

California Santa Monica'da Systems Development Corporation'da (SDS) psikolog ve dilbilimci olan Robert F. Simmons (1935-1994), kendi doğal dil işleme çalışmaları için daha görkemli hedefler benimsemişti. Simmons'ın Bilgisayar Bilimleri ve Psikoloji Profesörü olarak görev yaptığı Austin'deki Texas Üniversitesi'nde doktora öğrencilerinden Gordon Novak'ın kaleme aldığı "Anma Yazısı"na göre:[21]

> Simmons'ın hayali, "kitap sohbeti yapabilen" bilgisayarlardı; bilgisayar, kitabı okuyacak, ardından kullanıcı bilgisayarla sohbet edip bilgisayarın kitaptan anladıklarıyla yanıtlamasını istediği sorular soracaktı.

Bu "hayali" gerçekleştirmenin, YZ'nin kendisi kadar zor iş olduğu anlaşılacaktı. Teklif ettiği "Synthex" projesiyle ilgili 1961 tarihli notta Simon işe nasıl başlayacağını betimler:[22]

baum ve Julian Feldman, s. 217-233.

20 Charles K. Ogden, *Basic English: A General Introuction with Rules and Grammar*, 4. baskı, Londra: Kegan, Paul, Trench, Trubner & Co., Ltd., 1933. (Lindsay, 1700 sözcük derken, başka kaynaklar 850 sözcük der.)

21 Bkz. http://www.cs.utexas.edu/users/ai-lab/simmons.html.

22 Robert F. Simmons, "Synthex", *Communications of the ACM*, c. 4, no: 3, Mart 1961, s. 140.

> Bu projenin amacı, insanoğlunun karmaşık bilişsel işlevlerini sentezleyecek çok amaçlı bir bilgisayar sisteminin tasarımı ve inşası için hem bir araştırma yöntemi hem de bir araç geliştirmektir. İlk araç, *proto-synthex*, basit basılı yazıları okuyup temel İngilizceyle sorulan basit soruları yanıtlayan temel bir dil işleme cihazı olacak.

1965'e gelindiğinde, Simmons ile Lauren Doyle, Protosynthex sistemleriyle bazı deneyler yürütmüşlerdi. Trudi Bellardo Hahn'ın raporuna göre:[23] "Belli bir çocuk ansiklopedisinden (*Golden Book*) alınmış bölümlerin küçük bir tam metin veritabanı sisteme yüklenmişti. Protosynthex, basit İngilizce sorulara 'yanıt' verebiliyordu... metin erişimi konusunda doğal dil kullanma çabalarının öncüsüydü bu."

Bu esnada, MIT'de Marvin Minsky'nin doktora öğrencilerinden Daniel G. Bobrow (1935-2017), STUDENT sistemi denen bir dizi program yazmıştı; bu sistem, kısıtlı bir İngilizce altkümesiyle sorulan "öykümsü" cebir problemlerini çözebiliyordu. STUDENT'in çözebildiği bir problem örneği şöyledir:

> New York ile Los Angeles arası 3.000 mildir. Eğer jet uçağının ortalama hızı saatte 600 milse, New York'tan Los Angeles'a jetle gitme süresini bul.

STUDENT, bilinen hız ve uzaklık bağıntılarını kullanıp, uygun denklemleri kurup çözerek problemi yanıtlıyordu. Bobrow'un tezinde, STUDENT'in çözebildiği başka problem örnekleri ve kullandığı yöntemler de sunulmuştur.[24]

23 Trudi Bellardo Hahn, "Text Retrieval Online: Historical Perspective on Web Search Engines", *ASIS Bulletin*, Nisan/Mayıs 1998. Şu internet sitesinde: http://www.asis.org/Bulletin/Apr-98/hahn.html.

24 Daniel G. Bobrow, "Natural Language Input for a Computer Problem Solving System", MIT Artificial Intelligent Project Memo 66, Memorandum MAC-M-148, 30 Mart 1964. Şu internet sitesinde: http://dspace.mit.edu/bitstream/handle/1721.1/5922/AIM-066.pdf?sequence=2. Bu tezi temel alan bir makale için bkz. Marvin Minsky (ed.), *Semantic Information Processing*, 3. Bölüm, Cambridge, MA: MIT Press, 1968.

8
1960'LARIN ALTYAPISI

1960'lar boyunca gerçekleşmiş teknik gelişmeleri birtakım sistem destekleri ve toplumsal etkenler desteklemişti (aslında, mümkün kılmıştı denebilir). Yeni bilgisayar dilleri, YZ sistemleri kurmayı kolaylaştırmıştı. Matematikten, bilişsel bilimden, dilbilimden ve yakın gelecekte "bilgisayar bilimi" olarak adlandırılacak alandan gelen araştırmacılar, zeki davranışları makineleştirme sorunuyla boğuşmak amacıyla toplantılarda ve yeni kurulmuş laboratuvarlarda bir araya geliyordu. Buna ek olarak, bu yeni girişimde önemli çıkarları olduğunu düşünen devlet kurumları ve şirketler, gerekli araştırma desteğini sağlıyordu.

8.1 PROGRAMLAMA DİLLERİ

Newell ve Simon, zekâyı makineleştirme yaklaşımlarının tam kalbinde yer alan simgesel ifadeleri işlemek üzere uzmanlaşmış bir bilgisayar dilinin faydalı olacağını fark eden ilk kişiler arasındaydı. Simgesel ifadenin en temel türü, (7, B, 5) gibi bir simge listesidir. Simge listelerinin listesi ve listelerin listesinin listesi oluşturularak daha karmaşık yapılar da kurulabilir.

Sekizli bulmacanın simge yapılarıyla ilgili betimlememde, gerekli olan işleme türlerinden bahsetmiştim. Sekizli bulmacanın başlangıç konumunun şu ifadeyle gösterildiğini anımsayın:

((2, 8, 3) (1, 6, 4) (7, B, 5)).

Gerek duyulan şey, bulmacadaki hamlelere denk gelen konumları temsil edecek ifadeleri üretebilen programlar yazacak bir dildi. Örneğin, başlangıç konumundan itibaren yapılabilecek hamlelerden biri, şu ifadeyle temsil edilir:

((2, 8, 3) (1, 6, 4) (B, 7, 5))

Program, bu gösterimi işlemek için, başlangıç konumu gösterimini kopyalayıp, bu gösterimdeki üçüncü listenin birinci ve ikinci öğelerinin yerini değiştirmeli.

Newell, Shaw ve Simon, bu tür işlemelerin programlanabileceği bir dil geliştirmeye koyulmuşlardı. 1954 civarında RAND şirketinde işe koyularak, IPL (*information processing language* [bilgi işleme dili]) denen bir dizi dil yarattılar. Bu dilin birkaç sürümü üretildi. IPL-I gerçekleşmemiş, sadece bir tasarım tarifnamesi olarak hizmet etmiştir. IPL-II, RAND şirketinin JOHNNIAC Bilgisayarında 1955'te yürürlüğe sokuldu. Sonraki sürümleriyse (IPL-VI'ya kadar) Carnegie Tech'de gerçeklendi.

IPL dilleri, ilk YZ programlarından bazılarının programlanmasında kullanılmıştır; bunlar arasında LT (Logic Theorist [Mantık Kuramcısı]), GPÇ (Genel Problem Çözücü), NSS (Newell, Shaw, Simon satranç programı), ayrıca Quillian ve George Ernst gibi Newell'in ve Simon'un öğrencilerinin yazdığı programlar var. Dartmouth yaz projesinin ardından John McCarthy de liste işleme dillerinin kullanımı hakkında kafa yormaya başlamıştı. Gelernter'in geometri teoremi ispatlayan makinesinde FLP (liste işleme operasyonlarıyla güçlendirilmiş FORTRAN) kullanıldığının farkındaydı. Ama sonunda McCarthy, kullanımı IPL'den kolay ve FLP'den güçlü yeni bir dile gerek olduğuna inanmıştı.

MIT'de 1958 güzünden itibaren McCarthy, LISP (*list processing* [liste işleme]) dediği bir programlama dilini gerçeklemeye koyuldu. McCarthy bu dili (gevşek biçimde), matematiğin, özçağrılı (*recursive*) fonksiyon kuramı adlı özel öneme sahip bir hesaplama dalına dayandırmıştı. LISP, liste kopyalama, bu listeden öğe ayıklama, listeye öğe ekleme ve bir şeyin bu listenin öğesi olup olmadığını yoklama gibi temel işlemler barındırıyordu. Bu işlemlerden yola çıkılarak listelerde istenen karmaşık değişiklikler oluşturulabilir. LISP'in önemli bir özelliği, listeleri işleyen programların da birer liste olarak gösterilebilmesiydi. Bu tür programlar, başka listelerin öğeleri olabiliyor ve içlerinde altprogramlar barındırabilirdi. Bir programın içinde kendi kopyası bile gömülü olabiliyordu. Daha önce bahsettiğim gibi, işlemlerinin parçası olarak kendi kopyalarını etkinleştirebilen programlara "özçağrılı" program denir ve (sonsuz bir döngüye girmemesi için gerekli özen gösterildiği takdirde) bunlar çok faydalıdır.[1]

Kullanımı daha kolay olduğu için, LISP kısa sürede IPL'nin yerini alıp yapay zekâ araştırmalarının ve uygulamalarının birincil dili olmuştur. Minsky'nin öğrencileri Evans, Raphael, Bobrow, Slagle ve

1 LISP'in gelişimi konusunda McCarthy'nin kendi tarihçesi için bkz. http://www-formal.stanford.edu/jmc/history/lisp.html. Ayrıca Herbert Stoyan'ın LISP tarihçesi için bkz. http://www8.informatik.uni-erlangen.de/html.lisp-enter.html.

başkaları tarafından üretilmiş programların hepsi LISP'le yazılmıştı. (İlginçtir, Arthur Samuel, dama oynayan programlarını yazarken liste işleme dili kullanmamıştır. Yerine kahramanca davranarak, verimli çalışıp belleği idareli kullansınlar diye bunları basit makine işlemlerinin temel diliyle programlamıştır.)

McCarthy, LISP'i geliştirmenin ötesinde, "zaman-paylaşımı" adlı bir yöntem de önermiştir; bu yöntemde tek bilgisayar, aynı anda birkaç kullanıcıya hizmet edebiliyor, sanki her kullanıcının şahsi makinesiymiş gibi davranabiliyordu.[2] En başta Ed Fredkin'le Bolt, Beranek, Newman (BBN) adlı kurumda, ardından başkalarıyla birlikte çalışan McCarthy, MIT'de DEC PDP-1 bilgisayarını kullanarak ilk zaman-paylaşımı sistemlerinden birini geliştirdi.[3]

8.2 İLK YZ LABORATUVARLARI

Newell, Herb Simon'ın danışmanlığı altında endüstriyel yönetim alanında doktora yapmak amacıyla 1955'te RAND şirketinden ayrılıp 1967'de Carnegie Mellon Üniversitesi (CMU) olacak Carnegie Tech'e girdi. Doktorasını tamamlayan Newell Carnegie'de kalıp profesör oldu ve Simon'la birlikte pek çok doktora öğrencisine danışmanlık yapmaya başladılar; çalışmalarını betimlemek için "karmaşık bilgi işleme" (CIP; *complex information processing*) adını kullanıyorlardı. (Yıllar boyunca YZ deyiminden uzak durdular.) 1956 güzünde Herb Simon, CIP çalışmaları için kullanılan ilk bilgisayar olan IBM 650'yi teslim aldı. İleride IBM 704 kullanacak, ardından DEC makinelerine geçeceklerdi.

John McCarthy, 1958 güzünde Dartmouth'dan MIT'ye taşınmıştı. Minsky, bir yıl sonra MIT'ye katıldı. Minsky'ye kulak verelim:[4]

> [McCarthy'le birlikte] koridorda yürüyorduk ki Jerry Wiesner miydi, Zimmerman mıydı, yoksa başka biri miydi, tam hatırlamıyorum, biriyle karşılaştık. Nasıl olduğumuzu sordu, fena değiliz dedik, yapay zekâ fikirleri üzerinde çalıştığımızı ama biraz daha büyük bir odaya ve yüksek

2 Zaman paylaşımı sisteminin nasıl kurulduğuyla ilgili McCarthy'nin notu için bkz. http://www-formal.stanford.edu/jmc/history/timesharing-memo.html.

3 MIT'deki bilgisayar ve zaman paylaşımı çalışmalarının (ayrıca başka çalışmaların da) ilk günleri hakkında daha fazlası için, Charles Babbage Enstitüsü'nden William Aspray'ın 2 Mart 1989'da John McCarthy'yle yaptığı söyleyişiye bakınız. Şu internet adresinde: http://www.cbi.umn.edu/oh/display.phtml?id=92.

4 1 Kasım 1989'da Arhtur L. Norberg'in Charles Babbage Enstitüsü için yaptığı söyleşiden. Bkz.: http://www.cbi.umn.edu/oh/display.phtml?id=107.



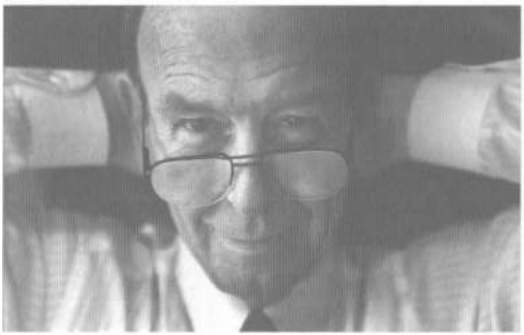
Resim 8.1. 1966'dan 1980'e dek Stanford YZ Laboratuvarının yeri. (Fotoğraf, Lester Earnest'in izniyle.)

lisans öğrencileri için desteğe ihtiyacımız olduğunu belirttik. Birkaç gün sonra bir oda ortaya çıktı...

Bu "oda" kısa süre içinde MIT Yapay Zekâ Projesine dönüşecekti. Ekip en başta MIT'nin IBM 704 bilgisayarını kullanıyordu, fakat bu bilgisayarın belleği, yazılan programlar için yeterli değildi. Böylece BBN'ye ait DEC PDP-1'i kullanmaya başladılar. MIT'de başka bir projeden mali kaynak bulunca kendi PDP-1'lerini satın aldılar; ardından PDP-6 ile PDP-10 bilgisayarları geldi. Ekibin doktora öğrencilerinden bazıları, çalışmalarını BBN'de ve Oliver Selfridge'in genel örüntü tanıma ve öğrenen makineler üzerine YZ araştırmaları yaptığı komşu Lincoln Laboratuvarı'nda sürdürdü. McCarthy, 1962'de Stanford'a geçip orada bir YZ projesine başladı. Jean Piaget'le çalışmış olan matematikçi Seymour Papert (1928-2016), 1963'te YZ Laboratuvarının eş yöneticisi olarak Minsky'ye katıldı.

1965'e gelindiğinde Stanford'da McCarthy ve çalışma arkadaşları, PDP-1 bilgisayarında Thor adlı bir zaman paylaşımı sistemi yaratmışlardı. Bu sistem, on iki Philco gösterim terminalini de içeriyordu ki bu dünyadaki ilk görüntü yönelimli zaman paylaşımı sistemiydi.

McCarthy, Lincoln Laboratuvarı'ndan Stanford'a geçmiş olan Lester Earnest'in (1930-) yardımıyla, 1965'te Stanford YZ Laboratuvarını (SAIL) kurdu. Kendi kampüs içi tesislerine artık sığmaz olan SAIL, 1966 yazında Stanford eteklerinde bir binaya taşındı (bkz. Resim 8.1). ARPA'nın ek desteğiyle bu Laboratuvar, DEC PDP-6 bilgisayarına,



Resim 8.2. Donald Michie
(Fotoğraf, Michie Ailesinin izniyle.)

daha sonra da PDP-10 bilgisayarına kavuştu. SAIL, YZ çalışmalarına (bunları ileriki bölümlerde anlatacağım) ek olarak, bilgisayarla ilişkili başka pek çok projede de yer almıştı, örneğin bilgisayar "pencerelerinin" (*windows*) bir öncüsünü geliştirmek ve ilk olarak herkesin bürosuna bir bilgisayar terminali kurmak gibi.[5]

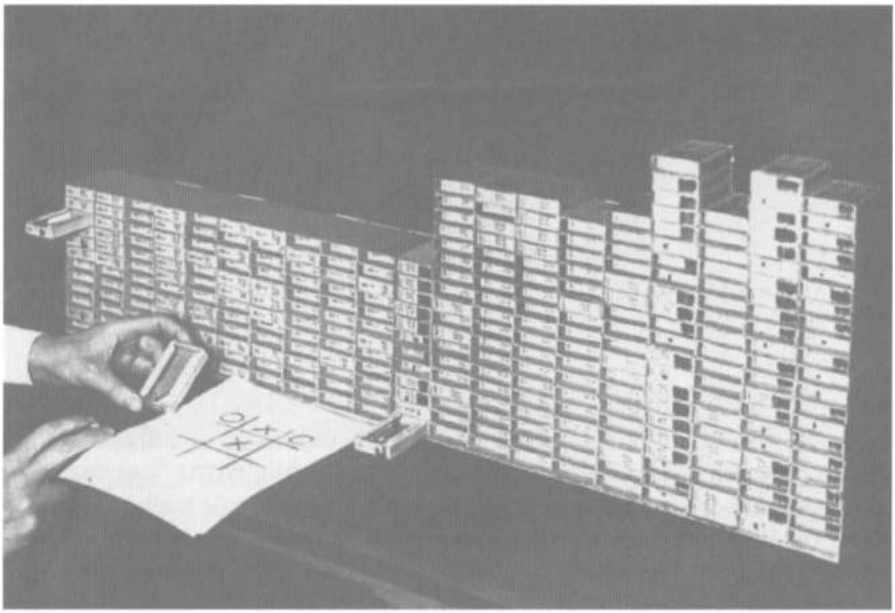
İlk günlerinden itibaren CMU, MIT ve Stanford ekipleri, YZ araştırmalarının önderleri arasında olmuştur. Genelde bu kurumların birinden mezun olanlar, ötekilerden birinde öğretim görevlisi oluyordu.

Yaklaşık 1965 yılı gibi, İskoçya'da Edinburgh Üniversitesi'nde birinci sınıf bir YZ merkezi daha boy gösterdi. Bu merkezi kuran kişi, İkinci Dünya Savaşı sırasında Bletchley Parkta Alan Turing ve I. J. (Jack) Good'la çalışmış olan Donald Michie'ydi (1923-2007; Resim 8.2). Bletchley Park'tayken Turing ve Good'la yaptığı zeki makine sohbetleri Michie'yi büyülemişti. Ekim 2002'de bir söyleşide belirttiği gibi: "Böyle bir girişim uygulanabilir hale gelir gelmez makine zekâsını hayatımın işi haline getirmeye kararlıydım."[6] 1940'ların ortasından sonuna dek bilgisayarlar ilkel ve nadir olduğu için, Michie genetikçi ve moleküler biyolog olmuştu.

Michie, makine zekâsına duyduğu ilgiyi saha kenarından da olsa izleyerek, 1960'da SOS (*noughts and crosses*) oynamayı öğrenebilen

5 Laboratuvarlarda 1973'e dek gerçekleştirilen YZ çalışmaları için bkz. Lester Earnest (ed.), "Final Report: The First Ten Years of Artificial Intelligence Research at Stanford", Stanford Artificial Intelligence Laboratory Memo AIM-228 and Stanford Computer Science Department Report no: STAN-CS-74-409, Temmuz 1973. (http://www-db.stanford.edu/pub/cstr/reports/cs/tr/74/409/CS-TR-74-409.pdf.) Başka SAIL tarihçeleri için Les Earnest'in "SAIL away" adlı makalesine bkz. http://www.stanford.edu/~learnest/sailaway.htm.

6 Bu söyleşinin dökümü için bkz. http://www.aiai.ed.ac.uk/events/ccs2002/CCS-early-british-ai-dmichie.pdf.



Resim 8.3. Michie'nin, SOS oynamayı öğrenen MENACE'ı.

bir "kibrit kutusu ve cam bilye düzeneği" kurmuştu. Bu "makine"ye (Matchbox Educable Noughts and Crosses Engine'nin [Eğitilebilir Kibrit Kutusu SOS Motoru] kısaltması olarak)[7] MENACE adını vermişti (bkz. Resim 8.3). (İleride açıklayacağım üzere, MENACE, günümüzde "pekiştirmeli öğrenme" denen şeyin müjdecisiydi.) Michie, 1960'ların ilk yarısında Stanford'daki bir senelik konukluğu sırasında (masraflarını Amerikan Donanma Araştırmaları Dairesi üstlenmişti) John McCarthy, Bernard Widrow ve YZ alanında çalışan başkalarıyla (bendeniz de dahil) tanıştı. Oradayken, motorlu bir araba üstünde sırık dengelemeyi öğrenen bir program üzerinde çalıştı.

1965'in Ocak ayında Michie, Birleşik Krallık'taki ilk YZ laboratuvarının, yani Edinburgh Üniversitesi Deneysel Programlama Birimi'nin müdürü oldu. Bu ekip, Ekim 1966'da Makine Zekâsı ve Algısı Bölümü'nü oluşturacaktı. Michie, Rod Burstall, Robin Popplestone ve John Collins dahil kimi seçkin bilgisayar yeteneklerini ekibin bünyesine katmıştı. Bu üçü, POP-2 denen bir liste işleme programı geliştirdi; POP-2, Birim'in üyelerince YZ programı yazmada kullanılan dildi. (Bu programlardan

7 Donald Michie, "Experiments on the Mechanisation on Game Learning: 1. Characterization of the Model and its Parameters", *Computer Journal*, c. 1, 1963, s. 232-263.

bazılarını ileride betimleyeceğim.) Michie'nin ekibi uzun yıllar boyunca, Edinburgh Üniversitesi'nde komşu bir ekiple, yani Bernard Meltzer'in (yaklaşık 1916-2008) yönetimindeki Meta-matematik Birimiyle işbirliği içinde çalıştı. Meta-matematik Birimi, Robert Boyer ile J. Strother Moore'un mekanikleştirilmiş teorem ispatlama ve Robert Kowalski'nin mantık programlama ilkeleri üzerine çalışmalarıyla ünlüdür.[8]

Poughskeepsie'deki IBM tesislerinde Nathan Rochester ve Herb Gelernter, Dartmouth çalıştayının ardından kısa süre boyunca YZ araştırmalarını sürdürdüler. Bu araştırmalar sonunda, geometri teoremi ispatlayan makine elde edilmiştir. Bununla birlikte, bilgisayar araştırmalarına verilen devlet desteklerini konu alan bir kitaba göre, bu başarının hemen ardından, "Rochester ve öteki IBM araştırmacılarının baştaki etkinliklerine rağmen, şirketin YZ'ye ilgisi soğudu. Her ne kadar bilgisayar tabanlı satranç ve dama üzerine çalışmalar sürdürülmüş olsa da, 1960 civarında hazırlanmış bir iç rapor, YZ'ye geniş destek verilmesine kuvvetle karşı çıkmıştı."[9] Belki de IBM, bilgisayarların, insanların *yerini almak* yerine, insan işlerine yardımcı olması gerektiğini vurgulamak istiyordu. McCarthy'nin tüm bu olup bitenler hakkındaki görüşü şöyle: "IBM, yapay zekânın [makinelerin insan kadar zeki olmasının], IBM imgesi için zararlı olduğunu düşünüyordu... Bu durum, imge sloganlarından biri olan 'veri hesaplamak değil, veri işlemek' sloganına bağlı olabilir."[10]

8.3 ARAŞTIRMA DESTEĞİ

YZ araştırmaları için gereken bilişim sistemleri gitgide daha büyük ve pahalı hale geldikçe ve YZ laboratuvarları kuruldukça, bireysel araştırmacıların bu alanda çalışmaya başladığı günlerde nazaran daha fazla maddi destek bulmak zorunlu hale gelmişti. 1950'lerin ikinci yarısı ve 1960'ların ilk yarısı boyunca başlıca mali kaynakların iki-

8 Edinburghlu bu ekiplerin tarihçesi için Jim Howe'un 1994 tarihli çevrimiçi makalesine bakınız: "Artificial Intelligence at Edinburgh University: A Perspective", http://www.dai.ed.ac.uk/AI_at_Edinburgh_perspctive.html.

9 National Research Council, *Funding a Revolution: Government Support for Computing Research*, Washington, DC: National Academy Press, 1999. (Bu kitabın, görece muhafazakâr bir YZ tarihçesi anlatısını içeren html versiyonu şu internet adresinde: http://www.nap.edu/catalog.php?record_id=6323#toc.)

10 William Aspray, "An Interview with John McCarthy", 2 Mart 1989, Palo Alto, CA, Charles Babbage Institute, The Center for the History of Information Processing, University of Minnesota, Minneapolis.

si, ABD Savunma Bakanlığı kuruluşları olan Donanma Araştırmaları Dairesi (ONR) ile İleri Araştırma Projeleri Kurumu'ydu (ARPA).

ONR, İkinci Dünya Savaşı'nın bitiminden kısa süre sonra kurulmuştu. Bunun görevi, "donanmanın gelecekteki gücünü korumaya ve ulusal güvenliğe ilişkin olarak, bilimsel araştırmanın olağanüstü öneminin bilinciyle davranarak araştırmalar planlamak, yürütmek, teşvik etmek"ti. ONR'nin Bilgi Sistemleri Dalı, 1950'lerin ortasında Marshall Yovits'in yönetiminde kurulmuştu. Bu dal, çeşitli kurumlarda YZ araştırmalarını desteklemiş, ayrıca öz-örgütlemeli (*self-organizing*) sistemler, sibernetik, optik yazı tanıma ve yapay zekâ konularında konferanslar, çalıştaylar düzenlemiştir. Tüm bunlar, söz konusu teknolojilerin genelde ABD Deniz Kuvvetleri'ne faydalı olacağı beklentisiyle yapılmıştı. (Sonraki yöneticilerden Marvin Denicoff, kimi araştırmalarıma ve yazdığım YZ ders kitabına destek vermiştir.)

ARPA'nın oluşumu kısmen, 1957'de Sovyet uydusu *Sputnik*'in başarıyla fırlatılmasına verilmiş bir tepkidir. ARPA'nın görevi, ABD savunması için önemli alanlardaki sorunları ele almak amacıyla büyük miktarda araştırma fonu temin etmekti. 1950'lerin ikinci yarısında en önemli projelerinden biri, balistik füzelerin atmosfere yeniden girişi sırasında çıkan ısıyı soğurup dağıtmak amacıyla, roketler için ergiyen uçlar geliştirmekti. ARPA'nın Bilgi İşleme Teknikleri Ofisi (IPTO), 1962'de J. C. R. (Lick) Licklider'in (1915-1990; Resim 8.4) yönetiminde kurulmuştur.

"Lick" (tanıyan herkes ona böyle seslenirdi), önce Lincoln Laboratuvarı'nda ve MIT'de, ardından BBN'de çalışmış bir psiko-akustikçiydi. Lick'in 1960 tarihli makalesi, "Man-Computer Symbiosis" (İnsan-Bilgisayar Ortakyaşamı), insanlarla bilgisayarların, "önceden belirlenmiş programlara katı şekilde bel bağlamaksızın karmaşık durumları denetlerken ve karar verirken işbirliği yapması" gerektiğini ileri sürüyordu.[11]

Lick, bilgisayarların ülke savunmasında çok önemli rol oynayacağına kanaat getirmişti; özellikle de insanlarla bilgisayarların birlikte çalıştığı uygulamalarda. ARPA'dayken MAC (Machine-Aided Cognition'ın ya da belki Multi-Access Computing'in ya da Man And Computers'in kısaltması) Projesinin oluşturulması için MIT'ye mali kaynak sağlamıştı. [İlk olarak Temmuz 1963'te başlayan MAC Projesi, ileride Bilgisayar Bilimi Laboratuvarı'na (LCS) dönüşecekti, daha da

11 J. C. R. Licklider, "Man-Computer Symbiosis", *IRE Transactions on Human Factors in Electronics*, HFE-1, 1960, s. 4-11. Şu internet adresinde: http://memex.org/licklider.html.



Resim 8.4. J. C. R. Licklider (Fotoğraf, MIT Koleksiyonundan Koby-Antupit'in [JCL8].)

ileride Bilgisayar Bilimi ve Yapay Zekâ Laboratuvarı'na (CSAIL) evrimleşecekti.] MAC Projesi, Minsky ile McCarthy'nin Yapay Zekâ Projesini kanatları altına alıp MIT'de Fernando Corbató yönetiminde oluşturulan Uyumlu Zaman Paylaşımı Sistemini (CTSS; Compatible Time-Sharing System) de desteklemişti. (CTSS çalışmaları, McCarthy'nin zaman paylaşımı projesinden ayrıydı.)

ARPA ödenekleri, bilgisayar biliminde "mükemmeliyet merkezleri" kurulmasına yardımcı oldu. MIT'nin yanı sıra bu merkezlere Stanford, Carnegie Mellon ve SRI dahildi. Ayrıca ARPA, başka şirketlerin yanı sıra RAND'ın, Systems Development'ın ve BBN'nin bilgisayar bilimi çalışmalarını da desteklemiştir. YZ, ARPA'nın ilgi alanlarından sadece biriydi. IPTO, grafikli kullanıcı arayüzlerini (ve bilgisayar farelerini), süper bilgisayarları, bilgisayar donanımını ve çok büyük ölçekli bütünleşik devreleri (VLSI) ortaya çıkaran araştırmaları, ve belki de en tanınmışı olan interneti doğuran araştırmaları da desteklemişti. Licklider'e göre "ARPA bütçeleri 1968'e dek YZ'yi ayrı bir kalem olarak içermiyordu bile."[12]

Ancak, YZ söz konusu olduğunda, Lick, Newell ve Simon, Minsky ve McCarthy'ye, büyük YZ projelerini desteklemeye yetecek araştırma fonları sağlanması gerektiğine inanıyordu. Stanford'daki (muhte-

12 J. C. R. Licklider, "The Early Years: Founding EPTO", *Expert Systems and Artificial Intelligence Applications and Management* içinde, Thomas C. Bartee (ed.), Indianapolis: Howard W. Sams, 1988, s. 220.

melen MIT ile CMU'daki) duruma ilişkin olarak Paul Edwards şunu açıklamıştır:[13]

> ARPA'dan gelen fonlar adeta otomatiğe bağlanmıştı; Licklider, McCarthy'ye ne istediğini açıkça sorar, sonra istediğini ona verirdi. Başka devlet kurumlarında hayal dahi edilemez bir yöntemdi bu. Licklider şunu anımsıyor: "McCarthy'nin, ARPA tarafından desteklenen bir laboratuvarı olması gerektiği bence açıktı... Ben de o zaman ona bir sözleşme hazırladım."

McCarthy tüm bunları biraz farklı anımsar. 1962'de Stanford'a gelişinin hemen ardından, Licklider'e "YZ çalışmak" için teklif getirmiş. McCarthy, en başta Licklider'in itiraz ettiğini, ama kendisi MIT'de, Licklider ise BBN'deyken aralarında kurulmuş olan yakın ilişkiye dayanarak "küçük bir sözleşme" sunduğunu ileri sürer.[14] Fakat, o zamanlar araştırmaların (örneğin Ulusal Bilim Vakfı tarafından) çoğunlukla nasıl desteklendiği düşünülürse, belki de bu sözleşme o kadar "küçük" değildi. Les Earnest, McCarthy'nin "15 Haziran 1963'ten itibaren ufak (6 kişilik) bir etkinlik için ARPA'dan mali destek elde ettiğini" söyler.[15]

İleride ARPA'nın adı, savunmayla ilgili araştırmalardaki rolünü vurgulamak amacıyla DARPA (Defense Advanced Research Projects Agency) olarak değiştirildi. DARPA projeleri ve ödenekleri, çoğunlukla ONR'lerinkinden büyüktü ve bilgisayarlarla başka ekipmanların satın alınmasına, ayrıca personel tutulmasına olanak veriyordu. Günümüzün bilgisayar tabanlı altyapısının önemli bir kısmı, DARPA araştırma desteklerinin sonucudur desek o kadar abartmış olmayız.

8.4 TAMAMEN HAZIR OLMAK VE YENİ DOĞRULTULAR

1960'ların ortasına gelindiğinde, YZ, yeni gelişmelere iyi hazırlanmıştı. İlk başarıların heyecanıyla dolup taşan YZ, 1960'ların geri kalanında ve 1970'lerde hızlı ilerlemelere de hazırdı. Doğrusu pek çok insan

13 Paul Edwards, *The Closed World: The Computers and the Politics of Discourse in Cold War America*, Cambridge, MA: MIT Press, 1996, s. 270.

14 "An Interview with John McCarthy", *a.g.e.*

15 Lester Earnest (ed.), "Final Report: The First Ten Years of Artificial Intelligence Research at Stanford", Stanford Artificial Intelligence Laboratory Memo AIM-228 and Stanford Computer Science Department Report no: STAN-CS-74-409, Temmuz 1973. (Şu internet adresinde: http://www-db.stanford.edu/pub/cstr/reports/cs/tr/74/409/CS-TR-74-409.pdf.)

taşkın tahminlerde bulunmuştur. Örneğin 1957'deki bir konuşmasında[16] Herb Simon, on sene içinde "sayısal bir bilgisayar, kurallar katılmasını engellemediği takdirde dünya satranç şampiyonu olacaktır" tahminini yapmıştı. Üç tahminde daha bulunmuştu. Ona göre on sene içinde bilgisayarlar müzik besteleyecek, matematik teoremlerini ispatlayacak ve program olarak bir ruh anlayışı barındıracaklardı. Şunu söylemişti: "Amacım sizi şaşırtmak ya da sarsmak değil... fakat özetlemenin en basit yolu, dünyada şu an düşünen, öğrenen, yaratan makineler olduğunu söylemektir. Dahası, görünür bir gelecekte bu bilgisayarların o işleri yapma yetkinliği hızla artacak, ta ki ele alabilecekleri sorunların yelpazesinin ölçeği, insan zihninin başa çıkabileceği sorun yelpazesinin ölçeğiyle aynı olana dek."[17] Daha sonra Simon, bu tahminlerinin, topluma "bilgisayarların ne anlama geldiği hakkında bir his verme" girişiminin parçası olduğunu söyleyecekti.

Simon'un müzik besteleyen ve matematik teoremi ispatlayan bilgisayarlar hakkındaki öngörülerinin, kısa süre içinde gerçekleştiği söylenebilir, fakat satranç şampiyonu bilgisayar kırk yıl sonrasına dek var olmayacaktı. Ayrıca bana kalırsa, insan zihninin başarabildikleriyle "aynı" şeyleri yapmaktan hâlâ uzağız.

Tek iyimser Simon değildi. Hubert Dreyfus'a göre, "MIT Yapay Zekâ Laboratuvarı'nın başı Marvin Minsky, 1968'de, Stanley Kubrick filmi *2001: A Space Odyssey* için yaptığı basın açıklamasında, 'zekâsı insan zekâsına denk makinelere 30 yıl içinde sahip olacağımızı' ilan etmişti."[18] Bu tür tahminleri değerlendirmenin zorluğu, "insan düzeyi zekânın" çok yönlü oluşudur. 2000'e gelindiğinde, YZ programları pek çok düşünsel beceride insanları aşmışken, çoğunda hâlâ çok gerideydi.

Yine de, başarılan işler etkileyici bir başlangıç oluşturuyordu. Makinelerin belirli zeki davranışlarının gösteriminden belki daha önemli olan şey, 1950'lerde ve 1960'ların ilk yarısında geliştirilmiş teknik zemindi. Günümüzde YZ araştırmacıları, bilgiyi, mantık formülleri olarak birtakım ağlarda ya da özgül problem alanlarına göre biçilmiş başka simge yapılarında kodlayarak temsil edecek araçlara sahip. Dahası, o bilgiyi yönlendirip kullanırken sezgisel arama süreçleri ve başka teknikler bakımından deneyim de kazanmışlardır. Aynı zamanda günümüz araştırmacılarının elinde IPL, LISP, POP-2 gibi yeni

16 Operation Research Society'nin (ORSA) on ikinci ulusal toplantısı, Pittsburgh.

17 Bu konuşmanın yayımlanmış versiyonu: Herbert Simon ve Allen Newell, "Heuristic Problem Solving: The Next Advance in Operations Research", *Operations Research*, c. 6, Ocak-Şubat 1958.

18 Hubert L. Dreyfus, "Overcoming the Myth of the Mental", *Topoi*, c. 25, 2006, s. 43-49.

programlama dilleri var; bunlar, simge işleyen programları yazmayı kolaylaştırıyor. Bu simge işleme teknolojisini tamamlayan şey, sinir ağları ve örüntü tanımaya ilişkin istatistiksel yaklaşımlardı. Bu teknik varlıklar, kurumsal ve mali olanlarla birlikte, YZ gelişiminin bir sonraki aşaması için sağlam bir zemin oluşturmuştur.

ÜÇÜNCÜ KISIM

ÇİÇEKLENME: 1960'LARIN ORTASINDAN 1970'LERİN ORTASINA

1960'lar boyunca ve 1970'lerin ortalarına dek, YZ araştırmaları çiçek açtı ve hızlı bir ilerleme kaydedildi. MIT, Carnegie Mellon, Stanford, SRI ve Edinburgh'da kurulmuş laboratuvarlar genişledi, ayrıca başka üniversitelerde ve şirketlerde yeni ekipler kuruldu. Önceki yıllarda yapılan işler, günümüzden bakılınca alçakgönüllü görünse de, aslında heyecan uyandırıp umutları yeşertmiş, ben de dahil yeni insanların aklını çelip bu alana girmesini sağlamıştı. Başarı konusunda pek çoğumuz, hızlı ilerlemeyle ilgili tahminlerini yaparken Herb Simon ve Marvin Minsky kadar iyimserdik.

YZ, pek çok yeni ve önemli icat doğuran bir çiçeklenme dönemine girmişti. Kimi fikirler, doktora tezi araştırma projeleri bağlamında doğmuştu. Kuramsal sorunlarla boğuşan araştırma laboratuvarlarında ve bireysel araştırmacıların aklında ortaya çıkmış fikirler de eksik değildi. Kitabın bu kısmında, önemli projelerin ve araştırma sonuçlarının bazılarını öne çıkarıyorum. Tam bir açıklama sayılmasa da, bu kadarı, o dönemde YZ alanında olup bitenin tipik örneğidir. Bilgisayarla görme konusundaki çalışmalardan başlıyorum.

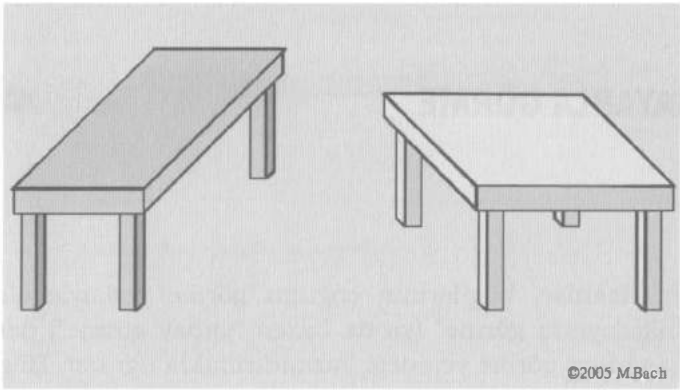
9
BİLGİSAYARLA GÖRME

Görebilen insanlar, bilgilerinin çoğunu görme yetisiyle elde eder. YZ'nin "bilgisayarla görme" (ya da bazen "yapay görme") denen kısmı, bilgisayarlara görme yeteneği kazandırmakla uğraşır. Bilgisayarla görme çalışmalarının çoğu, üç boyutlu dünyadan, bir ya da daha çok televizyon kamerasıyla toplanmış iki boyutlu görüntülerin işlenmesini temel alır. Bu görüntüler, üç boyutlu bir sahnenin iki boyutlu yansımaları olduğu için, görüntüleme sürecinde bilgi kaybı yaşanır. Yani, üç boyutlu farklı görüntüler, aynı iki boyutlu görüntüyü üretebilir. Dolayısıyla, bir görüntüden yola çıkıp görüntüyü aslına sadık biçimde oluşturmak prensipte imkânsızdır.

Yine de insanlar ve hayvanlar üç boyutlu dünyada gayet güzel idare edebilir. Retinalarında oluşan iki boyutlu görüntüleri, çevreleri hakkında kabul edilebilir oranda doğru ve faydalı bilgi verecek şekilde yorumlayabilirler.

İkili (stereo) görüş, yani iki gözün kullanımı, derinlik bilgisi sağlar. Bilgisayarla görme, aynı sahneye bakan en az iki kamerayı farklı noktalara yerleştirerek bu "ikili görüşten" faydalanabilir. (Aynı etki, tek kameranın farklı konumlara hareket ettirilmesiyle de elde edilebilir.) Örneğin, iki kamera kullanıldığında, bunların oluşturduğu görüntüler birbirlerine göre birazcık kaymıştır ve bu kaymadan sahnenin çeşitli kısımlarına olan uzaklığı hesaplamak amacıyla faydalanılabilir. Söz konusu hesaplama, görüntüde derinliği ölçülmek istenen nesnelerin görüntüde karşılık gelen göreli konumlarını karşılaştırmayı içerir. Bu "karşılıklılık problemi" çeşitli yollarla çözülür; bunlardan biri, bir görüntüdeki küçük alanlar ile başka bir görüntüdeki küçük alanlar arasındaki güçlü bağıntıları aramaktır. İki görüntüde, bir görüntü özelliğinin konumunun "aykırılığı" biliniyorsa, bu şekli oluşturan görüntünün o kısmına olan uzaklığı, trigonometri hesaplamaları kullanılarak hesaplanabilir (burada trigonometri hesaplarına girecek değilim elbette).[1]

1 Tam bir inceleme için bkz. David Forsyth ve Jean Ponce, *Computer Vision: A Modern Approach*, 13. Bölüm, Upper Saddle River, NJ: Prentince Hall, 2003.



Resim 9.1. İki masa. (Çizim, Michael Bach'ın izniyle.)

Belki şaşırtıcı ama, ikili görüşün dışında başka ipuçlarından da çokça derinlik bilgisi elde edilebilir. Bu ipuçlarından bazıları tek görüntüde saklıdır; bunlardan sonraki bölümlerde bahsedeceğim. Daha da önemlisi, görülmesi olası nesne türleri hakkındaki altyapı bilgisi, görüntüleri yorumlama becerimizi çoğunlukla açıklıyor. Örneğin, Resim 9.1'deki imgeye göz atalım.

Çoğu insan bu imgeyi, biri uzun ve dar, öteki karemsi iki masa olarak betimler. Ancak, bu imgedeki masa üstlerini ölçerseniz, bunların aynı şekle ve boyuta sahip olduğunu görüp şaşırırsınız! (Bu çizim, psikolog Roger Shepherd'ın "masaları döndürmek" dediği göz yanılmasını temel alır ve Shepherd'ın çiziminin Michael Bach versiyonundan uyarlanmıştır. Bach'ın internet sitesi http://www.michaelbach.de/ot/sze_shephardTables/ adresini ziyaret ederseniz, masa üstlerinden birinin hiç şekil değiştirmeksizin öteki masanın üstüne yerleştiğini izleyebilirsiniz.)

İmgenin dışında bir şey, Resim 9.1'deki iki boyutlu görüntüde yakalanmış üç boyutlu masaların şekli hakkında bizi çıkarımlar yapmaya iten bilgiyi sağlıyor. İleride göreceğimiz üzere, bu ek bilgi, iki şeyden oluşur: çeşitli ışıklandırma koşulları altında görüntü oluşturma süreci hakkındaki bilgi ve varlıkların türleriyle üç boyutlu dünyadaki yüzeyleri hakkındaki bilgi. Eğer bilgisayarları bu tür bilgiyle donatabilirsek, belki onlar da görebilir.

9.1 BİYOLOJİDEN GELEN İPUÇLARI

Görme yetisinin hayvanlarda nasıl işlediğini anlamaya çalışan bilimciler ile bilgisayarla görme üzerine çalışan mühendisler arasında sürekli olarak karşılıklı bilgi akışı vardır. Bu iki ilgi alanının kesişim noktasındaki çalışmalara dair erken bir örnek, MIT'li dört bilimcinin yazmış olduğu "What The Frog's Eye Tells the Frog's Brain" [Kurbağanın Gözünün Kurbağanın Beynine Anlattıkları] başlıklı makalede betimlenmiştir.[2] Önceki biyoloji çalışmalarının rehberlik ettiği Jerome Lettvin, H. R. Maturana, Warren McCulloch, Walter Pitts dörtlüsü, kurbağa beyninin görüntü işleyen kısımlarını incelemişti. Kurbağa görsel sisteminin, görsel alandaki sadece belirli şeylere tepki veren "sezici"lerden (*detector*) oluştuğunu bulmuşlardı. Küçük ve hareketli dışbükey nesnelere (örneğin sinekler) ve ışığın ani kararmasına (görme sahasına giren yırtıcılardan kaynaklanabilir) özgü sezicileri vardır. Birkaç basit seziciyle birlikte bunlar, kurbağaya besin ve tehlike hakkında bilgi verir. Özelde ise kurbağının görsel sistemi, belli ki, görsel manzarasının eksiksiz bir üç boyutlu modelini oluşturmuyor. Makalenin yazarlarının belirttiği gibi:

> Kurbağa, çevresindeki dünyanın durağan parçalarının ayrıntılarını görüyormuş ya da her nasılsa umursuyormuş gibi durmuyor. Çevreyi hareketsiz besinlerle çevrili olsa açlıktan ölür gider. Besin tercihi sadece boyutla ve hareketle sınırlı. Böcek ya da kurtçuk boyutundaki herhangi bir nesnenin üzerine atlar, yeter ki nesne onlar gibi hareket etsin. Sırf sarkıtılan bir et parçasıyla değil, hareketli herhangi bir küçük nesneyle kurbağayı kolayca kandırabilirsiniz. Cinsel yaşamı, ses ve dokunuş üzerinden yürüyor. Düşmanlarından kaçma yöntemi, daha karanlık olan yere sıçramaktan daha kurnazca değil. Sudayken de karadayken de yuvasında olduğu için, sıçradıktan sonra ineceği yer ya da gideceği yön neden önemli olsun ki?

Beynin, görsel imgeleri işleme biçimi hakkında ek bilgi ortaya çıkaran başka deneyler de yapılmıştır. Nörofizyologlar David Hubel (1926-2013) ve Torsten Wiesel (1924-), 1958'den başlayarak bir dizi deney yürütmüşler ve bu deneylerde, memeli görsel korteksinin belirli sinir hücrelerinin, görüntülere ve belli biçimdeki görüntü parçalarına seçi-

2 Lettvin ve diğ., "What the Frog's Eye Tells the Frog's Brain", *Proceedings of the IRE*, c. 47, no: 11, 1959, s. 1940-1951. [Şu eserin Yedinci Bölümü olarak yeniden basılmıştır: William C. Corning ve Martin Balaban (ed.), *The Mind: Biological Approaches to Its Functions*, 1968, s. 233-258.]

ci olarak tepki verdiğini göstermişlerdir. 1959'da, ilaçla uyutulmuş bir kedinin birincil görsel korteksine mikroelektrotlar yerleştirdiler. Kediye belli bir açıyla küçük çizgiler gösterdiklerinde belirli sinir hücrelerinin, başka bir açıyla küçük çizgiler gösterildiğinde başka sinir hücrelerinin ateşlendiğini gördüler. Aslında, sinir hücresinin yeriyle çizgi açısı arasında ilişki kurarak, kedi beyninin bu bölgesinin "harita"sını çıkarabildiler. Bu sinir hücrelerine "basit hücreler" demişlerdi, çünkü belirli doğrultularda hareket eden çizgilere seçici olarak tepki veren "karmaşık hücrelerden" bunları ayırmak istemişlerdi. Sonraki çalışmalar, köşeler, uzun hatlar ve büyük kenar çizgileri gibi daha karmaşık şekiller içeren görüntülere tepki vermek üzere uzmanlaşmış başka sinir hücrelerinin varlığını açığa çıkarmıştır.[3] Benzer uzmanlaşmış sinir hücrelerinin maymun beyninde de olduğunu buldular.[4] Hubel ve Wiesel, 1981'de Fizyoloji-Tıp Nobel Ödülü'ne layık görülmüşlerdir (başka çalışmalarından ötürü Roger Sperry'yle birlikte).[5]

İleriki kısımlarda anlatacağım üzere, bilgisayarla görme araştırmacıları, görüntülerden (hem büyük hem küçük) çizgileri çıkarma yöntemleri geliştiriyorlardı. Hubel ile Wiesel'in çalışması, görüntülerde çizgi bulmanın, görsel sürecin önemli bir kısmı olduğu görüşünü doğrulamaya yardımcı olmuştu. Ancak düz çizgiler, kedilerin (ve insanların) evrimleştiği doğal ortamlarda nadiren bulunur, o halde neden kedilerin (ve bizlerin) sinir hücreleri bu çizgileri sezmek üzere uzmanlaşmıştır? Aslında, 1992'de sinirbilimciler Horace B. Barlow ve David J. Tolhurst, "Why Do You Have Edge Detectors?" [Kenar Çizgisi Sezicileriniz Neden Var?] başlıklı bir makale yazmışlardı.[6] Bu soruya olası bir yanıt olarak Anthony J. Bell ve Terrence J. Sejnowski, doğal manzaraların, açıkça görünen kenar çizgileri olmasa bile, küçük kenar çizgilerinin ağırlıklı toplamı olarak çözümlenebileceğini matematiksel olarak göstereceklerdi.[7]

3 David H. Hubel ve Torsten N. Wiesel, "Receptive Fields, Bionucalar Interaction and Functional Architecture in the Cat's Visual Cortex", *Journal of Physiology*, c. 160, 1962, s. 106-154.

4 David H. Hubel ve Torsten N. Wiesel, "Receptive Fields and Functional Architecture of Monkey Striate Cortex", *Journal of Physiology*, c. 195, 1968, s. 215-243.

5 Hubel ile Wiesel'in, görsel imgeleri beynin nasıl işlediği hakkındaki çalışmalarıyla ve betimlemeleriyle ilgili ilginç bir açıklama için Hubel'in çevrimiçi kitabı *Eye, Brain, and Vision*'a bakınız. http://neuro.med.harvard.edu/site/dh/index.html.

6 Horace B. Barlow ve D. J. Tolhurst, "Why Do You Have Edge Detectors?", *Proceedings of the 1992 Optical Society of America Annual Meeting* içinde, Technical Digest Series, c. 23, Albuquerque, NM, Washington: Optical Society of America, 1992, s. 172.

7 Anthony J. Bell ve Terrence J. Sejnowski, "Edges Are the 'Independent Components'

9.2 YÜZ TANIMA

1960'ların ilk yarısında, Palo Alto'daki şirketi Panoramic Research'te Woodrow (Woody) W. Bledsoe (Texas Üniversitesi'nde otomatik teorem kanıtlama üzerine çalışacaktı ileride), Charles Bisson ve Helen Chan'la (daha sonra Helen Chan Wolf) birlikte, CIA projelerinin desteklediği yüz tanıma teknikleri geliştirmişlerdi.[8] Yaklaşımlarının, bir anma makalesinden alınmış betimlemesi şöyledir:[9]

> Bu [yüz tanıma] projesi, insan-makine diye etiketlenmişti, çünkü fotoğraflardan bir dizi özelliğin koordinatlarını çıkaranlar insanken, daha sonra bunları tanıma işinde kullanan bir bilgisayardı. GRAFACON ya da RAND TABLET kullanan operatör, gözbebeklerin merkezi, gözlerin iç köşeleri, gözlerin dış köşeleri, alında saçın üçgen yaptığı yer, vesaire gibi özelliklerin koordinatlarını çıkarır. Bu koordinatlardan yola çıkılarak, ağız genişliği ve gözbebeğinden gözbebeğine göz genişliği gibi 20 uzaklığı içeren bir liste hesaplanır. Bu operatörler, saatte yaklaşık 40 resim işleyebilir. Veritabanı oluşturulurken, fotoğraftaki insanın ismi, hesaplanan uzaklıkların listesiyle ilişkilendirilip bilgisayarda saklanır. Tanıma evresinde, uzaklık kümesi, her fotoğraftaki karşılık gelen uzaklıklarla kıyaslanır ve fotoğraf ile veritabanı kaydı arasında bir uzaklık belirlenir. En yakın kayıtlar çıktı olarak getirilir.

Bledsoe, 1966'da Panoramic'i bıraktıktan sonra bu çalışmayı Peter Hart'la birlikte devam ettirdi.[10]

Derken 1970'de, Stanfordlu doktora öğrencisi Michael D. Kelly, resimlerde yüz özelliklerini sezip bunu, insanları tanımak amacıyla kullanan bir program yazmıştır.[11] Bu programın görevi, Kelly'nin belirttiği gibi:

of Natural Scenes", *Advances in Neural Information Processing Systems*, c. 9, Cambridge, MA: MIT Press, 1996. Bkz. ftp://ftp.cnl.salk.edu/pub/tony/edge.ps.Z.

8 Woodrow W. Bledsoe ve Helen Chan, "A Man-Machine Facial Recognition System: Some Preliminary Results", Teknik Rapor PRI 19A, Panaromic Research, Inc., Palo Alto, CA, 1965.

9 Michael Ballntyne, Robert S. Boyer ve Larry Hines, "Woody Bledsoe: His Life and Legacy", *AI Magazine*, c. 17, no: 1, 1996, s. 7-20. Ayrıca şu internet adresinde: http://www.utexas.edu/faculty/council/1998-1999/memorials/Bledsoe/bledsoe.html.

10 Woodrow W. Bledsoe, "Semiautomatic Facial Recognition", Teknik Rapor SRI Projesi 6693, Stanford Araştırma Enstitüsü, Menlo Park, CA, 1968.

11 Michael D. Kelly, "Visual Identification of People by Computer", Stanford YZ Projesi, Stanford, CA, Teknik Raporu AI-130, 1970.

> TV kamerasıyla çekilmiş bir dizi insan fotoğrafından, aynı kişiyi gösteren resimleri seçmekti...
>
> Kısaca bu program, resimlerde gözler, burun ya da omuzlar gibi özelliklerin yerini bularak çalışıyordu... Bu tezde anlatılan çalışmanın ilginç ve zorlu kısmı, sayısal resimlerde bu özelliklerin saptanmasıdır. Ölçüm kümesi elde edilince, bireylerin tanınması için en yakın komşu yöntemi kullanılır.

Yüz tanıma alanında öncü çalışmalarda bulunmuş başka biri de, günümüzde Carnegie Mellon Üniversitesi profesörlerinden Takeo Kanade'dir. 2007'de On Birinci Uluslararası Bilgisayarla Görme Konferansında yaptığı konuşmada, bu sahadaki ilk çalışmalarını anlatır:[12] "Yüz tanıma programını bir çevirici diliyle yazmış, 10 mikrosaniye devir hızı ve 20 kB ana belleği olan bir bilgisayarda çalıştırmıştım. Programı 1.000 adet yüz görüntüsüyle sınamak bana gurur vermişti, çünkü 10 görüntüyle sınamanın 'büyük ölçekli' deney sayıldığı o günlerde bu nadir bir olaydı". (Yeri gelmişken belirtelim, Kanade, yüz tanıma çalışmalarını bugüne dek sürdürmüştür. Yüz tanımayla ilgili internet sitesi için bkz. http://www.ri.cmu.edu/labs/lab_51.html.)

1960 ve 1970'lerin yüz tanıma programlarının birtakım kısıtlamaları vardı. Çoğunlukla ölçeği, pozu, yüz ifadesi ve ışığı standart olan yüz imgelerine gereksinim duyuyorlardı. Elinizdeki kitabın sonuna doğru, çok daha gelişmiş otomatik yüz tanıma yöntemlerini doğuran araştırmaları betimliyorum.

9.3 ÜÇ BOYUTLU KATI NESNELER İÇİN BİLGİSAYARLA GÖRME

9.3.1. Erken Bir Görme Sistemi

Lincoln Laboratuvarı'nda çalışan MIT'li doktora öğrencisi Lawrence G. Roberts (1937-), siyah-beyaz (gri düzeyli) fotoğraflarda nesneleri saptayıp uzamdaki yönlenimlerini (*orientation*) ve konumlarını belirleyen bir program yazmış belki de ilk kişiydi. (Onun programı aynı zamanda "gizli-çizgi" algoritması kullanan ilk bilgisayardı; bu algoritma, bilgisayar grafiği hakkında sonraki çalışmalar için çok önemli olmuştur. ARPA'nın Bilgi İşleme Teknikleri Dairesinin baş bilimcisi ve sonra müdürü olarak Roberts, internetin öncüsü Arpanet'in yaratılışında önemli rol oynayacaktı.)

12 http://iccv2007.rutgers.edu/TakeoKanadeResponse.htm.

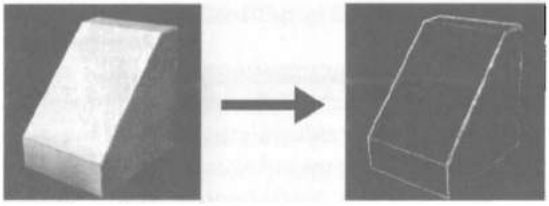
1963 MIT doktora tezinin giriş bölümünde Roberts şunu yazar:[13]

> Resimli verileri makinelerin tanıması sorunu, uzun süredir ulaşılması zor bir hedef olagelmiştir, fakat alfabe harflerinden daha karmaşık bir şeyde denenmesi nadiren gerçekleşmiştir. Pek çok insan, yazı karakteri tanıma araştırmalarının ilk adım olduğunu, bizi daha genel bir örüntü tanıma sistemine götüreceğini düşünüyordu. Bununla birlikte, yazı karakteri tanımayla ilgili çok sayıda girişim, benimki de dahil, pek yol alabilmiş değildi. Bana kalırsa bunun sebebi, iki boyutlu soyut biçimlerle ilgili çalışmaların, bizi, üç boyutlu nesnelerin tanınması için gerekli tekniklere yaklaştıracağına uzaklaştırmasıdır. Katı nesnelerin algılanması, üç boyutlu dönüşüm niteliklerine ve doğa yasalarına dayandırılabilecek bir süreçtir. Nesneleri saptamakla kalmayıp aynı zamanda nesnelerin uzamdaki yönlenimini ve konumunu belirleyen bir yöntem, bu niteliklerden özenle faydalanılarak geliştirilmiştir.

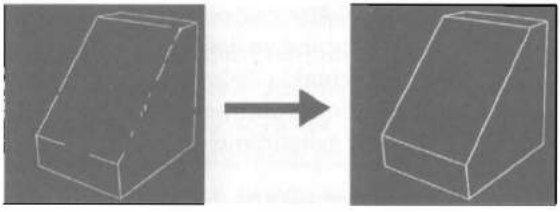
Roberts'in sistemi öncelikle manzara fotoğrafını işleyip çizgilerden bir gösterim (*representation*) yaratır. Ardından bu çizimi, üç boyutlu bir gösterime dönüştürür. Bu gösterimi, katı nesnelerin bilgisayarda saklanan gösterim listeleriyle eşleştirmek, programın, baktığı nesneyi sınıflandırmasını sağlar. Aynı zamanda, herhangi bir bakış açısından göründüğü biçimiyle nesnenin bilgisayar grafiği görüntüsünü de oluşturabilir.

Burada bizi ilgilendiren başlıca konu, Roberts'in fotoğraf görüntüsünü nasıl işlediğidir. Roberts, fotoğrafı taramasının ve bunun, yoğunluk değerlerini yansıtan rakamların (piksellerin) bir dizisi olarak gösterilmesinin ardından, ileride "Roberts Geçişi" denecek özel bir hesap kullanarak, dizideki her küçük 2 × 2'lik karenin, görüntü yoğunluğunda ani bir değişiklik gösteren görüntü parçasına karşılık gelip gelmediğini belirliyordu. (Roberts Geçişi, ileride "eğim operatörü" denecek şeyin ilk örneğiydi.) Sonra görüntüyü, yoğunluğu ani değişim geçirmiş kısımları "ışıklandırıp" yoğunluğu aşağı yukarı birörnek olan kısımları "karanlık" bırakarak gösteriyordu. Roberts'in tezinde kullanılmış tipik bir görüntüde gerçekleştirilen bu işlemin sonucu Resim 9.2'de gösteriliyor. Bu resimde görüldüğü üzere, görüntü yoğunluğundaki büyük değişimler çoğunlukla nesnenin kenar çizgileriyle iliş-

13 Lawrence G. Roberts, "Machine Perception of Three Dimensional Solids", MIT doktora tezi, 1963; Lincoln Laboratuvarı Teknik Rapor no: 315 olarak basılmıştır, 22 Mayıs, 1963; şu yayında: J. T. Tippett ve diğ. (ed.), *Optical and Electro-Optical Information Processing*, s. 159-197, Cambridge, MA: MIT Press, 1965. Şu internet adresinde: http://www.packet.cc/files/mach-per-3D-solids.html.



Resim 9.2. Yoğunluk değişikliklerini tespit etmek. (Fotoğraflar, Lawrence Roberts'in izniyle kullanılmıştır.)



Resim 9.3. Son çizimin oluşturulması. (Fotoğraflar, Lawrence Roberts'in izniyle kullanılmıştır.)

kilidir. Dolayısıyla Roberts Geçişi gibi eğim operatörlerine çoğunlukla "kenar sezicisi" denir.

Sağdaki görüntünün daha fazla işlenmesi, keskin yoğunluk değişikliklerini temsil eden noktaları birleştirerek önce küçük düz çizgiler, ardından daha uzun çizgiler oluşturmaya çalışır. Son olarak görüntünün bir çizimi oluşturulur. Bu son adım Resim 9.3'te gösteriliyor.

Roberts'in programı katı nesnelerin pek çok farklı fotoğrafını inceleyebiliyordu. Roberts şu yorumu yapar: "Resimden yola çıkıp çizime ulaşan süreç, her resim için uygun değil, ama basit resimlerde işe yarıyor". Roberts'in başarısı, görüntülerde çizgi bulan ve bu çizgilerle nesne gösterimleri oluşturan programlar üzerine başka çalışmaları teşvik etmiştir. Belki de Roberts'in, katı nesneleri seçmiş olmasından ötürü, sonraki çalışmaların pek çoğu oyuncak blokları (Britanya'da "tuğla" denir) kullanmıştır.

9.3.2 "Görme Yetisi Yaz Projesi"

Larry Roberts'in, Marvin Minsky'nin öğrencisi değil de MIT bilgi kuramı profesörü Peter Elias'ın öğrencisi olması ilginçtir. Fakat kısa süre içinde, Minsky'nin ekibi de bilgisayarla görme üzerine çalışmaya başlayacaktı. MIT Yapay Zekâ Ekibine yeni katılmış olan matematikçi ve psikolog Seymour Papert, 1966 yazında bir "görme yetisi yaz projesi" düzenledi. Bu projenin hedefi, bir "videsektör"den (bir tür tarayıcı) ge-

len resmi çözümleyip "bilinen nesneler [top, silindir, blok gibi] hakkında sözcük dağarcığıyla bunları eşleştirerek nesneleri isimlendiren" bir dizi program geliştirmekti. Projenin itici güçlerinden biri, "bir görme sisteminin önemli bir parçasını oluştururken yaz araştırmacılarımızdan etkin olarak faydalanmaktı."[14]

Elbette, "bir görme sisteminin önemli parçasını" oluşturma sorunu, Papert'in beklediğinden çok daha zordu. Bununla birlikte, proje, MIT'de kalıcı bir, bilgisayarla görme araştırması başlatması bakımından başarılıydı ve bu araştırmalar günümüze dek süregelmiştir.

MIT'deki bu ilk çabaların ardından (Stanford ve SRI'daki benzer çabalar kısaca betimlenecek), bilgisayarla görme araştırmaları iki alana odaklandı. İlkine, "alt düzey" görme diyebiliriz; yani, görece basit nesneler içeren bir görüntünün imgesinin çizim gösterimini amaçlayan görüntü işleme sürecinin ilk evreleri. İkinci alan, konumu belirlenip tanınan ayrı nesnelerin bir kümesi olarak çizimlerin nasıl çözümleneceğiyle ilgileniyordu. Alt düzey görmenin önemli bir parçası, "görüntü süzgeçleri"ydi; şimdi de bunu anlatacağım.

9.3.3 Görüntü Süzme

Bir görüntüyü sadeleştirip gürültüsünü atmak ve belirli görüntü özelliklerini pekiştirmek amacıyla süzgeçten geçirme fikri en az on yıldır biliniyordu. Örneğin, 1955'te Gerald Dinneen'in, gürültü yok etme ve kenar çizgilerini güçlendirme amacıyla görüntü işlediğinden bahsetmiştim. Russell Kirsch ile çalışma arkadaşları da görüntü işleme hakkında deneyler yapmışlardı.[15] (Sayısal fotoğraflarını bilgisayarla işleyen okurlar bu imge süzgeçlerinden bazılarını kullanmıştır.) İki boyutlu görüntüleri süzgeçten geçirmek, tek boyutlu elektronik sinyalleri süzgeçten geçirmekten (sıradan bir işlemdir) çok da farklı değil. Belki de betimlemesi en kolay işlem, "ortalama almaktır"; bu işlem, ince ayrıntıları muğlaklaştırır ve rasgele gürültü noktalarını ortadan kaldırır. Tüm ortalama alma işlemlerindeki gibi, görüntülerin ortalamasını alma işlemi, komşu değerleri hesaba katıp bunları birleştirir. Örneğin, Resim 9.4'te gösterilen, içerdiği 3 × 3'lük "ortalama alma

14 Bu proje MIT Yapay Zekâ Ekibi Görme Bildirisi no: 100'de betimlenmiştir; bkz. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-100.pdf.

15 Russell A. Kirsch ve diğ., "Experiments in Processing Pictorial Information with a Digital Computer", *Proceedings of the Eastern Joint Computer Conference*, Institute of Radio Engineers and Association for Computing Machinery, Aralık 1957, s. 221-229.

0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10
0	0	0	0	0	10	10	10	10	10

Resim 9.4. Görüntü yoğunluğu değerleri dizisi ve ortalama alan pencere.

-1	0	+1
-2	0	+2
-1	0	+1

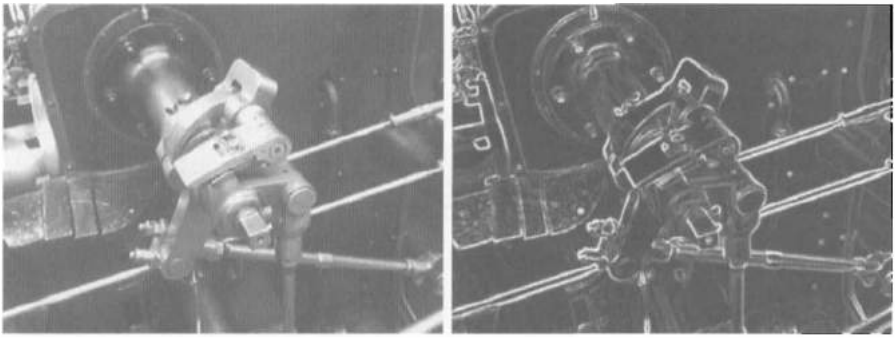
+1	+2	+1
0	0	0
-1	-2	-1

Resim 9.5. Sobel'in dikey (solda) ve yatay (sağda) süzgeçleri.

penceresi" kalın çizgilerle belirtilmiş, yoğunluk değerlerinin görüntü dizisine bakalım. Bu yoğunluk değerleri, sağ tarafı parlak, sol tarafı karanlık olup bu iki tarafı keskin bir çizgiyle ayrılan bir görüntüye karşılık gelir. (10 gibi büyük sayıların, görüntünün aydınlık kısımlarına, 0 sayısının ise karanlık kısma karşılık gelmesini bu örnekte ben de benimsedim.)

Ortalama alma işlemi, ortalama penceresini bütün görüntü üzerinde dolaştırır, böylece pencerenin merkezi sırayla her pikselin üzerine gelir. Pencerenin her yer değiştirişinde, merkezindeki yoğunluk değerinin yerine, pencere içindeki değerlerin ortalama yoğunluğu geçer (süzgeçli durumda). (Pencereyi görüntü üzerinde dolaştırma ve pencere içindeki rakamlara dayanarak hesap yapma sürecine *evrişim* [*convolution*] denir.) Bu örnekte, pencerenin merkezindeki 0 sayısının yerine 3,33 (belki üçe yuvarlanır) geçirilir. Ortalama alma işleminin, net kenar çizgisini bulandırdığı görülüyor; sağdan sola gittikçe 10 sayısı 7'ye (yuvarlanarak), 7 sayısı 3'e, 3 de 0'a düşer. Bununla birlikte, eşit ışıklandırılmış alanların içindeki rakamlar değişmez.

Görüntüde keskin parlaklık değişimlerini bulmaya yarayan önemli bir süzgeç işleminden, yani Roberts Geçişinden zaten bahsettim. Doktora öğrencisi Irwin Sobel, 1968'de başka bir süzgeçleme işlemi yarat-



Resim 9.6. İmge parlaklığındaki keskin değişikliklerin Sobel Operatörüyle bulunması. (Fotoğrafı çeken George Miller. GNU Free Documentation License şartlarında kullanılmıştır.)

mıştı. Stanford'da bir Bilgisayarla Görme dersinde bu işlemi betimleyen Raj Reddy buna "Sobel Operatörü" adını vermişti.[16] Operatör, iki süzgeç penceresi kullanır; bir tanesi dikey doğrultudaki büyük eğimler (yoğunluk değişiklikleri), ötekiyse yatay doğrultularda büyük eğimlere hassastır. Bunlar, Resim 9.5'te gösteriliyor.

Her noktadaki görüntü yoğunluğunun, tüm sayılar toplanmadan önce süzgeç penceresinin ilgili hücresindeki rakamla çarpılması dışında, Sobel süzgeçlerinden her biri ortalama alma süzgeci gibi çalışır. Eşit aydınlanan bölgelerde pencere içi toplamı 0 olur. Eğer dikey süzgeç, dikey kenar çizgisi (sağ taraf sol taraftan parlaktır) üzerinde ortalanırsa, toplam, artı bir değere kavuşur. (Öbür olasılıkları siz düşünün.) Herhangi bir doğrultuda keskin değişiklikleri sezmek amacıyla, iki süzgeçten de gelen sonuçlar matematiksel olarak birleştirilir.

Resim 9.6'daki görüntüler, Sobel Operatörünü örnekle gösteriyor. Sağdaki görüntü, soldaki görüntüye Sobel Operatörünü uygulayarak elde edilir.

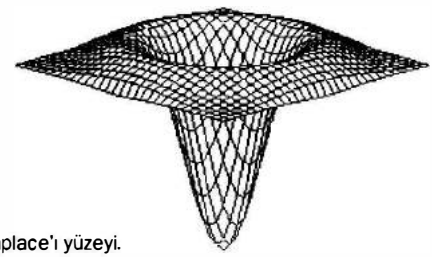
Görüntülerdeki nesnelerin kenarlarını, çizgilerini, köşelerini bulmak amacıyla, daha karmaşık ve gürbüz pek çok görüntü işleme işlemcisi önerilip kullanılmıştır.[17] Kenar çizgilerini bulan özellikle ilginç

16 Sobel'e göre o ve okul arkadaşı Gary Feldman, operatörü ilk kez 1968'de Stanford YZ seminerinde sunmuşlar. Daha sonra şurada betimlenecekti: Karl K. Pinger, "Visual Perception by a Computer", *Automatic Interpretation and Classification of Images* içinde, A. Grasselli (ed.), New York: Academic Press, 1969, s. 277-284. Ayrıca şurada da bahsedilmiştir: Richard O. Duda ve Peter E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973, s. 271-272.

17 Örneğin bkz. M. H. Hueckel, "An Operator Which Locates Edges in Digitized Pictures", *Journal of the ACM*, c. 18, no: 1, Ocak 1971, s. 113-125 ve Berthold K. P. Horn, "The

0	0	3	2	2	2	3	0	0
0	2	3	5	5	5	3	2	0
3	3	5	3	0	3	5	3	3
2	5	3	−12	−23	−12	3	5	2
2	5	0	−23	−40	−23	0	5	2
2	5	3	−12	−23	−12	3	5	2
3	3	5	3	0	3	5	3	3
0	2	3	5	5	5	3	2	0
0	0	3	2	2	2	3	0	0

Resim 9.7. Gauss'un Laplace'ı süzgeç penceresi.



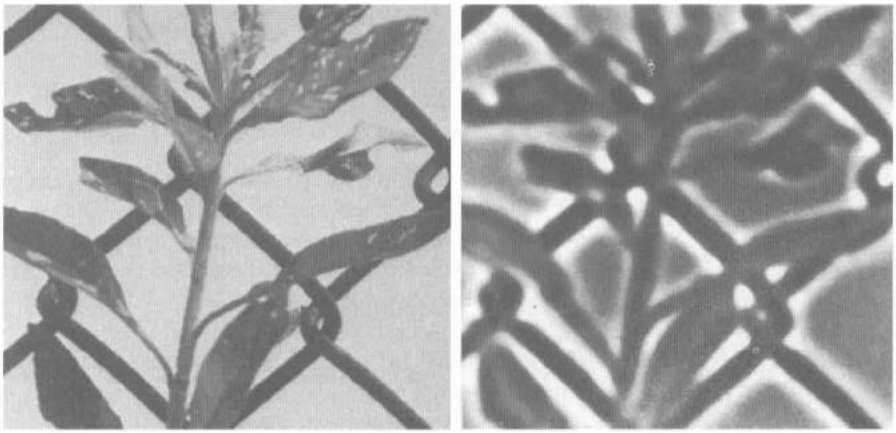
Resim 9.8. Gauss'un Laplace'ı yüzeyi.

bir tanesini Britanyalı sinirbilimci ve psikolog David Marr (1945-1980) ile Ellen Hidreth önermişlerdir.[18] Marr-Hildreth kenar saptayıcı, "Gauss'un Leplace"ı (LoG) adlı bir süzgeç penceresi kullanır. (Bu adın konmasının sebebi, "Laplace türü" bir matematik operatörünün "Gauss türü" bir çan eğrisi üzerinde kullanılmasıdır, böylece iki ünlü matematikçi Pierre-Simon Laplace ve Carl Friedrich Gauss anılmış oluyor). Resim 9.7'de, 9 × 9'luk bir süzgeç penceresinde LoG sayılarından bir örnek gösteriyorum. Bu pencere görüntü üzerinde gezdirilir ve görüntü sayılarını toplayıp çarpar, tıpkı daha önce bahsettiğim öbür süzgeç pencereleri gibi.

Eğer LoG sayıları, bir düzlem üzerindeki (ve altındaki) "yükselti"ler olarak resmedilirse, ilginç görünen bir yüzey ortaya çıkar. Resim 9.8'de

Bindford-Horn Line Finder", MIT AI Memo 285, MIT, Temmuz 1971 (Aralık 1973'te gözden geçirilmiştir. Şu internet adresinde: http://people.csail.mit.edu/bkph/AIM/AIM-285-OPT.pdf).

18 David Marr ve Ellen Hildreth, "Theory of Edge Detection", *Proceedings of the Royal Society of London*, Series B, Biological Sciences, c. 207, no: 1167, Şubat 1980, s. 187-217.



Resim 9.9. Bir imge (solda) ve bunun LoG'yle işlenmiş temsili (sağda). (Görüntüler şuradan: David Marr ve Ellen Hildreth, "Theory of Edge Detection", *Proceedings of the Royal Society of London*, Series B, Biological Sciences, c. 207, no. 1167, Şubat 1980, s. 187-217.)

bunun bir örneği gösteriliyor. Bu LoG fonksiyonuna çoğunlukla Meksika şapkası ya da sombrero fonksiyonu denmesi hiç de şaşırtıcı değil.

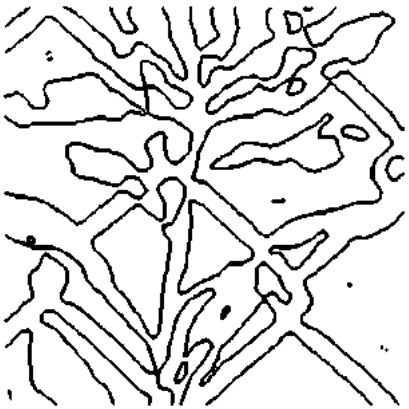
Marr ve Hildreth, LoG süzgeç penceresini birtakım örnek görüntüler üzerinde kullanmışlardı. Makalelerinden alınmış bir örnek, Resim 9.9'da gösteriliyor. Sağdaki görüntüde karanlık kısımları, beyazımsı bir şeridin çevrelediğine dikkat edin. Marr-Hildreth kenar saptayıcısı, Resim 9.10'da gösterildiği gibi bir son "çizim" yaratmak amacıyla, LoG'yle işlenmiş görüntüde aydınlıktan karanlığa (ve tersi) geçişleri arayan ikinci bir imge işleme sürecini devreye sokar.

Marr ile Hildreth'in çalışmasından bu yana kenar seziminde ilerleme kaydedilmiştir. Var olan sezicilerin en iyileri arasında, John Canny'nin ileri sürdüğü ve Canny kenar saptayıcısı denen saptayıcıyla ilintili olanlar da var.[19]

Nörofizyolog olarak Marr, özellikle insan beyninin, görüntüleri işleme şekliyle ilgileniyordu. 1976 tarihli makalesinde, görüntü işlemenin ilk adımında, "asal taslak" denilen şeyin üretildiğini ileri sürer.[20] Makalesinin özetinde belirttiği gibi:

19 John E. Canny, "A Computational Approach to Edge Detection", *IEEE Transactions Pattern Analysis and Machine Intelligence*, c. 8, 1986, s. 679-714.

20 David Marr, "Early Processing of Visual Information", *Philosophical Transactions of the Royal Society of London*, Series B, Biological Sciences, c. 275, no: 942, Ekim 1976, s. 483-519.



Resim 9.10. Marr-Hildrech kenar saptama işleminin sonucu. (David Marr ve Ellen Hildreth, "Theory of Edge Detection", *Proceedings of the Royal Society of London*, Series B, Biological Sciences, c. 207, no. 1167, Şubat 1980, s. 187-217.)

> Sonuç almada ilk adımın, görüntüde bulunan gri ton değişikliklerinin ilkel ama zengin bir betimlemesini hesaplamak olduğu ileri sürülür. Bu betimleme, yoğunluk değişikliği türleriyle ilgili sözcüklerle tanımlanır (KENAR, GÖLGELİ KENAR, UZUN KENAR, LEKE vs.)... Bu betimleme, sabit teknikler sayesinde bir yoğunluk dizisinden elde edilir ve buna asal taslak denir.

Marr ve Hildreth, kenar saptayıcılarını, asal taslağı oluştururken beynin kullandığı işlemlerden biri olarak öne sürer. Kuramlarının, "kimi temel fizyolojik bulguları açıkladığını ve ... basit [sinir] hücreleri hakkındaki fizyolojik modelin zeminini oluşturduğunu" söylemişlerdir.

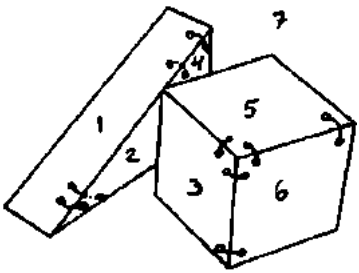
Marr'ın görme araştırmalarındaki parlak meslek hayatı, 1980'de kansere teslim olmasıyla sona erdi. Ömrünün son yıllarında, insan görme yetisiyle ilgili fikirlerini ayrıntılı olarak anlattığı önemli bir kitabı tamamlamıştır.[21] Marr'ın, görsel süreçlerin başka adımlarıyla ilgili kimi fikirlerini ileriki bir bölümde anlatacağım.

9.3.4 Çizimleri İşleme

Alt düzey görme programlarının, bir görüntünün çizgi sürümünü üretebileceğini belki de zamanından önce varsayan pek çok araştırmacı, görüntülerde nesneleri bulmak amacıyla çizimleri çözümleyen yöntemler geliştirmeye koyulmuştu.

Minsky'nin YZ ekibindeki öğrencilerden Adolfo Guzman-Arenas (1943-), bloklar içeren bir görüntü çiziminin bileşenlerine nasıl ayrıla-

21 David Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, San Francisco: W. H. Freeman an Co., 1982.



Resim 9.11. Örnek görüntüde SEE'nin belirlediği bağlar. (Çizim, Adolpho Guzman'ın izniyle kullanılmıştır.)

cağına odaklanmıştı; Guzman bu bileşenlere "cisimler" (*body*) diyordu. Bu ayırma işini başarmak için kullandığı LISP programının adı SEE'ydi ve bu programı MIT YZ Ekibinin PDP-6 bilgisayarında çalıştırmıştı.[22] SEE'ye sunulan girdi, görüntünün yüzeyler, çizgiler (iki yüzeyin bir araya geldiği yer) ve köşeler (iki çizginin bir araya geldiği yer) bağlamında çizimli gösterimiydi.

SEE'nin görüntü çözümlemesi, köşeleri pek çok farklı tipte sınıflandırmakla başlıyordu. SEE, türüne bağlı olarak her doruk noktası için, komşu düzlemsel yüzeyleri "bağ"larla birbirine iliştiriyordu. Yüzeyler arasındaki bağlar, o yüzeylerin aynı cisme ait olduğuna işaretti. Örneğin, SEE'nin çözümlemiş olduğu bir manzaranın kimi bağları Resim 9.11'de gösteriliyor.

SEE, çeşit çeşit çizimde görece iyi performans veriyordu. Örneğin, Resim 9.12'deki görüntünün tüm cisimlerini doğru şekilde bulmuştur.

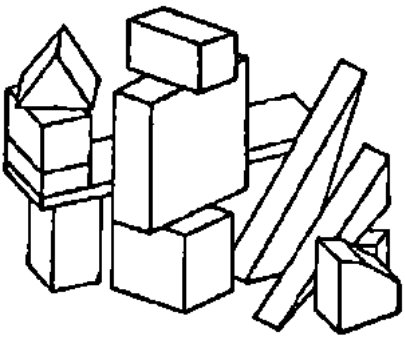
Çalışmasının büyük kısmında Guzman, gerek duyduğu çizimleri başka programların bir şekilde gerçek görüntülerden üreteceğini varsaymıştı. Araştırmasını betimleyen makalede yazdığı gibi:[23]

> Görüntü, görsel imge cihazından ya da parlaklık yoğunluğu dizilerinden elde edilmemiştir. Bunun yerine, bir tür önişlemenin gerçekleştirildiği ve çözümlenecek görüntünün... noktalar (köşeler), çizgiler (kenarlar) ve yüzeyler (alanlar) bağlamında simgesel bir formatta hazır olduğu varsayılır.

Buna ek olarak Guzman, görüntü cisimlere ayrıştırıldıktan sonra ne yapılabileceğini kendine dert etmez:

22 Guzman'ın 1968 yılındaki doktora tezinin başlığı: "Computer Recognition of Three Dimensional Objects in a Visual Scene". Şu internet adresinde: http://www.lcs.mit.edu/publications/pubs/pdf/MIT-LCS-TR-059.pdf.

23 Adolfo Guzman, "Decomposition of a Visual Scene into Three-Dimensional Bodies", *AFIPS*, c. 33, Washington, DC: Thompson Book Co., 1968, s. 291-304. MIT YZ Ekibinin notu olarak bkz. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-171.pdf.



Resim 9.12. SEE'nin çözümlediği bir görüntü. (Çizim, Adolpho Guzman'ın izniyle kullanılmıştır.)

> ... bu program, bir görüntüdeki "küpleri" ya da "evleri" bulamaz, çünkü "evin" ne olduğunu bilmez. SEE, bir görüntüyü cisimlere ayrıştırır ayrıştırmaz, başka bir program bunlar üzerinde çalışıp o cisimlerden hangilerinin "ev" olduğunu belirleyecektir.

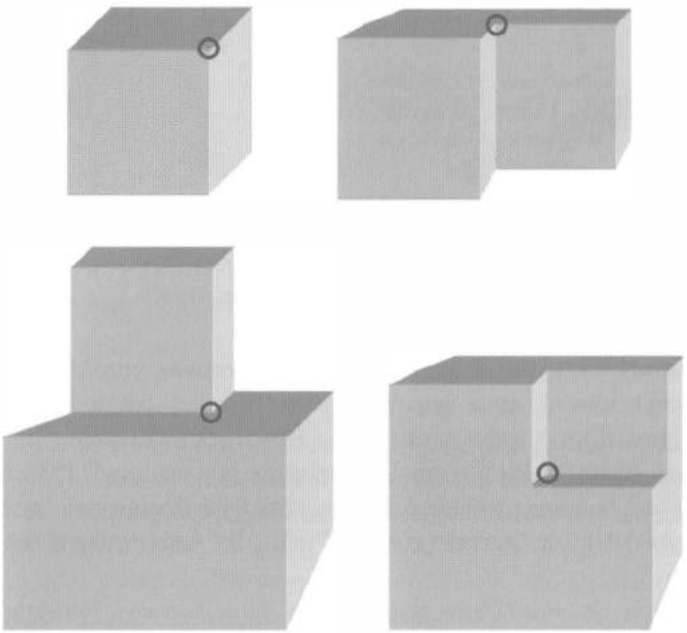
Guzman'ın tezinin son sürümünde bildirilen SEE'nin sonraki uzantıları, görüntü alma yöntemleri içeriyordu. Fakat daha önce bahsettiği gibi bu görüntüler, bilerek hazırlanmış manzaraların görüntüleriydi:[24]

> Başta SEE, elle çizilmiş görüntüler, "kusursuz" görüntüler (çizimler) üzerinde çalışıyordu...
>
> Daha sonra, çok yüzlü (genelde çarpık) ahşap cisimler inşa ettim, bunları siyaha boyadıktan sonra kenar çizgilerini özenle beyaza boyadım, birkaçını aynı yere yığıp manzaranın fotoğrafını çektim. Bu resimler tarandı, kenar çizgileri bulunup SEE'ye verildi. Program bu resimler üzerinde gayet güzel iş gördü.

Her ne kadar SEE, oldukça karmaşık görüntülerde cisimleri bulabiliyor olsa da, hatalar da yapıyor ve içlerinde delik olan blokları tanıyamıyordu.

Görüntülerin tanımlanması üzerine çalışan bir sonraki kişi, MIT Elektrik Mühendisliği profesörlerinden David Huffman'dı (1925-1999). (Huffman, henüz MIT'de yüksek lisans öğrencisiyken yaptığı, "Huffman kodlaması" adlı icadıyla meşhurdu; günümüzde bu etkin yöntem, sayısal verilerin sıkıştırılmasını ve iletilmesini içeren pek çok uygulamada kullanılır.) Hangi tür nesnelerin hangi tür çizimlere karşılık gelebileceği hakkında Guzman'ın eksik çözümlemesi diye nitelediği şey, Huffman'ın canını sıkıyordu. 1967'de MIT'yi, Santa Cruz California Üniversitesi'nde

24 Kişisel iletişim, 14 Eylül 2006.

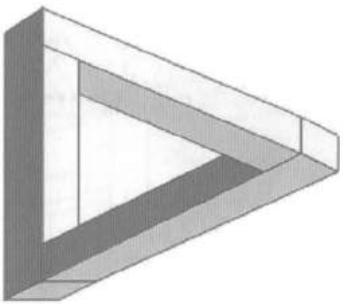


Resim 9.13. Üç yüzeyli cisimlerin dört farklı köşe türü.

Bilgi ve Bilgisayar Bilimi profesörü olmak üzere bıraktıktan sonra, üç yüzlü cisimlerin çizimlerinde çizgilere tanım atayan kuramı tamamladı; bu tür cisimlerde tamı tamına üç düzlemsel yüzey, nesnenin her köşesinde birleşir. Bu tanımlar, düzlemlerin köşelerde birleşme şekline göre değişiyordu. (O zamanlar Huffman'ı iyi tanıyordum, çünkü sık sık Stanford Araştırma Enstitüsü'ne danışmanlık yapardı.)

Huffman, üç düz yüzeyin bir köşede ancak dört şekilde bir araya gelebileceğini göstermiştir.[25] Bunlar Resim 9.13'te gösteriliyor. Bu dört köşe türüne ek olarak, görüntü, Huffman'ın "T-düğümü" dediği şeyleri de içerebilir; bir manzarada başka nesneyi örten nesnenin neden olduğu kesişim çizgileridir bunlar. Bunların hepsi, manzaradaki çizgiler için farklı farklı etiket türleri doğurur; bu etiketler, bu çizgilerin dışbükey, içbükey ya da kesişim kenar çizgilerine denk geldiğini tanımlar.

25 David A. Huffman, "Impossible Objects as Nonsense Sentences", *Machine Intelligence 6* içinde, B. Meltzer ve D. Michie (ed.), Edinburgh: Edinburgh University Press, 1971, s. 195-234; David A. Huffman, "Realizable Configurations of Lines in Pictures of Polyhedra", *Machine Intelligence 8* içinde, E. W. Elcock ve D. Michie (ed.), Chicester: Ellis Horwood, 1977, s. 493-509.



Resim 9.14. İmkânsız bir nesne.

Huffman, bir çizimdeki çizgilerin etiketlerinin, yerelde tutarlı (bazı köşeler çevresinde) ama genelde (tüm köşeler çevresinde) tutarsız olabileceğine dikkat çeker. Örneğin, Resim 9.14'te gösterilen, Roger Penrose'un ünlü "imkânsız nesne" çizimine göz atalım.[26] ("Genel konumu" dolayısıyla hiçbir nesnenin bu görüntüyü üretemeyişinden ötürü imkânsızdır.) Hiçbir "gerçek görüntü"nün, iki farklı etiketi olan çizgisi olamaz.

Britanya Sussex Üniversitesi'nden Max Clowes (yaklaşık 1944-1981), bağımsız olarak benzer fikirler geliştirmiştir,[27] ve günümüzde etiketleme yöntemi genelde Huffman-Clowes etiketlemesi olarak bilinir.

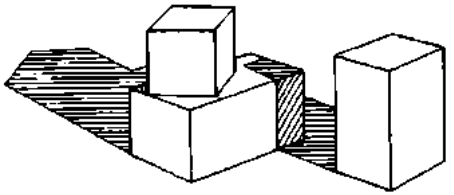
Bunun ardından David Waltz (1943-2012) geliyor. 1972 tarihli MIT doktora tezinde Waltz, Huffman-Clowes çizgi etkiletleme şemasının kapsamını, gölgeleri ve bitişik nesnelerin arasında olası "çatlakları" bulunan görüntülerin çizimini mümkün kılmak amacıyla genişletmiştir.[28] Waltz'ın önemli katkısı, tüm çizgilerin tek ama tek etiket barındırması kısıtlamasını yerine getiren etkin bir hesaplama yöntemi önerip gerçekleştirmesiydi. (Örneğin bir kenar çizgisi, bir ucunda dışbükey öteki ucunda içbükey olamaz.) Resim 9.15'te, Waltz'ın programının doğru şekilde bileşenlerine ayırdığı bir çizim örneği gösteriyorum.

MIT'de çizimlerin işlenmesiyle ilgili yapılmış çalışmaları özetleyen Patrick Winston, "Guzman deneyciydi, Huffman kuramcı, Waltz da

26 Wikipedia'ya göre bu imkânsız nesne ilk olarak İsveçli sanatçı Oscar Reutersvârd tarafından 1934'te çizilmiş.

27 Max B. Clowes, "On Seeing Things", *Artificial Intelligence*, c. 2, 1971, s. 79-116.

28 David L. Waltz, "Generating Semantic Descriptions from Drawings of Scenes with Shadows", MIT YZ Laboratuvarı Teknik Raporu no: AITR-271, 1 Kasım 1972. Şu internet adresinde: https://dspace.mit.edu/handle/1721.1/6911. Bunun özetlenmiş biçimi: Patrick Winston (ed.), *The Psychology of Computer Vision*, New York: McGraw-Hill, 1975, s. 19-91.



Resim 9.15. Waltz'ın programının çözümlediği gölgeli bir manzara. (Çizim, David Waltz'ın izniyle kullanılmıştır.)

ansiklopediciydi (çünkü Waltzman, çatlaklarla ve gölgelerle başa çıkabilmek uğruna binlerce kesişim noktası kataloglamak zorundaydı)" der.[29]

O zamanlar, üç boyutlu manzaralarda nesneleri bulma, tanıma ve betimleme hakkında benzer çalışmalar Stanford'da da yapılıyordu. 1972 yılına gelindiğinde Elektrik Mühendisliği doktora öğrencisi Gilbert Falk, Guzman'ın tekniklerinin uzantısı olan birtakım teknikler kullanarak, çizim görüntülerini ayrı nesnelere bölüştürebiliyordu.[30] 1973'e gelindiğindeyse, Bilgisayar Bilimi doktora öğrencisi Gunnar Grape, paralel yüzlü ve kama şekilli nesnelerin modellerinden faydalanarak, bu tür nesneler içeren görüntüleri öğelerine ayrıştırmıştı.[31]

Çokyüzlü şekiller içeren görüntülerin çözümlemesiyle ilgili başka çalışmalar, MIT YZ Laboratuvarında konukken Yoshiaki Shriai[32] ve Sussex Üniversitesi Deneysel Psikoloji Laboratuvarında Alan Mackworth tarafından gerçekleştirilmiştir.[33]

29 Kişisel iletişim, 20 Eylül 2006.

30 Gilbert Falk, "Computer Interpretation of Imperfect Line Data as a Three Dimensional Scene", Elektrik Mühendisliği doktora tezi, Stanford Üniversitesi, Yapay Zekâ Raporu, AIM-312 ve Bilgisayar Bilimi Raporu no: CS180, Ağustos 1970. Ayrıca bkz. Gilbert Falk, "Interpretation of Imperfect Line Data as a Three Dimensional Scene", *Artificial Intelligence*, c. 3, 1972, s. 101-144.

31 Gunnar Rutger Grape, "Model Based (Intermediate Level) Computer Vision", Stanford Bilgisayar Bilimi doktora tezi, Yapay Zekâ Raporu AIM-204 ve Bilgisayar Bilimi Raporu no: 266, Mayıs 1973.

32 Yoshiaki Shirai, "A Heterarchial Program for Recognition of Polyhedra", MIT YZ Raporu no: 263, Haziran 1972. Bkz. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-263.pdf.

33 Alan K. Mackworth, "Interpreting Pictures of Polyhedral Scenes", *Artificial Intelligence*, c. 4, no: 2, Haziran 1973, s. 121-137.

10
"EL-GÖZ" ARAŞTIRMALARI

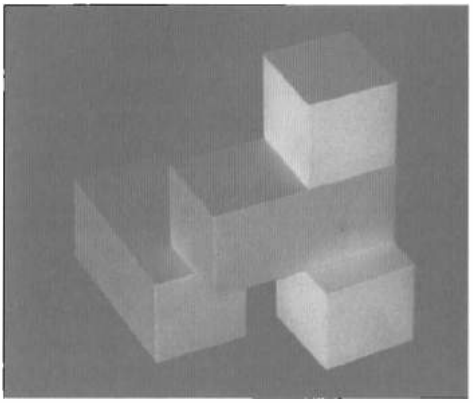
Anlatmış olduğum dönemdeki bilgisayarla görme araştırmalarının büyük kısmı için itici güç, robot kolunu yönlendirecek bilgi sağlamaktı. En iyi çözümlenen görüntüler, oyuncak bloklar gibi basit nesnelerin imgeleri olduğu için, çalışmalar blokları dizip dağıtacak robot kollar üzerinde yoğunlaşmıştı. Bu "el-göz" araştırmasının kimi tipik örneklerini anlatacağım ama, aslında "göz" içermeyen bir projeyle başlıyorum.

10.1 MIT'DE

1961'de Heinrich A. Ernst, MIT'deki Elektrik Mühendisliği doktora projesinin parçası olarak bilgisayar güdümlü mekanik bir "el" geliştirmişti.[1] (Danışmanı Claude Shannon'dı.) MH-1 adı verilen el, "TX-0 bilgisayarına uyarlanmış mekanik bir servomanipülatördü [Amerikan Makine ve Döküm modeli 8]". Yönlendirmek için ele monte edilmiş dokunma sensörleri kullanılıyordu; zira Ernst'in yazdığı gibi, "günümüz şartlarında, görme organları inşa etmek fazlasıyla zor"du. Ernst'in tezinin özeti, sistemin yapabileceklerinin bir bölümünü betimler:

> Dokuz komuttan oluşan bir program mekanik ele şunları yaptıracak: Masa üzerinde kutu aramak, kutunun konumunu hatırlamak, masada oyuncak blok aramak, bunları alıp kutuya yerleştirmek. Nesnelerin konumu, masada oldukları sürece konu dışıdır. Gömme mekanik zekânın sınanması amacıyla, el, blokları ararken, kutu alınıp başka yerlere yerleştirilmeli ve MH-1 kutunun yeni konumunu hatırlayıp bu durumdaki değişikliği, yani blokları ararken kutuya denk geldiğini fark eder etmez bununla çalışmayı sürdürmeli. Bu, o blok-ve-kutu işine özgü programda belirtilme ihtiyacı duyulmaksızın otomatik olarak yapılacaktır.

1 Heinrich A. Ernst, "MH-1, A Computer-Operated Mechanical Hand", doktora tezi, Massachusetts Teknoloji Enstitüsü, Elektrik Mühendisliği Bölümü, 1962. Bkz. http://dspace.mit.edu/bitstream/handle/1721.1/15735/09275630.pdf?sequence=1.



Resim 10.1. MIT kopyalama demosu için blok dizilişi. (Bethold Horn'un izniyle kullanılmıştır.)

Aslında MH-1, bilgisayar güdümlü ilk el değildi, gerçi hareketlerini yönlendirmek amacıyla dokunma sensörleri kullanan ilk cihazdı. Amerikalı mühendis George Devol böyle bir cihazı 1954'te geliştirip patentini almıştı. Bu icada dayanarak mühendis Joseph F. Engelberger'le birlikte Unimation Şirketi'ni kurdular. Kısa süre sonra, New Jersey Trenton yakınlarında General Motors Şirketi Ternstedt tesislerinde "Unimate" adlı ilk sanayi robotlarının prototipini oluşturdular.

MIT'ye dönecek olursak, doktora öğrencileri Patrick Winston (sonradan MIT YZ Laboratuvarı'nda profesör ve yönetici), Thomas O. Binford (sonradan Stanford profesörü), Berthold K. P. Horn (sonradan MIT profesörü) ve Eugene Freuder (sonradan Hampshire Üniversitesi'nde profesör), blok dizilişlerini bire bir taklit etmek amacıyla AMF Versatran robot kolu kullanan bir sistem geliştirmişlerdi. Bloklardan oluşan görüntü önce taranıp, ardından Binford ile Horn'un geliştirmiş olduğu "çizgi bulucu" sayesinde görüntüdeki hatlar belirlenir.[2] Bu hatlar sayesinde görüntüdeki nesneler tanınır, böylece robot kol için, manzaradaki blokları birbirinden ayırma planları yapılır. Ardından robot kol bu planı yürürlüğe koyup blokları özgün dizilişine göre düzenler. Bu sistem Aralık 1970'te çeşitli blok dizilişleri kullanılarak sergilendi. Bu "kopyalama demo"sunun başarıyla ele almış olduğu örnek bir blok dizilişi Resim 10.1'de gösterilmiştir. (Kopyalama demosunu iş başında gösteren *Eye of the Robot* [Robotun Gözü] adlı bir

2 Bildirildiği yer: Berthold K. P. Horn, "The Binford-Horn Line Finder", MIT YZ Raporu 285, Temmuz 1971 (1973 Aralığında gözden geçirilmiştir). Şu internet adresinde: http://people.csail.mit.edu/bkph/AIM/AIM-285-OPT.pdf.

film şuradan izlenebilir: http://projects.csail.mit.edu/films/aifilms/digitalFilms/9mpeg/88-eye.mpg.)

Bu sistem, hassas bir ışıklandırma ve özenle inşa edilmiş bloklar gerektiriyordu. Bilgisayarla görmenin menzilini daha az kısıtlı görüntülere uzatma girişimleri, MIT ve görme işleme çalışmalarının erken aşamalarını yaşayan başka yerlerde daha fazla yoğunlaşmaya yol açtı. Bu sorunlarla ilgili izleyen çalışmaların bazılarını ileride ayrıntılarıyla anlatıyorum.[3]

10.2 STANFORD'DA

Bu esnada John McCarthy'nin SAIL'inde, Profesör Jerome Feldman'ın (1938-) önderlik ettiği ekip, önce PDP-1, sonra PDP-6 ve PDP-10 bilgisayarlarını kullanarak el-göz projeleriyle ilgili çalışmalar yürütmüştür.[4] McCarthy daha sonra bana, bilgisayarla görmeye ilgi duyduğu için robotlarla ilgilendiğini anlatacaktı. Örüntü tanıma çalışmaları onu fazla heyecanlandırmıyordu; ona göre bu iş, "betimlemeden" daha çok "ayrım yapmakla" ilgiliydi. "Eğer bir şeyi elinize almak istiyorsanız, bu nesneyi sırf tanımakla kalmayıp onu betimlemelisiniz de."[5]

1966'da SAIL, Rancho Los Amigos Hastanesi için elektromekanik protez kol yapmıştı. 1967 baharına gelindiğinde, Karl Pingle, Jonathan Singer ve Bill Wichman'ın geliştirmiş olduğu el-göz sistemi, masa üzerine dağılmış blokların yerini belirlemek maksadıyla TV kamerası ve ilkel görme altprogramlarını kullanıyordu. Bu şekilde elde edilmiş bilgiden faydalanan sistem, blokları dizmek amacıyla robot kolu denetleyebiliyordu.[6] Sistemin yaratıcılarına kulak verelim:

3 Patrick Winston, yukarıda bahsedilen MIT programlarıyla ilgili hoş bir betimleme sunar (kopyalama demosu ve Winston'ın nesne dizilişlerinin yapısal tasvirlerini öğrenmeyle ilgili kendi tezi dahil). Bkz. Patrick H. Winston, "The MIT Robot", *Machine Intelligence* 7, Bernard Meltzer ve Donald Michie (ed.), New York: John Wiley & Sons, 1972, s. 431-463.

4 Kısa bir inceleme için bkz. Lester Earnest (ed.), "Final Report: The First Ten Years of Artificial Intelligence Research at Stanford", Stanford Artificial Intelligence Laboratory Memo AIM-228 and Stanford Computer Science Department Report no: STAN-CS-74-409, Temmuz 1973. (bkz. http://www-db.stanford.edu/pub/cstr/reports/c/tr/74/409/CS-TR-74-409.pdf.) Daha fazla ayrıntı için bkz. Jerome A. Feldman ve diğ., "The Stanford Hand-Eye Project", *Proceedings of the IJCAI*, Washington, DC, 1969, s. 521-526; Jerome A. Feldman ve diğ., "The Stanford Hand-Eye Project – Recent Results", Stockholm'de 1974'te IFIP konferansında sunulmuştur.

5 John McCarthy, kişisel iletişim, 11 Ağustos 2007.

6 Bu sistemin anlatıldığı makale için bkz. Karl P. Pingle, Jonathan A. Singer ve William

> Sistemin bir parçası, TV görüntüsünü tarayıp nesnelerin dış kenarlarını bulur, sonra da kenar çizgilerinin yer ve doğrultusunu bulmak amacıyla bir eğim operatörü [kenar çizgisi saptayıcısı] kullanarak nesnenin dış hatlarının izini sürer. Eğrilere oturan altprogramlar, düz çizgileri, dış hatlar üzerindeki noktaların listesine uydurup köşelerin konumunu hesaplar... Sistemin başka bir parçası, robot kolun denetimiyle ilgilidir... Kolu denetleyen kısımlar, her devindiricide gerekli olan açısal konumları hesaplayan bir çözüm programından ve kolu istenen konuma yönlendiren bir servo programından oluşur...

Les Earnest, bunun "robotbilimde ilk görsel geribildirim sistemi" olduğunu yazmıştı, gerçi "blokların sadece dış hatları gözlemleniyordu ve görsel yoklama işi bittikten sonra elin görüş alanından çıkarılması gerekiyordu..."[7]

Blok tanıyıp dizen bu sistemin birkaç sürümü sergilenmiştir. Bunlardan birinde sistem, masa üzerindeki renkli blokların yerini belirleyip bunları mavi ve kırmızı bloklar olarak ayrı raflara dizebiliyordu.[8]

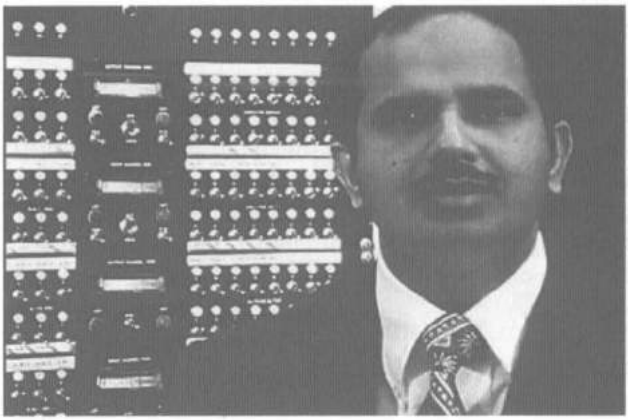
1971'e gelindiğinde görüş güdümlü blok dizme sistemi, "acil delilik" adlı bulmacayı çözebiliyordu.[9] Bu bulmacada her yüzü dört farklı renkten birine boyanmış dört küple öyle bir kule yapılmalıydı ki kulenin her yüzünde dört farklı renk olsun. SAIL'in PDP-10 bilgisayarında çalıştırılan sistem, masa üzerinde dört kübün yerini belirlemek için, dört mercekli bir kulecikle ve renk çarkıyla donatılmış bir TV kamerası kullanıyordu. Kol, her kübü alıp çevirerek tüm yüzlerini kameraya gösteriyordu. Her yüzün rengini bilen ve bulmacanın çözümünü bulmuş olan kol, çözümü sergilemek için küplerden kule yapıyordu. [16-mm'lik renkli, sessiz, altı dakikalık *Instant Insanity* adlı film, 1971'de Richard Paul ile Karl Pingle tarafından çekildi ve Londra'daki İkinci Uluslararası Ortak Yapay Zekâ Konferansında (IJCAI) gösterildi.

M. Wichman, "Computer Control of a Mechanical Arın through Visual Input", *Proceedings of the IFIP Congress (2)*, 1968, s. 1563-1569. Sistemin görmeyle ilgili kısmı içi bkz. William Wichman, "Use of Optical Feedback in the Computer Control of an Arın", Stanford Elektrik Mühendisliği Bölümü doktora tezi, Ağustos 1967 (ayrıca Stanford Yapay Zekâ Raporu AIM-56, 1967.)

7 Lester Earnest, *a.g.e.*

8 Bu tanıma sistemini iş başında gösteren *Butterfinger* adlı 8 dakikalık, 16 mm'lik renkli film 1968'de Gary Feldman tarafından üretilip yönetilmiştir. Film için bkz. http://projects.csail.mit.edu/films/aifilms/digitalFilms/1pm4/09-robot.mp4.

9 Bu sistem şu makalede betimlenmiştir: Jerome Feldman ve diğ., "The Use of Vision and Manipulation to Solve the 'Instant Insanity' Puzzle", *Proceedings of the IJCAI*, Londra: British Computer Society, Eylül 1971, s. 359-364.



Resim 10.2. Raj Reddy. (Fotoğraf, Raj Reddy'nin izniyle.)

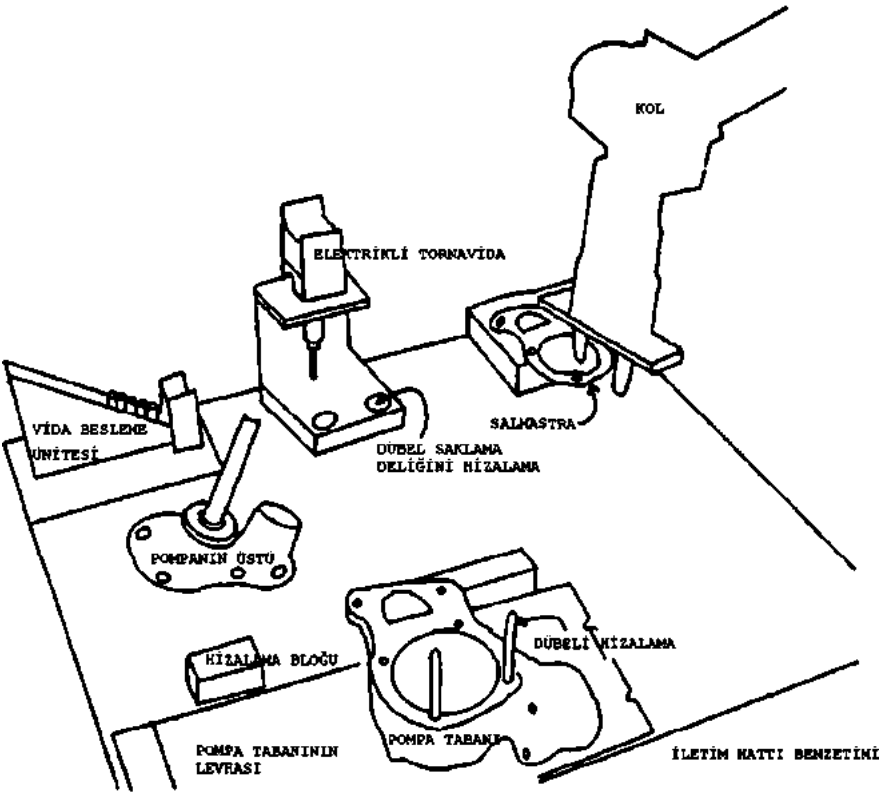
Filmi şu internet adresinde izleyebilirsiniz: http://www.youtube.com/watch?=O1oJzUSITeY.]

Dabblal Rajagopal "Raj" Reddy (1937- ; Resim 10.2), Stanford'un yeni Bilgisayar Bilimi Bölümü'nün ilk doktora öğrencisiydi. Onun tez araştırması, konuşma tanıma üzerineydi. Reddy, 1966'da doktorasını aldıktan sonra, Stanford'un öğretim kadrosuna girdi ve konuşma tanıma araştırmalarına SAIL'de devam etti. Oradayken, ses komutlarıyla bir el-göz sistemini denetleme projesine katılmıştı.[10] Proje incelemesinde bildirildiği üzere: "'Sol alt köşedeki küçük bloğu tut" gibi karmaşık komutlar tanınıyor ve bu görevler, bilgisayarın denetlediği kol tarafından gerçekleştiriliyor"du. (Bu sistem, 1969 tarihli on beş dakikalık, 16 mm renkli ve sesli bir filmde gösterilmiştir; filmde, Reddy'nin konuşma tanımaya dair çalışmalarının sonuçlarından biri sergileniyor. Filmin adı *Hear Here*'dır ve yapımcılığını Raj Reddy, Dave Espar, Art Eisensen üstlenmişlerdir. Şu internet adresinden izlenebilir: http://www.archive.org/details/sailfilm_hear.) Reddy, 1969'da CMU'ya geçip orada konuşma tanıma çalışmalarını sürdürdü; ileride CMU'nun Bilgisayar Bilimleri Kürsüsü'nün dekanı olacaktı.

1970'lerin ilk yarısında Stanford ekibi, bir görme sisteminden ve makine mühendisliği öğrencisi Vic Scheinman'ın tasarlamış olduğu yeni bir elektromekanik elden[11] faydalanarak, "T" Ford modeli yarış

10 Bu sistemin anlatıldığı makale için bkz. Les Earnest ve diğ., "A Computer with Hands, Eyes, and Ears", *Proceedings of the 1968 Fall Joint Computer Conference*, Washington, DC: Thompson, 1968.

11 Victor D. Scheinman, "Design of a Computer Manipulator", Stanford YZ Raporu AIM-



Resim 10.3. Su pompası montaj düzeneği. (Çizim, Robert Bolles'in izniyle kullanılmıştır.)

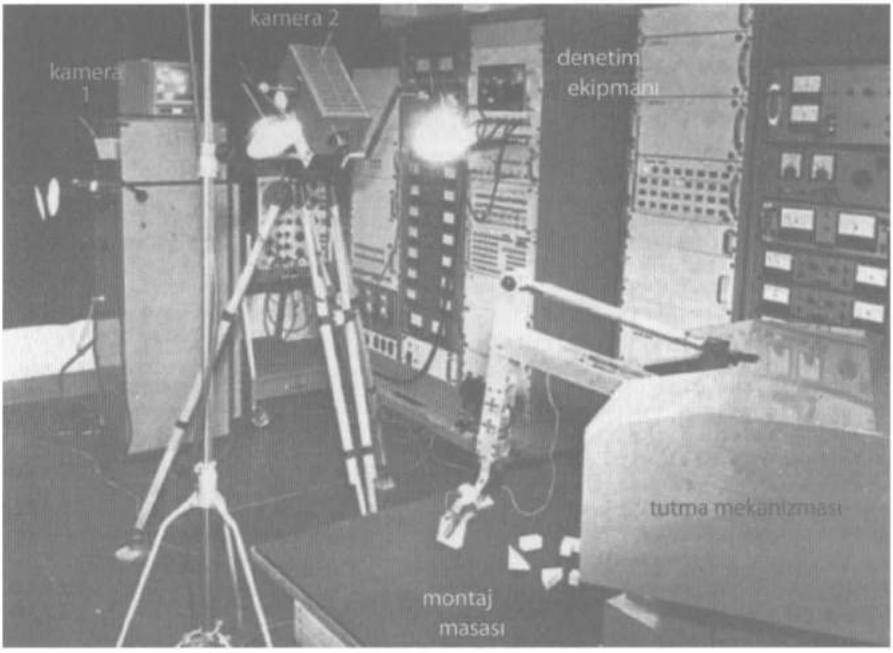
su pompası montajlıyordu.[12] Yatağın üzerine yayılmış sabit konumlu aletler, besleme ünitesindeki vidalar ve pompa gövdesiyle birlikte sanayi tarzı bir düzenek kullanılıyordu. Bu düzeneğin diyagramı Resim 10.3'te gösteriliyor.

El-göz sistemi, önceden hesaplanmış olan şu karmaşık adımları yerine getiriyordu:

> Pompa tabanının yerini belirle, bunu standart konuma getir, dokunma sayesinde son kavrama konumunu belirle, pompa tabanını standart konumuna yerleştir, salmastra ile kapağı aynı hizaya getirmek için iki dübel

92, 1 Haziran 1969.

12 Bu görev şurada anlatılıyor: Robert Bolles ve Richard Paul, "The Use of Sensory Feedback in a Programmable Assembly System", Stanford YZ Laboratuvarı Raporu AIM-220, Stanford Bilgisayar Bilimi Bölümü Raporu STAN-CS-396, Ekim 1973; bkz. ftp://reports.stanford.edu/pub/cstr/reports/cs/tr/73/396/CS-TR-73-396.pdf.



Resim 10.4. Hitachi'nin HIVIP robot montaj sistemi.

yerleştir, salmastrayı koy, görsel olarak salmastranın konumunu yokla, dokunma sayesinde kapağın yerini belirle, kapağı yön gösterici dübellerin üzerine koy, altıgen başlı elektrikli vidalama cihazını al, besleme ünitesinden vida al, ilk iki vidayı mıhla, hizalama dübellerini çıkar, son dört vidayı mıhla, son olarak rotoru döndürmek için gerekli gücü yokla.

Su pompası düzeneğinin filmi şu internet adresinde izlenebilir: http://www.archive.org/details/sailfilm_pump. Ayrıca, Stanford YZ Laboratuvarının başka filmleriyle birlikte şu adreste de var: http://www.saildart.org/films.

10.3 JAPONYA'DA

El-göz çalışmaları, Hitachi'nin Tokyo'daki Merkez Araştırma Laboratuvarı'nda da yürütülmüştür. Burada Masakazu Ejiri ve çalışma arkadaşları, HIVIP adlı bir robot sistemi geliştirmişlerdi; bu robot, GÖZ, BEYİN ve EL denen üç altsistemden oluşuyordu (bkz. Resim 10.4). GÖZ'ün iki televizyon kamerasından biri, blokların yerleşimini resmeden bir plan çizimine bakıyordu. Öteki kameraysa masadaki bloklara

bakıyordu. Ardından BEYİN, blokların çizimde belirtildiği gibi nasıl dizileceğini belirliyor, EL de bu yerleştirmeyi yapıyordu.[13]

10.4 EDİNBURGH'UN "FREDDY"Sİ

1960'ların ikinci yarısı boyunca ve 1970'lerin ortalarına dek, Edinburgh Üniversitesi Makine Zekâsı ve Algısı Bölümü'nde Profesör Donald Michie'nin yönetimi altında araştırmacılar, genelde "FREDDY" adını taşıyan robot sistemleri geliştirmiştir.[14] Bunların en çok tanınanı, el-göz sistemi FREDDY II'dir; bunun büyük bir robot kolu ve hareketli bir masanın üzerinde asılı duran iki TV kamerası vardı. Her ne kadar kol odaya göre hareketsiz olsa da, kendi "dünyası", yani masa üzerinde hareket ediyordu. Bu düzenek Resim 10.5'te gösteriliyor.

FREDDY II'nin gösteri görevlerinden biri, masaya yığılmış parçalardan oyuncak araba ya da tekne gibi tam montajlar üretmekti. Burada amaç, daha iyi endüstriyel montaj robotları, yani o günün kullanılan robotlarına kıyasla daha becerikli, güvenilir ve daha kolay programlanan robotlar için zemin oluşturabilecek YZ teknikleri geliştirmekti.

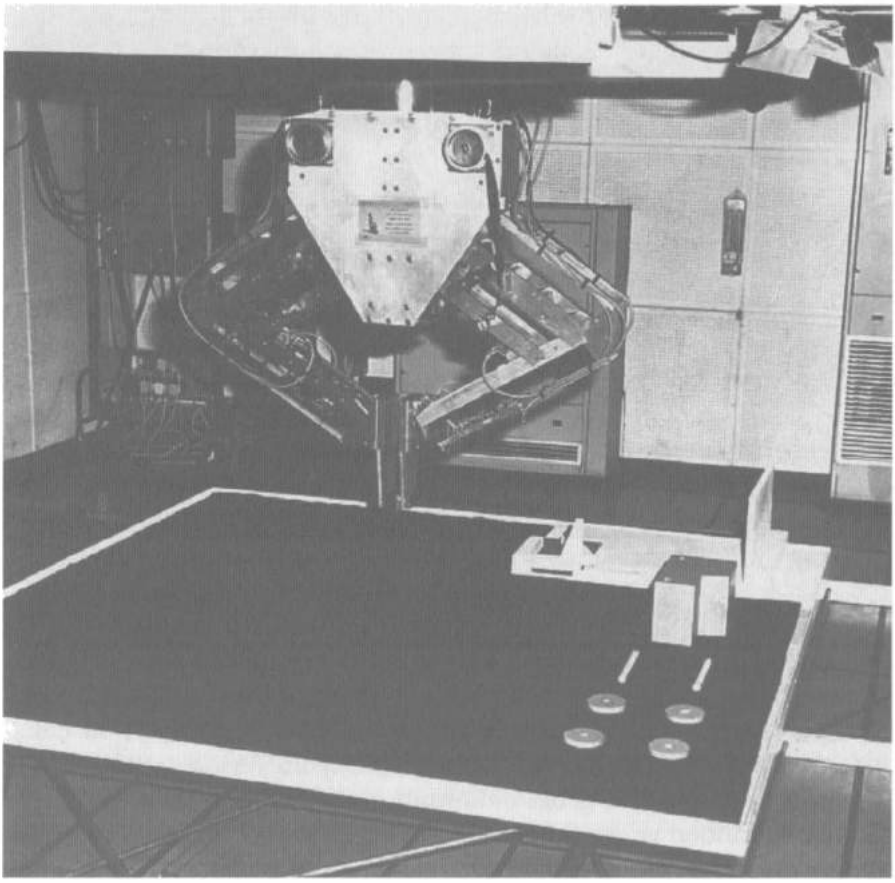
Başlangıçta bileşimler, düzensiz bir yığın halindedir. FREDDY, bunları bulup tanımalı ve düzenli bir sıraya koymalı. FREDDY, gerekli tüm parçaları bulduktan sonra, bir kıskaçla birlikte küçük bir çalışma istasyonundan faydalanarak montaj dizisini yerine getirir.

Ayrı ayrı parçalar, genel hatlarının özelliklerinden (köşeler, eğriler vs.), deliklerinden ve genel niteliklerinden tanınır. Bu aksam FREDDY'ye, önceki eğitim aşamasında her parça farklı açılardan gösterilerek öğretilmişti.

FREDDY, parça yığınlarıyla uğraşırken birtakım taktikler uyguluyordu: Yığın içinde ucu görünen parçaları bulup kavrayarak çıkarmaya çalışmak; en üstten (bilinmeyen) bir nesneyi havaya kaldırmak; ya da yığını küçük öbeklere bölmek amacıyla yığını elle saban gibi sürmek.

13 Bkz. Masakazu Eijiri ve diğ., "An Intelligent Robot with Cognition and Decision Making Ability", *Proceedings of the IJCAI*, Londra: British Computer Society, Eylül 1971, s. 350-358; Masakazu ve diğ., "A Prototype Intelligent Robot That Assembles Objects from Plan Drawings", *IEEE Transactions on Computers*, c. 21, no: 2, Şubat 1972, s. 161-170.

14 Donald Michie'ye göre bu, Frederick'in kısaltması; Frederick ise Eğitim, Sohbet ve Eğlence, Enformasyon Anımsama, Bilgi Derleme sözcüklerinin İngilizcelerinin baş harflerinden oluşturulmuş bir addır.



Resim 10.5. FREDDY II, Edinburgh Üniversitesi robotu. (Fotoğraf, Edinburgh Üniversitesi'nin izniyle.)

Düzeneği montajlamak, eğitim aşamasında etkileşimli biçimde programlanmış olan bir komutlar dizisi izlenerek gerçekleştirilir. Komutlardan bazıları basit devinimlerdi, fakat çok daha gelişmiş olup eldeki sensörlerin gücünden faydalanan komutlar da vardı. Örneğin, "kısıtlı bir devinimde" el, tuttuğu aksamı, dirençle karşılaşana dek bir yüzey üzerinde kaydırır; "deliğe oturtma" deviniminde el, bir parçayı (şaft gibi), başka bir parçanın (çark) deliğine, insanlar gibi, el yordamıyla yerleştirir.

FREDDY, iki takımın aksamı birbirine karıştırılıp masaya yığılsa bile hem araba hem de tekneyi montajlayabilmiştir. Bunu yapması yaklaşık dört saat sürmüştü; bunun başlıca sebebi, FREDDY'nin iki

bilgisayarının sınırlı gücüydü.[15] Sistemin ana bilgisayarı, 64k 24-bit sözcüklük (sonra 128k'ye çıkarıldı) ve 0,5 MHz saat hızlı Elliot 4130 bilgisayarıydı. Bu bilgisayar, POP-2 diliyle programlanmıştı. Robotun motorları ve kameraları, 4k 16-bit sözcüklük Honeywell H316 bilgisayarıyla denetleniyordu (ileride 8k sözcüğe yükseltilecekti; ek 8k byte bedeli yaklaşık 8.000 dolardı).

Bu çalışmanın kilit isimlerinden Harry Barlow (1943-), FREDDY'nin işleyişiyle ilgili şu açıklamada bulunmuştu:[16]

> TV kamerasından görüntü elde etmek birkaç saniye, görüntüyü işlemekse daha da uzun sürüyordu ve tek işletimde FREDDY 100 ve 150 arasında resim çekebiliyordu! Başarıyla havaya kaldırdığı her nesneyi yoklamak üzere kavradığında resmini çekiyordu, başarıyla kaldıramadığında ise yere bırakıyordu; üstelik bir alanın boş olduğunu doğrulamak için bir nesneyi oraya her koyduğunda da resim çekiyordu. Ayrıca, haritasını çıkarmak amacıyla bütün ortamı birkaç kez tarıyordu (bu da birden fazla resim çekmeyi gerektiriyordu) ve nesnelerin konumunu ve boyutunu stereo olarak kestirebilmek için her nesneye iki farklı kameradan bakıyordu. Aslında bu sistem, dünyadaki herhangi bir robot sistemine kıyasla görüntü verilerini en yoğun biçimde kullanır. Stanford ve MIT sistemleri, görevlerini yerine getirirken çok az sayıda resim çekiyor, büyük oranda gözü kapalı tahminlere ve işlerin ters gitmemesine bel bağlıyordu. Öte yandan bizler, işlerin ters gidebileceğini varsaymış (nesnelerin düşmesi, yuvarlanması vs.), dolayısıyla sistemimizi epey gürbüz kurmuştuk. O zamanların büyük olasılıkla en ileri el-göz sistemi olduğuna gerçekten inanıyorum.

FREDDY, günümüzde Edinburgh İskoçya Kraliyet Müzesi'nde, oyuncak arabaları ve tekneleri montajlandığını gösteren sürekli tekrar eden bir filmin eşliğinde kalıcı olarak sergileniyor.

Edinburgh'ta el-göz araştırması 1970'lerin ortasında kısmen askıya alınmıştır; bunun nedenlerinden biri, Britanya Bilim Araştırmaları Konseyi'nin yürüttüğü bir araştırmada bu sistemle ilgili beklentilerin olumsuz olarak değerlendirilmesidir. (Bu değerlendirme hakkında ileride bir iki sözüm olacak.)

15 http://www.aiai.ed.ac.uk/project/freddy/. Kilit kaynakça şu: A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall ve R. J. Popplestone, "A Versatile Computer-Controlled Assembly System", *Artificial Intelligence*, c. 6, 1975, s. 129-156.

16 Harry Barrow'dan e-posta notu, 3 Ocak 2009.

11
BİLGİ GÖSTERİMİ VE USLAMLAMA

Bir sistemin zeki olabilmesi için, kendi dünyası hakkında bilgisi ve bu bilgiden sonuç çıkarmaya ya da en azından bu bilgiyle harekete geçmeye yarayan olanakları olmalı. İnsanlar ve keza makineler benzer sebeplerle iç yapılarında bu bilginin, ister proteinle ister silikonla gösteriminin (*represent*) gerekli yollarını barındırmalı. Bilişsel bilimciler ve YZ araştırmacıları, bilgi gösteriminin başlıca iki yöntemi arasında bir fark belirler: yordamsal (*procedural*) ve bildirimsel (*declarative*) gösterimler. Hayvanlarda, örneğin tenis topuna vurmak gibi beceri gerektiren bir eylem için gerekli bilgiye yordamsal bilgi denir, çünkü o özel eylemi eşgüdümleyip denetleyen sinir sistemi devrelerine doğrudan işlenmiştir. Buna benzer olarak, uçaklardaki otomatik iniş sistemleri, kendi denetim programlarında uçuş rotaları, iniş hızı, uçak dinamiği, vesaire hakkında yordamsal bilgi içerir. Bunun aksine, "Kaç yaşındasın?" gibi bir soruyu, bildirimsel bir cümleyle yanıtlarız, örneğin "yirmi dört yaşımdayım" gibi. Doğal olarak bildirimsel bir cümleyle gösterilen her bilgiye, bildirimsel bilgi denir.

YZ araştırmalarında (ve genel olarak bilgisayar biliminde), yordamsal bilgi, o bilgiden faydalanan programlarda doğrudan gösterilir, oysa bildirimsel bilgi, bu bilgiyi simge yapılarında kullanabilecek pek çok farklı programdan aşağı yukarı ayrı olan yine bu simge yapılarıyla gösterilir. Bildirimsel bilgi simge yapılarına örnek, mantıksal yargıları (dünya hakkındaki bilginin gösterilişine dair McCarthy'nin savunduğu türde yargılar) ve anlamsal ağları (Raphael'inkiler ya da Quillian'ınkiler gibi) kodlayan simge yapılarıdır. Tipik olarak yordamsal gösterimler belirli görevler için uzmanlaşmış oldukları için, (o görevleri yerine getirirken) daha verimlidir, oysa farklı farklı programların faydalanabileceği bildirimsel gösterimler genelde daha faydalıdır. Bu bölümde, bildirimsel bilgiyle akıl yürütmeye ve bunların gösterilmesine dair, bu dönemde ortaya konmuş fikirleri anlatacağım.

11.1 SİMGESEL MANTIKTA TÜMDENGELİM

Aristoteles, mantığı, tasım çözümlemesiyle başlatır. 19. yüzyılda George Boole, önermeli mantığın temellerini oluşturdu ve Gottlob Frege, önermelerin parçası olarak iç bileşenler ("terim" deniyordu) içerebilen bir dil önererek mantığın ifade gücünü geliştirdi. Sonradan çeşitli mantıkçıkların sağladığı gelişimler bize, günümüzde yüklemcil biçimsel dizge (*predicate calculus*) dediğimiz şeyi kazandırmıştır; McCarthy'nin, zeki bir sistemin gereksinim duyduğu bilgiyi temsil etmek için önerdiği dil budur.

Yüklemcil biçimsel dizge diliyle ifade edilmiş, Aristoteles tasımlarından biri şöyledir:

1. (∀x)[İnsan(x) ⊃ Ölümlü(x)]
   ("(∀x)" ifadesi, "tüm *x*'ler için" yazmanın bir yoludur; "⊃" ifadesi "ise" yazmanın bir yoludur. "İnsan(x)", "*x*, insandır" yazmanın ve "Ölümlü(x)", "*x* ölümlüdür" yazmanın bir yoludur. Dolayısıyla, ifadenin bütünü, "tüm *x*'ler için, *x* bir insan ise *x* ölümlüdür" ya da "tüm insanlar ölümlüdür" yazmanın bir yoludur.)
2. İnsan(Sokrates)
   (Sokrates insandır.)
3. Öyleyse Ölümlü(Sokrates)
   (Sokrates ölümlüdür.)

"Öyleyse" sözcüğünün ardından gelen üçüncü önerme, mantıkta *tümdengelime* örnektir. McCarthy, zeki bir etmenin (*agent*) özgül bir durumda gereksinim duyabileceği bilginin, daha önce verilmiş genel bilgiden çıkarılabileceğini ileri sürmüştü. Dolayısıyla McCarthy tarzı YZ için, sadece bir dile (belki yüklemcil biçimsel dizge diline) gereksinim duymakla kalmayız, ayrıca bu dildeki yargılardan gerekli tümdengelimleri yapma yöntemine de gereksinim duyarız.

Mantıkçılar, "çıkarsama kuralları" dedikleri şeyi temel alan çeşitli tümdengelim yöntemleri oluşturmuşlardır. Örneğin, önemli çıkarsama kurallarından biri *modus ponens*'dir ("doğrulama kipi"nin Latincesi). P ve P ⊃ Q gibi iki mantık yargımız varsa, Q yargısını çıkarsamakta haklı olduğumuzu ileri sürer.

1960'lara gelindiğinde, yüklemcil biçimsel dizge teoremlerini ispatlamak amacıyla çıkarsama kurallarından faydalanabilen programlar yazılmıştı. Bunların başlıcaları arasında, IBM'de Paul Gilmore'un,[1]

1 Paul C. Gilmore, "A Proof Method for Quantification Theory: Its Justification and

yine IBM'de Hao Wang'ın[2] ve Dag Prawitz'in[3] (günümüzde Stockholm Üniversitesi'ndedir) yazdıkları programlar bulunuyordu. Bunların programları basit teoremleri ispatlıyor olsa da, daha karmaşık teoremleri kanıtlamak, çok daha fazla araştırma gerektirecekti.[4]

İleride Black-Scholes opsiyon fiyatlama denkleminin eş-mucidi olacak[5] Harvardlı doktora öğrencisi Fisher Black (1938-1995), McCarthy'nin kimi fikirlerini hayata geçiren birtakım erken çalışmalar yapmıştı.[6] Fakat McCarthy'nin tavsiyesine daha fazla uyan QA3 sistemini programlayan kişi, Stanfordlı doktora öğrencisi ve SRI araştırmacısı C. Cordell Green olacaktı. Dış dünya hakkında bilgiyi mantıksal yargılar olarak göstermek zor olmasa da, Black'ın çalışmasının zamanında eksik olan şey, bu yargılardan sonuçlara ulaşacak etkin mekanik yöntemlerdi. Green, John Alan Robinson'ın geliştirmiş olduğu yeni ve etkili uslamlama yöntemini kullanabilmişti.

1960'ların ilk yarısı boyunca İngiliz (ve Amerikalı) matematikçi ve mantıkçı John Alan Robinson (1930-2016), özellikle bilgisayarda çalışmaya uygun bir çıkarım (*deduction*) yöntemi geliştirdi. Bu yöntem, Robinson'ın "çözümleme" (*resolution*) dediği çıkarsama kuralını temel alıyordu.[7] Her ne kadar "çözümleme"nin tam anlatımı çok fazla teknik ayrıntı istese de, bu, uygulandığında, iki yargıdan yeni bir yargı türeten bir kuraldır (*modus ponens*'in kural oluşu gibi). Örneğin, $\neg P$

Realization", *IBM Journal of Research and Development*, c. 4, 1960, s. 28-35.

2 Hao Wang, "Proving Theorems by Pattern Recognition", *Communications of the ACM*, c. 4, no: 3, s. 229-243, 1960; Hao Wang, "Toward Mechanical Mathematics", *IBM Journal of Research and Development*, c. 4, 1960, s. 2-21.

3 D. Prawitz, H. Prawitz ve N. Voghera, "A Mathematical Proof Procedure and Its Realization in an Electronic Computer", *Journal of the Association for Computing Machinery*, c. 7, 1960, s. 102-128.

4 Otomatik çıkarım hakkında ek temel bilgi ve tarih bilgisi için bkz. Wolfgang Bibel, "Early History and Perspectives of Automated Deduction", *Proceedings of the 30th Annual German Conference on Artificial Intelligence (KI-2007)* içinde, J. Hertzberg, M. Beetz ve R. Englert (ed.), Yapay Zekâ seminer notları, Berlin: Springer-Verlag, 2007, s. 2-18.

5 1997'de Myron Scholes ve Robert C. Merton, opsiyon fiyatlama çalışmalarından ötürü Nobel ekonomi ödülü aldılar. Black 1995'te kanserden ölmüştü. Nobel Ödülü ölmüş kişilere sonradan verilmez ama ödülün duyurulmasında Nobel komitesi Black'ın kilit rolünü üstüne basa basa belirtmişti.

6 Fischer Black, "A Deductive Question-Answering System", doktora tezi, Harvard Üniversitesi, Haziran 1964. Tekrar baskısı için bkz. Marvin Minsky (ed.), *Semantic Information Processing*, Cambridge, MA: MIT Press, 1968, s. 354-402.

7 John Alan Robinson, "A Machine-Oriented Logic Based on the Resolution Principle", *Journal of the ACM*, c. 12, no: 1, 1965, s. 23-41.

$\vee$ Q ve P yargılarına uygulanan "çözümleme", Q yargısını üretir. ("$\neg$" simgesi, "değildir" yazmanın bir yoludur ve "$\vee$" simgesi "ya da" yazmanın bir yoludur.) "Çözümleme", bu iki yargıda P ile $\neg$P'yi sıfırlamak olarak düşünülebilir. ("Çözümleme" bir tür genel *modus ponens*'tir; $\neg$P $\vee$ Q yargısının mantıken P $\supset$ Q yargısına eşit olmasından anlayabiliriz bunu.) Bu örnek özellikle basit, çünkü yargılarda iç terimler yok. Robinson'ın kilit katkısı, "çözümleme"nin, yüklemcil biçimsel dizgede $\neg$P(x) $\vee$ Q(x) gibi iç terimleri olan genel yargılara nasıl uygulanabileceğini göstermesidir.

"Çözümleme"nin üstünlüğü, bir mantıksal yargılar kümesinden çıkarım yapan programlarla kolayca gerçekleştirilebilmesidir. Bunun için, yargılar öncelikle, mantıkçıların "koşul" dediği şeylerden oluşan özel bir biçime dönüştürülmeli. (Üstünkörü anlatmak gerekirse, koşul, sadece $\vee$'ler ve $\neg$'ler kullanan bir formüldür.) Her mantıksal yargı, koşul biçimine dönüştürülebilir (gerçi John McCarthy gibi kimileri, bu dönüştürme işlemi sırasında, yargıların mantıksal çıkarımlarda en iyi nasıl kullanılacağı hakkında ipuçlarının kaybolabileceğinden yakınır).

"Çözümleme" ilk kez, matematik teoremlerini ispatlayan bilgisayar programlarında kullanılmıştır. (Teknik anlamda "teorem", kesin çözüm gibi bir çıkarsama kuralının "aksiyom" denen mantıksal yargı kümesinin üyelerine ve aksiyomlardan çıkarsanmış yargılara ardı ardına uygulanarak elde edilebilen herhangi bir mantıksal yargıdır.) Argonne Ulusal Laboratuvarı'nın (Lawrence Wos yönetiminde), Austin'deki Texas Üniversitesi'nin (Woody Bledsoe yönetiminde) ve Edinburgh Üniversitesi'nin (Bernard Meltzer yönetiminde) araştırma ekipleri, kesin çözüm tekniğine dayalı, teorem ispatlayan programlar geliştirmeye başlamıştı kısa süre içinde. Bu programlar, önceden "elle" kanıtlanmış teoremleri, hatta bazı yeni ve önceden kanıtlanmamış matematik teoremlerini ispatlayabiliyordu.[8] Bu yeni teoremlerden biri, Herbert Robbins'e ait bir bir varsayımla ilgiliydi; onun varsayımına göre Robbins cebiri, bir Boole cebiridir. Bu varsayımı William McCune, otomatik teorem ispatlayıcı kullanarak 1996'da kanıtlamıştır.[9]

8 Çalışmalarından bazılarının betimlemesi için bkz. Larry Wos, Ross Overbeek, Ewing Lusk ve Jim Boyle, *Automated Reasoning: Introuction and Applications*, 2. baskı, New York: McGraw-Hill, 1992. Daha yeni çalışmalar için Larry Wos'un internet sayfasını ziyaret edin: http://www.mcs.anl.gov/~wos/

9 William McCune, "Solution of the Robbins Problem", *Journal of Automated Reasoning*, c. 19, no: 3, 1997, s. 263-276.

Burada bizi ilgilendiren konuysa, zeki sistemlerin gereksinim duyduğu uslamlamayı otomatik hale getirecek çıkarım yöntemlerinin kullanımıdır. 1968 civarında Green (yine Stanford öğrencisi Robert Yates'in yardımıyla), çözümlemeyi temel alan bir tümdengelim sistemini LISP dilinde programladı; QA3 adlı bu sistem, SRI'nın zaman paylaşımlı SDS 940 bilgisayarında çalışıyordu. (Green'in ilk çabası olup SRI'da Bertram Raphael'in kılavuzluk ettiği QA1, Raphel'in eski SIR sistemini iyileştirme girişimiydi. QA2, Green'in kesin çözüme dayanan ilk sistemiydi ve QA3 bunun daha gelişmiş ve uygulamaya yönelik mirasçısıydı.) "QA" adı, sistemin itici gücü uygulamalarından biri olan "soru yanıtlama"dan (*question answering*) geliyordu.

QA3'ün soru yanıtlama becerisi hakkında Green'in Stanford tezinden alınmış kısa ve açıklayıcı bir örnek sunacağım.[10] Önce sisteme iki yargı verilir, yani:

1. ROBOT(Rob)
   (Rob, robottur.)

2. $(\forall x)$[MAKİNE(x) $\supset$ $\neg$HAYVAN(x)]
   (tüm x'ler için eğer x makine ise x hayvan değildir.)

Ardından sisteme, şu yargıyı çıkarmasını sağlamaya çalışılarak "Her şey hayvan mıdır" diye sorulur:

3. $(\forall x)$HAYVAN(x)

QA3, böyle bir tümdengelimin imkânsız olduğunu bulup "HAYIR" yanıtını vermekle kalmaz, aynı zamanda bu soruya yanıt olarak "karşıt örnek" de verir:

4. x=Rob
   (Bu, $\neg$HAYVAN(Rob) ifadesinin, çıkarması istenen şeyle çeliştiğini gösteriyor.)

10 SRI Teknik Notu olarak ulaşılabilir: C. Green, "Application of Theorem Proving to Problem Solving", Technical Note, 4, AI Center, SRI International, 333, Ravenswoo Ave, Menlo Park, CA 94025, Mart 1969. Çevrimiçi versiyonu: http://www.ai.sri.com/pubs/files/tn004-green69.pdf. Ayrıca bkz. C. Green, "Theorem Proving by Resolution as a Basis for Question-Answering Systems", *Machine Intelligence 4* içinde, B. Meltzer ve D. Michie, Edinburgh: Edinburgh University Press, 1969, s. 183vd ve C. Green, "Applications of Theorem Proving to Problem Solving", yeniden baskısı 1969 tarihli IJCAI konferansı makalesinde, B. L. Webber ve N. J. Nilsson (ed.), *Readings in Artificial Intelligence* içinde, San Francisco: Morgan Kaufmann, 1981, s. 202-222.

Büyük bir mantıksal yargı kümesinden belli bir sonuç çıkarmak amacıyla çözümlemenin kullanımı, tıpkı herhangi bir çıkarsama kuralının kullanımı gibi, pek çok olasılık içinde bu kuralın hangi iki yargıya uygulanması gerektiğine karar verme zorunluluğunu içerir. Ardından, umulan ve istenen sonuç elde edilene dek tekrar tekrar benzer kararlar alınır. Dama oynayan, bulmaca çözen, geometri teoremleri kanıtlayan programlar gibi, çıkarım programları da, çözüm arayışlarında pek çok olasılığı deneme gereksinimiyle karşı karşıya gelir. Başka programlardaki gibi, çıkarım programları için de çeşitli sezgisel arama yöntemleri geliştirilmiştir.

11.2 DURUM KALKÜLÜSÜ

Green, "soru yanıtlamanın" oldukça geniş bir konu başlığı olduğunu fark etmişti. Neredeyse her şey hakkında soru sorulabilir. Örneğin, "sayı listelerini, artan bir düzenle sıralayan program nedir?" diye sorulabilir. "Oyuncak bloklardan kule inşa eden robotların izlemesi gereken adımlar nedir?" diye de sorulabilir. Bu tür soruları yanıtlarken QA3'ü uygulamanın kilit noktası, McCarthy'nin "durum kalkülüsü"nü kullanmaktır.

McCarthy, "durum kalkülüsü" dediği bir mantık sürümünü önermişti; bu kalkülüste, bir şeyin ya da aksinin geçerli olduğu durumu açıkça belirten mantıksal yargılar yazılabilir. Örneğin, oyuncak blok bir durumda başka bir bloğun üzerinde duruyor olabilir ama başka bir durumda öyle değildir. Green, bu McCarthy mantığının sürümünü geliştirmişti; bu sürümde, belli bir şeyin geçerli olduğu durum, o şeyin geçerli olduğunu bildiren ifadede koşullardan biri olarak belirir. Örneğin, S durumunda A bloğunun B bloğu üzerinde durduğunu (başka durumlarda bunun geçerli olmayabileceği göz önüne alınarak) söylemek için Green şöyle yazar:

```
Üzerinde(A, B, S),
```

A bloğunun tüm durumlarda mavi olduğunu Green şöyle yazar:

```
(∀s)Mavi(A, s),
```

A bloğunun B bloğu üzerinde durduğu bir durumun var olduğunu söylemek için Green şöyle yazar:

```
(∃s)Üzerinde(A, B, s).
```

Burada "(∃s)" ifadesi, "öyle bir *s* vardır ki..." yazmanın bir yoludur.

QA3, yargılar çıkarmakla kalmıyordu; varoluşsal yargı denen şeyi (az önce bahsedilen gibi) çıkardığında, "var olduğu" varsayılan şey için bir örnek bulabiliyordu. Dolayısıyla, (∃s)Üzerinde(A, B, s) önermesine ulaştığında, bu çıkarımın hangi durumda geçerli olduğunu da hesaplıyordu. Green, bu değerin, başlangıçtaki durumu, varılan yargının geçerli olacağı duruma çevirecek robot için bir eylem dizisi bağlamında ifade etmenin yolunu tasarlamıştı. Dolayısıyla, örneğin QA3, robot eylemlerini planlamak amacıyla kullanılabiliyordu. Bu amaçla nasıl kullanıldığını ileride göreceğiz.

11.3 MANTIK PROGRAMLAMA

QA3, robot planlama amacıyla kullanılabildiği gibi, basit bilgisayar programları da oluşturulabilir. Green, 1969 tarihli makalesinde şöyle yazar:

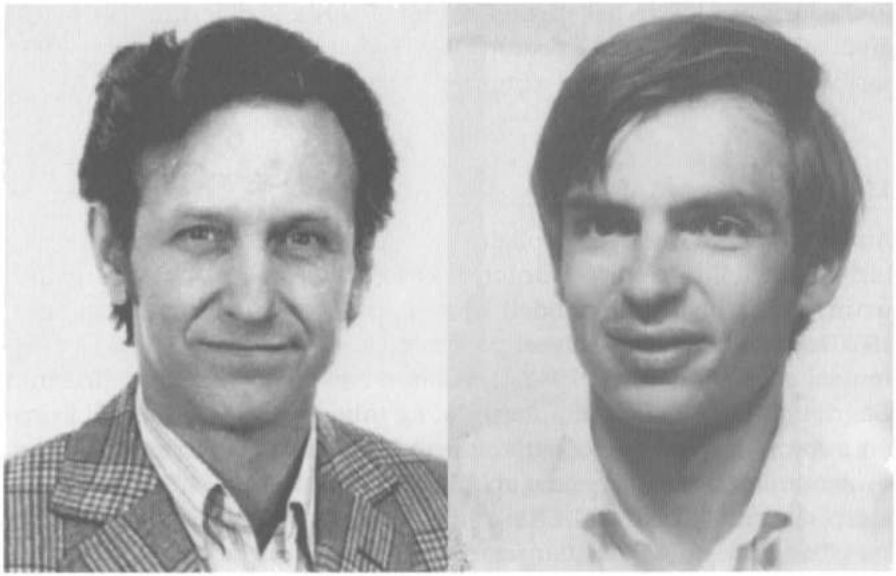
> Burada sunulan formel yaklaşım, özçağrılı programlar dahil otomatik program oluşturma sorununu tam olarak ifade etmek ve çözmek amacıyla kullanılabilir, ayrıca bunun yanı sıra bu programların sağlamasını da yapabilir. Dolayısıyla, bu yöntemle yazılmış hiçbir programda hata yoktur.

Green'in otomatik programlamayla ilgili çalışması, mantıksal yargılar kullanan programlar yazmaya yönelik ilk girişimdi. O dönemde, Edinburgh Üniversitesi'nden doktorasını henüz almış olan Amerikalı Robert Kowalski (1941- ; Resim 11.1) ve Donald Kuehner, Robinson'ın kesin çözüm yönteminin, "SL-çözümleme" dedikleri daha etkin bir sürümünü geliştirdi.[11] 1972 yazında Kowalski, Marsilya'da Centre National de la Recherche Scientifique [Ulusal Bilimsel Araştırma Merkezi] ile Aix-Marseille Üniversitesi'nde Groupe d'Intelligence Artificielle'in (GIA; Yapay Zekâ Ekibi) başı Alain Colmaurer'i (1941-2017 ; Resim 11.1) ziyaret etti. Kowalski şöyle yazmıştır: "Anladığımız haliyle mantık programlama, bu ikinci ziyarette doğdu."[12]

Colmerauer ve doktora öğrencisi Philippe Roussel, yeni programlama dili PROLOG'u 1972'de geliştirdi (Roussel bu adı, "PROgrammation en LOGique" ifadesinin kısaltması olarak seçmişti). PROLOG'da programlar, düzenli bir mantıksal yargı dizisinden meydana geliyordu. Başka yapıların yanı sıra, bu yargıların yazılma sırası, programın

11 Robert A. Kowalski ve Donald Kuehner, "Linear Resolution with Selection Function", *Artificial Intelligence*, c. 2, no: 3-4, 1971, s. 227-260.

12 Kowalski'nin internet sayfalarından birinde: http://www.doc.ic.ac.uk/~rak/history.html.



Resim 11.1. Robert Kowalski (solda) ve Alain Colmerauer (sağda). (Fotoğraflar, Robert Kowalsky ve Alain Colmerauer'in izniyle.)

çalışmasında kilit noktaydı. PROLOG, SL-çözümleme yöntemiyle yapılmış çıkarımların düzenlenişini temel alır. PROLOG'un itibarını Kowalski, Colmerauer ve Roussel paylaşıyor, ama Kowalski şunu kabul eder: "... dürüst olmak gerekirse, benim katkılarımın genelde felsefi, Alain'in katkılarıysa daha uygulamaya yönelikti."[13]

PROLOG dilinin önemi gitgide arttı ve LISP'e rakip oldu, gerçi çoğunlukla ABD dışındaki YZ araştırmacıları kullanıyordu bunu. Bazı Amerikalı araştırmacılar, özellikle de MIT'dekiler, PROLOG'a (ve çözümlemeye dayalı öbür çıkarım sistemlerine) karşı çıkıyor, tümdengelime dayalı hesaplamanın verimli olmadığını ileri sürüyordu (haklı bazı gerekçelerle). Aramayı azaltmak amacıyla, çözülecek problem ve bu problemin çözümü hakkındaki bilginin doğrudan program içine yerleştirilmesiyle denetlenen hesaplamaları savunuyorlardı. "Bilginin bu şekilde yordamsal olarak gömülmesi", Carl Hewitt ile çalışma ar-

13 http://www.doc.ic.ac.uk/~rak/history.html. PROLOG'un doğuşu hakkında Colmerauer ile Roussel'in açıklaması için bkz. Alan Colmerauer ve Philippe Roussel, "The Birth of PROLOG", *Programming Languages* içinde, Thomas J. Bergin ve Richard G. Gibson (ed.), New York: ACM Press, Addison Wesley, 1996. Bkz. http://alain.colmerauer.free.fr/ArchivesPublications/HistoireProlog/19november92.pdf.

kadaşlarının MIT'de geliştirmiş olduğu PLANNER dillerinin bir özelliğiydi. (Hewitt, "bilginin yordamsal olarak gömülmesi" sözünü 1971 tarihli makalesinde ortaya atmıştı.)[14]

11.4 ANLAMSAL AĞLAR

Anlamsal ağlar, bildirimsel bilginin gösteriminde önemli formatlardan biriydi (ve hâlâ öyledir). Bunların Ross Quillian tarafından, insanın uzun süreli belleğinin modeli olarak kullanımından bahsetmiştim. 1970'lerde Stanfordlu bilişsel psikolog Gordon Bower (1932-) ve öğrencisi John Anderson (1947-), *Human Associative Memory* [İnsanın Çağrışımlı Belleği] adlı kitaplarında, ağ tabanlı bir insan belleği kuramı sunmuşlardı.[15] Anderson'ın yaşamöyküsünün özetine göre kitap, "o zamanlar bu alanda çalışan herkesin hemen dikkatini çekti. Bu kitap, önermeli anlamsal ağların, bellekte gösterimin temeli olarak ve bu gibi ağlardaki bağlantılar üzerinden yayılan etkinliğin, bellekte bilgi erişiminin temeli olarak belirlenmesinde önemli rol oynamıştır."[16]

Bu kuram, HAM (Human Associative Memory'nin [İnsanın Çağrışımlı Belleği] kısaltması) adlı bilgisayar benzetiminde kısmen hayata geçirilmişti. HAM, basit önermeli cümleleri çözümleyip bunları anlamsal ağ yapısında saklayabiliyordu. HAM, biriktirdiği belleği kullanarak basit soruları yanıtlayabiliyordu.

1960'ların ikinci yarısı ve 1970'lerin ilk yarısı boyunca ağ tabanlı başka gösterimler keşfedilmişti. Robert F. Simmons, SDC'den Austin Texas Üniversitesi'ne geçtikten sonra, doğal dili bilgisayarların anlaması için gereken yapıların ve işlemlerin bir hesaplamalı dilbilim kuramı olarak anlamsal ağları kullanmaya başladı. Şunu yazmıştır: "Anlamsal ağlar, bilgisayarlardaki faydalı dil işleme uygulamalarını destekleyen rahat bir hesaplama biçiminde, İngilizce söz dizilerinin anlamlarının kimi yönlerini temsil eden basit, hatta zarif yapılardır."[17]

14 Bkz. Carl Hewitt, "Procedural Embedding of Knowledge in PLANNER", *Proceedings of the Second International Joint Conference on Artificial Intelligence*, s. 167-182, Los Altos, CA: Morgan Kaufmann Publishing, Co., 1971.

15 John R. Anderson ve Gordon H. Bower, *Human Associative Memory*, Washington, DC: Winston and Sons, 1973.

16 Şu internet adresinden: http://rumelhartprize.org/john.htm.

17 Robert F. Simmons, "Semantic Networks: Computation and Use for Understanding English Sentences", *Computer Models of Thought and Language* içinde, Roger Schrank ve Kenneth Colby (ed.), San Francisco: W. H. Freeman and Co., 1973, s. 63-113.



Resim 11.2. Roger Schank. (Fotoğraf, Roger Schank'in izniyle.)

O zamanlar Madison'daki Wisconsin Üniversitesi'nde olan Stuart C. Shapiro (1944-), 1971'de, anlamsal bilgi depolayan MENS (MEmory Net Structure) adlı ağ yapısını tanıttı.[18] MENTAL (MEmory Net That Answers and Learns [Yanıtlayıp Öğrenen Bellek Ağı]) adlı yardımcı sistem, kullanıcıyla ve MEMS'le etkileşime giriyordu. MENTAL, saklanmış bilgiden yola çıkarak yeni bilgiye ulaşmada MEMS'e yardımcı oluyordu. Shapiro, MENTAL'in, MEMS'de saklanmış bilgiden faydalanarak kullanıcı sorularına yanıt verebileceğini düşünmüştü.

Shapiro daha sonra Buffalo'daki New York Devlet Üniversitesi'ne geçmiştir; burada çalışma arkadaşlarıyla birlikte SNePS (Semantic NEtwork Processing System) adlı bir dizi sistem geliştirmeyi sürdürüyor.[19] SNePS, mantıksal gösterimlerin özellikleriyle ağ gösterimlerininkileri birleştirir, ayrıca doğal dili anlayıp üretme ve başka uygulamalar için de kullanılmıştır.[20]

Roger C. Schank (1946- ; Resim 11.2), Austin kentindeki Texas Üniversitesi'nde dilbilim konulu doktora tezi araştırmalarında, "doğal dil cümleleri için kavramsal bağımlılık gösterimleri" dediği şeyi geliştirmeye başlamıştı.[21] Daha sonra, Stanford ve Yale'de profesörken, mes-

18 Stuart C. Shapiro, "A Net Structure for Semantic Information Storage, Deduction and Retrieval", *Proceedings of the Second International Joint Conference on Artificial Intelligence*, Los Altos, CA: Morgan Kaufmann Publishing Co., 1971, s. 512-523.

19 Eski makalelerden biri: Stuart C. Shapiro, "The SNePS Semantic Network Processing System", *Associative Networks: The Representation and Use of Knowledge by Computers* içinde, Nicholas V. Findler (ed.), New York: Academic Press, 1979, s. 179-203.

20 SNePS internet sayfası: http://www.cse.buffalo.edu/sneps/.

21 Roger C. Shank, "A Conceptual Dependency Representation for a Computer-Oriented Semantics", doktora tezi, Austin Texas Üniversitesi, 1969. Stanford YZ Raporu 83 ya da Bilgisayar Bilimi Teknik Notu 130, Bilgisayar Bilimi Bölümü, Stanford Üniversitesi,

lektaşlarıyla birlikte bu fikirleri geliştirmeyi sürdürdü. Schank'ın çalışmasının temeli, aslen ifade edildikleri lisandan bağımsız olarak doğal dil cümlelerinin insanlar tarafından "kavramsal yapılara" dönüştürüldüğüne inanmasıydı. Onun iddiasına göre bu kavramsal yapılar, cümlelerdeki bilginin anlaşılma ve anımsanma şekliydi. Örneğin, dilden dile cümle çevrilirken, önce bu cümlenin bilgi içeriği kavramsal yapı olarak gösterilir, sonra da söylenen şey hakkında akıl yürütmek ya da bilgiyi başka bir dilde cümle olarak ifade etmek için bu yapıdan faydalanılır. Makalelerinden birinde Schank'ın dediği gibi: "Aynı anlama gelen herhangi iki söz, ister aynı dilde ister ayrı dillerde olsunlar, kavramsal yapılar tarafından yalnızca tek biçimde tanımlanabilir."[22]

Schank'ın kendi kavramsal yapıları (zaman zaman "kavramsal bağımlılık grafikleri" denir) için kullandığı gösterim, 1970'lerde şekillenmiştir.[23] Makalelerinden alınmış bir örnek, Resim 11.3, *John threw the pencil to Sam* (John Sam'e kurşunkalem fırlattı) cümlesini nasıl temsil ettiğini gösteriyor. Bu yapı, Schank'ın bu gösterimler için tanımlamış olduğu "ilkel eylemler"den üçünü kullanıyor. Bunlar, sahip olmanın geçişi anlamına gelen ATRANS; fiziksel yerin geçişi anlamına gelen PTRANS; ve nesneye güç uygulamak anlamına gelen PROPEL'dir. Schank, devinim, ilgi gösterme, konuşma, fikir aktarma, vesaireyi göstermek amacıyla başka ilkel eylemler de belirlemişti.

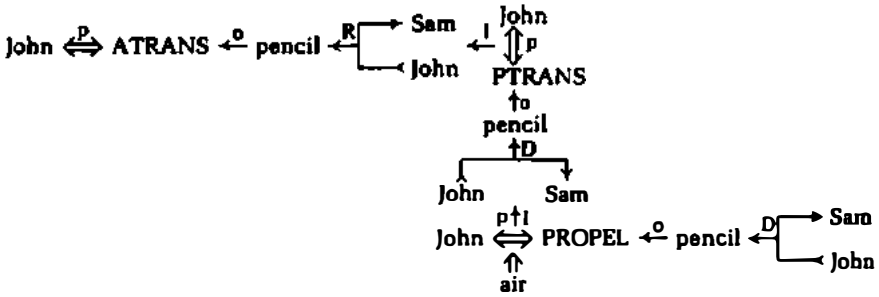
Resimdeki yapının gösterdiği şeyi bire bir okursak, anlamı yaklaşık şöyle olur: "John, kurşunkaleme fiziksel kuvvet uyguladı; bu kuvvet sayesinde kurşunkalem, John'un bulunduğu yerden Sam'in bulunduğu yere doğru havadan ilerledi, böylece Sam'in sahip olmasına neden oldu". Anlam gösterimi dilleriyle ilgilenen pek çok kişi gibi Schank de, bu gösterimlerin, çıkarım yapmak ve sorular yanıtlamak amacıyla doğrudan kullanılabileceğini belirtir. Örneğin, "Sam kurşunkalemi nasıl elde etti?" ve "John fırlattıktan sonra kurşunkalem kime geçti?" gibi soruların yanıtlarına kolayca ulaşılabilir.

Her ne kadar ağ yapıları, bunlar hakkındaki makalelerde grafiklerle açıklansa da, bilgisayarlarca işlenmeleri için LISP dilinde kodlanırlar.

Stanford, CA, 1969.

22 Roger C. Schank, "Identification of Conceptualizations Underlying Natural Language", *Computer Models of Thought and Language* içinde, Roger Schank ve Kenneth Colby (ed.), s. 187-247, San Francisco: W. H. Freeman and Co., 1973.

23 İlgili okurlar bu sözü, Schank'ın çeşitli kitaplarına ve makalelerine atfedebilir; örneğin, Roger C. Schank, "Conceptual Dependency: A Theory of Natural Language Understanding", *Cognitive Psychology*, c. 3, 1972, s. 552-631 ve Roger C. Schank, *Conceptual Information Processing*, New York: Elsevier, 1975.



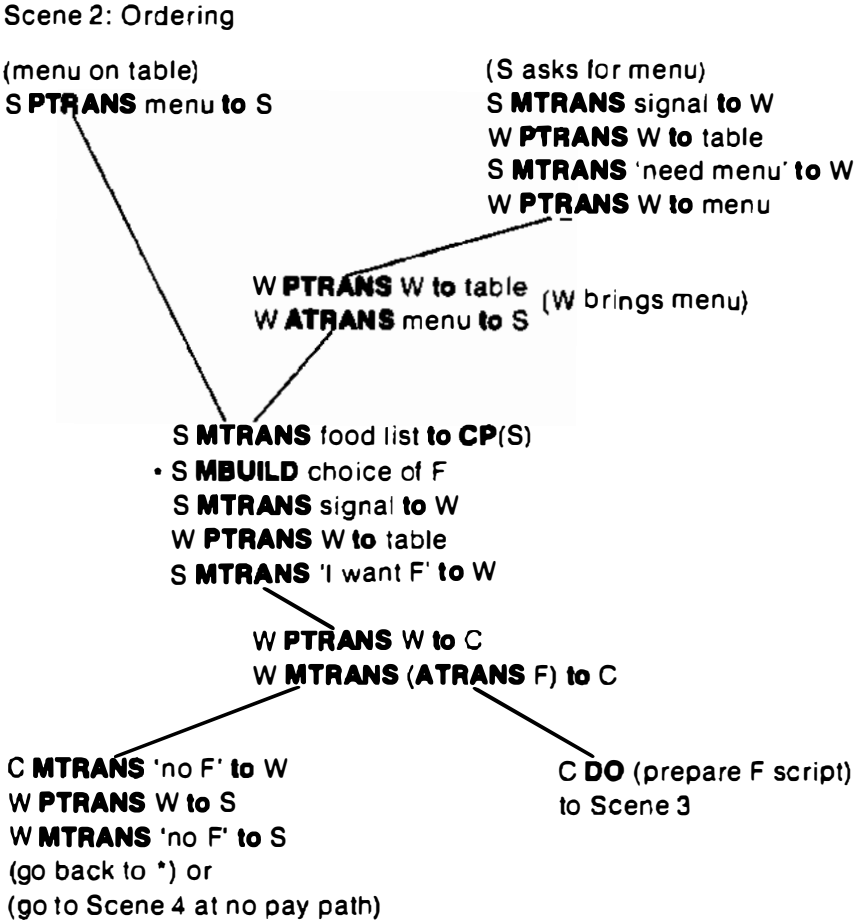
Resim 11.3. *John threw the pencil to Sam* (John Sam'e kurşunkalem fırlattı) cümlesinin kavramsal yapısı. (Şu makaleden: Roger C. Schank, "Identification of Conceptualizations Underlying Natural Language", *Computer Models of Thought and Language* içinde, Roger Schank ve Kenneth Colby (ed.), San Francisco: W. H. Freeman and Co., 1973, s. 187-247.)

11.5 KOMUT DOSYALARI VE ÇERÇEVELER

Anlamsal ağlar ve kavramsal yapılar gibi çizgisel bilgi gösterimleri, birbirleriyle ilgili birimleri topluluklar halinde bir araya getirir. Hesaplama bakımından bu gibi kümelenmeler iyidir, çünkü birbirleriyle ilgili birimler genelde aynı uslamlama zinciri içinde yer alır. Böyle bir birime ulaşıldığında, yakınındaki birimlere de ulaşmak kolaylaşır. Roger Schank ile Robert Abelson, "komut dosyası" (*script*) kavramını önererek bu düşüncenin kapsamını genişletmiştir.[24] Komut dosyası, onların "özgül bilgi", yani "pek çok kez yaşadığımız" bir durum ya da olay hakkındaki ayrıntılı bilgi dedikleri şeyi göstermenin bir yoludur. Özgül bilginin karşısına "genel bilgi"yi koyarlar; genel bilgi, pek çok durumda yararlı olan çok miktarda altyapı ya da sağduyu bilgisinden oluşmuştur.

"Restoran" adlı komut dosyaları ("Kafe versiyonu"), verdikleri en ünlü açıklayıcı örnektir. Bu komut dosyası, dört "sahneden" oluşur, yani giriş, ısmarlama, yeme ve çıkış. Komut dosyasının "destek" birimleri Masalar, Mönü, Yemek, Hesap ve Paradır. "Rolleri" ise S-Müşteri, W-Garson, C-Aşçı, M-kasiyer ve O-Mekân Sahibi. "Giriş koşulları" şöyledir: S açtır ve S'nin parası vardır. "Sonuç"ta, S'nin parası azalır, O'nun parası artar, S artık aç değildir ve S memnundur (isteğe bağlı). Resim 11.4, "Ismarlama" sahnesinin komut dosyasını gösteriyor.

24 Roger C. Schank ve Robert P. Abelson, *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1977.



Resim 11.4. Restoran komut dosyasında bir sahne. (Roger C. Schank ve Robert P. Abelson, *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures* içinde, Hillsdale, NJ: Lawrence Erlbaum Associates, 1977, s. 43.)

PTRANS (yer değiştirme) ve ATRANS (mülk değiştirme) eylemlerinin yanı sıra, bu komut dosyası iki ilkel eylem daha kullanıyor: MTRANS (bilgi değiştirme) ve MBUILD (düşünce yaratmak ya da birleştirmek). CP(S), S'nin düşünme işinin meydana geldiği "kavram işlemcisi" anlamına gelir, ve DO, bu düşünmenin ardından gelen şeyin tanımladığı "yapmacık eylem" anlamına gelir. Diyagramdaki çizgiler, komut dosyası içinde olası yolları gösterir. Örneğin, eğer mönü zaten masadaysa, komut dosyası sol üst köşeden başlar, yoksa sağ üst köşeden başlayacaktır. Komut dosyasının büyük oranda anlaşılır olduğunu

düşünüyorum, fakat orta noktasında neler olup bittiğini açıklamada yardımcı olacağım. S, "yemek listesini" ana işlemcisine getirir; burada, yemek tercihine zihnen karar verebilir (oluşturabilir). Ardından S, garsona, "masaya gel" bilgisini nakleder ve garson masaya gelir. Böylece S, kendi yemek tercihi hakkındaki bilgiyi garsona nakleder. Bu işlem, aşçı ya ısmarlanan yemeğin kalmadığını garsona söyleyene ya da yemeği hazırlayana dek sürer. Restoran komut dosyasındaki öteki üç sahne, Schank ile Abelson'ın kitabında benzer şekilde açıklanıyor.[25] Restoran komut dosyasının birtakım çeşitlemeleri (dört farklı restoran türü vs.) de olasıdır.

Komut dosyaları, bir öykü duyduğumuzda otomatik olarak yaptığımız uslamlamayı kısmen açıklamaya yardımcı olur. Örneğin, John'ın kafeye gidip lazanya ısmarladığını duyduğumuzda, mönüde lazanya olduğunu varsayabiliriz. Bir sonraki sefer John'ın başka bir şey ısmarladığını duyduğumuzda, kafede o an lazanya kalmadığını varsayarız. Schank ve Abelson, küçük çocukların bile bu tür komut dosyaları inşa ettiğine ve insanların karşılaştıkları durumlarda yollarını bulup mantık yürütmek için çok sayıda komut dosyası barındırması gerektiği hakkında hikâyeleri kanıt olarak sunarlar.

Schank daha sonra komut dosyalarının ve bununla ilişkili fikirlerin kapsamını başka bir kitapta genişletecekti; o kitabında, komut dosyalarına göre durumları daha çapraşık ve dinamik biçimde betimleyen "bellek örgütleme paketleri" (BÖP) [*memory organization packets* (MOPS)] fikrini tanıtmıştır.[26] Bunların eğitim alanındaki uygulamalarıyla ilgili başka bir kitapta bu fikirlere "yeniden değinecekti", ki Schank, eğitim alanına önemli katkılar yapmıştır.[27]

Schank ve bu iddiaları, YZ araştırmacıları arasında epey tartışma yaratmıştır. Örneğin, 1983 yılında restoranın birinde (acaba mönüyü beklerken miydi?) onunla tartıştığımızı hatırlıyorum; onun programının ve SRI'da bizim programlarımızın doğal dil anlama performanslarını kıyaslamıştık. Anımsadığım kadarıyla, kendi programlarının yapabilecekleri hakkında, benim inanmaya hazır olduğumdan ya da kendi programlarımızla ilgili iddialarımdan çok daha abartılı iddialarda bulunmaya hevesliydi. Tufts Üniversitesi filozoflarından Daniel Dennett'in şu sözü aktarılır: "Schank'ın sinek gibi rahatsız eden, her

25 *A.g.e.*

26 Roger C. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge: Cambridge University Press, 1982.

27 Roger C. Schank, *Dynamic Memory Revisited*, Cambridge: Cambridge University Press, 1999.

şeye karşı çıkan, bilişsel bilim âleminin gerillası havalarını, daima büyük sorular soruşunu, eski çabalarını çöpe atmak isteyip ilginç sebeplerden ötürü bunların kökten yetersiz olduğunu söylemesini hep beğenmişimdir. O sineğin tekiydi ama esaslı bir sinekti."[28] Bana kalırsa, komut dosyalarıyla ilgili temel fikri zamanının ötesindeydi. Ayrıca, çok sayıda öğrenci yetiştirmiştir. "YZ Soykütüğü" (AI Genealogy) internet sitesi,[29] neredeyse dört düzine Schank öğrencisini sıralıyor, ki bunların pek çoğunun seçkin birer meslek yaşantısı olmuştur.

Schank'ın çalışmasıyla hemen hemen aynı zamanda Marvin Minsky, durumlar hakkındaki bilginin, "çerçeve" dediği yapılarla gösterilebileceğini ileri sürmüştü.[30] Schank'ın fikirlerinin (başka fikirler içinde), mantık dilindeki cümleler gibi, "bilgiyi, ayrı, basit parçacıkların bir derlemesi olarak göstermeye çalışmaktan" uzaklaşmanın örneği olduğunu söylemişti. Minsky, Çerçeveyi şöyle tanımlar:

> *Çerçeve*, belirli bir oturma odası türünde bulunmak ya da bir çocuğun doğum günü partisine gitmek benzeri çok bilinen durumları gösteren veri-yapısıdır. Her çerçeveye birkaç bilgi türü iliştirilmiştir. Bu bilgilerden bazıları, çerçevenin nasıl kullanılacağı hakkındadır. Bazılarıysa, hemen şimdi olacaklar hakkında beklentilerle ilgilidir. Kimileri de, eğer bu beklentiler boşa çıkarsa ne yapılacağını tanımlar.
>
> ...
>
> Birbiriyle ilintili çerçevelerin derlemeleri *çerçeve-sistemleri* olarak bağlanır. Önemli eylemlerin etkileri, bir sistemin çerçeveleri arasındaki dönüşümlerle yansıtılır. Bunlar, belirli hesaplamaları az hesapla yapmak, vurgu ve ilgideki değişiklikleri temsil etmek ve "imgelem"in etkinliğini açıklamak amacıyla kullanılır.

Minsky'nin makalesi, çerçeve sistemlerinin, görmeye ve görüntülere, dilsel ve başka tür anlayışlara, bellek oluşturmaya, bilgi erişimi ve denetimine nasıl uygulanabileceğini anlatır. Makalesi fikir bakımından zengin olsa da, Minsky, herhangi bir çerçeve sistemini hayata geçirmemiştir. Bundan birkaç yıl sonra öğrencilerinden bazıları ve kimi eski öğrencileri, çerçeve benzeri sistemler gerçeklemiştir. Bunlardan biri olan FRL'yi (Frame Representation Language [Çerçeve Gösterim

28 Bkz. http://www.edge.org/3rd_culture/bios/schank.html.

29 Bkz. http://aigp.csres.utexas.edu/~aigp/researcher/show/192.

30 Marvin Minsky, "A Framework for Representing Knowledge", MIT AI Laboratory Memo 306, Haziran 1974. Yeniden baskısı: Patrick Winston (ed.), *The Psychology of Computer Vision*, New York: McGraw-Hill, 1975. Şu internet adresinde: http://web.media.mit.edu/~minsky/papers/Frames/frames.html.

Dili]), R. Bruce Roberts ile Ira P. Goldstein geliştirmişti.[31] Daniel Bobrow ile Terry Winograd (Winograd, Papert'in öğrencisiydi), KRL (Knowledge Representation Language [Bilgi Gösterim Dili]) adıyla daha da iddialı bir sistem gerçeklemiştir.[32]

Çerçeve sistemlerinin benimsediği uslamlama tarzı, tıpkı Minsky'nin yapacağı gibi, "özellikle tanımlanmamış" ayrıntılar hakkında varsayımda bulunur, böylece "mantığı devre dışı bırakır". Bu tarz daha önce Raphael'in SIR sisteminde kullanılmıştı (Altıncı Bölüme bakınız), ayrıca bilgi gösteriminde mantık dillerinin kullanımını savunan araştırmacılar da bu tarzı benimsemek için mantığın kapsamını genişletecekti. Yine de, Minsky'nin çerçevelerle ilgili makalesinin son kısmı ("Mantık Yaklaşımının Eleştirisi" başlığını taşıyordu), "sıradan bilgiyi pek çok küçük, başlı başına 'doğru' savlar biçiminde göstermenin uygulanabilirliğinden" insanın (Minsky'yle birlikte) şüphe etmesini sağlayan pek çok sebep sıralar.

31 R. Bruce Roberts ve Ira P. Goldstein, *The FRL Primer*, Massachusetts, Institute of Technology AI Laboratory Technical Report AIM-408, Temmuz 1977. Şu internet adresinde: ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-408.pdf.

32 Daniel G. Bobrow ve Terry A. Winograd, "An overview of KRL, a Knowledge Representation Language", Rapor no: CS-TR-76-581, Bilgisayar Bilimi Bölümü, Stanford Üniversitesi, Kasım 1976. Şu internet adresinde: ftp://reports.stanford.edu/pub/cstr/reports/cs/tr/76/581/CS-TR-76-581.pdf. Daha sonra şu biçimde basılmıştır: Daniel Bobrow ve Terry Winograd, "An Overview of KRL, a Knowledge Representation Language", Cognitive Science, c. 1, no: 1, Ocak 1977, s. 3-46.

12
HAREKETLİ ROBOTLAR

Daha önce anlatmış olduğumuz el-göz sistemleri, "robot" sayılabilir, ama bunlar, sabit konumlarını değiştiremez. Bugüne dek hareketli robotlar konusunda çok az çalışma yapılmıştır, gerçi bilim kurguda hareketli robotlar epey göz önüne çıkar. Grey Walter'ın "tosbağaları"ndan bahsetmiştim; bunlar hareketli özerk robotların ilk çeşitleriydi. 1960'ların ilk yarısında Johns Hopkins Üniversitesi Uygulamalı Fizik Laboratuvarı araştırmacıları "Beast" [Canavar] adını verdikleri hareketli robotu imal etmişti (Resim 12.1). Üzerine yerleştirilmiş elektronik devrelerle denetlenen ve yolunu sonar, ışık sensörleri ve "duvar kirişine dokunan" kollarla bulan bu robot, beyaz duvarlı koridorlarda gezinerek koyu renkli prizleri arardı. Priz bulduğunda, üstelik pilleri de azalmışsa, kendini prize takıp pillerini doldururdu. Bu sistem, Hans Moravec'in bir kitabında betimlenmiştir.[1]

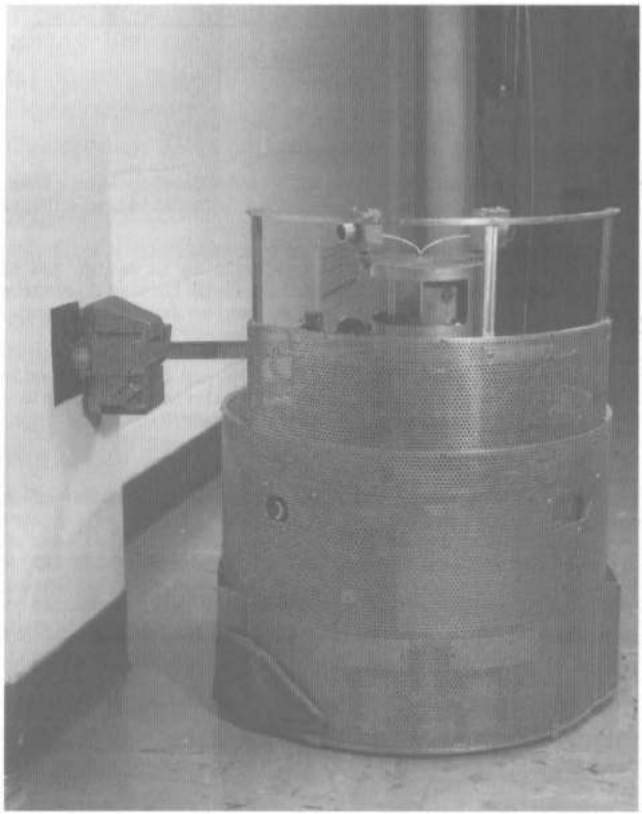
1960'ların ortasından başlayarak birkaç ekip hareketli robotlar üzerinde çalışmaya başladı. Bunlar arasında, SRI ve Stanford YZ Laboratuvarları da vardı. SRI robotunun kapsamlı bir betimlemesiyle başlıyorum; çünkü bu, birtakım önemli YZ teknolojilerinin icadı ve bütünleştirilmesi için itici güç olmuştur.

12.1 SRI ROBOTU SHAKEY

SRI'daki sinir ağı araştırmalarının önderi Charles Rosen, Kasım 1963'te yazdığı bilgilendirme notunda, sinir ağlarının örüntü tanıma ve bellek becerilerini, MIT'de, Stanford'da, CMU'da ve başka yerlerde geliştirilen programlar gibi üst düzey YZ programlarıyla birleştirecek hareketli bir "otomat" geliştirmeyi önermişti. Rosen daha önce, UCLA'da LISP'le ilgili bir yaz dönemi dersine katılmıştı; bu dersi, (SIR'de) doktorasını bitirmekte olan Bertram Raphael vermişti.

Rosen, ben ve ekibindekiler, hemen hareketli robotlar üzerine kafa yormaya başladık. Ayrıca danışman olarak bize yardımcı olması için

1 Hans P. Moravec, *Robot: Mere Machine to Transcendent Mind*, Oxford: Oxford University Press, 1999, s. 18-19.



Resim 12.1. Johns Hopkins "Beast"i (Johns Hopkins Üniversitesi Uygulamalı Fizik Laboratuvarı'nın izniyle.)

Marvin Minsky'yi seçmiştik. Minsky, Ağustos 1964'te SRI'da iki hafta geçirmişti. SRI'de hareketli robot araştırmalarına destek bulmak amacıyla ARPA bürosuna (o zamanlar Pentagon'daydı) düzenlediğimiz pek çok yolculuğun ilkini gerçekleştirdik. Ayrıca Savunma Araştırmaları ve Mühendisliği Dairesi'nin (DDR&E) müdürü Ruth Davis'le de konuştuk; tüm Savunma araştırmaları bu dairenin sorumluluğundaydı. Nisan 1964'te DDR&E'ye "Zeki Otomat Araştırmaları (Faz I)" başlıklı teklifimizi sunduk; iddiamıza göre bu araştırma, "günümüzde insan zekâsı gerektirdiği düşünülen görevleri yerine getiren makinelerin en son amaç olarak geliştirilmesini sağlayacaktı."[2] Birkaç ziyaret ve konuşmayla

2 Bu teklifin nüshası şu internet adresinde: http://www.ai.sri.com/pubs/files/1320.pdf. Kapak sayfasında "Hazırlayan Nils J. Nilsson" yazar, ama aslında bir ekip ça-

A RESEARCH AND DEVELOPMENT PROGRAM IN APPLICATIONS OF INTELLIGENT AUTOMATA TO RECONNAISSANCE

Goals

The long-range goal of this program will be to develop automatons capable of gathering processing and transmitting information in a hostile environment. The time period involved is 1970-1980.

The first short-range goal of the program will be to design and develop a mobile automaton to accomplish non-trivial missions in a real environment. External control will be exercised over the automaton from a computer. The automaton will have at least visual and tactile sensor capability.

The second short-range goal will be to design and develop a mobile automaton to accomplish non-trivial missions in a real environment in a self-contained mode, e.g., with little or no external control.

.....

Such a long-range goal attained by stepping through a number of intermediary ones is believed essential to knit together as many of the constituent subject areas of "artificial intelligence" as possible. It has been so stated as to require the successful application of many techniques with all the attendant problems of interaction and feed-back. It is difficult of realization but by the same token it will provide solutions to existing pressing military problems.

Resim 12.2. Otomat çalışma bildirisinin daktilo metninden bir alıntı.

birlikte bu teklif, Kasım 1964'te ARPA Enformasyon İşleme Teknikleri Dairesi'nin o zamanki müdürü Ivan Sutherland'ın tebliğ ettiği "çalışma bildirisi" biçimine ulaştı. Resim 12.2'deki alıntı, bu programın hedeflerini betimliyor.[3]

Bu sırada Bertram Raphael, 1964'te MIT'de doktorasını tamamlayıp tek akademik yıl için UC Berkeley'de çalışmaya başlamıştı. Nisan 1965'te SRI'ye katılma teklifimizi kabul edip ekibimize gereken YZ uzmanlığını sağlamış oldu. Birkaç araştırma teklifinden ve Savunma Bakanlığı'ndaki ilişkili dairelerde kimi insanlarla konuştuktan sonra (o dönem Ivan Sutherland'in ARPA'dan ayrılmış olması işleri güçleştirmişti), SRI sonunda, temelde Sutherland'ın çalışma bildirisini temel alan görece büyük (o zamanlar için) bir sözleşmeyle ödüllendirildi. New York eyaletinin Rome şehrindeki Rome Havacılık Geliştirme Merkezi (RADC) tarafından ARPA için belirlenmiş olan, projenin "işbaşı" tarihi 17 Mart 1966'ydı. (Rastlantıya bakın ki, 1961'de SRI'ye katılmadan

lışmasıydı ve buradaki fikirlerin pek çoğu, Rosen'ın 1963 tarihli notunda geçen fikirlerin incelik kazandırılmış haliydi.

3 Bu bildirinin tamamı için bkz. http://ai.stanford.edu/~nilsson/automaton-work-statement.pdf.

önce, RADC'de Havacı Kuvvetleri Teğmeni olarak radar sistemleri için istatistiksel sinyal işleme teknikleriyle ilgili üç senelik görevimi henüz bitirmiştim.) Ruth Davis, projenin başlamasında ARPA ve RADC'yi devreye sokmakta belirgin rol oynamıştı. Kimi ayrı YZ tekniklerini "birbirine dokumak", SRI otomasyon projesinin önündeki başlıca zorluktu ama en büyük katkılarından biri bu olmuştur.[4]

Projede görevlerden biri, eylemleri bir dizi program tarafından denetlenecek bir robotu fiziksel olarak gerçekleştirmekti. Mühendisliğe özgü kimi tuhaflıklar yüzünden, ani duruşlarda bu cihaz sallanıyordu. Dolayısıyla cihaza "Shakey" [Titrek] der olmuştuk, gerçi araştırmacılardan biri bu lakabın saygısızca olduğunu düşünüyordu. [Shakey, 2004 yılında "Ünlü Robotlar Geçidi"ne alındı (C-3PO ve başka robotlarla birlikte).[5] Ayrıca *Wired Magazine,* Ocak 2006'da Shakey'yi (elli robot arasından) en iyi beşinci robot seçti. *Wired*'ın 2 ve 4 numaraları kurmaca robotlardı, "Spirit" ve "Oppurtunity" (Mars robotları) 3 numaraydı ve (2005 DARPA "Büyük İddia"nın kazananı) "Stanley" gelmiş geçmiş en iyi robot seçilmişti. Günümüzde Shakey, California Mountain View'deki Bilgisayar Tarihi Müzesi'nde sergileniyor.][6]

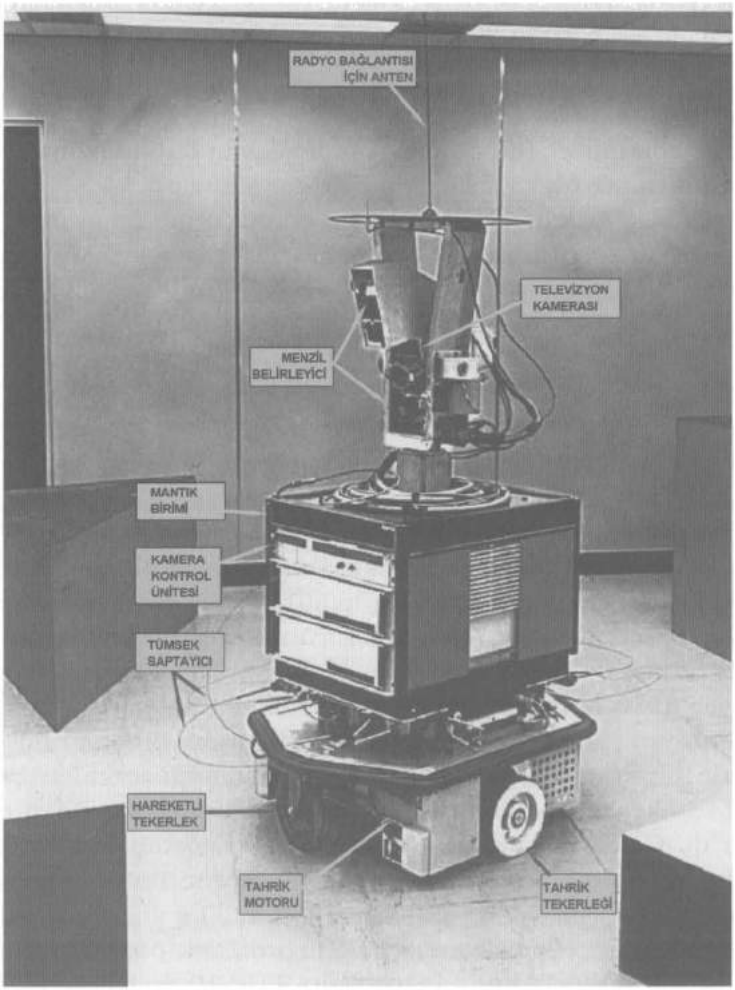
Shakey'nin üzerinde, çevresini görüntüleyen bir televizyon kamerası, duvarlara ve başka nesnelere uzaklığını ölçen lazerli bir mesafe belirleyici (uçuş süresine değil, nirengi noktalarına göre) ve kedi bıyığına benzeyen tümsek saptayıcıları bulunuyordu. Shakey'nin çevresi, geçitlerle birbirine bağlanan ama normalde biz üzerinden bakabilsek de Shakey'nin bakamadığı alçak duvarlarla birbirinden ayrılan "odalar"dı. Odaların bazıları, Resim 12.3'teki gösterildiği gibi büyük nesneler içeriyordu. Shakey'nin boyutunu, Resim 12.4'e bakarak kestirebilirsiniz.

Shakey'yi denetlemek amacıyla geliştirdiğimiz programların çoğu, DEC PDP-10 bilgisayarında çalışıyordu. PDP-10 ile hareketli cihazın kendisi arasında PDP-15 adlı alt bilgisayar (alt düzey iletişimi ve donanıma gönderilen komutları idare etmek için) ve iki yönlü radyo ile video bağlantısı vardı. PDP-10 programları, "üç katmanlı" hiyerarşi dediğimiz yapı içinde örgütlenmişti. En alt düzeydeki programlar, tüm motorları çalıştırıyor ve duyusal bilgi topluyordu. Ara düzeydeki programlar, önceden belirlenmiş bir konuma gitmek gibi ilkel eylemleri

4 SRI'nın otomasyon projesiyle ilgili tekliflerinin, izleyen ilerleme raporlarının ve ilgili makalelerin çevrimiçi nüshaları için bkz. http://www.robothalloffame.org/.

5 Bkz. http://www.robothalloffame.org.

6 Bkz. http://computerhistory.org/timeline/?category=rai.



Resim 12.3. Shakey, Kasım 1968'deki haliyle (bazı bileşenleri etiketlenmiş). (Fotoğraf, SRI International'ın izniyle.)

denetliyor, ayrıca Shakey'nin TV kamerasından gelen görüntüleri işliyordu. Bir dizi ara düzey eylemin yerine getirmesini gerektiren daha karmaşık eylemleri planlama işi, hiyerarşinin en tepesindeki programlar tarafından yapılıyordu. Shakey projesi, arama tekniklerinde, eylemlerin sağlam şekilde denetlenmesinde, planlamada, öğrenmede ve görmede yapılmış birtakım yeni icatların bütünleştirilmesini içeriyordu. Günümüzde bu fikirlerin pek çoğu yaygınlıkla kullanılıyor. İzleyen birkaç altbaşlıkta bu fikirleri anlatacağım.

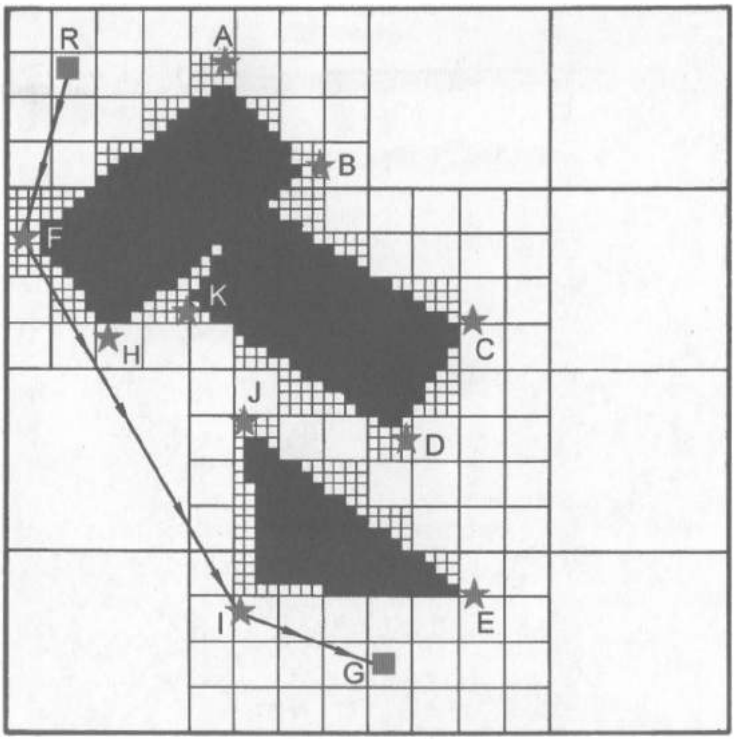


Resim 12.4. Charles A. Rosen'la birlikte Shakey. (Fotoğraf, SRI International'ın izniyle.)

12.1.1 A*: Yeni bir Sezgisel Arama Yöntemi

Üzerinde kafa yorduğumuz ilk sorunlardan biri, Shakey'nin, bir yerden bir yere giderken kullanabileceği "yol noktaları" dizisini nasıl planlayacağıydı. Başlangıç konumu ile hedef konum arasındaki tek engelin etrafından dolaşırken, Shakey önce, yolunu tıkayan engelin kenarında bir noktaya yönelir, sonra da açık hedef noktasına dümdüz ilerler. Ancak, eğer ortam birkaç engelle doluysa, durum daha da güçleşir; dolayısıyla, daha zorlu olan bu soruna genel bir çözüm arıyorduk.

Shakey, engellerin konumu ve kendi konumuyla ilgili bilgiyi, Resim 12.5'te gösterilene benzer bir "ızgara modelinde" tutuyordu. (Gerekli hassasiyeti elde etmek için, ızgara hücreleri nesnelerin yakınla-



Resim 12.5. Shakey için yön bulma problemi. (Çizim, SRI International'ın izniyle kullanılmıştır.)

rında daha küçük hücrelere bölünüyordu. Bence bu, robot devinimi planlamasında uyarlanımsal hücre ayrışımına (*adaptive cell decomposition*) yönelik ilk uygulamalardan biriydi ve günümüzde yaygınlıkla kullanılan bir tekniktir.) Örneğin, Shakey'nin R noktasından G noktasına giderken yönünü bulma problemine göz atalım (burada R ile G, gri karelerle gösteriliyor). Yolculuğuna başlamadan önce bir rota planlamak için, ızgara modelinin bilgisayar gösterimini kullanabilir; peki ama nasıl? Harita, çarpmadan geçilmesi gereken üç nesnenin konumunu gösteriyor. Nesnelerin köşelerine yakın bazı aday yol noktalarının konumunu hesaplamak o kadar zor değil. (Bu yol noktaları, köşelerden yeterince uzak olmalı ki Shakey nesnelere çarpmasın.) Yol noktaları, gri yıldızlarla gösteriliyor ve "A", "B"den "K"ya kadar harflerle etiketlenmiş. Bilgisayar grafiği alanında günümüzde bilinen teknikler sayesinde, R'den G'ye engelsiz ve düz bir yoldan gitmek için hangi yol noktalarından geçileceğini hesaplamak zor değil.

Bu açıdan bakıldığında, Shakey'nin yön bulma problemi, daha

önce bahsettiklerime benzer bir arama problemidir. Bir arama ağacı oluşturup R'den G'ye giden en kısa yolu aramayla nasıl bulunacağı aşağıda anlatılıyor. Önce, R'den çıkan engelsiz ve düz yollarla A ile F'ye doğrudan ulaşmak mümkün olduğu için, arama ağacında bunlar, R'nin doğrudan çocuğu olan "düğüm noktaları" olarak belirlenir. Hem A'dan hem F'den (engelsiz, düz yollarla) uzanan düğüm noktalarını hesaplama işlemini, ağaca G noktası eklenene dek sürdürürüz. Ardından, R'den G'ye giden en kısa yolu bulmak kolay bir iştir.

Arama ağaçlarıyla (ve daha genel kuzenleri olan çizgilerle) ilgili birtakım yöntemler 1960'ların ortasında zaten kullanımdaydı. Bilinen yöntemlerin lehine olan bir nokta, Shakey'nin yön bulma problemlerini çözerken en kısa yolu kesin buluyor olmasıydı. Ancak, zor problemlerde hesap açısından verimsizdi. Elbette, basit yön bulma problemlerini (diyagramda gösterilene benzer problemleri) çözmek çok fazla arama yapmayı gerektirmez, dolayısıyla her arama yöntemi bu tür problemleri hızlıca çözecektir. Fakat bizler, daha büyük ve zor problemlerde verimli çalışacak genel yöntemlerle ilgileniyorduk. Hareketli karelerden oluşan sekiz parçalı bulmacanın çözümüne dair J. Doran ile Donald Michie'nin önerdiği sezgisel arama (*heuristic search*) yöntemini biliyordum. Arama ağacındaki her düğüm noktası için, o noktadan hedefe ulaşmanın yaklaşık zorluğuna göre sayısal bir değer saptıyorlardı. En düşük puanı olan düğüm noktası, yeni düğüm noktaları türetmek üzere seçiliyordu.[7]

Bir yol noktasından hedefe ulaşmanın zorluğuna dair iyi bir "sezgisel" tahminin (ek aramalar yapmaksızın), aradaki engelleri tamamen göz ardı edip, o noktadan hedefe giden "kuş uçuşu mesafe" olacağı mantığını yürüttüm. Bu tahmini, arama ağacındaki karşılık gelen düğüm noktasının puanı olarak kullanmayı önerdim. O dönem Shakey araştırmasını yöneten Bertram Raphael, puan için o ana dek başlangıç noktasından itibaren katedilmiş uzaklıkla robotun kalan yoluna dair sezgisel tahmini toplamının daha iyi bir puan olacağını belirtti.[8]

Raphael'le birlikte bu fikri, Stanford'dan doktorasını alıp SRI'daki ekibimize yeni katılmış olan Peter Hart'a açtık. Hart, "o gün eve gittiğini, bir sandalyede oturup bir saati aşkın süre duvara gözlerini diktiği-

7 J. Doran ve Donald Michie, "Experiments with the Graph Traverser Program", *Proceedings of the Royal Society of London,* A Serisi, c. 294, 1966, s. 235-259.

8 Bu fikirle ilgili ilk yazılı açıklama: Charles A. Rosen ve Nils Nilsson, "Application of Intelligent Automata to Reconnaissance", s. 21-22, SRI Report, Aralık 1967. Bkz. http://www.ai.sri.com/pubs/files/rosen67-p5953-interim 3.pdf.

ni ve şu sonuca vardığını" hatırlıyor:[9] Eğer kalan uzaklık için yapılan tahmin (artık ne olursa), kalan mesafeden asla büyük değilse, o zaman yeni puanlama şemamızda böyle bir tahmini kullanmak, hedefe giden en kısa yolu *daima* bulur. (Elbette, bahsettiğim sezgisel kuş uçuşu mesafesi, Hart'ın bahsettiği daha-genel koşulu karşılıyordu.) Dahası, böyle bir yöntemin, en kısa yolu bulması kesin olup bizimkilerden daha iyi olmayan sezgisel yöntemler kullanan başka yöntemlerden daha büyük bir arama ağacı yaratmayacağını da düşünmüştü.

Hart, Raphael ve ben hep birlikte bu iddialar için kanıtlar oluşturabildik ve ortaya çıkan arama işlemini "A*" diye isimlendirdik. ("A", algoritma anlamına geliyordu, "*" ise en kısa yolu bulma özelliğine işaret ediyordu. Bence kanıtları bulma işinde ağır yükü Hart ile Raphael sırtlanmıştı.) Yolların bedelleri yalnızca uzaklıktan başka şeylere de bağlı olduğunda ve puanları hesaplarken (uzaklık yerine) böyle bedeller hesaba katıldığında, A*'nın en düşük bedelli yolu bulacağı kesindir.[10]

Hedefe kalan uzaklığın (ya da bedelin) tahminini işe dahil etmek, hedefin genel yönünün aranmasına katkıda bulunur. O ana dek katedilmiş uzaklığın (ya da bedelin) dahil edilmesi, arama işleminin daima, umut vaat eden ama bir yere çıkmayan yollara sapmamasını ve engellerin etrafından "süzülüp geçmesini" güvence altına alır.

A*'nın kapsamı pek çok bakımdan genişletilmiştir; özellikle de, bilgisayar belleği kısıtlı olduğunda bu süreç pratikte daha iyi çalışsın diye Richard Korf tarafından.[11] Günümüzde A* pek çok uygulamada kullanılıyor; bunlar arasında doğal dil çözümleme,[12] sürüş doğrultularının hesaplanması[13] ve etkileşimli bilgisayar oyunları var.[14]

9 Kişisel iletişim, 24 Ekim 2006.

10 Bkz. Peter Hart, Nils Nilsson ve Bertram Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions System Science and Cybernetics*, c. 4, no: 2, s. 100-107, 1968 ve Peter Hart, Nils Nilsson ve Bertram Raphael, "Correction to 'A Formal Basis for the Heuristic Determination of Minimum-Cost Paths,'" *SIGART Newsletter*, no: 37, s. 28-29, Aralık 1972.

11 Örneğin Korf'un yayınlarına bkz. http://www.cs.ucla.edu/~korf/publications.html.

12 2003 tarihli ve "A* Parsing: Fast Exact Viterbi Parse Selection" başlıklı makalelerinde Dan Klein ve Christopher Manning şöyle yazmıştır: "A* arama yönteminin kullanımı, çözümlemeyi tamamlamanın olasılıklarını tutucu biçimde tahmin ederek, en iyi çözümlemeyi bulmak için gereken zamanı epey kısaltabilir."

13 Bir araç yön bulma şirketinin baş mühendisi Brian Smart, 27 Mart 2002 tarihinde Peter Hart'a gönderdiği e-postada şöyle yazar: "'Konum temelli hizmetler' ve 'araç yön bulma' sanayisinin çoğunluğu gibi bizler de, araç ve yaya yön bulma uygulamalarında rota hesaplamak için A*'nın bir çeşitlemesini kullanıyoruz."

14 Bilgisayar oyunlarında YZ kullanımı danışmanı olan Steven Woodcock 14 Haziran

12.1.2 Gürbüz Eylem Gerçekleştirme

A* algoritması Shakey'nin programlarına, engellerle dolu odada bir yerden bir yere giderken yolunu bulabilsin ve bir nesneyi oradan oraya itebilsin diye yerleştirilmişti. Yön bulma programları, başka programlarla birlikte, Shakey programları hiyerarşisinde ara düzeydeydi. Bu ara düzey programlarının hepsi, belirli hedeflere ulaşmak üzere tasarlanmıştı, örneğin bir nesneyi kapı girişinin önüne getirmek gibi. Aynı zamanda bunları, görünmez zorluklar karşısında bile "denemekten" kolay kolay yılmıyordu. Örneğin, eğer itilen nesne kazara "itme çubuğu"nun önünden kayarsa, itme programı bu sorunu fark edip (itme çubuğundaki sensörler sayesinde) Shakey'nin konumunu değiştiriyordu ki nesneyi yine önüne alıp itebilsin.

Bu yılmama haline nasıl ulaşacağımızı düşünürken, hem Miller, Galanter ve Pribram'ın TOTE birimleri hem de homeostazi (iç denge) fikri bana ilham verdi. (Çivi çakan TOTE biriminin, çivi, yerine tamamen girene dek çakmayı sürdürdüğünü ve homeostazi birimlerinin, algıladıkları çevresel çalkantılar karşısında kararlı hale dönmek üzere harekete geçtiğini anımsayın.) Ara düzey programların, hem hedefe ulaşmaya "en yakın" olanı hem de o an gerçekleştirilebilecek eylemi arayıp yerine getirmesini istiyordum. Bu eylem gerçekleştirildiğinde, öngörüldüğü üzere, hedefe ulaşmaya daha da yakın bir eylemin gerçekleştirileceği bir durum ortaya çıkıyorsa ne güzel; ara düzey program en azından ilerliyor diyebiliriz. Bu durumu yaratmıyorsa ya da beklenmedik bir şey aksaklık çıkarıyorsa, işleri yeniden rayına oturtmak için başka bir eylem gerçekleştirilir. Richard Duda ve ben, "çabalamaktan" yılmama özelliği taşıyan bu ara düzey programları yazmak için "Markov tabloları" denen bir biçem geliştirdik.[15]

12.1.3 STRIPS: Yeni Bir Planlama Yöntemi

Ara düzey programlar pek çok basit işi yapabiliyordu, örneğin Shakey'yi aynı odada bir yerden bir yere ulaştırmak, nesneleri itmek, Shakey'yi kapıdan yan odaya geçirmek gibi. Ancak, uzak bir odaya gidip ora-

2003'te bana gönderdiği e-postada şöyle yazıyor: "A*, günümüz oyunlarında uzak ara en çok kullanılan... ve en faydalı... yol bulma algoritması. 1999'dan beri GDC yuvarlak masalarında oyun geliştiricileri, yol bulma konusunda başka araçlara kıyasla en fazla A*'dan faydalandıklarını belirtiyor."

15 Bkz. Bertram Raphael ve diğ., "Research and Applications–Artificial Intelligence", SRI Report, Nisan 1971, s. 27-32. Şu internet adresinde: http://www.ai.sri.com/pubs/files/raphael71-p8973-semi.pdf.

da bir nesneyi önceden planlanmış yere itmek, belki otuz kadar ara programın bir araya getirilmesini gerektiriyordu. Tıpkı zaman zaman insanların işlerini yapmadan önce plan yapıp öyle harekete geçmesi gibi, Shakey'nin de bir eylem planı oluşturup sonra da planı hayata geçirmesini istiyorduk. Plan, gerçekleştirilecek programların listesinden oluşacaktı.

Planlama için gerekli bilgi, "aksiyom modeli" denen yapıda saklanıyordu. Bu model, yüklemcil biçimsel dizge dilinde mantıksal ifadeler barındırıyordu (daha önce bu dizgeden bahsetmiştim). Örneğin Shakey'nin yeri, `YER(ROBOT, 7, 5)` ifadesiyle gösteriliyordu; kutunun itilebilir olduğu gerçeği, `İTİLEBİLİR(KUTU1)` ifadesiyle, R1 ile R2 odaları arasında D1 adlı bir geçit olduğuysa `ODABİRLEŞTİRİR(D1, R1,R2)` ifadesiyle gösteriliyordu. Aksiyom modelinde bunlar gibi 200'e yakın ifade vardı ve bunlar, Shakey'nin uslamlama ve planlama yetilerinin zeminini oluşturuyordu.

Shakey için plan kurmaya dair ilk girişimimiz, QA3 çıkarım sistemini ve durum kalkülüsünü kullanmaktı. QA3'ten, Shakey'nin hedefinin (örneğin uzak bir odada olmak gibi) geçerli olduğu bir durumun var olduğunu kanıtlamasını isteyecektik (bir aksiyom modeli sürümü kullanarak). Çıkarımın sonucu (eğer başarılı olursa), bu durumu, gerçekleştirilecek ara düzey eylemlerin listesi bağlamında ifade edecekti.[16]

Ara düzey eylemlerin nasıl bir araya getirileceğini planlamada durum kalkülüsünün kullanımı, belli durumlarda bu eylemlerinin etkisini betimleyen mantık ifadelerini kullanmayı içeriyordu. Ara düzey eylemin ortamda belirli şeyleri nasıl değiştirdiğini betimlemekle kalmayıp, pek çok şeyi etkilemediğini de ifade etmeliydik. Örneğin, Shakey bir nesneyi ittiğinde, sonuçta o nesnenin konumu değişmiş oluyordu, fakat öteki nesnelerin konumu değişmiyordu. Shakey'nin dünyasında çoğu şeyin yerinin değişmediği, mantıksal ifadelerle açıkça gösterilmeliydi ve daha kötüsü, QA3'ün bunlar hakkında mantık yürütmesi gerekiyordu. "Çerçeve problemi" denen bu zorluk, YZ'de çok fazla araştırma konusu olmuştur ve bu problemi çözemese de hafifletmeye çalışan pek çok girişimde bulunulmuştur.[17] Çerçeve problemi yüzünden, QA3, ancak iki ya

16 QA3'ün, Shakey için üç nesneyi aynı yere itme planını nasıl geliştirdiğinin betimlenişi için örneğin bkz. Nils J. Nilsson, "Research on Intelligent Automata", Stanford Research Institute Report 7494, Şubat 1969, s. 10vd; şu internet adresinde: http://www.ai.sri.com/pubs/files/nilsson69-p7494-interim1.pdf.

17 McCarthy ile Hayes bu problemi ilk olarak şu yayında betimlemiştir: John McCarthy ve Patrick Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence", *Machine Intelligence* içinde, Donald Michie ve Bernard Meltzer (ed.), c. 4,

da üç adımlık basit planların kurulmasında kullanılabilirdi. Çok daha uzun planlar yaratmaya bilgisayarın belleği yetmezdi.

Durum kalkülüsünün (o zamanki haliyle) sorunu, değişmeyeceği açıkça ifade edilmediği müddetçe her şeyin değişebileceğini varsaymasıydı. Değiştikleri açıkça ifade edilene dek hiçbir şeyin değişmediğini varsaymanın daha iyi bir düzen olacağını düşünmüştüm. Böyle bir düzeni yürürlüğe sokmak için, belli bir durumu betimleyen mantıksal yargıların derlemesini farklı bir şekilde güncellemeyi önerdim. Bu fikir, belirli olguların, özellikle de eyleme geçtikten sonraki değil de önceki olguların silinmesini ve belirli yeni olguların, yani eylemin sonucunda ortaya çıkan olguların eklenmesini içeriyordu. Tüm öteki olgular (silinmek üzere bir kenara ayrılmamış olanlar), yeni durumu betimleyen derlemeye olduğu gibi kopyalanacaktı. Bir eylemin *etkilerini* bu şekilde betimlemenin yanı sıra, her eylemin bir de *önkoşulu* olacaktı, yani eylemi o durumda gerçekleştirmek için bu durumda geçerli olacak şeylerin ifade edilmesi gerekiyordu. (Yaklaşık bir yıl önce, MIT doktora öğrencilerinden Carl Hewitt, benzer güncelleme türleri için mekanizmalar barındıran, PLANNER adlı bir programlama dili geliştiriyordu.)[18]

Örneğin, Shakey'yi (X1,Y1) konumundan başka bir (X2,Y2) konumuna hareket ettirmeye yarayan `git((X1,Y1),(X2,Y2))` programının etkilerini betimlemek amacıyla, `YER(ROBOT, X1,Y1)` mantık yargısı silinmeli ve `YER(ROBOT, X2,Y2)` yargısı eklenmeli, ayrıca tüm öbür yargılar olduğu gibi kalmalıdır. Elbette, `git((X1,Y1),(X2,Y2))` programının uygulanması için, Shakey'nin (X1,Y1) konumunda olması şart; yani, aksiyom modeli `YER(ROBOT, X1,Y1)` önkoşul yargısını içeriyor olmalı ya da en azından `YER(ROBOT, X1,Y1)` yargısıyla kanıtlanabilecek yargılar içermeli.

Yaklaşık o zamanlarda (1969), Richard Fikes (1942-), Carnegie'de Alan Newell'ın yanında doktorasını henüz tamamlayıp SRI'daki ekibimize katılmıştı. Fikes'ın tezinde, QA3'teki gibi mantık kullanmak yerine birtakım başka yöntemler kullanılarak problem çözmenin yeni yolları keşfedilmişti. Fikes ve ben, eylemleri betimlemek amacıyla önkoşullar-

1969, s. 463-502. Yeniden baskısı: Matthew Ginsberg (ed.), *Readings in Nonmonotonic Reasoning*, San Francisco: Morgan Kaufmann Publishers, Inc., 1987, s. 26-45. Önbaskısı şu internet adresinde: http://www-formal.stanford.edu/jmc/mcchay69/mcchay69.html.

18 Bkz. Carl Hewitt, "PLANNER: A Language for Proving Theorems in Robots", *Proceedings of the First International Joint Conference on Artificial Intelligence*, 1969, s. 295-301.

dan faydalanan, listeler silen ve yeni listeler ekleyen (hepsi mantıksal yargı olarak ifade ediliyordu) planlama sistemleri tasarlama işinde birlikte çalıştık. Fikes, sistemin, hedefi tutturan bir eylem dizisi ararken, Newell, Shaw ve Simon'un Genel Problem Çözücüsünde (GPÇ) temel yeti olan "araç-amaç" çözümlemeli sezgisel yöntemi kullanması gerektiğini ileri sürdü. Araç-amaç çözümlemesini kullanan arama işlemi, eklenmiş listelerinde, hedef koşullarının belirlenmesini kolaylaştıran yargıları içeren eylemleri saptayarak işe başlayacaktı. Bu eylemlerin önkoşulları, althedef olarak belirlenecek ve geriye dönük bu uslamlama süreci, başlangıç durumunu, hedefi karşılayacak duruma dönüştüren bir eylem dizisi sonunda bulunana dek devam edecekti.

1970 civarında Fikes, yeni planlama sistemimizi programlamayı bitirdi (LISP dilinde). Bu sisteme, *Stanford Research Institute Problem Solver* [Stanford Araştırma Enstitüsü Problem Çözücü] adının kısaltması olarak STRIPS dedik.[19] STRIPS, tamamlanışından sonra, Shakey'nin eylem planı yaratma sistemi olarak QA3'ün yerini aldı. Yaklaşık altı adet ara düzey eylemden oluşan tipik bir plan, PDP-10'da aşağı yukarı iki dakikada yaratılabilir.

STRIPS planlama sistemi, daha etkin YZ planlayıcılara yol açmıştır, fakat bunların birçoğu hâlâ, eylemleri, önkoşullardan ve listeleri silip eklemeden oluşan "STRIPS operatörü" (bazen "STRIPS kuralları" da denir) bağlamında betimler.

12.1.4 Plan Öğrenme ve Gerçekleştirme

Plan yapmak ayrı, planı doğru düzgün uygulamak ayrı bir iş. Ayrıca, STRIPS'in zaten yapmış olduğu planları, gelecekte olası kullanımlar için saklayabilmek de istiyorduk. Planları temsil etmek için "üçgen tablosu" denen yapıyı bulmuştuk; bu yapı, sırf planları gerçekleştirmeye değil saklamaya da yarıyordu. (Robot eylemlerinin etkilerini ve koşullarını üçgen bir tabloda kümelendirmeyi öneren ilk kişi John Munson'dı. 1970 civarında Munson, Richard Fikes, Peter Hart ve ben, STRIPS operatörlerinden meydana gelen planları temsil etmek için üçgen tablo biçemini geliştirdik.) Üçgen tablo, planın doğru düzgün icra edilip edilmediğini izleyebilmek için plandaki her eylemin önkoşullarını ve etkilerini tutuyordu.

19 Bkz. Richard Fikes ve Nils Nilsson, "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving", *Artificial Intelligence*, c. 2, no: 3-4, 1971, s. 189-208. Bkz. http://ai.stanford.edu/users/nilsson/OnlinePubs-Nils/PublishedPapers/strips.pdf.

STRIPS'in yarattığı planlarda eylemlerin kendi parametreleri için özgül değerleri bulunuyordu. Örneğin, eğer bir git eylemi, belli bir planın parçasıysa, Shakey'nin olduğu yerden gideceği yeri belirlemek için gerçek yer koordinatları kullanılıyordu; `git((3,7), (8,14))` gibi. Her ne kadar bu özgül git komutunu bileşeni olarak barındıran planı belki saklamak istesek de, genel uygulamaya daha açık bir planın git bileşeni, özgül hedefe bağlı olarak özgül parametrelerle yer değiştirebilecek, özgül olmayan parametreler barındırabilirdi. Yani, `git((3,7), (8,14))` gibi bir şeyi `git((x1,y1), (x2,y2))` olarak genelleştirmek istiyorduk. Sabit değerlerin yerine değişkenleri öylesine koyamayız, fakat bu tür bir genellemenin, tüm değişken değerler için geçerli ve uygulanabilir planlar yaratmasını sağlamalıyız. Doğru genellemeler yaratan bir yöntem bulabilmiştik ve üçgen tablosunda temsil edilenler bu genellenmiş planlardı.

Bir plan oluşturulduktan, genellendikten ve üçgen tabloda temsil edildikten sonra, Shakey'nin "PLANEX" denen genel gerçekleştirme programı, bu planın uygulanışını gözetip denetler.[20] Shakey'nin iş gördüğü ortamda, planın uygulanışı bazen duraksar, ama üçgen tablosundan faydalanan PLANEX, Shakey'yi asıl hedefe doğru nasıl tekrar rayına sokacağına karar verir. PLANEX, planın uygulanmasında, tıpkı Markov tablolarının ara düzey eylemlerde ısrarcı oluşu gibi, yılmadan "çabalanmasını" sağlıyor.

12.1.5 Shakey'nin Görme Altprogramları

Shakey'nin ortamı, üzerinde hareket ettiği zeminden, odaları çevreleyen duvarlardan, odalar arası geçitlerden ve bazı odaların zeminindeki büyük doğrusal nesnelerden meydana geliyordu. "Görme işini" Shakey için kolaylaştırmak üzere elimizden geleni yapmıştık. Koyu renkli süpürgelik, açık renkli zemini açık renkli duvardan ayırıyordu. Nesneler kırmızının çeşitli tonlarıyla boyanmıştı ve vidikon kamerasında koyu renk, kızıl ötesi lazerli menzil saptayıcıda açık renk olarak görünüyordu. Yine de görsel işleme, zorlu sorunlar doğuruyordu.

Çalışmamız, görsel sahnelerin tam çözümlemesini yapmak yerine, Shakey'nin görevlerini başarmak için gereksinim duyduğu özgül bilgiyi elde etmek amacıyla görme yetisini kullanmaya yoğunlaşmıştı.

20 Genelleme ve performans mekanizmaları şu yayında betimleniyor: Richard Fikes, Peter Hart ve Nils Nilsson, "Learning and Executing Generalized Robot Plans", *Artificial Intelligence*, c. 3, no: 4, 1972, s. 251-288. SRI raporu olarak şu internet adresinde: http://www.ai.sri.com/pubs/files/tn070-fikes72.pdf.

Bu bilgi, Shakey'nin yeri ile nesnelerin varlığını ve konumlarını içeriyordu; bu, ara düzey eylemlerin gerek duyduğu türde bilgiydi. Bu bilgiyi toplamak üzere tasarlanmış görsel altprogramlar, o eylemleri uygulamaya yarayan programlara iliştirilmişti. Bu altprogramlarda Shakey'nin ortamıyla ilgili bilinen özelliklerden faydalanılıyordu.

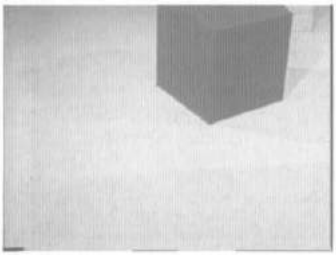
Ekibimizden Claude Brice ve Claude Fennema, nesnelerin, zeminin ve duvarın görece sabit aydınlatması olan düzlemler barındırmasından yararlanarak, bir imgede birörnek yoğunluğu olan bölgeleri bulan, imge işleme altprogramları geliştirdi.[21] Tek düzlem üzerindeki, örneğin bir nesnenin yüzündeki ışıklandırma, bölge içinde yavaş yavaş değişebileceği için, bölge bulma altprogramı önce görece küçük bölgeleri saptar. Sonra bu küçük bölgeler, eğer sınırın iki yakasında yoğunluk farkı çok büyük değilse, imgedeki bölge sınırları boyunca birleştirilir. Sonunda imge, sahnedeki düzlemleri makul şekilde temsil eden pek çok büyük bölgeye ayrılır. Ardından bu bölgelerin sınırlarına, düz çizgiler oturtulur.

Başka bir görme altprogramı, imgedeki düz çizgili kesitleri derhal saptayabiliyordu. Richard Duda ile Peter Hart, "Hough dönüşümü"nün modern bir biçimini temel alarak bu işi yapan bir yöntem geliştirmişti.[22] Kenar çizgisi saptama işlemi, kısa çizgili kesitlerin konumunu ve doğrultusunu bulduktan sonra, Hough dönüşümü, kısa çizgili kesitler bulgu olarak göz önüne alındığında, istatistiksel olarak en olası uzun çizgileri belirlemek için kullanılıyordu.

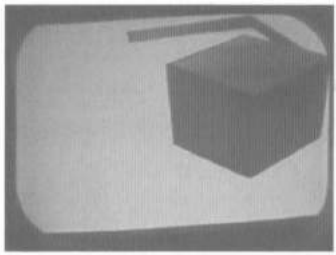
Hem bölge bulma hem de çizgi sezme işlemi, ara düzey işlemlerinin çeşitli görme altprogramlarında kullanılmıştı. Bu altprogramlardan biri, `obloc`, konumu ancak kabaca bilinen bir nesnenin kesin yerini belirlemek amacıyla kullanılıyordu. Resim 12.6'daki resimler, bir kutuyu, kutunun Shakey kamerasında TV görüntüsü olarak nasıl göründüğünü ve `obloc`'un işlemlerinden iki aşamayı gösteriyor. Kutuya ve zemine karşılık gelen bölgelerden yola çıkarak (ve Shakey'nin kutuyla aynı zeminde olduğu gerçeğinden faydalanarak), basit geometri hesaplamaları, kutuyu ve kutunun konumunu, Shakey'nin modellerine ekleyebilir.

21 Bkz. Claude Brice ve Claude Fennema, "Scene Analysis Using Regions", *Artificial Intelligence*, c. 1, no: 3, 1970, s. 205-226.

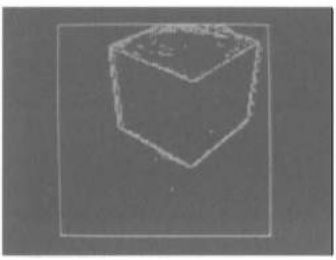
22 Richard O. Duda ve Peter E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures", *Communications of the ACM*, c. 15, Ocak 1972, s. 11-15. Ayrıca bkz. Peter E. Hart, "How the Hough Transform Was Invented", *IEEE Signal Processing Magazine*, Kasım 2009.



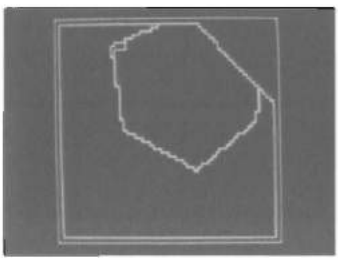
Kutu



TV ekranında görünüşü



Kenar çizgisi saptama

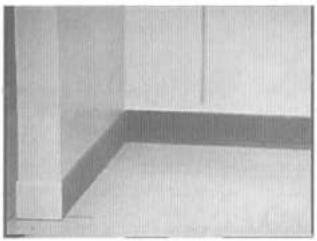


Bölge bulma

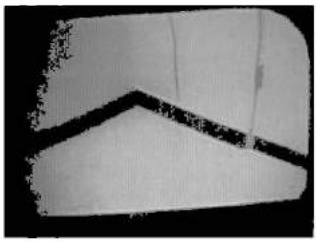
Resim 12.6. Bir nesnenin yerinin belirlenmesinde görme yetisinden faydalanmak. (*Shakey: An Experiment in Robot Planning Learning* adlı filmden. SRI International'ın izniyle kullanılmıştır.)

Shakey, kendi konumunun izini çoğunlukla gözü kapalı tahminle (teker devrini sayarak) sürüyordu, fakat bu tahmin tarzı, hataları kademeli olarak biriktirir. Shakey, kendi konumunu güncellemeye karar verdiğinde, `picloc` denen başka bir görme altprogramından faydalanıyordu. Odanın köşesi gibi yakın bir "işaret noktası", Shakey'nin bu noktaya göre konumunu güncellemek amacıyla kullanılır. Resim 12.7'deki resimler, `obloc`'un odadaki süpürgeliği takip edip duvarlara ve zemine karşılık gelen bölgeleri nasıl bulduğunu gösteriyor. Son resim, Shakey'nin tahmin edilen köşe konumu (Shakey'nin kendi konumu hakkında tahminine dayanarak) ile `picloc`'a dayanan gerçek konumu arasındaki farklılığı gösteriyor. Bu ayrılık, Shakey'nin kendi konumuyla ilgili tahminini düzeltmek için kullanılır.

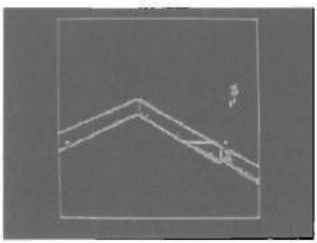
Shakey, engellerin var olduğunun bilindiği bir odada doğrusal bir harekete başlamadan önce, yolun açık olup olmadığını belirlemeye yarayan `clearpath` adlı bir altprogram kullanıyordu. Bu altprogram, zeminde gideceği yolun (ikiz kenar yamuk şekilli bölge) görüntüsüne, bir engelin varlığına işaret olabilecek parlaklık değişiklikleri bakımından bakar.



Odanın köşesi



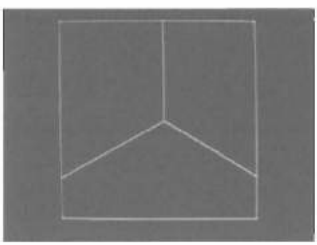
TV ekranından görünüşü



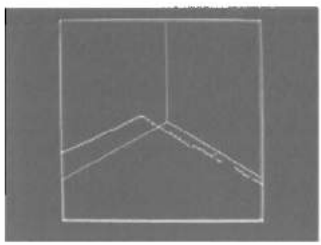
Saptanan süpürgelik



Zemin ile yer arasındaki sınır



Tahmin edilen köşe



Hata düzeltmekte kullanılan farklılık

Resim 12.7. Konumu güncellemede görme yetisinden faydalanmak. (*Shakey: An Experiment in Robot Planning Learning* adlı filmden. SRI International'ın izniyle kullanılmıştır.)

Shakey'nin görsel performansını ölçerken, bu aletin oldukça ilkel olduğunu ve ona özel tasarlanmış ortamında bile pek çok hata yaptığını belirtmek önemli. Bir raporun kabul ettiği gibi, "Köşede buluşan iki duvar gibi ayrı tutmak istediğimiz bölgeler sık sık kaynaştırılıyor ve aslında birleştirilmesi gereken anlamlı bölge kesitleri çoğunlukla ayrı tutuluyor". Aynı rapor, örneğin `clearpath` için şunu belirtir: "... gölgeler ve yansımalar yine de yanlış uyarıya sebep oluyor ve bu sorunlardan bazılarının tek çözümü, daha kapsamlı bir görüntü çözüm-

lemesi yapmaktır."[23] Bununla birlikte görme yetisi, Shakey'nin toplam performansında önemli paya sahipti ve Shakey projesi sırasında geliştirilmiş görsel tekniklerin pek çoğu hâlâ kullanılır (günümüzdeki sonradan yapılan iyileştirmelerle birlikte).[24]

12.1.6 Shakey'yle Bazı Deneyler

Shakey'nin planlama ve plan uygulayıp öğrenme yöntemlerini iş başında göstermek için belirlediğimiz görevde Shakey, komşu olmayan bir odanın önceden belirlenmiş kapısına belirli bir kutuyu itecekti. Bunu başarmak için Shakey, o odaya gidip kutuyu itmeyi içeren planı, STRIPS sayesinde yapmak zorundaydı. Shakey, planı uygulamaya başlamadan evvel, daha önceden betimlenmiş genel bir formda saklıyordu bunu. Planı uygulama sürecinde Shakey'nin önüne beklenmedik bir engel çıkardık. Shakey, yılmaksızın plan uygulama yöntemini sergileyerek, işine devam edebileceği hedef odaya gitmek üzere farklı bir rota çizen, genel planın farklı bir çeşidini oluşturabiliyordu.[25]

Shakey projesinde çalışan araştırmacılardan biri, Carnegie Mellon Üniversitesi'nde Herb Simon'un yanında doğal dil işleme konulu çalışmasıyla doktora derecesini daha yeni almış olan L. Stephen Coles'du (1941-2014). Coles, Shakey'ye İngilizce komutlar vermek istiyordu. Basit İngilizce komutları STRIPS'in mantıksal yargılarına dönüştüren bir çözümleyici ve anlamsal çözümleme sistemi geliştirmişti. Örneğin, yukarıda bahsettiğimiz kutu itme görevi, Shakey'ye şu şekilde veriliyordu:

```
DPDPCLK kapısını kapamak için RCLK odasındaki KUTU2'yi kullan.
```

23 Bertram Raphael ve diğ., "Research and Applications – Artificial Intelligence", Kısım 5, SRI Sonuç Raporu, Aralık 1971. Şu internet adresinde: http://www.ai.sri.com/pubs/files/raphael71-p8973-final.pdf.

24 Yukarıda bahsedilen sonuç raporuna ek olarak Shakey'nin görsel altprogramları hakkında daha fazla bilgi için bkz. Richard O. Duda, "Some Current Techniques for Scene Analysis", SRI Artificial Intelligence Group Technical Note 46, Ekim 1970. Şu internet adresinde: http://www.ai.sri.com/pubs/files/tn046-duda70.pdf.

25 Bu deney ve Shakey hakkında bilgi için: Bertram Raphael ve diğ., "Research and Applications–Artificial Intelligence", Kısım 5, SRI Son Rapor, Aralık 1971; şu internet adresinde: http://www.ai.sri.com/pubs/files/raphael71-p8973-final.pdf; "Shakey The Robot" içinde, SRI Technical Report 323, Nils Nilsson (ed.), Nisan 1984, şu internet adresinde: http://www.ai.sri.com/pubs/files/629.pdf. Ayrıca 1972 tarihli film *Shakey: An Experiment in Robot Planning Learning*; şu internet adresinde: http://www.ai.sri.com/movies/Shakey.ram.

(KUTU2, DPDPCLK ve RCLK, Shakey'nin söz konusu kutuyu, kapıyı ve odayı nitelemek için kullandığı isimlerdi. Robota görev verildiğinde, eşyaları Shakey'nin isimleriyle adlandırmayı şart koşuyorduk.)

Cole'un ENGROB adlı programı,[26] bu İngilizce komutu, geçerli olması için şu koşula dönüştürüyordu (yüklemcil biçimsel dizge diliyle ifade edilmiştir):

```
BLOCKED(DPDPCLK, RCLK, KUTU2)
```

Ardından bu koşul hedefe ulaşacak için gerekli planı yapması için STRIPS'e veriliyordu.

Coles ayrıca, Shakey'e dolaylı uslamlama gerektiren problemler de çözdürmek istiyordu. Shakey'nin, kutuyu yükseltilmiş bir düzlemden düşürmesini gerektiren bir deney düzeneği kurmuştu. Bu düzenekte Shakey, düzlemin yanına bir rampa itmesi gerektiğini bulmalı, sonra da oraya çıkıp kutuyu itmeliydi. Bu görev Shakey'e şu komutla verildi: "Düzlemin üzerinde duran kutuyu zemine it." Bu görev başarıyla yerine getirilmiş ve Shakey'nin teknik raporlarından birinde betimlenmiştir.[27]

"Kutuyu düzlemden aşağı it" görevi, Shakey'nin "maymun-muz" tarzı problemleri çözebileceğini, Cole'un kanıtlama yoluydu. Tümdengelimli uslamlamaya örnek olarak John McCarthy'nin meşhur ettiği bu problem, bir maymun, bir kutu ve ulaşılamayacak bir yerde muzlara erişmek için kutuyu muzların altına itip üzerine tırmanmayı ve böylece muzları almayı akıl edebilmesi gerekir.[28] McCarthy'nin, 1948 Caltech Hixon sempozyumunda, şempanzelerin problemleri zekâ gerektiren yollarla çözmesine kanıt olarak benzer bir problemi betimleyen Karl Lashley'in konuşmasını dinlediği söylenir.

Shakey'nin etkilediği kişilerden biri de, daha sonra Microsoft'un eşkurucularından olacak Bill Gates'di. 1972'de lise ikinci sınıftayken Shakey filmini seyretmiş ve Shakey'ye sırf bir göz atmak için

26 L. Stephen Coles, "Talking with a Robot in English", *Proceedings of the International Joint Conference on Artificial Intelligence*, Washington, DC, 7-9 Mayıs, Bedford, MA: The MITRE Corporation, 1969.

27 Bkz. L. Stephen Coles vediğ., "Application of Intelligent Automata to Reconnaiscance", SRI Sonuç Raporu, Kasım 1969. Bkz. http://www.ai.sri.com/pubs/files/coles69-p7494-final.pdf.

28 Bu problemi McCarthy Temmuz 1963 tarihli bilgilendirme notunda anlatır: "Situations, Actions, and Causal Laws", yeniden baskısı "Program with Commonsense" isimli makalesinin 7.2 numaralı kısmında, *Semantic Information Processing* içinde, Marvin Minsky (ed.), Cambridge, MA: MIT Press, 1968, s. 403-418.

Seattle'dan SRI'a kadar (Microsoft'un öbür eşkurucusu Paul Allen'la birlikte) direksiyon sallamış. Bir kaynağa göre, "Shakey'nin eşyaları oradan oraya itip kendine rampa yapışından özellikle etkilenmişti."[29]

12.1.7 Shakey Mali Sorunlar Yaşıyor

Shakey, planlama, uslamlama ve öğrenme yetileri barındıran, çevresini algılamak için görme yetisini, uzaklık algılama ve dokunma sensörlerini kullanan, planlarının uygulanmasını denetleyen ilk robot sistemiydi. Belki de zamanının biraz ötesindeydi. Bunun gibi yetenekleri olan robotların gerçek hayatta uygulamaya geçilmesinden önce çok daha fazla araştırma yapılması (ve genel olarak bilgisayar teknolojisinde ilerleme kaydedilmesi) gerekiyordu. O zamanların robot araştırma projelerinin sınırlayıcı varsayımlarından bazılarını Shakey'yle ilgili bir raporumuzda anlatmıştık:

> [Robotun] Problem ortamı genelde, değişiklik yaratacak tek etmenin (*agent*) robot olduğu sıkıcı bir yerdir; robot harekete geçene dek zaman bile ilerlemez. Robotun kafası kolay karışır; ilk problemi bitirmeden ikincisi verilemez, iki problem yakından bağlantılı olsa bile. Son olarak, çoğu robot sistemi, henüz koşullu ifadeler ya da döngüler içeren bir plan kuramaz.

SRI araştırmacılarının Shakey'ye dair büyük planları olsa da, DARPA bunları kabul etmedi ve proje 1972'de sona erdi. Projenin bu şekilde bitmesi talihsizlikti, çünkü robot sistemlerinde planlama, görme, öğrenme ve bunların bütünleştirilmesi çalışmaları büyük hız kazanmış, SRI araştırmacıları arasında heyecan yaratmıştı. Dahası, planlama ve görsel algı için yeni fikirler araştırılıyordu. Bu fikirlerden pek çoğu, Shakey projesinin son raporunda ayrıntılarıyla betimlenmiştir.[30]

Bu fikirler arasında özellikle önemli bir tanesi, planları hiyerarşik tarzla kurma teknikleri içeriyordu. Bunun için önce, salt "üst düzey" eylemlerden meydana gelen genel bir planın oluşturulması şart. Bu tür bir plan, gerekli tüm alt düzey eylemlerden oluşan plana kıyasla çok daha az aramayla bulunabilir. Örneğin, kişinin işe gitme planı,

29 Eric Horvitz'in 12 Mayıs 2003 tarihli e-postası.

30 Peter Hart ve diğ., "Artificial Intelligence–Research and Applications", Technical Report, Stanford Research Institute, Aralık 1972. (Şu internet adresinde: http://www.ai.sri.com/pubs/files/hart72-p1530-annual.pdf.) Ayrıca bkz. Richard E. Fikes, Peter E. Hart ve Nils J. Nilsson, "Some New Directions in Robot Problem Solving", *Machine Intelligene 7* içinde, Bernard Meltzer ve Donald Michie (ed.), Edinburgh: Edinburgh University Press, 1972, s. 405-430. (SRI Teknik Notu 68 şu internet adresinde: http://www.ai.sri.com/pubs/files/1484.pdf.)

metroya binme ya da arabasını sürme kararını içerebilir. Ardından, kademeli olarak üst düzey plana, gitgide artan ayrıntılarla incelik katılmalı, ta ki en alt düzey eylemlerin (hangi araba anahtarı kullanılmalı gibi) içi en sonunda doldurulana dek.

SRI'de çalışan Stanford bilgisayar bilimi mezunu Earl Sacerdoti (1948-), hiyerarşik planlama için iki yeni yöntem önermişti. Birincisi (yüksek lisans tezinin parçası olarak), programlamış olduğu ABSTRIPS adlı programdı.[31] Bu program, STRIPS'in bir dizi uygulamasından meydana geliyordu; en önemli operatör önkoşulları dışında tüm önkoşulları görmezden gelen, oluşturulması kolay bir planla başlıyordu işe. Önceden yaratılmış üst düzey planların yönlendirdiği sonraki STRIPS uygulamaları, daha ayrıntılı önkoşulları yavaş yavaş hesaba katar. Sonuçta iyice ayrıntılı bir dizi plan ortaya çıkar ve gerçekten uygulanabilecek bir planla sonuçlanır.

Sacerdoti, doktora çalışması için NOAH (Nets of Action Hierarchies [Eylem Hiyerarşi Ağları]) adlı daha güçlü bir hiyerarşik planlama sistemi geliştirmişti.[32] Tüm eylem operatörlerinin aynı ayrıntı düzeyinde bulunduğu (gerçi seçerek görmezden gelinebilecek önkoşullarla birlikte) ABSTRIPS'in aksine NOAH, farklı ayrıntı düzeylerinde eylem operatörleri devreye sokuyordu. Her operatör, daha düşük bir ayrıntı düzeyindeki operatörler sayesinde nasıl incelik kazanabileceğine dair tanımlarla donatılmış oluyordu. Dahası, Sacerdoti'nin "yordamsal ağ" dediği biçimde NOAH'ın plan gösterimi, bir düzeyde hayata geçirilecek plan adımlarının sırasına dair belirsizliğe olanak tanıyordu. Sıraya dizmeyle ilgili bu "ertelenmiş karar", bir düzeydeki düzensiz planlara daha ayrıntılı incelik kazandırma adımlarının bir alt düzeyde boş bırakılmasına izin veriyordu, böylece sonuçta toplam verim artıyordu.

Sacerdoti, bu hiyerarşik planlama fikirlerini Shakey projesinde kullanmayı umuyordu, dolayısıyla DARPA'nın sonraki projeleri desteklemeyişi onda da, biz SRI çalışanlarında da oldukça hayal kırıklığı

31 Earl D. Sacerdoti, "Planning in a Hierarchy of Abstraction Spaces", s. 412-422, *Proceedings of the Third International Joint Conference on Artificial Intelligence*, 1973. (SRI YZ Merkezi Teknik Notu 78, şu internet adresinde: http://www.ai.sri.com/pubs/files/1501.pdf.)

32 Earl D. Sacerdoti, "The Non-Linear Nature of Plans", *Proceedings of the International Joint Conferences on Artificial Intelligence*, 1975. (SRI YZ Merkezi Teknik Notu 101 şu internet adresinde: http://www.ai.sri.com/pubs/files/1385.pdf.) Ayrıca bkz. Earl D. Sacerdoti, *A Structure for Plans and Behavior*, New York: Elsevier North-Holland, 1977. (SRI YZ Merkezi Teknik Notu 109 şu internet adresinde: http://www.ai.sri.com/pubs/files/762.pdf.)

yaratmıştır. (Robotlarla ilgili temel araştırmalar, 1970'lerin ilk yarısından itibaren DARPA'nın uygulama çalışmalarına önem verişinin yol açtığı ilk zaiyatlardan biriydi.) Bununla birlikte, belirgin askerî önemi olan ama otomatik planlama, görme ve plan gerçekleşleştirme üzerine çalışmamıza olanak tanıyacak bir proje hakkında DARPA'yla konuşabildik. İlginçtir ki, bu proje, Shakey üzerine yaptığımız çalışmaların aşağı yukarı devamıydı, fakat planlanmış görevleri robot yerine insan uygulayacaktı. Buna, "bilgisayar tabanlı danışman (CBC; computer-based consultant) projesi" adını verdik. Bu projeyi, bir sonraki bölümde anlatıyorum.

Hiyerarşik planlamanın önemini tek kavrayanlar Sacerdoti ve SRI araştırmacıları değildi. Austin Tate (1951-), Edinburgh Üniversitesi'ndeki doktora çalışmasının parçası olarak, INTERPLAN adıyla, ağ-tabanlı bir planlama sistemi geliştiriyordu.[33] Britanya Bilim Araştırmaları Konseyi'nin desteklediği Tate ve meslektaşları, 1975'te ve 1976'da, NONLIN adlı hiyerarşik planlamacıyı yarattılar.[34] Bu planlamacı, adını, tıpkı NOAH gibi bazı plan adımlarının, hiyerarşide alt basamaklarda ayrıntı kazanana dek düzensiz bırakılması olgusundan alıyordu.

NOAH ve NONLIN geleneği, başka planlama sistemlerine de can vermiştir. Bir tanesi, SRI International'de David E. Wilkins'in geliştirmiş olduğu etkileşimli plan yaratma ve plan uygulama sistemi SIPE-2'ydi.[35] Edinburgh Üniversitesi Yapay Zekâ Uygulamaları Enstitüsü'nde (AIAI) Tate ile çalışma arkadaşlarının geliştirdiği O-PLAN da böyle bir sistemdi.[36] Bu sistemler yaygınlıkla kullanılmış, geliştirilmiş, uygulanmıştır.[37]

33 Austin Tate, "Interacting Goals and Their Use", *Proceedings of the Fourth International Joint Conference on Artificial Intelligence (IJCAI-75)*, Tiflis, SSCB, Eylül 1975, s. 215-218; bkz. http://www.aiai.ed.ac.uk/project/oplan/documents/1990-PRE/1975-ijcai-tate-interacting-goals.pdf. Austin Tate, "Using Goal Structure to Direct Search in a Problem Solver", doktora tezi, Edinburgh Üniversitesi, Eylül 1975, şu internet adresinde: http:// www.aiai.ed.ac.uk/project/oplan/documents/1990-PRE/.

34 Austin Tate, "Generating Project Networks", *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*, Boston, MA, Ağustos 1979, s. 888-893; bkz. http://www.aiai.ed.ac.uk/project/oplan/documents/1990-PRE/1977-ijcai-tate-generating-projects-networks.pdf.

35 SIPE-2 internet sayfasına bakınız: http://www.ai.sri.com/~sipe/.

36 Ken Currie ve Austin Tate, "O-PLAN: The Open Planning Architecture", *Artificial Intelligence*, c. 52, 1991, s. 49-86. Şu internet adresinde: http://www.aiai.ed.ac.uk/project/oplan/documents/1991/91-aij-oplan-as-published.pdf.

37 Örneğin bkz. AIAI "Planning and Activity Management" internet sayfası: http://www.aiai.ed.ac.uk/project/plan/.



Resim 12.8. Stanford arabası. (Fotoğraf, Lester Earnest'ın izniyle.)

12.2 STANFORD ARABASI

1960'ların ilk yarısında, Stanford makine mühendisliği yüksek lisans öğrencilerinden James Adams (ileride Stanford'da profesör olacaktı), dört tekerli, TV kameralı ve radyo kumandalı, hareketli bir araba üzerinde çalışmaya başladı. Lester Earnest (bu arabadan faydalanan projelerinin tarihçesinde) şöyle yazmıştır: "Başka şeylerin yanı sıra Adams, tezinde, sinyalin Ay'a gidiş gelişine karşılık gelen iletişim gecikmesi (yaklaşık iki buçuk saniye) koşulunda, eğer araç 0,2 m/s'ten (0,3 km/s) hızlı giderse güvenli biçimde denetlenemeyeceğini de göstermişti."[38]

Earnest, Stanford YZ Laboratuvarına katıldıktan sonra, elektrik mühendisliği doktora öğrencisi Rodney Schmidt'le birlikte, "denetimli ışıklandırma koşulları altında yaklaşık 0,8 m/s (1.3 km/s) hızıyla yüksek kontrastlı beyaz çizgiyi (Laboratuvar etrafındaki yolda) izleyen", arabanın geliştirilmiş bir çeşidini yaptı. 1970'lerin ilk yarısı boyunca başka YZ öğrencileri de zaman zaman bu arabayla deneyler yürütmüştür. Arabanın fotoğraflarından biri (o zamanki haliyle) Resim 12.8'de gösteriliyor.

38 Lester Earnest, "Stanford Cart", Ağustos 2005; şu internet adresinde: http://www.stanford.edu/~learnest/cart.htm.

Hans Moravec, Stanford'a gelip görsel yol bulmaya dair doktora çalışmaları kapsamında bu araba üzerinde çalışmaya başladı, "ama Ekim 1973'te, araba elle idare edilirken çıkış rampasında devrildi ve cihazın elektronik devrelerini aküden dökülen asit yaktı". 1979'a gelindiğinde Moravec, yenilenmiş arabasına kavuşmuştu; artık araba stereo kamerayla donatılmıştı; böylece insan müdahalesi olmaksızın nesnelerle dolu bir odayı boydan boya aşabiliyordu; fakat bu işi çok yavaş yapıyordu. Moravec'e göre,[39]

> Sistem, kısa mesafelerde güvenilirdir ama yavaştı. Araba, 10 ilâ 15 dakikada durup kalkarak 1 metre ilerleyebiliyordu. Bit metreyi aştıktan sonra duruyor, birtakım fotoğraflar çekiyor ve bunlar hakkında uzun süre kafa yoruyordu. Sonra yeni bir rota planlıyor, bunun bir kısmını uygulamaya geçiriyor ve yine duraksıyordu. Üç ya da dört sakınımlı yön değiştirme gerektirecek kadar karmaşık 20 metrelik kulvarlarda (her biri 5 saat sürmüştü) araba başarıyla sürülmüştü; başka denemelerde çok göze batan bir biçimde başarısız olmuştur.

Arabayı iş başında gösteren kısa bir video şu internet adresinde izlenebilir: http://www.frc.ri.cmu.edu/users/hpm/talks/Cart.1979/Cart.final.mov. Stanford Arabası, Shakey'yle birlikte California Mountain View'deki Bilgisayar Tarihi Müzesi'nde. Onlar, ileriki bölümlerde ele alacağımız uzun bir robot araç soyunun atalarıdır.

39 Hans P. Moravec, "The Stanford Cart and the CMU Rover", *Proceedings of the IEEE*, c. 71, no: 7, Temmuz 1983, s. 872-884.

13
DOĞAL DİL İŞLEMEDE İLERLEME

Daha önce bahsedildiği gibi, (bilgisayar dili değil de) doğal insan dillerinin içeriğini anlama, üretme ve çeviri sorunları, doğal dil işleme (DDİ) başlığı altına girer. YZ araştırmalarının "ilk keşifler" evresinde DDİ sorunlarıyla ilgili kimi sağlam başlangıçlar yapılmıştı. 1960'ların ikinci yarısından 1970'lerin ilk yarısına uzanan sonraki evrede, bu temeller üzerine yeni çalışmalar eklenmiştir; bunları, kitabın bu kısmında anlatıyorum.

13.1 YAPAY ÇEVİRİ

Yapay çevirinin (YÇ) tarihçesi hakkında çok yazmış olan W. John Hutchins, 1967'den 1976'ya kadar geçen süreyi "sessiz onyıl" olarak nitelemiştir.[1] Bu dönem boyunca alandaki hareketsizliğin nedeni kısmen, ALPAC raporuydu; daha önce belirttiğim gibi bu rapor, yapay çeviriyle ilgili beklentiler konusunda karamsardı. Hutchins şunu ileri sürmüştü: "ALPAC raporunun etkisi derindi. ABD'de on yılı aşkın bir süre boyunca YÇ araştırmalarına uygulamada son verdi ve yıllar boyunca YÇ, tam bir başarısızlık olarak algılanıyordu... YÇ etkinliğinin odağı, ABD'den Kanada'ya ve Avrupa'ya kaymıştı."[2]

ABD'deki bu on yıllık durgunluğun tek istinası, Petr Toma'nın Systran (System Translator) adlı çeviri programını geliştirmesi olmuştur; Macaristan doğumlu bilgisayar bilimcisi ve dilbilim araştırmacısı Toma, Georgetown Rusçadan İngilizceye çeviri sistemi üzerinde çalışmıştı. 1968'de Toma, daha önce Almanya'dayken başlamış olduğu Systran geliştirme işini sürdürmek amacıyla California La Jolla'da Latsec isimli bir şirketi kurdu. ABD Hava Kuvvetleri, Rusçadan İngilizceye çeviri sistemini geliştirmesi için bu şirketle sözleşme yaptı. Bu sistem, 1969'un ilk yarısında Ohio Dayton'da Wright-Patterson Hava Kuvvet-

1 W. John Hutchins, "Machine Translation: A Brief History", *Concise History of the Language Sciences: From the Sumerians to the Cognitivists* içinde, E. F. K. Koerner ve R. E. Asher (ed.), Oxford: Pergamon Press, 1995, s. 431-445. (Ayrıca şu internet adresinde: http://www.hutchinsweb.me.uk/ConcHistoryLangSci-1995.pdf.)

2 *A.g.e.*

leri Üssünde denendi; "günümüzde de ABD Hava Kuvvetleri Teknoloji Dairesi için Rusça-İngilizce çeviriler yapmayı sürdürüyor."[3] Systran, evrimleşip başlıca otomatik çeviri sistemlerinden biri olmuştur. Florida Tampa'da Imageforce Şirketi tarafından pazarlanmaktadır.[4]

Peki, Systran'ın çevirileri ne kadar iyi? Bu, performansın nasıl ölçüleceğine bağlı. Margaret Boden, iki ölçüden bahseder, yani "anlaşılırlık" ve "doğruluk". Bu iki ölçü de insan hükmüne göre değişir. İlk ölçü için, "Çeviri genel olarak anlaşılıyor mu?" sorusu sorulabilir. İkinci ölçü için, "Bir insan 'editör'ün çeviriyi düzeltmesi gerekiyor mu?" sorusu sorulabilir. Boden, "1976'dan 1978'e dek iki yıllık dönemde, Systran'ın yaptığı çevirilerin anlaşılırlığı yüzde 45'ten yüzde 78'e fırladı [ham metin girdisi bakımından]..." der. Ayrıca, insan çevirilerinin başarısının da yüzde yüz olmadığını, ancak yüzde 98'le 99 arasında olduğunu da belirtir. Çevirinin doğruluğuna gelince, Boden şunu söyler: "1978'de 'insan editörlerin' dokunmadığı sözcüklerin oranı yalnızca %64'tü. Yine de, Systran çıktısının bir sayfasını düzeltmek 1980'lerin ortalarında sadece yirmi dakika sürerken, normal (bütünüyle insana ait) çevirinin düzeltilmesi bir saat sürüyordu."[5]

13.2 ANLAMA

Her ne kadar 1960'ların ikinci yarısı ve 1970'lerin ilk yarısı yapay çeviri açısından "sessiz onyıl" olmuşsa da, başka DDİ (doğal dil işleme) çalışmaları için çok faal bir dönemdi. Bu yıllar boyunca araştırmacılar, doğal dil anlama sorununa çok daha güçlü sözdizim, anlambilim ve çıkarım yetenekleri uygulamıştır. Yeni yaklaşımın tipik gözlemi, 1960'ların ikinci yarısı boyunca MIT'de doktora öğrencisi olan Terry Winograd tarafından yapılmıştır:[6]

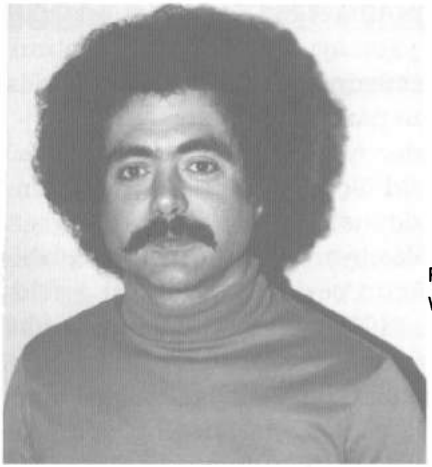
> Eğer bilgisayarların bizi gerçekten anlamasını istiyorsak, onlara daha fazla bilgi anlama yetisi kazandırmalıyız. Dilbilgisine ek olarak, konuştukları konuyla ilgili her türlü bilgiye ihtiyaç duyarlar ve bir cümleyi alıp

3 W. John Hutchins, *Machine Translation: Past, Present, Future*, Chichester: Ellis Horwood, 1986. Güncellenmiş (2003) versiyonu şu internet adresinde: http://www.hutchinsweb.me.uk/PPF-TOC.htm. Systran'ın işleyişine dair kimi ayrıntılar bu kitapta betimlenmiştir.

4 http://www.translationsoftware4u.com/.

5 Margaret A. Boden, *Mind as Machine: A History of Cognitive Science*, Oxford: Oxford University Press, 2006, s. 683.

6 Bu alıntı, Winograd'ın doktora tezinin önsözündendir.



Resim 13.1. Terry Winograd (Fotoğraf, Terry Winograd'ın izniyle.)

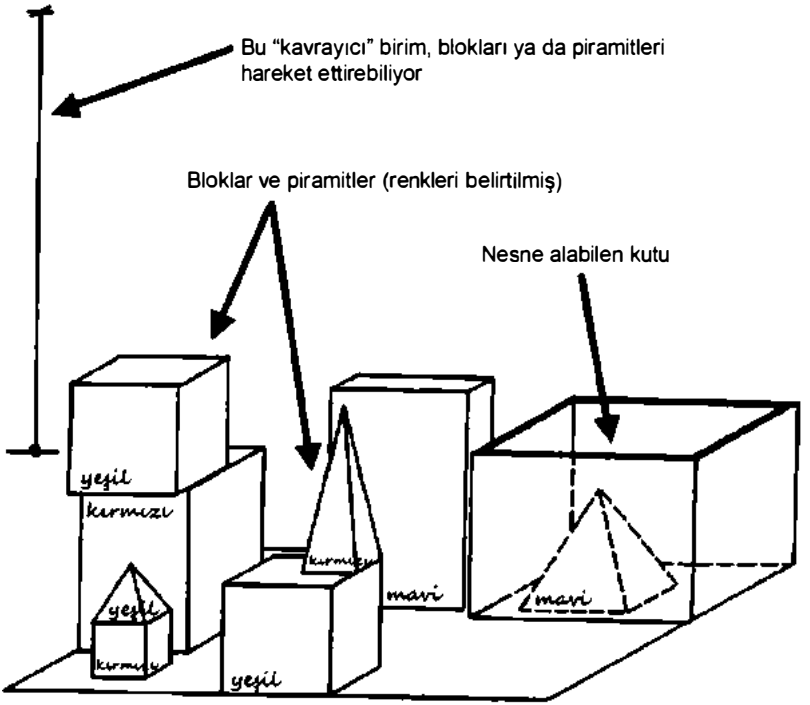
buna düzgün tepki vermek amacıyla olguları bir araya getirirken uslamlamadan faydalanmaları gerekir. Bir cümleyi anlama süreci, dilbilgisini, anlambilimi ve uslamlamayı iç içe geçecekleri biçimde birleştirmeli, her parçayı öbürlerine yardım etmeye çağırmalıdır.

13.2.1 SHRDLU

Belki de en fazla heyecan uyandırmış DDİ uygulaması, MIT'deki doktora tezi (Seymourt Papert'ın danışmanlığında) için Terry Winograd'ın (1946- ; Resim 13.1) programladığı doğal dil diyalog sistemi SHRDLU'ydu.[7]

SHRDLU, Winograd'ın "mikrodünya" dediği, oyuncak bloklardan ve bu blokları hareket ettiren kavrayıcıdan oluşan dünya hakkında ikili bir sohbet yürütebiliyordu. MIT'nin ve Stanford'un önceki el-göz araştırmasında kullanılmış gerçek bloklardan oluşan dünyanın aksi-

7 SHRDLU, Winograd'ın tezinde anlatılıyor: "Procedures as a Representiation for Data in a Computer Program for Understanding Natural Language". MIT YZ Teknik Raporu no: 235 olarak 23 Şubat 1971'de yayımlanmıştır ve şu internet adresinde: http://dspace.mit.edu/bitstream/1721.1/7095/3/AITR-235.pdf. Bu tez aynı zamanda *Cognitive Psychology* dergisinde tam sayı olarak basılmıştır: c. 3, no: 1, 1972. Ayrıca kitap olarak da basılmıştır: *Understanding Natural Language*, New York: Academic Press, 1972. SHRDLU'daki harfler, linotip makineleri tuşlarının ikinci sütununu meydana getirir ve bilgisayarlardan önce yazı tipi bunlarla belirlenirdi. Bu anlamsız sözcük, Winograd'ın gençliğinde okuduğu *MAD* dergisinde sık sık kullanılırdı. Kendi sistemini adlandırmak için uygun bir kısaltma bulamayan Winograd, SHRDLU'yu kullanmış. Winograd'ın açıklaması için bkz. http://hci.stanford.edu/~winograd/shrdlu/name.html.



Resim 13.2. SHRDLU'nun dünyası. (Çizim, Terry Winograd'ın izniyle.)

ne, Winograd'ın bloklar dünyası, DEC PDP-10 bilgisayarında benzetimi yapılmış ve Resim 13.2'de gösterilen gibi bir DEC 340 siyah-beyaz grafik ekranında görselleştirilmişti. SHRDLU'ya gündelik İngilizceyle yazılmış komutlar verilip kavrayıcı birimini kullanarak bloklarla iş yapması sağlanabiliyordu. Ayrıca, dünyasının durumu hakkında soru da sorulabiliyordu. SHRDLU, bu komutları ve soruları "anlayabiliyor", istenen işleri yapabiliyor ve anlamadığı sorular ya da komutlar olduğunda, kullanıcıya açıklayıcı sorular sorabiliyordu.

SHRDLU tasarımının öğelerini betimlemeden önce, kısmi bir diyalog parçasına göz atalım.[8] Aşağıdaki örnekte kullanıcı girdisi numaralandırılıp küçük harflerle gösterilmiş, SHRDLU'nun yanıtları büyük harflerle yazılmış ve Winograd'ın yorumlarıysa aynı yazı tarzıyla yazılmıştır; benim yorumlarım da normal tarzla parantez içindedir:

8 Winograd'ın tezinin 1.3 numaralı kısmından alınmıştır.

1. Büyük kırmızı bloğu al.

Sistem, kırmızı bloğu almak için, yeşil bloğun durduğu alanı bulup yeşil bloğu oradan çekerek kırmızı bloğa yer açmak zorunda.

TAMAM.

Sistem, komutu yerine getirdiğinde "TAMAM" yanıtı verir.

(Diyalog ilerlerken blok dizilişinin nasıl göründüğünü göstermekle uğraşmayacağım, işleri kendi zihinsel "model"inizle izleyebileceğinizi varsayıyorum. Şu an kavrayıcı kırmızı bloğu tutuyor.)

2. Piramidi kavra.

Sahnede üç piramit var, dolayısıya program bir şey yapmıyor.

HANGİ PİRAMİDİ KASTETTİĞİNİ ANLAMIYORUM.

3. Tuttuğun bloktan daha iri olan bir blok bulup onu kutuya koy.

(Şimdi kullanıcı piramit kavramakla ilgili fikrini değiştirmiş görünüyor.)

Sistemin, "bir" ve "onu" sözcüğüyle ne kastedildiğini anlaması gerekiyor, ayrıca hangi bloğu tutacağına karar vereceğini çıkarsamalıdır.

"ONU" SÖZCÜĞÜYLE, ELİMDE TUTTUĞUM BLOKTAN DAHA İRİ OLAN BLOĞU KASTETTİĞİNİ VARSAYIYORUM.

Bizi bilgilendirmek amacıyla bunun çıktısını verir.

Mekanik el, kırmızı bloğu bırakıp iri mavi bloğu almalı.

TAMAM.

4. Kutunun içinde ne var?

MAVİ PİRAMİT VE MAVİ BLOK.

5. Piramidi ne destekliyor?

Daha önce sistem, "piramit" sözcüğüne, "hangi piramidi kastettiğini anlamıyorum" diyerek yanıt vermişti. Şimdi, az önce bahsettiğimiz piramidi kastettiğimizi biliyor.

KUTU

SHRDLU'nun becerilerini sergileyen bu diyalog, kırk iki kullanıcı girdisi ve sistemin yanıtları boyunca sürüyor.[9]

9 Diyaloğun bütününü merak eden okurlar ya Winograd'ın tezine ya da onun internet sitelerinden birine göz atabilir: http://hci.stanford.edu/~winograd/shrdlu/.

Bu tür diyaloglar için SHRDLU'nun "bilgiye" ihtiyacı var. İngilizcenin kullandığı kadarının sözdizimini bilmeli, ayrıca kullanımları bağlamında çeşitli sözcüklerin ve sözcük kombinasyonlarının ne anlama geldiğini de bilmeli. Üstelik, kendi blok dünyası hakkında da bilgisi olmalı; örneğin, bloklarla nasıl iş yapılır ve bir nesnenin "kutu" içinde olması ne anlama gelir, gibi. Diyaloğun seyrini de takip edebilmeli ki "o" benzeri sözcüklerle, daha önce bahsedilmiş hangi nesnelere atıfta bulunulduğuna karar verebilsin.

Tüm bu bilgiler, LISP programlarıyla ya da Winograd'ın deyişiyle "yordamlar"la gösteriliyordu. Sözdizimi hakkındaki bilgi, "sistemli dilbilgisi" ilkelerini temel alan yordamlar derlemesi olarak gösteriliyordu.[10] Sözcüklerin bağlam içindeki anlamlarına dair bilgi, sözcük anlamları sözlüğüne, sözcüğünün kullanıldığı cümlenin öbür öğelerine ve söyleme atıfta bulunan yordamlarla gösteriliyordu. Bloklar dünyası hakkındaki bilgi, iki şekilde gösteriliyordu: Tüm nesnelerin yerini belirten bir model ve kavrayıcının çeşitli nesneleri kavrayışının (bu model içinde) öngörülen etkilerini çıkarsayabilen yordamlar vardı. Nesneleri hareket ettirmekle ilgili yordamlarda, bu işlemlerin hem önkoşulu hem de etkileri hakkında bilgi bulunuyordu. Bu yordamlar, Hewitt'in PLANNER dilinin bir çeşidiyle kodlanmıştı; daha önce de belirtildiği gibi bu, STRIPS operatörleriyle benzerlikler içeriyordu. PLANNER dilinde ek yordamlar, sistemin gereksinim duyduğu başka çıkarsama türleri için kullanılıyordu. Mantık kuralları, hem ileriye hem geriye yönelik çıkarımlar yapabilen programlar olarak ifade ediliyordu.

SHRDLU'nun dil anlama süreçleri üç kısma ayrılabilir: sözdizimi, anlambilim, çıkarsama; fakat bu ayrım bir parça yanıltıcıdır, çünkü bu öğeler arasındaki etkileşim, sistemin kilit özelliğidir. Winograd'ın ifade ettiği gibi, "Her bilgi parçası bir yordam olabileceği için, herhangi bir bilgi parçasını çağrılabilir". Örneğin, Winograd, "Herhangi bir sözdizimi yapısını bulduğunda, önce inanılır mı diye, sonra da (eğer mümkünse) bu sistemin dünya hakkındaki hem özgül hem de genel bilgisiyle uyumlu mu diye sözdizimsel yorumu yoklar" diye yazmıştır.

Winograd'ın bahsettiği yordamsal bilgi gösterimi (bu tür gösterimleri kodlayan, Hewitt'in PLANNER diliyle birlikte), bilgiyi bildirimsel tarzla gösteren McCarthy'nin mantık formülleriyle zıt görünebilir. SHRDLU'nun başarısı, yordamsal ve bildirimsel olmak üzere bu iki bilgi gösterimi stratejisinin artıları ve eksileri hakkında, YZ araştırmacıları

10 Winograd, çeşitli kaynakçaların yanı sıra şuna da atıfta bulunuyor: M. A. K. Halliday, "Categories of the Theory of Grammar", *Word*, c. 17, no: 3, 1961, s. 241-292.

arasında bir tartışma alevlendirdi. Aslında, yordamların gösteriminde LISP kullanımı, bu ayrımı bir ölçüye kadar muğlaklaştırıyor, çünkü, Winograd'ın belirttiği gibi, "LISP, programlara veri, verilere program muamelesi yapmamıza olanak tanır". O halde, SHRDLU'nun bilgisi yordamsal olarak temsil edilmişse de, kendi yordamlarına bildirimsel kimi yeni bilgileri yerleştirebilirdi (programa İngilizce cümleler olarak sunuluyordu bu bilgiler).

Doğrusu, SHRDLU'nun performansı oldukça etkileyiciydi ve bazı doğal dil araştırmacılarını gelecekteki başarılar konusunda iyimser kılmıştır.[11] Bununla birlikte, kısa süre içinde Winograd, bu araştırma yönünü terk edip bilgisayarlar ile insanların iletişimi hakkında çalışacaktı. Belki de bloklar dünyası gibi basit bir şeyde bile dili başarıyla anlamak için ne çok bilginin gerekli olduğuyla ilgili ilk elden yaşadığı deneyimlerden ötürü, bilgisayarlara, insanların sözel yeterliğini kopyalayacak kadar bilgi verilebileceğinden şüpheye düşmüştü. 2004 tarihli bir e-postasında Winograd, SHRDLU'nun yetilerini insan yetileri bağlamına oturtur:[12]

> SHRDLU'nun ve benzerlerinin işleyiş biçimi ile beynimizin içinde olup bitenler arasında temel uçurumlar vardır. Güncel araştırmaların bu uçurumu aşma konusunda çok ilerlediğini sanmıyorum, ayrıca bununla ilgili bilim dallarının, başlangıçtaki iddiaların hayata geçirileceği noktaya ulaşmaları onlarca yıl sürebilir. Bu esnada YZ, daha iddiasız alanlarda çalışan ya da insanın yetilerine ulaşmayan sonuçları (çevirideki gibi) kabul eden daha yapılabilir hedefler benimsemiştir.

13.2.2 LUNAR

Aya ilk kez insan indirme görevinden dönen Apollo 11 astronotları, bilimsel incelemeler için yanlarında birkaç kilo ay taşı getirmişti. Bu taşlarla ilgili çeşitli veriler, jeologların ya da başka bilimcilerin erişebileceği veritabanlarında saklanıyordu. Ay jeologlarının bu bilgiye ulaşmasını kolaylaştırmak amacıyla NASA, BBN'nin genç bilgisayar bilimcilerinden William A. Woods'a, veritabanlarını tuhaf bir bilgisayar kodu yerine İngilizce sorgulamayı mümkün kılacak bir tür doğal dil "ön-yüz" tasarlama olasılığını sordu. Woods, Harvard'ta soru-yanıt sistemleri konulu doktora tezini henüz tamamlamıştı.[13]

11 SHRDLU'yu iş başında gösteren kısa bir film için bkz. http://projects.csail.mit.edu/films/aifilms/digitalFilms/3mpeg/26-robot.mpg.

12 Şu internet adresinden: http://www.semaphorecorp.com/misc/shrdlu.html.

13 William A. Woods, "Semantics for a Question-Answering System", doktora tezi,

NASA İnsanlı Uzay Aracı Merkezi tarafından desteklenen Woods ve BBN'den çalışma arkadaşları Ron Kaplan ile Bonnie Webber, Ay taşlarıyla ilgili soruları yanıtlayan, "LUNAR" adlı bir sistem geliştirdi.[14] LUNAR, İngilizce soruları ay taşı veritabanı sorularına dönüştürmek amacıyla hem sözdizimsel hem de anlamsal süreçlerden faydalanıyordu. Sözdizimsel çözümleme için, "genişletilmiş geçiş ağları" (GGA) kullanılıyordu; bu yöntemi Woods, Harvard'ta doktora araştırması yaparken geliştirmişti. (GGA'ların ne iş yaptığını kısaca anlatacağım.) GGA'dan türeyen ayrıştırma ağaçlarının yönlendirdiği anlamsal bileşen, İngilizce cümleleri, Woods'un "anlam gösterim dili" (ATD) dediği yapıya dönüştürüyordu. Bu dil, bir mantık diliydi (yüklemcil biçimsel dizge dili gibi) ama uygulanabilen yordamlarla kapsamı genişletilmişti. ATD'yi aslen Woods, Harvard'ta bulmuştu ve BBN'de daha da geliştirilmişti.

LUNAR, çeşit çeşit soruyu "anlayıp" yanıtlayabiliyordu, örneğin:

> "Yüksek alkali kayalarda ortalama aluminyum yoğunluğu nedir?"
> "Olivinde kaç breş var?"
> "Bunlar nedir?" (LUNAR, "bunlar" sözcüğünün, son sorunun yanıtında adı geçen breşlere atıfta bulunduğunu bilir.)

LUNAR, performans verilerini yayımlayan ilk soru yanıtlama sistemiydi. Ocak 1971'de Houston'da düzenlenen İkinci Yıllık Ay Bilimi Konferansına katılmış jeologların sorduğu soruların %78'ine başarıyla yanıt verebilmişti. Bildirilene göre, sistemde yapılacak "ufak düzeltmelerle" bunların %90'ının yanıtlayabilirmiş.

Haziran 2006'da LUNAR hakkında yaptığımız sohbette Woods,[15] sistemin bazı kısıtlamalarından bahsetmişti. Aşağıdaki diyalog, yetersizliklerinden birini göz önüne seriyor:

Kullanıcı	: Breş nedir?
LUNAR	: S10018.
Kullanıcı	: S10018 nedir?
LUNAR	: S10018.

Harvard Üniversitesi, Ağustos 1967. Şu serinin bir sayısı olarak yeniden basılmıştır: *Outstanding Dissertations in the Computer Sciences*, New York: Garland Publishing, 1979.

14 William A. Woods, Ron M. Kaplan ve Bonnie Nash-Webber, "The Lunar Sciences Natural Language Information System: Final Report", BBN, Cambridge, MA, 1 Haziran 1972. Ayrıca bkz. A. Woods, "Progress in Natural Language Understanding–An Application to Lunar Geology", *AFIPS Conference Proceedings*, c. 42, Montvale, New Jersey: AFIPS Press, 1973, s. 441-450.

15 Bkz. http://www.ils.albany.edu/IQA06/Files/Bill_Woods_IQA06.pdf.

Woods, "LUNAR, sadece ifadelerle atıfta bulunulan şeyleri bulup bunların ismini belirtiyor; kullanıcı sorusunun ardındaki amaca ya da farklı amaçların farklı tür yanıtlarına dair bir model yok" demiştir.

Her ne kadar LUNAR, aynı soru farklı şekillerde ifade edildiğinde bunu anlayabilse de, Woods, "dilbilgisinin çözümleyebilmesi için (güncel dilbilgisinin kısıtlamaları yüzünden) belli biçimlerde ifade edilmesi gereken istekler ve anlamsal yorumcunun, ancak belirli biçimlerle ifade edildiğinde anlayabildiği sorular da olduğunu" söylemiştir.[16]

13.2.3 Genişletilmiş Geçiş Ağları

Pek çok insan, bağlamdan yoksun dilbilgilerinin (daha önce bahsettiklerim gibi), çoğu doğal dil işleme uygulaması için fazlasıyla zayıf kaldığını fark etmişti. Örneğin, kitabın 7.1'inci kısmında betimlediğim açıklayıcı dilbilgisinin kapsamını genişletip ("fırlatma", "vurdu" ve "adam"a ek olarak) "fırlatmak", "fırlatıyor" ve "vuruyor" gibi şimdiki zaman çekimleri ile "adamlar" gibi çoğul bir isim ekleseydik, "adamlar topa vuruyor" ve "adam topu fırlatıyor" sözcük zincirleri, yersiz biçimde dilbilgisine uygun cümleler kabul edilirdi. Bağlamdan yoksun dilbilgisini, isimlere ve yüklemlere ihtiyaç duyacak şekilde genişletmek, başa çıkılamayacak kadar çok kural içerir. Ayrıca, "topa adam tarafından vuruldu" benzeri edilgen çatılı cümleler, daha da fazla ayrıntı gerektirir. Açıkçası, jeologların ay taşları hakkında sorabileceği türde cümleler, Woods ve kimilerinin geliştirmeye başladığı, genişletilmiş geçiş ağları gibi daha güçlü dilbilgisi gerektiriyordu.

1957 tarihli kitabında Chomsky,[17] bağlamdan yoksun dilbilgilerinin, örneklerden sadece biri olduğu hiyerarşik bir dilbilgisi sistemi önermişti. Chomsky'nin daha güçlü dilbilgilerinde "dönüşüm bileşeni" vardı ve örneğin, "topa adam tarafından vuruldu" gibi bir cümleyi çözümleyip "adam topa vurdu" cümlesiyle aynı "derin yapı"yı verebiliyordu. Genişletilmiş geçiş ağlı dilbilgileri de bu tür dönüşümleri gerçekleştirebiliyordu ama hesaplama bakımından daha uygun biçimde.

Genişletilmiş geçiş ağı, harita benzeri çizgesel bir yapıdır ve bu yapıda düğüm noktaları, çözümleme sürecindeki ilerleme noktalarını; iki düğüm noktasını birleştiren yollar ise sözdizimsel kategorileri gösterir. Cümle çözümlemeyi, (şimdilik hiç ilerleme kaydedilmemiş)

16 William A. Woods, "Progress in Natural Language Understanding–An Application to Lunar Geology", *AFIPS Conference Proceedings*, c. 42, Montvale, New Jersey: AFIPS Press, 1973, s. 441-450.

17 Noam Chomsky, *Syntactic Structures*, 's-Gravenhage: Mouton & Co., 1957.

başlangıç düğüm noktasından (cümlenin başarıyla çözümlendiği) bitiş düğüm noktasına ulaşırken ağ boyunca geçilen bir yola benzetebiliriz. Bu yolda ilerlemek, cümlenin sözdizim yapısını ayrıştırma ağacı biçiminde inşa eder. Cümlenin çözümlenmesi, soldan sağa olacak şekilde sözcüklerin ayıklanmasını ve ağda hangi yoldan gidileceğini belirlerken bu sözcüklerden faydalanılmasını içerir.

Sözdizimsel çözümleme, tek sözcüğü bir kenara ayırıp sözlükten bakarak bunun isim mi, belirteç mi, ek mi, sıfat mı ya da başka bir sözdizimsel "son-ek" kategorisi mi olduğunun belirlenmesiyle başlar; ya da bir sözcük öbeği alınır ve acaba bunlar isim öbeği mi, yüklem öbeği mi, edat öbeği mi ya da elimizde ne varsa o mu diye yoklanır. İlk durumda, tek sözcüğün kategorisine bağlı olarak, başlangıç düğüm noktasından itibaren bu kategoriye denk gelen bir yola gireriz. İkinci durumu ele almaya gelince, isim öbeğine ve öteki olası üst düzey kategorilere denk gelen olası yollar söz konusu olacaktır.

Ancak, örneğin isim öbeği yolundan gidip gidemeyeceğimize nasıl karar veriyoruz? Woods ile birtakım başkalarının önerdiği yanıt, bu üst düzey kategorilere karşılık gelen ek geçiş ağlarının devreye sokulmasıydı. Ana geçiş ağında isim öbeği yolundan gitmemize ancak, isim öbeği ağını başarıyla aşarsak izin verilir. İsim öbeği ağındaki yol, edat öbeğiyle başlayabileceği için, (isim öbeği ağındaki) bu yola girip giremeyeceğimizi, edat öbeği ağını başarıyla aşmamıza göre belirleriz. Bu süreç, bir programın başka programları muhtemelen *özçağrılı* biçimde yürürlüğe sokmasına (ya da "çağırmasına") benzer tarzda bir ağın başka ağları "çağırma"sıyla devam eder. (Özçağrılı programlar hakkında anlattıklarımı anımsarsınız: Kendi sürümlerini çağırabilen programlar.) Bu nedenle, bunlar gibi ağların oluşturduğu kümelere özçağrılı geçiş ağları denir.

İskoçya'nın Edinburgh Üniversitesi'nde James Thorne, Paul Bratley ve Hamish Dewar bu tür ağların ilkini geliştirdi.[18] Daha sonra Dan Bobrow ile Bruce Fraser, İskoç geçiş ağına incelik katan başka bir geçiş ağı önerecekti.[19] Bu iki sistem de, ağlar üzerinde ilerlemeye yardımcı olan hesaplamalar yapar. Bu "genişletmeler", çözümlenen cümlenin

18 James Thorne, Paul Bratley ve Hamish Dewar, "The Syntactic Analysis of English by Machine", *Machine Intelligence 3* içinde, D. Michie (ed.), New York: American Elsevier Publishing Co., 1968, s. 281-309; Hamish Dewar, Paul Bratley ve James Thorne, "A Program for the Syntactic Analysis of English Sentences", *Communications of the ACM*, c. 12, no: 8, Ağustos 1969, s. 476-479.

19 Daniel Bobrow ve Bruce Fraser, "An Augmented State Transition Network Analysis Procedure", *Proceedings of the International Joint Conference on Artificial Intelligence*, Washington, DC, 1969, s. 557-567.

"derin yapı" gösteriminin kurulmasına olanak tanır. Woods'un "genişletilmiş, özçağrılı geçiş ağları" üzerine çalışması, bu fikirler üzerine kurulup bunları geliştirmiştir; ayrıca şık bir ağ tanım dili önermiştir.[20]

Örnek olarak Woods, kendi ağlarından birinin, "John was believed to have been shot." ["John'un vurulduğuna inanılıyordu"], cümlesini nasıl çözümlediğini betimler.[21] Yan ağların hepsi göreve çağrıldıktan ve tüm yardımcı hesaplamalar yapıldıktan sonra, Resim 13.3'te gösterilen ayrıştırma ağacı elde edilmiş. Bu ayrıştırma ağacında iki şey gözlemliyoruz. Birincisi, zaman eki olarak "GEÇMİŞ ZAMAN" [PAST] ile "MİŞLİ GEÇMİŞ ZAMAN"ın [PAST PERFECT] varlığına dikkat edin. İkincisi, özgün cümledeki edilgen çatının, farazi bir adıl, "SOMEONE" [BİRİSİ] sözcüğü kullanılarak etken çatılı cümleye dönüştürüldüğünü gözden kaçırmayın. Woods'un belirttiği gibi, cümle şu şekilde de ifade edilebilir: "Someone believed that someone had shot John" ["Birisi inanıyordu ki birisi John'u vurmuş"]. Ağ dilbilgileri, cümleleri standart biçime dönüştürerek cümlelerin "derin yapı"sına ulaşır.

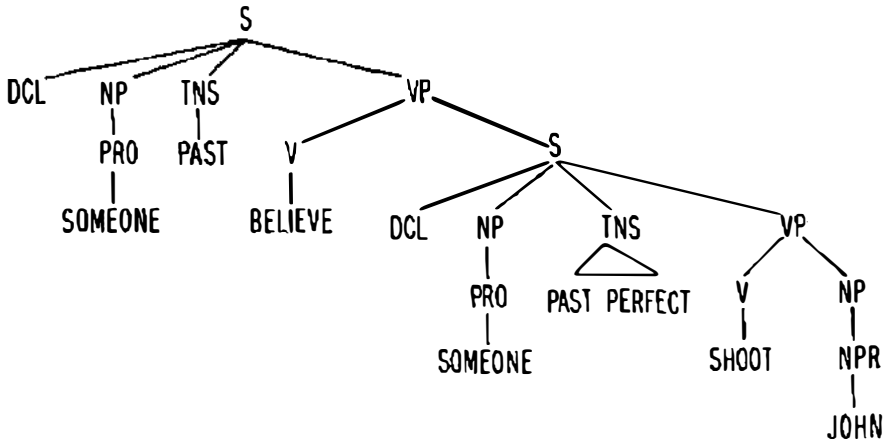
13.2.4 GUS

BBN'de DDİ (doğal dil işleme) üzerine çalışan araştırmacılardan ikisi, Dan Bobrow ile Ronald Kaplan, 1970'lerin ortalarında yeni kurulan Xerox Palo Alto Araştırma Merkezi'ne (PARC) geçti. Onların ve bu merkezdeki başka araştırmacıların üzerinde çalıştığı projelerden biri GUS'du (Genial Understander System); bu sistemin "amacı, canayakın ve işbirliğine açık bir insanla, çok kısıtlı bir söylem alanıyla kısıtlanmış belli bir hedefe yönelik İngilizce sohbet etmekti."[22] Bu kısıtlı alan, "California eyaletinde tek şehre basit bir dönüş yolculuğu yapmak isteyen müşteriyle" seyahat acentası yetkilisinin girdiği sohbetti. Gerçek bir diyaloğun aşağıdaki dökümü, sistemin kimi yeteneklerini göz önüne seriyor:

20 William A. Woods, "Augmented Transition Networks for Natural Language Analysis", Rapor CS-1, Aiken Computation Laboratory, Harvard University, Cambridge, MA, Aralık 1969; William A. Woods, "Transition Network Grammars for Natural Language Analysis", *Communications of the ACM*, c. 13, no: 10, 1970, s. 591-606. ACM makalesi şurada yeniden basılmıştır: Yoh-Han Pao ve George W. Ernest (ed.), *Tutorial: Context-Directed Pattern Recognition and Machine Intelligence Techniques for Information Processing*, Silver Spring, MD. IEEE Computer Society Press, 1982; ve Barbara Grosz, Karen Sparck Jones ve Bonnie Weber (ed.), *Readings in Natural Language Processing*, San Mateo, CA: Morgan Kaufmann, 1986.

21 William A. Woods, *a.g.e.*, s. 602vd.

22 Daniel G. Bobrow ve diğ., "GUS, A Frame-Driven Dialog System", *Artificial Intelligence*, c. 8, 1977, s. 155-173.



Resim 13.3. "John was believed to have been shut." cümlesi için elde edilen ayrıştırma ağacı. (William A. Woods, "Transition Network Grammars for Natural Language Analysis", *Communications of the ACM*, sayı 13, no. 10, 1970, s. 591-606.)

GUS: Merhaba. Adım GUS. Basit bir hava yolculuğu planlamanıza yardımcı olabilirim. Nereye gitmek istersiniz?
(1) *Müşteri*: 28 Mayıs günü San Diego'ya gitmek istiyorum.
GUS: Saat kaçta yola çıkmak istersiniz?
(2) *Müşteri*: Sabah 10'dan önce San Diego'da olmam gerekiyor.
GUS: Saat 9:15'te varan 102 numaralı P.S.A. seferini ister misiniz?
(3) *Müşteri*: Bir sonraki uçuş ne zaman?
GUS: 310 numaralı Air California seferi; 8:30'da kalkıyor ve 10:00'da varıyor.
(4) *Müşteri*: İlkiyle gideyim.
GUS: Yerinizi hangi isimle ayırtayım acaba?
(5) *Müşteri*: Dan Bobrow.
GUS: Şu seferi teyit ediyorum: P.S.A. seferi 102, 28 Mayıs Çarşamba, San Jose'den San Diego'ya sabah 7:30'da kalkıyor, 9:15'te varıyor. Hangi tarihte dönmek istersiniz?
(6) *Müşteri*: Cuma akşamı.
GUS: Akşam 7:45 seferi var, ister misiniz?
(7) *Müşteri*: Uygun.
GUS: Şu seferi teyit ediyorum: P.S.A. seferi 307, 30 Mayıs Cuma, San Diego'dan San Jose'ye akşam 7:45'te kalkıyor, 9:30'da varıyor. Aradığınız için teşekkürler. Hoşçakalın.

Sistemin becerileri gerçekten kısıtlı olsa da, GUS pek çok sorunla başa çıkabiliyordu. Bu sorunlardan biri, DDİ (doğal dil işleme) araştırmacılarının "çözüm anaforu" dedikleri durumu ilgilendirir; bununla kastettikleri şey, bir diyalogta çeşitli sözcüklerin ya da tamlamaların hangi nesnelere ya da olaylara atıfta bulunduğuna karar verme meselesidir. Yukarıdaki diyalogtaki numaralı cümlelere ilişkin kimi örnekler, GUS hakkındaki makalede belirtilmiş:

> Örneğin (3) numaralı satırda müşterinin sorusu, GUS'un az önce bahsettiği uçuşa ilişkindir. (4) numaralı satırda, sohbette daha önce bahsedilmiş olan uçuşa atıfta bulunulur [(2) numaralı satır]. (3) numaralı satırda geçen "bir sonraki uçuş"un hava yolları rehberindeki uçuş sırasına göre yorumlandığına, oysa (4) numaralı satırda bahsedilen "ilk uçuş"un, adı geçen uçuşlara gönderme yaptığına dikkat edin. Başka bir örtük göndermeyse, (6) numaralı satırda tarih belirtmek için "Cuma" sözcüğünün kullanımıdır. Bu gönderme hakkında karar vermek, sohbetin hem içeriğini hem de bağlamını ilgilendiren karışık bir uslamlama gerektiriyor. 28 Mayıs gidiş tarihi olarak belirtildiği için, müşterinin aklında o hafta içindeki Cuma günü olmalı. Öte yandan, bu tanımların terse çevrildiğini ve Cumanın (1) numaralı satırda gidiş tarihi olarak verildiğini düşünün. O zaman, hemen izleyen Cuma gününe atıfta bulunulduğu yorumu yapılabilirdi.

GUS, birkaç iletişim altsisteminin, sözcük bileşenleriyle başa çıkan bir biçimbilim çözümleyicisinin, ayrıştırma ağaçları yaratan bir sözdizimi çözümlemecisinin, kullanıcının kastettiklerini ve niyetlerini anlayan bir "uslamlamacı"nın ve yanıt veren bir dil üreticisinin birleşimiydi. Bu bileşenleri denetleme işini bir "gündem" mekanizması üstlenmişti. Sistemin yaratıcıları şöyle açıklıyor:

> GUS, gündemini incelediği, sıradaki işi seçip yaptığı bir döngü içinde iş görür. Genel olarak, seçili görevin yerine getirilmesi, yeni görev kalemlerinin yaratılıp gündeme yerleştirilmesini gerektirir. Çıktı metinlerini yaratmak, herhangi bir anda uslamlama süreçlerince harekete geçirilebilir ve müşteriden gelen girdiler, sisteme ulaşır ulaşmaz ele alınır. Daha ileri bir aşamadan gelen bilginin (örneğin anlambilimle ilgili), daha erken bir aşamayı (çözümleyici gibi) beslediği yerler vardır. Denetçi bir süreç, herhangi bir an gündemi baştan düzenleyebilir.

GUS'un sözdizim bileşeninin, "3.000'i aşkın kök sözcükten ve basit deyimden oluşan ana bir sözlüğe erişimi" vardı. Sözdizim çözümleyicisi, daha önce Ronald Kaplan tarafından geliştirilmiş olan bir sistemi temel alıyordu; geçiş ağı dilbilgisi kullanan bu sisteme "Genel Sözdizim

Çözümleyicisi" deniyordu.[23] Müşterinin cümleleri, "çerçeve" içinde kodlanıyordu (bunlar, Minsky'nin çerçeveleriyle ilişkilidir ama biçimleri anlamsal ağlara daha yakındır.) Bazı çerçeveler, normal diyalog dizisini betimlerken, başka çerçeveler bir tarihin, yolculuk planının ya da yolcunun özelliklerini gösteriyordu. GUS'un uslamlama bileşeni, müşteri cümlelerinin en iyi nasıl yorumlanacağını belirlemek amacıyla çerçevelerin içeriğini ve yapısını kullanıyordu.

Bu makale, anaforun ötesinde GUS'un başa çıkabildiği başka sorunlardan da bahseder. Bununla birlikte makale, "[örnek diyalogtan] yola çıkılarak, GUS'un, sahip olduğundan çok daha fazla problem çözümü barındırdığı gibi hatalı bir düşünce çıkarmak fazlasıyla kolaydır", uyarısını da yapar. İnsan müşteriler ile GUS rolü oynayan insanlar arasında geçen kaydedilmiş örnek diyaloglar, GUS bilgisayarının başarısız olduğu pek çok durumu gösteriyor. Makalenin yazarları şu sonuca varmıştır: Eğer GUS gibi sistemlerin kullanıcıları, "onlardan beklenen davranışın en ince ayrıntısından ayrılırsa ya da yapılarında görünürde önemsiz ayarlamalar yapılırsa", bu sistemler sanki "afazik" ya da ölmüş gibi davranır. Yazarlar şunu kabul ediyor: "Bizzat GUS çok zeki değil, ama [zeki bir dil anlama] sisteminin yaşamsal bileşenleri olduğuna inandığımız şeyi göz önüne seriyor... Sistemin, yüksek nitelikli bir çözümleyicisi, bir uslamlama bileşeni ve iyi yapılandırılmış bir bilgi veritabanı olmak zorunda". PARC'da ve pek çok başka yerde DDİ üzerine gerçekleştirilmiş sonraki çalışmalar, tüm bu bileşenleri iyileştirmeyi denemiştir.

Winograd, Woods, Bobrow ve meslektaşlarının geliştirdiği sistemler, bilgisayarlarla İngilizce sohbet etmeye yönelik çok etkileyici adımlardı. Yine de, doğal dil anlama sistemlerinin, doktora tezinin önsözünde Winograd'ın öngördüğü performansa ulaşmasına daha uzun bir yol var:

> Bilgisayarları, işlerine uygun biçimde komut alacakları şekilde kullanmayı hayal edelim. Araştırma asistanları, kütüphaneciler ya da sekreterle konuşur gibi konuşacağız onlarla; komutlarımızı yerine getirip istediğimiz bilgileri temin edecekler. Eğer talimatlarımız yeterince açık olmazsa, istediğimiz şeyi yapmadan önce daha fazla bilgi vermemizi isteyecekler; üstelik bu diyalog bütünüyle İngilizce (doğal dilde) olacak.

23 Ronald Kaplan, "A General Syntactic Processor", *Natural Language Processing* içinde, R. Rustin (ed.), New York: Algorithmics Press, 1973.

14
OYUN OYNAMAK

Satranç ve dama gibi masaüstü oyunları oynayan bilgisayarlar programlamaya yönelik girişimlerden daha önce bahsetmiştim. Bunlar arasında en başarılısı, Arthur Samuel'in dama oynayan programıdır. 1967'de Samuel, programının gelişmiş bir sürümünü betimleyen makaleyi yayımladı.[1] Programın arama yöntemine incelik kazandırıp, daha iyi "kitabi öğrenme" becerileri yerleştirmişti, ayrıca bir konumun tahmini değerini ağırlıklı özellik değerlerini toplayarak hesaplamak yerine hiyerarşik örgütlenmiş tablolardan faydalanmıştı. Richard Sutton'a göre, "Bu sürüm, 1959 tarihli programdan çok daha iyi dama oynamayı öğrenmişti, gerçi, hâlâ usta düzeyinde değildi."[2]

1959 ile 1962 yılları arasında, John McCarthy'nin danışmanlık yaptığı birtakım MIT öğrencileri, satranç oynayan bir program geliştirdi. Bu program, McCarthy'nin IBM 704 için yazmış olduğu eski programları temel alıyordu. O öğrencilerden Alan Kotok (1941-2006), söz konusu programı MIT lisans tezinde betimlemiştir.[3] Bu program, FORTRAN dilinin ve makine (çevirme) kodunun bir birleşimiyle yazılmış, MIT'de IBM 7090 bilgisayarında çalıştırılmıştı. Arama ağacının, sonucu değiştirmeksizin elenebilecek dallarını yaratmaktan kaçınmak amacıyla (daha önce bahsettiğim) alfa-beta yöntemini kullanıyordu. Kotok, programı için şunu ileri sürmüştü: Herhangi bir oyunu tamamlamıyor ama "100 oyunluk deneyimi olan bir amatör kadar satranç oynadığı dört uzun oyun parçası oynadı... Makinenin hamle-

1 Arthur L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers II – Recent Progress", *IBM Journal of Research and Development*, c. 11, no: 6, 1967, s. 601-617.

2 http://www.cs.ualberta.ca/~sutton/book/11/node3.html.

3 Alan Kotok, "A Chess Playing Program for the IBM 7090 Computer", MIT Elektrik Mühendisliği lisans tezi, Haziran 1962. Tezin çevrimiçi versiyonları için bkz. http://www.kotok.org/AK-Thesis-1962.pdf ve http://www.kotok.org/AI_MEmo_41.html. (İkincisi bir MIT raporudur ve Kotok bu raporda şuna işaret ediyor: "Bu rapor benim tarafımdan yazılmış olsa da, benden, Elwyn R. Berlekamp'tan (ilk yıl için), Michael Liberman'dan, Charles Niessen'den ve Robert A. Wagner'den (üçüncü yıl için) oluşan 'satranç ekibi'nin ortak çalışmasını temsil eder. Hepimiz MIT 1962 (lisans) sınıfının öğrencileriyiz.)

lerinin çoğu ne harikadır ne de aptalca. Ara sıra saçmaladığını kabul etmek gerekiyor."[4] McCarthy Stanford'a geçtiğinde, bu programı yanında götürüp üzerinde çalışmayı sürdürmüştü.

Bu esnada, Moskova Kuramsal ve Deneysel Fizik Enstitüsü'nde (KDFE) Aleksander Kronrod'un laboratuvarında Georgi Adelson-Velskiy ve çalışma arkadaşları, bir bilgisayar satranç programı geliştirdi.[5] 1965'te Sovyetler Birliği'ne yaptığı ziyarette McCarthy, gelen meydan okuma karşısında, Kotok-McCarthy programının Sovyet programıyla maç yapmasını kabul etti. 22 Nisan 1967'den itibaren dokuz ay boyunca Kotok-McCarthy programı (Stanford'da DEC PDP-6 bilgisayarında çalışıyordu), Rus programıyla satranç oynadı (KDFE'de Rus M-20 bilgisayarında çalışıyordu); bilgisayarın bilgisayara karşı satranç oynadığı ilk maçtı bu. İlk iki oyunda Stanford programı, Rus programının zayıf sürümüyle berabere kaldı (üzerinde anlaşılmış 40 hamle sınırına kadar yenilmeyerek). Fakat, KDFE programının güçlü sürümü karşısında son iki oyunu kaybetti. McCarthy ileride, Stanford'un bilgisayarı daha iyi olsa da KDFE'de daha iyi programlar olduğunu söyleyecekti.[6] KDFE'nin programı, ileride Moskova Kontrol Bilimi Enstitüsü'nde Mişa Donskoy, Vladimir Arlazarov ve Aleksander Uşkov'un yarattıkları çok daha gelişmiş Kaissa programının öncülüydü.

MIT YZ Laboratuvarının uzman programcılarından Richard Greenblatt, Kotok'un satranç programını geliştirebileceğini düşünmüştü. Bilgisayar satrancı üzerine çalışmaları sonucunda MAC HACK VI adını verdiği program ortaya çıkmıştı.[7] Bizzat satranç uzmanı olan Greenblatt, programına, hamleleri seçip değerlendirmeden geçirecek mükemmel sezgisel yöntemler yerleştirebilmişti. YZ Laboratuvarının DEC PDP-6 bilgisayarında çalışan ve etkin makine koduyla yazılmış olan MAC HACK VI, turnuvalarda insan oyunculara karşı satranç oynayan ilk makineydi. 1967 yılının Nisan ayında düzenlenen bir turnuvada, iki oyunu ka-

4 Bilgisayar Tarihi Müzesi, Kotok'un "sözlü tarih" videosunu barındırıyor; bkz. http://www.computehistory.org/chess/alan_kotok.oral_history_highlight.102645440/index.php?iid=orl-433444ecc827d.

5 G. M. Adelson-Velskiy, V. L. Arlazarov, A. R. Bitman, A. A. Zhivotovskii ve A. V. Uskov, "Programming a Computer to Play Chess", *Russian Mathematical Surveys* 25, Mart-Nisan 1970, Londra: Cleaver-Hume Press, s. 221-262. (*Proceedings of the 1st Summer School on Mathematical Programming*, c. 2, 1969, s. 216-252.)

6 Bilgisayar satrancının tarihçesi hakkında sözlü sunum için bkz. http://video.google.com/videoplay?docid=-1583888480148765375.

7 Richard D. Greenblatt, Donald E. Eastlake III ve Stephen D. Crocker, "The Greenblatt Chess Program", YZ Raporu 174, Nisan 1969. Bkz. http://dspace.mit.edu/bitstream/1721.1/6176/2/AIM-174.pdf.

zanıp iki oyunu berabere bitirmiş, böylece ABD Satranç Federasyonu puanlama tablosunda 1.450 puana ulaşmıştı; bu puan, amatör oyuncular düzeyindedir. (Satranç oyuncularının uluslararası puanlama sistemine göre en yüksek düzey Büyük Usta'dır. Ardından Uluslararası Ustalar, Ulusal Ustalar, Uzmanlar, A Sınıfı, B Sınıfı vesaire gelir. MAC HACK VI, üst düzey C Sınıfı ila alt düzey B Sınıfı düzeyinde oynuyordu ki usta düzeyinden oldukça uzaktır bu.) Program, ABD Satranç Federasyonu ve Massachusetts Satranç Birliği'ninin fahri üyesi seçildi. 1967'de MIT'de düzenlenen meşhur maçta,[8] Greenblatt'ın programı, daha önce "hiçbir satranç programı amatör düzeyde bile oynayamaz"[9] demiş olan YZ eleştirmeni Hubert Dreyfus'u (1929-) yendi. Her ne kadar Dreyfus'un saptaması 1965 şartlarında muhtemelen doğru olsa da, Greenblatt'ın MAC HACK VI'sı, iki sene sonra amatör düzeyde satranç oynayabiliyordu.

Belki de MAC HACK'in maharetinden cesaret bulan Donald Michie ve John McCarthy, 1968'de, İskoçyalı Uluslararası Usta David Levy'yle (1945-), on sene içinde bir bilgisayarın onu yeneceğine 250'şer sterlinine bahse girdiler. (Ertesi yıl bu bahse Seymour Papert ve 1971'de Davis California Üniversitesi'nden Ed Kozdrowicki de girdi, böylece toplam bahis 1.000 sterline yükselmiş oldu. 1974'te Donald Michie toplamı 1.250 sterline yükseltti.) 1978'de Levy bahis parasını kazanmıştır; ileride bunu göreceğiz.[10]

1970 civarında Illinois Northwestern Üniversitesi'nde üç öğrenci, David Slate, Larry Atkin ve Keith Gorlen, bir dizi satranç programı yazmaya başladı. Bunların ilki, CDC 6400 bilgisayarında çalışan CHESS 3.0, 1970'de New York'ta, ACM'nin (Association for Computing Machinery [Bilgi İşleyen Makineler Birliği]) ilk bilgisayar satrancı turnuvasını (bilgisayarlara karşı bilgisayarlar) kazandı. Turnuvada altı katılımcı vardı; gerçi MAC HACK VI katılımcılar arasında değildi. David Levy'ye göre, "CHESS 3.0 saniyede yaklaşık 100 konumu değerlendiriyordu ve ABD Satranç Federasyonu'nun puanlama cetvelinde 1.400 puan düzeyinde satranç oynuyordu". İzleyen Nortwestern programları, CHESS 4.6'ya varana dek, bu yıllık etkinlikte galibiyet serileri yakalamıştır. Fakat bu arada CHESS 4.2, 1974'te Stockholm'de toplanan

8 Bkz. *SIGART Newsletter*, Aralık 1968.

9 Hubert L. Dreyfus, "Alchemy and Artificial Intelligence", RAND makalesi P-3244, The RAND Corporation, Santa Monica, CA, Aralık 1965, s. 10. Şu internet adresinde: http://www.rand.org/pubs/papers/2006/P3244.pdf.

10 Bu bahisle ilgili birinci elden ayrıntılar için bkz. David Levy, *Robots Unlimited: Life in a Virtual Age*, Wellesley, MA: A K Peters, Ltd., 2006, s. 83.

Uluslararası Enformasyon İşleme Dernekleri Federasyonu'nun (IFIPS) düzenlediği ilk Dünya Bilgisayar Satrancı Şampiyonası'nın erken turlarında yenildi. Rus programı Kaissa, o turnavada dört maçını da kazanıp dünya bilgisayar satrancı şampiyonu oldu.[11]

1960'ların ikinci yarısından 1970'lerin ortalarına uzanan bu yıllar, bilgisayar satranç programlarının aşama aşama gelişerek başlangıç düzeyinden orta düzey oyunculuğa yükseldiğine şahit oldu. İlerideki bir bölümde göreceğimiz üzere, sonraki yirmi yıl boyunca bilgisayar satrancı üzerine çalışmalar, sonunda usta oyuncu düzeyine ulaşacaktı. Bu hızlı ilerlemeye rağmen, bilgisayarların satranç oynama şekli ile insanların satranç oynama şekli arasında büyük farklılık olduğu çoktan ortaya çıkmaya başlamıştı. Hem satranç uzmanı hem de satranç programlama uzmanı olan Hans Berliner'in, *Nature* adlı dergide çıkan makalesinde şöyle belirtmiştir:[12]

> [İnsan], örüntü tanıma sürecinde olağanüstü miktarda bilgiden [düzgün bir hamleye karar vermek için] ve önerilen çözümün o anki durumda iyi olduğunu doğrulamak amacıyla az miktarda hesaplamadan faydalanır... Fakat bilgisayar aynı hamleyi, gözden geçirdiği yüzbinlerce olasılık içinde, devam etmesi en yararlı hamle olduğunu bulduğu geniş kapsamlı aramanın sonunda yapar. CHESS 4.6, bugüne kadar, hamle hakkında, uygulanışının koşulları vesaire hakkında en ufak bilgisi olmaksızın ama sırf hamlenin sonucunun iyi olduğunu bilirek, iyi bilinen hamleler yapmıştır.

Berliner bu farklılığı şunu söyleyerek özetler: "İnsan satrancının gücü, bilgisayarların aksine, *birikmiş bilgiye* yaslanır" (vurgu bana ait). Çözülmesi istenen problem hakkındaki özgül bilgi, problemin çözümü için devasa bir arama işlemine kalkışmanın aksine, bu dönemde yapay zekâ araştırmalarının başlıca konusu olagelmiştir. (Fakat devasa arama süreci ileride yeniden önem kazanacaktı.) Problem çözmede bilgi kullanımının belki de en etkili yandaşları, Edward Feigenbaum ile Stanford'daki çalışma arkadaşlarıydı. Şimdi de bunların özgün çalışmalarına göz atacağız.

11 Bilgisayar Tarihi Müzesi'nin bilgisayar satrancının tarihine dair sergileri için bkz. http://computerhistory.org/chess/index.php. Bilgisayar satrancı tarihçesinin Bill Wall tarafından derlenmiş özlü bir zaman çizelgesi için bkz. http://www.geocites.com/SiliconValley/Lab/7378/comphis.htm.

12 Hans J. Berliner, "Computer Chess", *Nature*, c. 274, Ağustos 1978, s. 747.

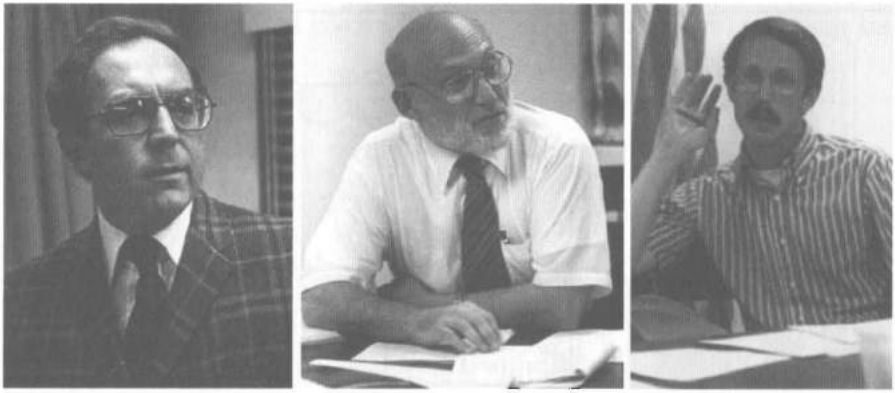
15
DENDRAL PROJESİ

Ed Feigenbaum, 1965'te Berkeley'den Stanford'a geçtikten sonra, "bilimcilerin düşünce süreçlerinin modellerini, özellikle verilerden hipotez ve kuramların çıkarsandığı görgül tümevarım süreçlerinin modellerini yaratmakla" ilgileniyordu. Şöyle der: "Gereksinim duyduğum şey, bu konuları somut olarak inceleyebileceğim özel bir görev ortamıydı."[1] Feigenbaum, Stanford'da Davranış Bilimleri çalıştayına katılıp Stanford Genetik Bölümü'nün kurucusu Nobel ödüllü genetikçi Joshua Lederberg'in (1925-2008; Resim 15.1) konuşmasını dinlemiş. Lederberg, atom bileşenlerinden ve kütle spektrogramından yola çıkarak bir kimyasal bileşiğinin yapısını belirleme problemi hakkında konuşmuş. Bu, Feigenbaum'un aradığı türde bir probleme benziyormuş ve kısa süre içinde Lederberg'le işbirliği yapma konusunda anlaşmışlar.[2]

Kimyasal moleküller, atom bileşenlerini gösteren formüllerle tanımlanır. Örneğin, propanın formülü C_3H_8, bu molekülün üç karbon ve sekiz hidrojen atomundan meydana geldiğine gösterir; fakat bir bileşik hakkında bildiklerimiz, onu oluşturan atomlardan fazlasını içerir. Molekülü oluşturan atomlar belli bir geometrik yapıya göre dizilir, kimyacılar da bu yapının ne olduğunu bilmek ister. Örneğin, propandaki karbon atomları birbirlerine zincir oluşturacak şekilde bağlanmıştır. Zincirin uçlarındaki iki karbon atomuna üçer hidrojen bağlıdır ve zincirin ortasındaki tek karbon atomuna iki hidrojen atomu yapışmıştır. Kimyacılar bu yapıyı, Resim 15.2'deki diyagramla temsil eder.

1 Bu alıntı: "Comments by Edward A. Feigenbaum" şu yayında: Edward H. Shortliffe ve Thomas C. Rindfleisch, "Presentation of the Morris F. Collen Award to Joshua Lederberg", *Journal of the American Medical Informatics Association*, c. 7, no: 3, Mayıs-Haziran 2000, s. 326-332. Şu internet adresinde: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=61437.

2 İşbirliklerinin ilginç bir açıklaması için bkz. "How DENDRAL Was Conceived Born", yazan Joshua Lederberg, 5 Kasım 1987 tarihinde Ulusal Tıp Kütüphanesinde düzenlenen Tıp Bilişiminin Tarihçesi konulu Association for Computing Machinery (ACM) Sempozyumunda sunulmuş makale. Daha sonra şurada yayımlanmıştır: Bruce I. Blum ve Karen Duncan (ed.), *A History of Medical Informatics*, New York: Association for Computing Machinery Press, 1990, s. 14-44. Metin şu internet adresinde: http://profiles.nlm.nih.gov/BB/A/L/Y/P/_/bbalyp.pdf.

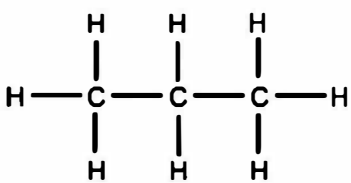


Resim 15.1. Edward Feigenbaum (solda), Joshua Lederberg (ortada), Bruce Buchanan (sağda). (Fotoğraflar, Edward Feigenbaum'un izniyle.)

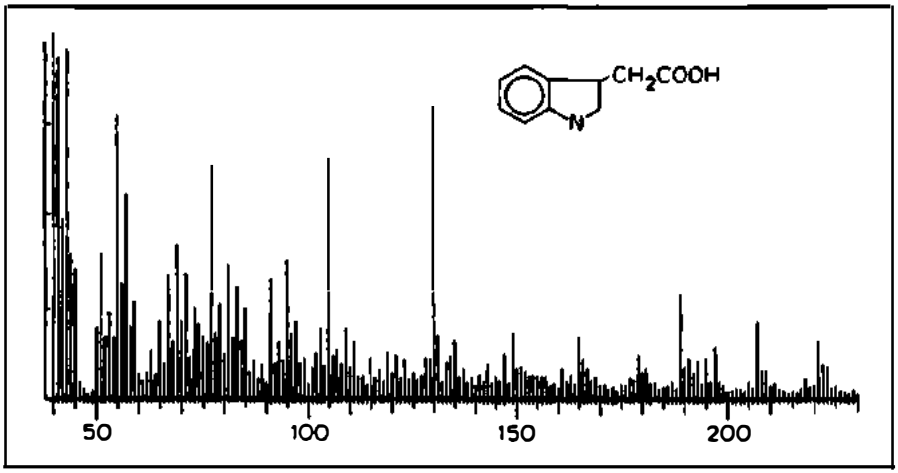
Kimyacılar, propan gibi basit bileşiklerin yapısını belirlemenin fazla zor olmadığını bulmuştur. Bununla birlikte, 2-metil-heksan-3-bir, yani kimyasal formülü C_7H_{140} olan bu keton gibi daha karmaşık bileşikler söz konusu olunca, iş zorlaşır. Bir bileşiğin yapısını tahmin etmek amacıyla kimyacıların kullandığı yöntemlerden biri, bileşiği kütle spektrometresinde yüksek enerjili elektronlarla bombalamaktır. Kütle spektrometresinin elektron ışını, bileşiği parçalara ayırır ve ortaya çıkan bu parçalar, spektrometre içindeki manyetik alan tarafından kütlelerine göre sıralanır. Resim 15.3'te, örnek bir kütle spektrogramı gösteriliyor.

Kütle spektrometresinin ürettiği bileşenler, bileşiğin sağlam alt-birimlerinden meydana gelme eğilimi taşır; bu alt-birimlerin kütleleri, ana yapı hakkında ipucu verir. Deneyimli bir kimyacı, bir bileşiğin yapısı hakkında doğru düzgün tahminler yapmak için, kütle sprektrometresinde bileşiklerin nasıl parçalandığına dair "birikmiş bilgi"den (Berliner'in deyimini kullanacak olursak) faydalanır.

Feigenbaum ve Lederberg, Michigan Üniversitesi'nden felsefe doktorasını aldıktan sonra 1966'da Stanford'a girmiş olan meslektaş-



Resim 15.2. Propan molekülünün yapısı.



Resim 15.3. Kütle spektrogramı. (Çizim, Edward Feigenbaum'un izniyle kullanılmıştır.)

ları Bruce Buchanan'la (1940-) birlikte, bileşiklerin yapısına "ışık tutmak" (bu sözcüğü kullanmışlardı) için bileşiğin kimyasal formülüyle birlikte kütle spektrogramından yararlanan bilgisayar programları oluşturmaya koyuldular.

Lederberg zaten Dendral (Dendric Algorithm'in kısaltması) adlı bir bilgisayar yordamı geliştirmişti; bu yordam, atomların başka atomlara nasıl bağlandığına dair temel kimya bilgisi ve kimyasal formüller verildiğinde, incelenen bileşiğin, topoloji bakımından mümkün tüm düz yapılarını yaratabiliyordu. (Düz yapı, herhangi bir halka yapısı içermez. Bildiğiniz gibi, benzen molekülü, altıgen şeklinde düzenlenmiş altı karbon atomundan meydana gelir ve kimyacılar buna halka der. Bu halkada her karbon atomuna bir hidrojen atomu bağlıdır.) Lederberg'in algoritması, ana formülden kısmi yapılar üretip, sonra bu yapılardan daha ayrıntılı kısmi yapılar ortaya çıkararak, bir ağacın büyümesi gibi adım adım ilerliyordu. Ağacın yaprakları ya da uçları, topografik bakımdan mümkün, tam ayrıntılı son yapıları içeriyordu. Bir bileşiğin asıl yapısını (en azından en inanılır gerçek yapılarını) bulmak, ağaçta uygun ucu ya da uçları aramaya benzetilebilir.

Feigenbaum ve çalışma arkadaşları, kütle spektrometresi verilerini yorumlarken vasıflı kimyacıların kullandığı bilgiden faydalanmayı önermişti. Kimyacılar, spektrogramda belirli özelliklerin, incelenen moleküllerde belirli altyapıları içerdiğini, belirli altyapılarıysa içermediğini gösterdiğini biliyordu. Bu bilgi, Lederberg'in Dendral algoritmasının yarattığı olası yapıların sayısını sınırlamak üzere kullanılabilirdi. Bu

tür bilgi, "kural" olarak gösterilir. Dendral kurallarından bir örneğe bakalım:

```
Kural 74:
EĞER        Molekülün tayfında X1 ve X2 kütlelerinde öyle iki zirve
vardır ki
            X1 + X2 = M + 28
                    ve
            X1 - 28 yüksek bir zirvedir
                    ve
            X2 - 28 yüksek bir zirvedir
                    ve
            X1 ya da X2'den en az biri yüksektir, doğruysa
O ZAMAN     Molekülde keton grubu vardır.
```

Bu tür bilgiyi kullanan ilk programa HEURISTIC DENDRAL [SEZGİSEL DENDRAL] denmişti. ("Sezgisel" sıfatının kullanılmasının nedeni, kimyacılardan alınan bilgiden, Dendral ağacındaki aramayı denetlerken faydalanılmasıydı.) Bu program, girdi olarak kimyasal formülleri ve kütle spektrometresi verilerini (bazen de nükleer-manyetik-rezonans verilerini) kullanıyor, çıktı olarak da, verileri açıklamak üzere öne sürülen, düzenli bir kimyasal yapı tarifi kümesi veriyordu. HEURISTIC DENDRAL'le yapılan ilk çalışmalar, düz bileşiklerin yapısını aydınlatmakla sınırlıydı, çünkü Lederberg'in algoritması yalnızca bu tür moleküllerle başa çıkabiliyordu. Bunlar arasında doymuş düz eterler, alkoller, tioeterler, tioller ve aminler vardı. Onların ilk programının gücüne dair bir örneğe bakalım: *N,N*-dimetil-1-oktadesil aminin 14.715.813 olası yapısı vardır. HEURISTIC DENDRAL, bu bileşiğin kütle spektrogramını kullanarak bu rakamı 1.284.792'ye indirmiştir. Kütle tayfı ve nükleer-manyetik-rezonans verilerinden faydalanınca, olası tek yapı kalmıştır.[3]

"DENDRAL" adı, 1970'lerin sonuna kadar sürmüş olan Dendral projesi sırasında geliştirilmiş, yapıları açıklığa kavuşturmaya yönelik programların koca bir derlemesini betimler olmuştur. Günümüzde kimyacılar bu programların pek çoğundan faydalanıyor. Bu proje üzerinde çalışan bilgisayar bilimcileri ve kimyacılar, Lederberg'in algoritmasını genişletip halkalı bileşiklerle başa çıkmasını sağlamıştır.

3 Robert K. Lindsay, Bruce G. Buchanan, Edward A. Feigenbaum ve Joshua Lederberg, *Applications of Artificial Intelligence for Organic Chemistry: The Dendral Project*, New York: McGraw-Hill Book Co., 1980, s. 70.

Lederberg, Stanfordlu kimyacı Carl Djerassi'yi projeye katılmaya ikna ettikten sonra, programın performansı hem genişlik hem derinlik bakımından epey artmıştır.[4]

Dendral projesi sırasında gerçekleştirilen önemli bir yenilik, kütle spektrometresinde kimyasal yapıların nasıl parçalanacağının benzetimiydi. HEURISTIC DENDRAL, belirli bir bileşik için bazı adaylar belirledikten sonra, bu yapılar, kütle spektrometresi benzetiminde çözümlenir. Sonra bu çıktılar, gerçek kütle spektrometresi çıktılarıyla karşılaştırılır. Yapıların hangisinin spektrogram benzetimi, gerçek spektrograma en yakınsa, büyük olasılıkla bileşiğin gerçek yapısı budur. Bu "bileşimle çözümleme" süreci, yapay zekâda, özellikle de bilgisayarla görmede yaygınlıkla kullanılır olmuştur.

DENDRAL üzerine çalıştığı yıllardaki deneyimlerine dayanan Feigenbaum, YZ uygulamalarının problem alanı hakkında özel bilginin önemini savunmayı sürdürdü (genel çıkarım yöntemlerinin kullanılması karşısında). Önce "bilgi güçtür" hipotezi, sonra "bilgi ilkesi" dediği düşünceyi ileri sürdü.[5] Bu ilkeyi ileride şöyle betimleyecekti:[6]

> Bugüne dek yaşadığımız deneyimlerden yola çıkarak, zeki bir etmenin (*agent*) performansında sergilenen problem çözme gücünün, öncelikle bu etmen tarafından devreye sokulan uzmanlık bilgisinin sonucu olduğunu ve devreye sokulan çıkarım yönteminin genel geçerliğiyle ve gücüyle ancak yan bir ilişkisi bulunduğunu varsaymalıyız. Etmenlerimiz, yöntem bakımından zayıf olsa bile bilgi bakımından zengin olmak zorundadır.

İleride göreceğimiz üzere, YZ programlarına bilgi uzmanları yerleştirmek, pek çok "uzman sistemin" ortaya çıkmasına yol açtı. Aynı zamanda, özgül ve epey sınırlı problemlere yoğunlaşmayı artırıp nasıl olursa olsun genel zekâ mekanizmalarına odaklanmayı azalttı.

4 Dendral projesinin başarılarının tam bir açıklaması için adı geçen esere bakınız.

5 Bu hipotez şu yayında örtük biçimde var: Edward A. Feigenbaum, "Artificial Intelligence: Themes in the Second Decade", *Supplement to Proceedings of the IFIP 68 International Congress*, Edinburgh, Ağustos 1968. Yayımlandığı yer: A. J. H. Morrell (ed.), *Information Processing 68*, c. II, Amsterdam: North-Holland, 1969, s. 1008-1022.

6 Edward A. Feigenbaum, "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering", *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 1977, s. 1014-1029. Ayrıca bkz. Edward A. Feigenbaum, "The Art of Artificial Intelligence: I. Themes and Case Studies of Knowledge Engineering", Stanford Sezgisel Programlama Projesi Raporu HPP-77-25, Ağustos 1977; şu internet adresinde: http://infolab.stanford.edu/pub/cstr/reports/cs/tr/77/621/CS-TR-77-621.pdf.

16
KONFERANSLAR, KİTAPLAR, ÖDENEKLER

Bu dönemde, yapay zekâdaki ilerlemelerin yanında, yeni konferanslar ve çalıştaylar düzenlenmiş, ders kitapları yazılmış, temel araştırmalar için mali destek önce artıp sonra bir parça azalmıştır.

Sırf yapay zekâya adanmış ilk büyük konferans Mayıs 1969'da Washington DC'de düzenlenmiştir. MITRE şirketinden Donald E. Walker (1928-1999) ile Washington Üniversitesi'nden Alistair Holden'ın (1930-1999) düzenlediği bu konferansa Uluslararası Yapay Zekâ Ortak Konferansı (IJCAI; *International Joint Conference on Artificial Intelligence*) denmişti. Bu konferansı ABD, Avrupa ve Japonya'dan on altı farklı teknik dernek (alt topluluklarıyla birlikte) destekledi. Konferansa yaklaşık 600 kişi katıldı, dokuz farklı ülkeden gelen araştırmacıların altmış üç makalesi sunuldu. Bu makaleler bir bildiri kitabında toplandı ve bu kitap konferansta tüm katılımcılara dağıtıldı.

Bu ilk konferansın başarısından ötürü, 1971'de Londra'da ikinci bir konferans yapılmasına karar verildi. İlk yıllarda konferanslar görece gelişigüzel düzenleniyordu; gelecekte düzenlenecek konferanslara, toplantıların düzenlenmesinde boy gösteren ve alanın kimi önderlerinden oluşan çekirdek bir ekip karar veriyordu. Bu ekibin Londra'da 1971'de yaptığı toplantıda, sıradaki konferansın nerede ve ne zaman düzenleneceği konuşulurken bir an için odadan çıkmıştım. Odaya döndüğümde, 1973'te Stanford'da yapılacak bir sonraki toplantının "çarı" seçildiğim söylendi bana. İleride, "Uluslararası Yapay Zekâ Ortak Konferansları"nı idare etmek amacıyla, Başkanı, Yönetim Kurulu, Sekreteryası olan daha kurumsal bir düzen belirlenecekti.[1] İlk toplantıdan itibaren IJCAI konferansları iki yılda bir (tek sayılı yıllarda), Kuzey Amerika ile dünyanın geri kalanı arasında (yaklaşık olarak) sırayla değişen toplantı yerlerinde düzenlenmiştir. İlk konferanstaki gibi, tüm konferanslarda bildiri kitabı dağıtılmıştır. (Bunlardan bazıları çeşitli kitapçılarda var ve India internet sitesi Sayısal Kütüphanesinde bulunabilir: http://202.41.92.139/.)

Bilgi İşleyen Makineler Birliği'nin (ACM) en eski "özel ilgi topluluklarından" biri SIGART'tır (Special Interest Group for ARTificial intelligence). Bu topluluk, 1966'da Haber Bülteni yayımlamaya başladı;

1 Bkz. http://www.ijcai.org/IJCAItrustees.php.

bu bülten (SIGART internet sitesinin belirttiği üzere), "2001'e dek farklı farklı kisvelerde devam etmiştir (SIGART Bülteni, Intelligence Magazine)". Günümüzde SIGART çeşitli konferans ve çalıştayları destekliyor ve YZ doktora öğrencilerinin, bitmek üzere olan çalışmalarını, yorum ve eleştiri yapsınlar diye akranlarına ve kıdemli araştırmacılara sunabildiği toplantılar düzenliyor.

YZ alanı, tekniklerini ve yöntemlerini geliştirmeye başladıkça, bazı üniversitelerde yapay zekâ konusunda lisansüstü dersleri açılıyordu. Dolayısıyla, bu dersleri veren kimilerimiz, YZ hakkında kitap yazmaya ya da derlemeye değeceğini düşünüyordu. 1963'te, o zamanlar UC Berkeley'de yardımcı doçent olan Edward Feigenbaum ve Julian Feldman, daha önce pek çok farklı yerde çıkmış eski YZ ve bilişsel bilim makalelerinin bir derlemesini yayımladılar. *Computers and Thought* (Bilgisayarlar ve Düşünce) adlı bu kitabı, YZ'nin ilk öğrencileri için zorunlu okumaydı (ben de dahil).[2] 1995 baskısında Feigenbaum'un yazdığı gibi, "Bazı makaleler, içerdikleri temel fikirlerden ötürü, 1950'lerin ikinci yarısında ve 1960'ların ilk yarısında yazıldıklarında nasıl önemlilerse, bugün de öyle önemliler. Diğer makalelerse, kapsamını genişletip kökten değiştirdikleri alanların ilk kilometre taşları olarak ilginçtir."

1965'te, örüntü tanımada kullanılan sinir ağı ve istatistik yöntemleriyle ilgili bir kitap yayımladım.[3] Bu kitabın ardından 1971'de, YZ arama yöntemleriyle ilgili bir kitap geldi.[4] Yaklaşık aynı zamanda, ikisi de MIT'de Marvin Minsky'nin eski doktora öğrencisi olan James Slagle[5] ve Bertram Raphael'in[6] kitapları yayımlandı.

1969'da Marvin Minsky ve Seymour Papert, değindikleri başka konuların yanı sıra, Rosenblatt'ın kimi perceptron sürümlerinin önemli kısıtlamaları olduğunu kanıtladıkları önemli bir kitap yayımladılar.[7]

2 Edward A. Feigenbaum ve Julian Feldman, *Computers and Thought*, New York: McGraw-Hill Book Co., 1963. (McGraw-Hill edisyonu artık basılmıyor; AAAIPress/MIT Press, 1995 baskısı aracılığıyla ulaşılabilir.)

3 Nils J. Nilsson, *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, New York: McGraw-Hill Book Co., 1965; yeniden baskısı: *The Mathematical Foundations of Learning Machines*, San Francisco: Morgan Kaufmann Publishers, 1990.

4 Nils J. Nilsson, *Problem-Solving Methods in Artificial Intelligence*, New York: McGraw-Hill Book Co., 1971.

5 James R. Slagle, *Artificial Intelligence: The Heuristic Programming Approach*, New York: McGraw-Hill Book Co., 1971.

6 Bertram Raphael, *The Thinking Computer: Mind Inside Matter*, New York: W. H. Freeman, 1976.

7 Marvin Minsky ve Seymour Papert, *Perceptrons: An Introduction to Computational*

Minsky-Papert kitabının, sinir ağı araştırmalarına gösterilen ilginin sönüşünün nedeni olduğunu iddia edenler olmuştur ama bundan kuşkuluyum. Öncelikle, bizzat Rosenblatt, 1969'dan çok önce başka konulara yoğunlaşmaya başlamıştı,[8] ayrıca sezgisel programlama yöntemlerinin başarısı, 1960'ların ortasında ilginin (benimki dahil) sinir ağlarından uzaklaşmasına yol açmıştı.

1965'te Edinburgh Üniversitesi'nde Donald Michie, sadece davetle gelinen "Makine Zekâsı" çalıştaylarının ilkini düzenledi. Bu ilk çalıştay, Amerikalı ve Avrupalı araştırmacıların katılımıyla Edinburgh'ta yapıldı. Katılımcılar bu çalıştayda makaleler sunmuş ve bu makalelerin hepsi, N. L. Collins ile Donald Michie'nin hazırladıkları kitapta 1967'de yayımlanmıştır. İkinci çalıştay, Eylül 1966'da yine Edinburgh Üniversitesi'nde düzenlendi. Sonraki çalıştaylar, 1971'e dek her yıl Edinburgh'da yapıldı. Ardından, çalıştaylar birkaç senede bir, farklı yerlerde düzenlendi. Her çalıştayın sonucunda *Machine Intelligence N* başlıklı bir kitap basılmıştır; burada *N*, çalıştay ve kitap numarasını gösteriyor.[9] Son birkaç cilt, *Electronic Transactions on Artificial Intelligence* tarafından internette yayınlanmıştır.[10] Bu kitaplar, YZ alanının erken tarihinde en çok atıfta bulunulan ve en önemli makalelerinden bazılarını içeriyor.

O yıllarda, ABD Vietnam'da savaşmaya başladı ve Kongre, ABD Savunma Bakanlığı'nın desteklediği araştımaların askerî işlerle ilişkili olmasını istiyordu. Bu baskılara tepki olarak Kongre, 19 Kasım 1969'da, 1970 Savunma Satın Alımı Yetkisi Akti (Kamu Kanunu 91-121) için "Mansfield Ek Yasası"nı geçirdi; buna göre Savunma Bakanlığı, temel araştırmalara verdiği desteği, "belirli bir askerî işlevle ya da operasyonla doğrudan ve görünür ilişkisi olan" projelerle sınırlayacaktı. 23 Mart 1972'de İleri Araştırma Projeleri Dairesinin (ARPA) adı değiştirilip İleri Savunma Araştırmaları Projeleri Dairesi (DARPA) yapıldı; bu isim, artık askerî becerileri artıracak projelere ağırlık verildiğini yansıtıyordu. (Kurumun ismi 1993'te İleri Araştırma Projeleri Dairesine çevrilmiş olsa da 1996'da yine İleri Savunma Araştırmaları Projeleri Dairesine dönüştürüldü.)[11]

Geometry, Cambridge, MA: MIT Press, 1969.

8 Bkz. Frank Rosenblatt, J. T. Farrow ve S. Rhine, "The Transfer of Learned Behavior from Trained to Untrained Rats by Means of Brain Extracts. I", *Proceedings of the National Academy of Sciences*, c. 55, no: 3, Mart 1966, s. 548-555.

9 Bu serinin internet sayfası: http://www.cs.york.ac.uk/mlg/MI/mi.html.

10 Bkz. http://www.etai.info/mi/.

11 Bkz. http://www.darpa.mil/body/arpa_darpa.html.

Atlas Okyanusu'nun öteki yakasında Britanyalı YZ araştırmacıları da ödenek buhranı yaşıyordu. Birleşik Krallık'ın üniversite araştırmaları için başlıca ödenek kuruluşlarından olan Bilim Araştırmaları Konseyi, Cambridge Üniversitesi'nin ünlü hidrodinamikçisi Profesör James Lighthill'den, yapay zekâ araştırmalarını değerlendirecek bir incelemeyi üstlenmesini istemişti. Lighthill'in "Yapay Zekâ: Genel Bir İnceleme" başlıklı raporu, YZ araştırmalarını, biraz da kendine özgü biçimde üç kategoriye böler, yani ileri otomatlar, merkezî sinir sisteminin bilgisayar tabanlı araştırmaları ve bunlar arasındaki köprüler. Bu kategorileri sırayla A, B, C diye isimlendirmişti. Her ne kadar Lighthill, A ve C kategorilerinde çalışmaların sürdürülmesinden yana olsa da, robotik ve B kategorisine soktuğu dil işleme çalışmaları dahil, çoğu temel YZ araştırmalarını oldukça eleştiriyordu. "Bu alanın hiçbir kısmında, şimdiye dek yapılmış keşifler o zamanlar [1960 civarı] söz verilmiş ana etkiyi doğurmadı" diye yazmıştır. Var olan YZ arama tekniklerinin (oyuncak problemler üzerinde çalışan tekniklerin), gerçek sorunlarla başa çıkacak ölçeğe ulaşamayacağı, çünkü birleşimsel patlama yüzünden bu tekniklerin yetersiz kalacağı sonucuna varmıştı.[12]

Lighthill'in raporu yüzünden Birleşik Krallık'ta YZ araştırmaları büyük oranda budandı. Özellikle, sebep olduğu zaiyatlardan biri robot FREDDY ve Edinburgh'da Donald Michie'nin öteki YZ çalışmalarıdır. Bu raporun etkilerine dair Michie'nin daha sonraki yorumlarından biri şöyle olacaktı:[13]

> Yetenekli genç insanların harika çalışmaları, niteliksiz bilim diye kötülenmiş ve deneyler daha yarı yoldayken sona erdirilmiştir. İnsanların bu işbirliği düzeneğinin ve pek çok elin özenli emeğinin bu şekilde yok edilmesi, başka yerlerde aksilik olarak betimlenir. Fakat açık konuşmak gerekirse, bu bir zulümdü. İleride, bilimin değerleri ve yöntemleri gelişip karşıt siyasetin değerleri ve yöntemleri geri çekilince, öyle olduğu görülecektir.

DARPA'nın kısa vadeli uygulamalı araştırmalara kayması, Lighthill raporuyla ve çeşitli gözlemcilerin eleştirileriyle birlikte, sonraki birkaç yıl boyunca temel YZ araştırmaları için zorluklar çıkaracaktı. Bunun-

12 Önde gelen Britanyalı YZ araştırmacılarının yorumları ve eleştirileri eşliğinde bu raporun metni 1972'de şu kitapta basılmıştır: James Lighthill ve diğ. (ed.), *Artificial Intelligence: A Paper Symposium*, Londra: Science Research Council of Great Britain, 1972.

13 Donald Michie, *Machine Intelligence and Related Topics: An Information Scientist's Weekend Book*, New York: Gordon and Breach Science Publishers, 1982, s. 220.

la birlikte, Lighthill'in değerlendirmesinin aksine, pek çok YZ yöntemi gerçek problemlerde uygulanır oldu ve YZ araştırmaları çalışmalarında da bir genişleme dönemi başladı; önümüzdeki birkaç bölümde bunlara göz atacağız.

DÖRDÜNCÜ KISIM

UYGULAMALAR VE UZMANLAŞMALAR: 1970'LERDEN 1980'LERİN İLK YARISINA

Yaklaşık 1970'lerin ilk yarısına dek çoğu YZ araştırması, Seymour Papert'ın "oyuncak" problemler (bulmaca ya da oyun çözen programlar) dediği problemlerle uğraştı ya da araştırmacılar, yüksek denetimli laboratuvar ortamında yürütülen projelerin peşinden gitti. (Elbette kayda değer kimi istisnalar olmuştur, örneğin yapay çeviri, DENDRAL ve LUNAR gibi.) Fakat kısa süre sonra YZ çalışmaları, açık biçimde uygulama alanlarına kaymaya başlayıp gerçek dünyada önemli problemlerle karşı karşıya geldi. Başarılı uygulama çalışmalarının, doğal dil işleme, uzman sistemler ve bilgisayarla görme gibi altdisiplinlerde uzmanlaşmayı cesaretlendirmesi kaçınılmazdı.

Uygulamalara ilginin artmasının sebeplerinden biri, YZ yöntemlerinin gücünün artıp gerçekçi uygulamaların ulaşılabilir göründüğü noktaya varmasıydı. Fakat belki de daha önemlisi, ABD Savunma Bakanlığı'ndaki YZ araştırmaları destekçilerinin, 1969 "Mansfield Ek Yasası"nın getirdiği kısıtlamalarla başa çıkmak zorunda kalmasıydı; bu yasaya göre temel araştırmalar, askerî işlerle ilgili olmalıydı. Uygulamalara artık daha fazla ağırlık verilmesinin örneklerinden biri, 1972'de Enformasyon Bilimleri Enstitüsü'nün (UCS-ISI) kurulması ve burada özellikle uygulamaların peşinden koşulmasıdır. California'nın Marina Del Rey kentinde kurulmuş bu enstitü, Southern California Üniversitesi'yle ortak çalışır ve başlangıçta aldığı desteğin büyük kısmı DARPA'dan gelmiştir. Ayrıca, kimi büyük şirketler YZ'nin ticari potansiyelini keşfetmeye başlamıştı.

Elbette kuramsal ve temel araştırmalar da sürüyordu, var olanlara yeni üniversite ekipleri katılmıştı. Yeni ekiplerin kısa listesinde, Toronto, Rochester, Texas, Maryland, British Columbia, California ve Washington Üniversitelerindeki ekipleri sayabiliriz. Avrupa'da ve Asya'da da ekipler kurulmuştur. (1981'de Çin'de YZ seminerleri ver-

meye davet edildim; o zamanlar Çin, "Kültürel Devrim"in etkilerinden yeni yeni kurtuluyor, "Açılma ve Reform" programına henüz başlıyordu.) Ancak üniversitelerde bile, temel araştırmaların büyük kısmının itici gücü uygulamalardı. Kitabın bu kısmında, 1970'lerden 1980'lerin ilk yarısına dek üstlenilen bazı YZ uygulamaları çalışmalarını anlatacağım.

17
KONUŞMA TANIMA VE ANLAMA SİSTEMLERİ

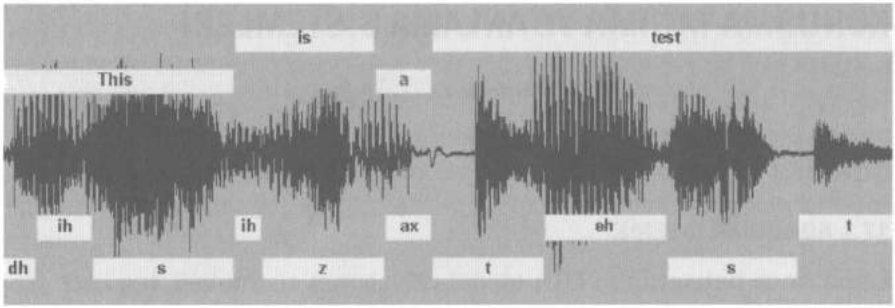
17.1 KONUŞMA İŞLEME

Daha önce bahsettiğim DDİ (doğal dil işleme) sistemleri, İngilizce girdilerin metin formatında olmasını gerektiriyordu. Yine de bilgisayara konuşmanın, klavyeyle yazmaya yeğleneceği durumlar vardır. İnsanlar genelde, yazabildiklerinden hızlı konuşur (saniyede yaklaşık üç sözcüğe karşı saniyede yaklaşık bir sözcük), üstelik hareketliyken de konuşulabilir. Ayrıca, konuşmak elleri ya da gözleri tek işe bağlamaz.

Bilgisayarların konuşma işlemesinden bahsederken, bazı ayrımları belirlemek önemli. Bu ayrımlardan biri, tek başına telaffuz edilen bir sözcüğü tanıma ile kesintisiz bir konuşma akışını işleme arasındaki farklılıktır. Çoğu YZ araştırması, bu sorunlardan ikincisine ve daha zor olanına yoğunlaşmıştır. Başka bir ayrım ise, konuşma *tanıma* ile konuşma *anlama* arasındadır.

Konuşma tanımayla kastedilen şey, mikrofon ve ilintili elektronik donanım sayesinde elde edilmiş akustik konuşma girdisini, o konuşmanın sözcük bileşenlerinden oluşan bir metin gösterimine dönüştürmektir. Bu sürecin zorlu oluşunun sebebi, pek çok akustik ses öbeğinin, kulağa benzer gelen ama oldukça farklı sözcüklerden meydana gelmesidir. (Örneğin şu cümlelerin konuşulduğunu düşünün: *There are many ways to recognize speech* ve *There are many ways to wreck a nice beach.*) Bunun aksine konuşma anlama, konuşulan şeyin *anlaşılmasını* gerektirir. Bir söz eğer uygun bir eyleme ya da tepkiye yol açıyorsa, o sözün anlaşıldığı söylenebilir, ki o sözün *tüm* sözcükleri anlaşılmadan bile yapılabilir.

Konuşma anlama, metin anlamadan daha zor, çünkü konuşma dalgabiçimini (*waveform*) işleyip söylenmiş sözcükleri belirlemek gibi ek bir sorun vardır. Mikrofonun yakaladığı konuşma, osiloskopta gösterilen elektronik sinyale ya da dalgabiçimine dönüştürülür. Resim 17.1'de, *This is a test* (bu bir deneme) diyen kişinin yarattığı dalgabiçimini gösteriyorum. Bu diyagram, zaman içinde konuşmanın genliğindeki (voltaj) değişiklikleri gösteriyor. Dalgabiçiminde, sözcüklere denk gelen kısımlar, diyagramın üst tarafında kutuya alınmış. Alttaki kutucuklar, bu sözcüklerin, "ses" (*phone*) denen akustik öğelerini gösteriyor.



Resim 17.1. Konuşma dalgabiçimi. (Gunish Rai Chawla'nın izniyle kullanılmıştır.)

Genel olarak sesler, sesli ya da sessiz harflere denk gelen seslerdir. İngilizce konuşmanın yaklaşık kırk farklı sesten meydana geldiği düşünülüyor. Seslerin gösterimi için özel alfabeler tasarlanmıştır. Bir tanesi, bilinen tüm dillerdeki sesleri içeren Uluslararası Sesbilgisel Alfabe'dir (IPA; International Phonetic Alphabet). IPA, standart bilgisayar kodları (ASCII) olmayan bazı özel yazı karakterleri kullanır. Sadece Amerikan İngilizcesindeki sesleri içerip yalnızca standart yazı karakterleri kullanan alfabeyse ARPAbet'tir; bu alfabe, DARPA'nın desteklediği konuşma işleme araştırmaları sırasında geliştirildi. Resim 17.1'de kutucuk içine alınmış sesler, ARPAbet işaretleri kullanıyor. Resim 17.2'deki tabloysa, ARPAbet seslerini ve bunları içeren örnek sözcükleri gösteriyor.

Konuşma tanıma sistemlerinin ilk örnekleri, konuşma dalgabiçimini ses bileşenlerine ayırıp, bu seslerden sözcük oluşturmaya çalışıyordu. Bu amaçla, konuşma sinyali önce sayısal hale getirilir ve frekans ya da ses perdesi gibi çeşitli parametreleri belirlenir. Bu parametlerin değerlerinin zamanla değişme şekli, dalgabiçimini, ses içeren birimlere ayrıştırmak üzere kullanılır. Dalgabiçimi parametrelerinin değerlerini seslerle, sesleriyse sözcüklerle ilişkilendiren sözlükler kullanılarak, sonuçta dalgabiçimi, metne dönüştürülür. Bu süreç kulağa basitmiş gibi geliyor ama aslında oldukça karmaşıktır, çünkü, başka etkenlerin yanı sıra, telaffuz edilen sözcüklerin ve bunları oluşturan seslerin başları ve sonları, karmaşık örüntüler içinde üst üste biner ve insanlar çoğunlukla aynı sözcüğü farklı şekillerde telaffuz eder. Örneğin *you* sözcüğü, *are you* [aa r y uw] ve *did you* [d ih d jh uh] ifadelerinde farklı telaffuz edilebilir.

Konuşma tanıma girişimleri, 1930'larda Bell Laboratuvarları'nda başladı. 1952'de Bell Laboratuvarları mühendisleri, tek konuşmacının dile getirdiği, "sıfır"dan "dokuz"a kadar rakamları tanıyan bir sistem

Simge	Örnek Ses	Simge	Örnek Ses
Sessizler		Sesliler	
[p]	**p**at	[iy]	l**i**ly
[t]	**t**om	[ih]	m**i**ss
[k]	**c**at	[ey]	l**a**zy
[b]	**b**oy	[eh]	m**e**ss
[d]	**d**ip	[ae]	**a**fter
[g]	**g**arment	[aa]	**o**p
[m]	**m**at	[ao]	**o**rchestra
[n]	**n**ut	[uh]	w**oo**d
[ng]	si**ng**	[ow]	l**o**tus
[f]	**f**ive	[uw]	t**u**lip
[v]	**v**ove	[uh]	b**u**tter
[th]	**th**istle	[er]	b**ir**d
[dh]	**th**ather	[ay]	**i**tem
[s]	**s**at	[aw]	fl**ou**r
[z]	ha**z**e	[oy]	t**oi**l
[sh]	**sh**ash	[y uw]	**few**
[zh]	ambro**s**ia	[ax]	ruff**i**an
[ch]	**ch**ic	[ix]	l**i**p
[jh]	**j**ail	[axr]	leath**er**
[l]	**l**ick	[ux]	d**u**de
[w]	**w**ick		
[r]	**r**ace		
[y]	**y**ew		
[h]	**h**orse		
[q]	uh-oh (glottal stop)		
[dx]	bu**tt**er		
[nx]	wi**n**ter		
[el]	thist**le**		

Resim 17.2. ARPAbet sesbilgisi alfabesindeki sesli ve sessiz harfler.

kurdu.[1] 1950'lerde ve 1960'larda, RCA Laboratuvarları, MIT, Japonya, İngiltere ve Sovyetler Birliği'nde de çalışmalar yapıldı.[2] 1970'lerde çalışmalar ivme kazandı; şimdi bunlardan bazılarını anlatacağım.

1 K. H. Davis, R. Biddulph ve S. Balashek, "Automatic Recognition of Spoken Digits", *Journal of the Acoustical Society of America*, c. 24, no: 6, 1952, s. 627-642.

2 İlk çalışmaların tarihçesi için bkz. B. H. Juang ve Lawrence R. Rabiner, "Automatic Speech Recognition–A Brief History of the Technology Development"; bkz. http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/354_LALI-ASRHistory-final-10-8.pdf; ya da Sadaoki Furui, "50 Years of Progress in Speech and Speaker Recognition"; bz. http://www.furui.cs.titech.ac.jp/publication/2005/SPCOM05.pdf.

17.2 KONUŞMA ANLAMA ÇALIŞMA EKİBİ

1966'nın ikinci yarısında "baş uzman" olarak DARPA Enformasyon İşleme Teknikleri Dairesi'ne (IPTO) girip daha sonra buranın müdürü olan Larry Roberts, konuşma anlayan sistemler kurmaya merak sarmıştı. O zamanlar ABD Kara Kuvvetlerinde teğmen olarak hizmet veren Cordell Green, 1970'lerin ilk yarısında Roberts'in yönetimindeki IPTO'ya atandı ve YZ araştırma projelerine mali destek bulup gözetme görevine getirildi. Green'in aktardığına göre Roberts ona, "Konuşmayı tanıyabilen bir sistem hakkında uygulanabilirlik incelemesi yap" demiş.[3]

Böylece, Mart 1970'in sonlarında Green, Carnegie Mellon Üniversitesi'nde bazı DARPA üstlenicileri ve konuşma işlemeyle ilgilenen kişilerle bilgisayarların konuşma anlamasının olabilirliğini tartışmak için bir toplantı düzenledi. Toplantıya katılanlar arasında, SDS, Lincoln Laboratuvarı, MIT, CMU, SRI ve BBN araştırmacıları vardı. Toplantıda, bir "çalışma ekibi" kurulması kararlaştırıldı; ekip, bu dalın o anki durumunu belirleyip konuşma anlama konusunda DARPA destekli büyük bir proje başlatmayla ilgili önerilerde bulunacaktı. Ekibin başkanlığını CMU'dan Allen Newell yapacaktı.[4]

Mart ayındaki bu toplantı sırasında Roberts, kendi aklındaki konuşma-anlama sisteminden bahsetmeye ikna olmuştu. Çalışma ekibinin, Roberts'ın yorumları hakkında değerlendirmesine göre, Roberts, pek çok ortak kullanıcıdan gelen kesintisiz konuşmaları telefon üzerinden 10.000 sözcüklük kelime dağarı kullanan, gerçek zamanın birkaç katı yavaşlıkta %10'un altında anlamsal hata yapacak ve 1973'te kullanıma hazır bir sistem düşünüyordu.

Çalışma ekibi ilk toplantısını 26, 27 Mayıs 1970 tarihlerinde BBN'de düzenledi. O toplantıda çalışma ekibi, anlama sisteminin başa çıkabileceği bazı özel görevleri ele aldı. Bunlar arasında, veri idaresiyle ilgili soruları yanıtlamak, bir bilgisayarın işleyişi hakkındaki soruları yanıtlamak ve bir bilgisayar işletim sistemiyle ilgili danışmanlık yapmak vardı.

Ekibin son toplantısı 26-28 Temmuz 1970'te Santa Monica'da SDC'de yapıldı. Ekibin tavsiyesi (kısaca), "genel Amerikan lehçesi"yle

3 C. Cordell Green, "AI During IPTO's Middle Years", *Expert Systems and Artificial Intelligence: Applications and Management* içinde, Thomas C. Bartee (ed.), Indianapolis: Howard W. Sams & Co., 1988, s. 240.

4 Ekibin öteki üyeleri: Systems Development şirketinden Jeffrey Barnett, Lincoln Laboratuvarı'ndan James Forgie, o zamanlar DARPA'da görevli ABD Kara Kuvvetleri teğmeni olan C. Cordell Green, MIT'den Dennis Klatt, o zamanlar MIT'de olan J. C. R. Licklider, SRI'den John Munson, CMU'dan Raj Reddy ve BBN'den William Woods.

konuşan pek çok ortak konuşmacının kesintisiz konuşmasını yüksek kaliteli bir mikrofon (telefon değil) üzerinden tanıyabilen, seçilmiş 1.000 sözcüklük (10.000 sözcük değil) kelime dağarını "son derece yapay sözdizimiyle" birlikte kullanan, gerçek zamanın birkaç katı yavaşlıkta %10'dan az hatayla veri idaresi ya da bilgisayarın durumunu bildirme (ama danışma değil) gibi görevlerle ilgilenen ve mütevazı bir başarı şansıyla 1976'da (1973'te değil) kullanılabilecek bir sistemin hedeflenmesiydi. Çalışma ekibinin son raporu toplantıdan sonra yazıldı, DARPA'ya ulaştırıldı ve sonunda 1973'te yayımlandı.[5]

Bilgisayarla konuşma işleme konusunda geçmişte çok araştırma yapılmış olsa da (çalışma ekibinin raporunda gayet güzel özetlenmiştir), bunun başarılacağı konusunda herkes iyimser değildi. Konuşma tanıma üzerinde epey çalışmış olan Bell Laboratuvarları araştırmacılarından John R. Pierce kötümserler arasındaydı. Pierce, *Journal of the Acoustical Society of America* dergisine 1969'da yazdığı makalede,[6] konuşma tanıma üzerine çalışan çoğu insanın "deli bilimci ve güvenilmez mühendis" gibi davrandığını ileri sürüp şunu eklemiştir: "Tanıma üzerine çalışan tipik kişi, 'problemi' çözebileceğini iyice kafasına sokar." Gerçi aynı mektupta şunu da yazar:

> ... tanıma cihazı, söylenen şeyi anadili kolaylığında (yani dili akıcı konuşan bir yabancıdan daha iyi) *anlamadığı* müddetçe, sistemin performansı çok kısıtlı kalacaktır. O halde, insanlar konuşma tanıma konusunda çalışmayı sürdürmeli mi? Belki de bu, o alandaki insanların karar vereceği bir şey. [Vurgu bana ait.]

17.3 DARPA KONUŞMA ANLAMA ARAŞTIRMA PROGRAMI

Gerçekten de kararı o alandaki insanlar vermiştir. Ekim 1971'de Roberts, DARPA'da, büyük oranda çalışma ekibinin çabalarını temel alan beş yıllık bir Konuşma Anlama Araştırma (SUR) programı başlattı. Bu

5 Bu rapor, *Artificial Intelligence* dergisinin özel sayısı olarak yayımlanmıştır. Allen Newell ve diğ., *Speech Understanding Systems: Final Report of a Study Group*, New York: American Elsevier Publishing Co., Inc., 1973. Raporun taslağı Newell koleksiyonunda; bkz. http://diva.library.cmu.edu/webapp/newell/item.jsp?q=box00105/fld08162/bd10001/doc0001/.

6 J. R. Pierce, "Whither Speech Recognition?", *Journal of the Acoustical Society of America*, c. 46, no: 4, İkinci Kısım, 1969, s. 1049-1051. Ayrıca Arthur Samuel'in şerhine ve Pierce'ın Samuel'e yanıtına ve başka şerhlere bakınız: *Journal of the Acoustical Society of America*, c. 47, no: 6, İkinci Kısım, 1970, s. 1616-1617.

programın yıllık bütçesi yaklaşık 3 milyon dolardı. CMU, Lincoln Laboratuvarı, BBN, SDC ve SRI'yla, belli sistemleri kurmaları için anlaşılmıştı. Tamamlayıcı araştırmalar, Haskins Laboratuvarları, Konuşma İletişim Araştırma Laboratuvarı, Sperry Univac Konuşma İletişim Dairesi ve Berkeley California Üniversitesi'nde yürütülecekti.

1976'da bu uğraşların bazılarının gösterimi yapıldı ve programın hedefleriyle sınanan sistemler elde edildi. Bunlardan ikisini, HARPY ve HEARSAY-II'yi CMU geliştirmişti. BBN, HWIM'i (Hear What I Mean) yaratmıştı. SRI ile SDC bir ortaklık kurmuş, ve bu ortaklıkta SDC, akustik işleme bileşenlerini, SRI ise çözümleme ve anlam bileşenlerini geliştirmişti. Ancak, SDC'nin uğraşı, bilgisayar erişiminde zorluklar yaşadı; dolayısıyla birleşik SRI/SDC sistemi asla resmen sınanmamıştır. BBN'nin çalışmalarını kısaca özetledikten sonra CMU çalışmalarını ayrıntılarıyla betimleyeceğim.[7]

17.3.1 BBN'deki Çalışmalar

SPEECHLIS, BBN'de geliştirilmiş ilk konuşma anlama sistemiydi. Ay taşı veritabanıyla ilgili sözlü sorulara yanıt vermesi için tasarlanmıştı (BBN'nin LUNAR sistemiyle aynı veritabanından faydalanıyordu). Bu sistem, görece yavaştı ve sistemli biçimde sınanmamıştır.[8]

HWIM, bir yolculuk bütçesi müdürünün otomatik yardımcısı olmak üzere tasarlanmıştı ve "Konuşma anlama bütçesinde ne kadar para kaldı?" gibi sorulara yanıt verebiliyordu.[9] Bu sürümünde HWIM, her biri üç erkek konuşmacının dile getirdiği altmış dört farklı sözden oluşan iki sürümle sınandı. Bu cümlelerin otuz biri, henüz tasarım aşamasındayken sistem tarafından kullanılmıştı, dolayısıyla program, o cümlelerle başa çıkabilecek (kasıtlı olmasa da) örtük ve yerleşik bir ek yeti barındırıyor olabilir. Cümlelerin uzunluğu, üç ila on üç sözcüktü. HWIM, cümlelerin %41'ine doğru yanıt verirken, %23'üne doğruya "yakın" ya-

7 SRI çalışmasının betimlemesi için bkz. Donald E. Walker (ed.), *Understanding Spoken Language*, New York:Elsevier North-Holland, Inc., 1978.

8 Daha fazla ayrıntı için bkz. William A. Woods, "Motivaton and Overview of BBN SPEECHLIS: An Experimental Prototype for Speech Understanding Research", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, c. ASSP-23, no: 1, Şubat 1975, s. 2-9.

9 Bkz. J. Wolf ve William A. Woods, "The HWIM Speech Understanding System", *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'77*, c. 2, Mayıs 1977, s. 784-787. Ayrıca (tüm ayrıntılar için) bkz. William A. Woods ve diğ., *Speech Understanding Systems – Final Report*, BBN Report no: 3438, c. I-V, Bolt, Beranek, and Newman, Inc., Cambridge, MA, 1976.

nıt verdi. Cümlelerin %20'sine yanıt bile vermedi. Her ne kadar hem SPEECHLIS hem de HWIM, konuşma anlama sürecinde yeni ve önemli yöntemlerin öncülüğünü yapmış olsa da, HWIM'nin performansının, genelde DARPA'nın asıl hedeflerini karşılamadığı düşünülür. (Tasarımcılar, bu denemenin, HWIM'nin potansiyelini göstermediğini ve zaman olsa daha iyi iş çıkartabileceklerini iddia etmiştir.)

17.3.2 CMU'daki Çalışmalar

Raj Reddy 1969'da Stanford'dan ayrılıp Carnegie Mellon Üniversitesi'nde öğretim üyesi oldu. O ve meslektaşlarının CMU'da üzerinde çalıştıkları ilk konuşma sistemlerinden biri HEARSAY'di (ileride buna HEARSAY-I denecekti).[10] Bu sistem, satrançta belli bir konumdan yapılan, örneğin "c2'deki piyon c4'e" gibi sözlü hamleleri tanımak için birbirinden bağımsız pek çok hesaplama sürecinden faydalanıyordu. Bu çalışmanın ilk safhaları sırasında, DARPA, Konuşma Anlama Çalışma Ekibini kurup konuşma anlama konusunda çalışmalar başlatmıştı. Kesintisiz konuşmaları tanıyan HEARSAY-I'in kamuya açık gösterimi Haziran 1972'de düzenlendi.

CMU'da DARPA konuşma anlama araştırmaları şemsiyesi altında, konuşma tanımaya ve anlamaya yönelik üç farklı sistem geliştirildi. Bunlar, DRAGON, HARPY ve HEARSAY-II'ydi ve hepsi, önemli YZ fikirleri öne sürerek alana katkıda bulunmuştur. Bu sistemlerle ilgili çalışmalara Allen Newell, Raj Reddy, James Baker, Bruce Lowerre, Lee Erman, Victor Lesser ve Rick Hayes-Roth önderlik etmiştir.[11]

A. DRAGON

CMU'nun konuşma anlama araştırmalarının ilk günlerinde, doktora öğrencisi James K. Baker, "DRAGON" adını verdiği konuşma anlama sistemi üzerinde çalışmaya başladı.[12] (Allen Newell'a göre DRAGON

10 D. Raj Reddy, Lee D. Erman ve Richard B. Neely, "A Model and a System for Machine Recogntion of Speech", *IEEE Transactions on Audio and Electroacoustics*, c. AU-21, no: 3, Haziran 1973, s. 229-238; ve D. Raj Reddy, Lee D. Erman, R. D. Fennell ve Richard B. Neely, "The HEARSAY Speech Understanding System: An Example of the Recogntion Processes", *Proceedings of the 3rd Internatonal Joint Conference on Artificial Intelligence* içinde, Stanford, CA, Ağustos 1973, s. 185-183.

11 Bu dönem boyunca CMU'daki konuşma işleme çalışmalarının geçmişi için bkz. Lee D. Erman, "Overview of the HEARSAY Speech Understanding Research", *SIGART Newsletter*, no: 56, Şubat 1976, s. 9-16.

12 James K. Baker, "Stochastic Modelling as a Means of Automatic Speech Recognition", doktora tezi, Bilgisayar Bilimi Bölümü, Carnegie Mellon Üniversitesi, Pittsburgh, PA,

adı, bunun, "konuşmayla ilgili çabaların geri kalanında kafa yorulan YZ sistemlerinden bütünüyle farklı bir yaratık olduğuna işaret ediyordu."[13]) HEARSAY-I gibi DRAGON da, satranç hamleleriyle ilgili cümleleri anlaması için tasarlanmıştı.

DRAGON, konuşma işleme konusunda yeni güçlü teknikler getirmişti; bunların geliştirilmiş hali, günümüzün çoğu konuşma tanıma sisteminde kullanılır. Bu sistem, gözlenen konuşma sinyalini üretmiş olabilecek en olası sözcük zincirleri hakkında tahminler yapmaya yarayan istatistiksel teknikleri kullanıyordu. DRAGON, YZ'ye olasılık gösterimlerini ve bununla ilişkili hesaplama yöntemlerini ithal etmenin ilk örneklerinden biriydi. Bunları ileriki bölümlerde sık sık göreceğiz.

Bununla ilgili başlıca fikirleri, matematiğe fazla dalmadan anlatmaya çalışacağım. Bölüm 2.3.2'de tanıttığımız gösterimi kullanmak gerekirse, diyelim ki x bir sözcük zinciri, y de x dile getirildiğinde ortaya çıkan konuşma dalgabiçimini göstersin. (Aslında y, çeşitli frekans bantlarında dalgabiçiminin içerdiği enerji miktarı gibi kolaylıkla ölçülebilir nitelikler bağlamında, dalgabiçiminin bilgi-içeren bir gösterimidir. İşi sadeleştirmek için, aslında bir gösterim olduğunu kastetsem bile, y'ye dalgabiçimi diyeceğim; y, farklı konuşma anlama sistemleri için farklı olabilir.)

Aynı konuşmacı farklı durumlarda aynı sözcükleri bir parça farklı telaffuz ettiği için, ayrıca farklı konuşmacılar aynı sözcüğü kesinlikle farklı telaffuz edeceği için, sözcük zinciri x, konuşma dalgabiçimi y'nin ne olacağını bütünüyle belirlemez. Yani belirli bir x için, sadece farklı y'lerin olasılığının ne olabileceğini söyleyebiliriz. İkinci Bölüm'de anlatıldığı gibi, bu olasılıklar, $p(y|x)$ olarak bir fonksiyon biçiminde yazılır ("verilmiş bir x için y olasılığı" olarak okuyun). Prensipte, verilmiş bir x için, örneğin $x = X$ için $p(y|x)$'nin gerçek değerleri, örneğin, X sözcük zincirini birçok konuşmacının pek çok farklı kez telaffuz etmesiyle ve farklı y konuşma dalgabiçimleri sıklığının çizelgeye yerleştirilmesiyle tahmin edilebilir. Bu süreç, pek çok farklı sözcük zinciri için tekrarlanmak zorundadır. DRAGON, bu zorlu çizelgeleme işinden, kısaca açıklayacağımız biçimde kaçınır.

1975 ve James K. Baker, "The DRAGON System–An Overview", *IEEE Transactons on Acoustics, Speech, and Signal Processing*, c. ASSP-23, no:1, Şubat 1975.

13 Allen Newell, "Happy, Production Systems and Human Cognition", *Perception and Production of Fluent Speech* içinde, Ronald A Cole (ed.), Hillsdale, NJ: Lawrence Erlbaum Associates, 1980. Carnegie Mellon Üniversitesi Teknik Raporu CMU-CS-78-140 olarak şu internet adresinde: http://diva.library.cmu.edu/webapp/newell/item.jsp?q=box00089/fld06145/bdl0001/doc0001/.

Bununla birlikte, konuşma tanımada verilmiş bir konuşma sinyali y için, sözcük zinciri x'in olasılığını bilmek isteriz, böylece en olası x'i seçebiliriz. Yani $p(y|x)$ yerine $p(x|y)$'yi isteriz. İstenen olasılığı aşağıdaki gibi belirtmek için, önceki gibi Bayes kuralından faydalanabiliriz:

$$p(x|y) = p(y|x)p(x)/p(y)$$

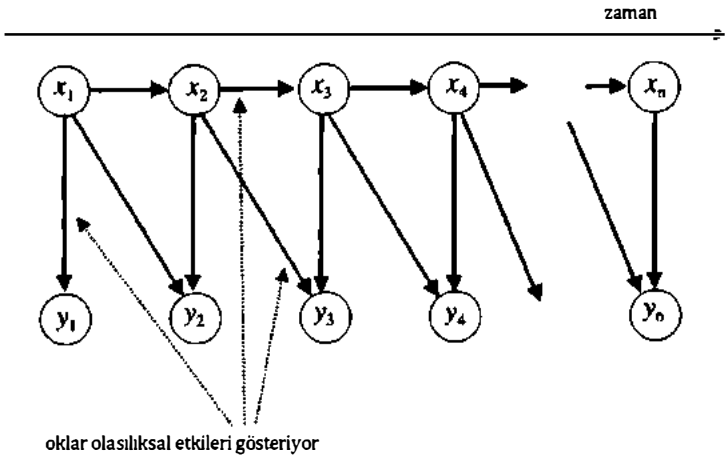
Diyelim ki $y = Y$ gibi belirli bir dalgabiçimini gözlemlediğimizde, hangi sözcük zinciri x'in söylenmiş olduğunu belirlemek amacıyla şu nicelikleri kullanırız:

1. Gözden geçirdiğimiz tüm x değerleri için $p(Y|x)$'in tüm değerlerine göz at. (Bu işi, olası *tüm* sözcük zincirleri için yapmak zorunda değiliz; sadece, konuşma anlama görevine uygun düşen özelleşmiş alanın kelime dağarının ve sözdiziminin izin verdiği sözcük zincirleri için yaparız, örneğin DRAGON sisteminde satranç hamleleri gibi.)
2. Tüm bu değerleri $p(x)$'le çarp. (Karar, olasılığı yüksek sözcük zincirlerini yeğlemeli.)
3. Çarpımın en yüksek olduğu x'i seç, örneğin X gibi. [$p(y)$'ye bölmeyi göz ardı edebiliriz, çünkü bunun değeri, hangi $p(x|Y)$'nin daha büyük olacağını etkilemez.]

Her ne kadar bu süreç prensipte işe yarasa da, hesaplamalı olarak uygulaması elverişli değildir. Bunun yerine DRAGON ve günümüzün konuşma tanıma sistemleri, konuşma dalgabiçiminin yaratılma şekli olduğu düşünülen hiyerarşik yapıdan yararlanır. Bu hiyerarşide, tanımlanabilecek çeşitli düzeyler bulunur. Biraz sadeleştirmek gerekirse, hiyerarşinin tepesinde belli bir anlamsal fikir, dilin sözdizimi kurallarına uyan bir sözcük zinciriyle ifade edilir. Ardından bu sözcük zinciri, bir ses zinciri doğurur: sesbilgisi birimleridir bunlar. Son olarak, hiyerarşinin dibinde ses zinciri, bir konuşma dalgabiçimiyle ifade edilir.

Her düzeyde, $x_1, x_2, \ldots x_n$ gibi diziler $y_1, y_2, \ldots y_n$ benzeri başka diziler yaratır. Bu süreci, Resim 17.3'teki gibi resmedebiliriz.

DRAGON sistemi, sadeleştirici kimi varsayımlarda bulunmuştur. x dizisindeki her x_i'nin sadece bir önceki x_{i-1}'den etkileneceğini, önceki x_i'den etkilenmeyeceğini varsayar. Bu varsayıma, Markov varsayımı denir. [Andrey Andreyeviç Markov (1856-1922), Rus matematikçiydi. Puşkin'in *Yevgeni Onegin* adlı romanından alınmış 20.000 Rusça harflik bir dizinin istatistiğini çözümlemek üzere bir Markov modelin-



Resim 17.3. Konuşma yaratımında iki hiyerarşik düzey.

den faydalanmıştı (adına ileride böyle denecekti).[14] Markov modelleri fizik ve mühendislikte yaygınlıkla kullanılır. Örneğin Google, sayfaları sıralarken Markov varsayımından faydalanır.] Elbette, bir dizideki her sözcüğün, kendisinden hemen önce gelen sözcüğe daha fazla bağlı olduğunu biliyoruz. Yine de Markov varsayımı, hesaplamaları basitleştirip yüksek performans alınmasını sağlar.

Dahası, her y_i'nin sadece x_i'den ve x_{i-1}'den etkilendiği varsayılmıştı. Tüm bu "etkiler" olasılıksaldır. Yani, x_3 ve x_4 nicelikleri belli olduğunda, y_4 değeri bütünüyle belirlenmiş olmaz. Bu durumda, sadece y_4 değerinin olasılıklarının ne olabileceği söylenir; bu olasılıklar, $p(y_4 \mid x_3, x_4)$ fonksiyon ifadesiyle belirtilir. Dolayısıyla y'ler için olasılık değerleri, "Markov sürecinin olasılık fonksiyonu" denen şey tarafından verilir. Bu olasılıkların tahminlerini oluşturmak amacıyla, bir "öğrenme süreci" sırasında istatistiksel veriler toplanabilir (bu süreçte konuşmacı, bir dizi alıştırma cümlesi söyler).

DRAGON, bu ayrı düzeyleri birleştirip, Markov süreçlerinin olasılık fonksiyonları hiyerarşisinden meydana gelen bir ağ oluşturur. Konuşma dalgabiçiminin bölümlerini gösteren öğeler hiyerarşinin en altında, sesleri gösteren öğeler ortada, sözcükleri gösteren öğeler en üsttedir. Her düzeyde, belli y'ler için x olasılıklarını hesaplamak ama-

14 İngilizce çevirisi için bkz. A. A. Markov, "An Example of Statistical Investigation of the Text *Eugene Onegin* Concerning the Connection of Samples in Chains", *Science in Context*, c. 19, no: 4, 2006, s. 591-600.

cıyla Bayes kuralı kullanılır. Sadece alt düzeydeki konuşma dalgabiçimi gerçekten gözlemlendiği için, seslerin ve sözcüklerin "saklı" olduğu söylenir. Bu sebeple bütün ağ, Saklı Markov Modellerini (SMM) kullanır. DRAGON, YZ alanında SMM kullanımının ilk örneğidir. Oysa bu modeller, daha önce başka amaçlar uyarınca geliştirilmişti.[15]

Bu ağ kullanılarak, bir sözün tanınması, bu ağda en olası yolun bulunmasıyla başarılıyordu. Gözlenen konuşma dalgabiçiminin belirli kısımları için, sözdizimi açısından geçerli sözcük dizilerinin olasılığını hesaplamak, daha önce tanımlamış olduğum probleme, yani FORTRAN kodu sayfalarında yazı karakterlerinin doğruluk payını hesaplamaya benzer. Yine, dinamik programlamaya dayanan bir yöntem kullanılmıştır. Baker'ın yazdığı gibi, "En iyi yolu bulan algoritma, pratikte tüm yolları araştırmıştır."[16] Sürecin sonunda, geçerli sözdizimi bakımından en olası sözcük zinciri tespit edilir. Bu hesaplamalarda kullanılan matematik işlemleri, burada açıklanamayacak kadar karmaşıktır ama konuşma tanımayı uygulanabilir kılacak biçimde gerçekleştirilebilirler.

Her ne kadar DRAGON sistemi, DARPA'nın konuşma anlama sistemleri için belirlemiş olduğu hedefler için sınanmış sistemler arasında olmasa da, Baker, başlangıç sonuçlarının "çok umut vaat ettiğini" ve "canlı konuşma girdisi kullanılan ilk testte sistemin dokuz cümledeki her sözcüğü doğru şekilde tanıdığını" iddia etti.[17] DRAGON, Baker ile eşi Janet'in kurduğu Dragon Systems adlı şirketin geliştirip pazarladığı ticari ürün "Dragon Naturally Speaking"in temeli olmuştur.

B. HARPY

HARPY, CMU'da DARPA'nın konuşma anlama araştırmaları kapsamında üretilen ikinci bir sistemdi. Bruce T. Lowerre bu sistemi, doktora araştırmasının bir parçası olarak tasarlayıp hayata geçirmişti.[18] HARPY, HEARSAY-I'de ve DRAGON'da faydalanılan kimi fikirleri bir araya getirmişti. Tıpkı DRAGON gibi, HARPY de sözlü ifade edilmiş cümleleri tanımak amacıyla bir ağ içindeki yolları araştırıyordu, fakat

15 Bkz. L. E. Baum ve J. A. Eagon, "An Inequality with Applications to Statistical Estimation for Probabilistic Functions of a Markov Process and to a Model for Ecology", *Bulletin of the American Medical Society*, c. 73, 1967, s. 360-363. Baker, Markov sürecinin olasılıksal fonksiyonunu kendisine Baum'un tanıttığını belirtir.

16 James K. Baker, "The DRAGON System–An Overview", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, c. ASSP-23, no: 1, Şubat 1975, s. 24.

17 *A.g.e.*, s. 29.

18 Bruce T. Lowerre, "The HARPY Speech Recognition System", doktora tezi, Bilgisayar Bilimi Bölümü, Carnegie Mellon Üniversitesi, Pittsburgh, PA, Nisan 1976.

DRAGON'un aksine, ağdaki düğüm noktaları ile geçiş olasılıkları arasındaki bağlantıları açıklamıyordu. Ayrıca HARPY, tıpkı HEARSAY-I gibi, sezgisel arama yöntemlerinden faydalanıyordu.

Farklı görev alanlarıyla ilgili sözlü cümleleri anlayacak çeşitli HARPY sürümleri geliştirilmiştir. Başlıcalarından biri, YZ makalelerinin özetlerini ("özetçe" [*abstracts*] denir) içeren bir veritabanıyla ilgili soruları yanıtlayıp bu veritabanındaki belgeleri kullanıcı önüne getiriyordu. Bazı örnekler şöyledir:

"Hangi özetçelerde hesaplama kuramına atıfta bulunuluyor?"
"O makaleleri sırala."
"Aralarında hiç Feigenbaum ve Feldman makalesi var mı?"
"Bin dokuz yüz yetmiş dörtten bu yana McCarthy ne yazmış?"

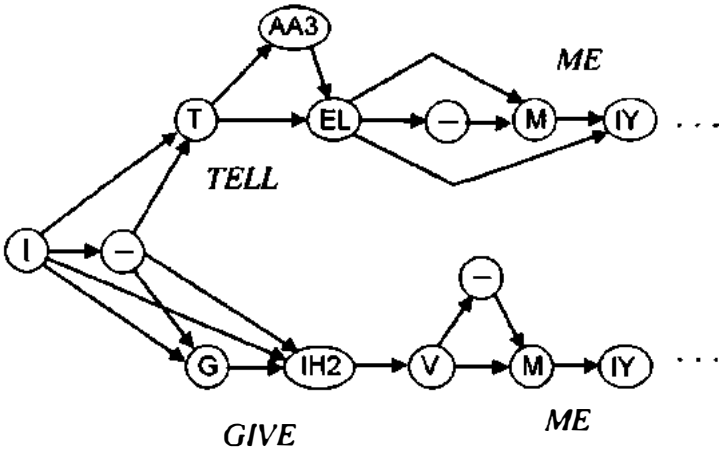
HARPY, 1.011 sözcüklük bir dağarcıkla iş görüyordu. İsim, sıfat, vesaire gibi geleneksel sözdizimi kategorileri barındıran bir dilbilgisi kullanmak yerine, HARPY, "anlamsal dilbilgisi" kullanıyordu; bu dilbilgisinin, Konu, Yazar, Yıl, Yayıncı gibi anlamsal açıdan ilgi alanlarıyla, yani YZ makalelerine dair verilerle ilintili kapsamlı kategorileri vardı. HARPY'nin dilbilgisi, HARPY'nin tanıması beklenen yazarlar ve makaleler hakkındaki cümlelerle başa çıkmakla sınırlıydı.

Sistemin ağı, "bilgi kaynakları" (BK) dediğimiz şeylerden inşa edilmişti; bilgi kaynakları, tanıma süreci için gerekli bilgilerden meydana geliyordu.[19] Bunların ilki, dilbilgisi hakkında sözdizimi bilgisini kodluyordu.

HARPY'nin kullandığı ikinci bilgi kaynağı, HARPY'nin kelime dağarındaki her sözcüğün nasıl tellafuz edilebileceğini betimliyordu. Üstelik, sözlü dilde sözcükler arası sınırlar, ilgili sözcüklere bağlı olarak örtüştüğü için, başarılı bir tanıma işlemi, bu gibi durumlarla başa çıkacak üçüncü bir bilgi kaynağı gerektirir. Dördüncü bir bilgi kaynağıysa, sözcüklerin telaffuzuyla ve sözcükler arası geçişlerle ilgili sesleri belirliyordu.

HARPY tüm bu bilgileri birleştirip, sözdizimi bakımından geçerli cümlelerin dile getirilişin tüm olası yollarını temsil eden seslerden meydana gelen devasa bir ağ oluşturur. Bu ağda her "ses düğüm noktası", konuşma dalgabiçiminin "tayf kalıbı" denilen ve özellikle o sesle ilintili olması beklenen bir kesitinin gösterimiyle eşleştirilir. Bu

19 HARPY betimlememi şu makaleye dayandırıyorum: Bruce Lowerre ve Raj Reddy, "The HARPY Speech Understanding System", *Trends in Speech Recognition*, Prentice Hall. Yeniden baskısı: A. Waibel ve K. Lee (ed.), *Readings in Speech Recognition*, San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1990, s. 576-586.



Resim 17.4. Sözlü cümlede görülebilecek seslerin kısmi ağı.

kalıplar en başta bir konuşmacıya 700 cümle okutularak elde edilir. Ayrıca "öğrenme" seansı sırasında okura yaklaşık 20 seçilmiş cümle okutturulur; bunlar yeni konuşmacıya göre de "ayarlanabilir". Resim 17.4'te, genel fikri açıklamak amacıyla bir ses ağı parçası gösteriliyor. HARPY'nin asıl ağında 15.000 düğüm noktası vardır. Resimdeki ağ, "Tell me..." ve "Give me..." sözleriyle başlayan cümlelerin bu kesiti içindir. Düğüm noktaları içindeki simgeler, sesleri gösteriyor; bu simgelerde DRAGON'un işaretleri kullanılmış. Oklar, bir sesten öbürüne olası geçişleri gösteriyor. Sözcükleri telaffuz etmenin farklı biçimlerine karşılık gelen birden çok yol olduğuna dikkat edin.

Sözlü cümle içindeki sözcükleri tanımak için, gözlemlenen konuşma dalgabiçimi önce dalgabiçimindeki seslere denk geldiği tahmin edilen değişik uzunlukta kesitlere bölünür. Bu kesitlerin her biri için bir tayf kalıbı hesaplanır. Ardından tanıma süreci aşağıdaki gibi ilerler: Konuşma dalgabiçimindeki ilk tayf kesitine karşılık gelen tayf kalıbı, ağın başlangıcındaki tüm seslere karşılık gelen tüm kalıplarla kıyaslanır. Resim 17.4 üzerinden anlatacak olursak, -, T, G, IH2 kalıplarıyla kıyaslanır, çünkü bunlar, başlangıç düğüm noktasından, yani [. noktasından itibaren tek adımda ulaşılacak ağ düğüm noktaları arasındadır. (Elbette, örnekte sunulan ağ parçası yerine bütün bir ağ kullanıldığında, başlangıç düğüm noktasından itibaren tek adımda ulaşılabilecek ek düğüm noktalarının kalıplarıyla daha fazla kıyaslama yapılmış olur.) En iyi uyan birkaçı not edilir ve bu düğüm noktalarına giden yollar, ağ içinde en iyi tek adımlık yol parçası olarak belirlenir. Sonraki aşamada,

sıradaki dalgabiçimi kesitinin tayf kalıbı, en iyi tek adımlık yolları bir adım daha uzatarak ulaşılan tüm ses düğüm noktalarıyla karşılaştırılır. Şimdiye dek hesaplanmış kıyaslamaların değerleri kullanılarak, en iyi iki adımlık yol parçası kümesi belirlenir. Bu süreç, ağın sonuna ulaşılana dek sürer. O ana kadar bulunmuş en iyi yol, bu yol üzerindeki düğüm noktalarıyla ilişkili sözcüklerle ilişkilendirilir. Sonra bu sözcük dizisi, HARPY'nin tanıma kararı olarak belirlenir.

HARPY'nin bir ağda en iyi yolu arama yöntemini, daha önce betimlemiş olduğumuz A* sezgisel arama süreciyle karşılaştırılabiliriz. A*, arama sırasında "ön cephesini" olası tüm aramalar için hazır bekletirken, HARPY ön cephesinde sadece, o ana dek bulunmuş en iyi birkaç yolu tutar. (Ön tarafta tutulan düğüm noktalarının sayısı, arama sürecini denetleme gereksinimi uyarınca belirlenebilecek bir parametredir.) HARPY'nin tasarımcıları bu tekniğe "ışınlı arama" demişti, çünkü arama sürecinin dokunduğu düğüm noktaları, ağ boyunca dar bir ışınla sınırlıydı. Süreç devam ederken ışın içinde yer almayan düğüm noktaları elendiği için, HARPY'nin bulduğu en iyi tam yol, ağ genelinde en iyi yol olmayabilir. (Elenmiş düğüm noktalarından biri, bu en iyi yolun üzerinde olabilir.) Yine de, elde edilen yol çoğunlukla sözlü cümlenin doğru telaffuzuna denk gelir.

DARPA konuşma anlama projesinin sonunda HARPY, üç erkek ve iki kadın konuşmacının söylediği 100 cümleyle sınandı. HARPY, bu cümlelerin %95'ten fazlasını doğru anlayabilmiş, böylece DARPA'nın %10'dan düşük hata hedefini tutturmuştu. HARPY, bir saniyelik konuşmayı işlemek için ortalama yaklaşık 30 milyon bilgisayar komutu uyguluyordu. Saniyede 0,4 milyon komut (0,4 MIPS) gerçekleştiren bir makine (DEC PDP-KA10) kullanarak, bir saniyelik konuşmayı bir dakikayı aşkın sürede işleyebiliyordu; her ne kadar bu, gerçek zaman performansından oldukça kötü olsa da, DARPA'nın "gerçek zamanın birkaç katı" hedefine ulaşmıştı (eğer "birkaç" sözcüğünü gönlümüzce yorumlarsak). Manzaraya gerçek zamanı sokmak gerekirse, günümüz bilgisayarları saniyede milyarlarca komut işleyebiliyor. HARPY, DARPA'nın hedeflerini karşılayan tek sistemdi.

C. HEARSAY-II

Son olarak, HEARSAY-I'in yeniden tasarlanıp geliştirilmiş sürümü HEARSAY-II, CMU'nun belki de en iddialı konuşma projesiydi.[20] Tıpkı

20 Lee D. Erman, Frederick Hayes-Roth. Victor R. Lesser ve D. Raj. Reddy, "The HEARSAY-II Speech-Understanding System: Integrating Knowledge to Resolve Uncer-

HARPY gibi HEARSAY-II de YZ konulu makalelerin özetlerini içeren bir veritabanıyla ilgili soruları yanıtlayıp bu veritabanından belge bulmak üzere tasarlanmıştı. (Bundan önceki görevi, haber ajanslarının haberlerini tarayıp getirmekti.) Bu sistem de 1.011 sözcükle sınırlıydı ve o konuda uzmanlaşmış anlamsal bir dilbilgisinden faydalanıyordu.

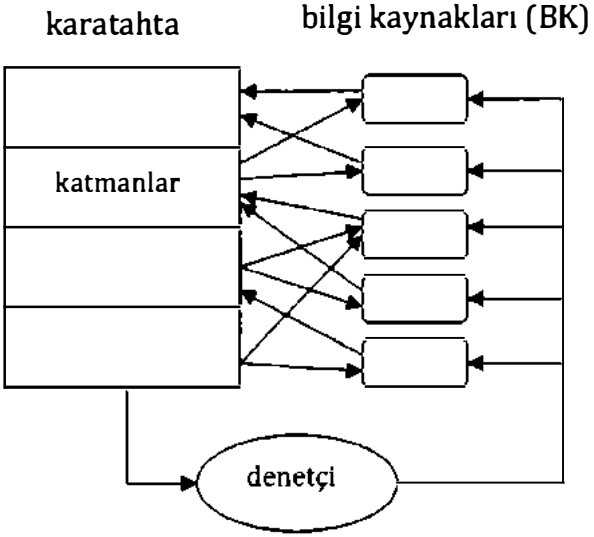
HEARSAY'in sözleri işlemesinde ilk adımlar, konuşma dalgabiçimini kesitlere ayırmayı ve her kesitte var olduğu tahmin edilen sesleri etiketlemeyi içeriyordu. Ardından HEARSAY, bu bileşenlerden hece, hecelerden sözcük, sözcüklerden sözcük dizileri ve son olarak sözcük dizilerinden sözcük öbekleri oluşturmak amacıyla yepyeni bir yöntem kullanıyordu. Bu öbekler de, YZ makalelerinin veritabanına erişmek için uygun altprogramlara dönüştürülüyordu.[21]

HEARSAY'in yararlandığı işleme yöntemi, "Karatahta" denen katmanlı bir yapı barındırıyordu. Var olduğu tahmin edilen seslerin etiketleri, varoluş olasılıklarıyla ilintili sayılarla birlikte, Karatahtanın alt katmanlarından birine "yazılıyordu". Seslerden hece oluşturmayı "bilen" uzmanlaşmış bilgi kaynağı altprogramları, bu etiketleri "okuyup" söz içinde hangi hecelerin var olduğu hakkında tahminler hesaplıyordu. Ardından bu tahminler, doğruluk paylarını ya da olabilirliklerini ölçen rakamlarla birlikte, Karatahtanın hece katmanına yazılıyordu. Hecelerden sözcük oluşturmayı bilen başka uzmanlaşmış bilgi kaynağı altprogramlarıysa, Karatahtadaki bilgiyi okuyup bunun sözcük katmanına, sözcüklerle ilgili tahminleri işliyordu. Hesaplama böyle sürüp gidiyordu. HEARSAY-II'de bu bilgi kaynaklarından yaklaşık 40 tane vardı. Sistemin ana fikri Resim 17.5'te gösteriliyor.

Prensipte, bir bilgi kaynağı Karatahtanın ilintili herhangi bir katmanından bilgi okuyabilir ya da oraya bilgi işleyebilir. Dahası, bu işi "eşzamansız" yapabilir; yani başka bilgi kaynaklarının okuma ve yazma işlerine bağlı değildir. Üstelik, sözcük katmanında zaten yazılı olan sözcükleri ve başka katmanlardaki bilgileri temel alarak, yeni sözcüklere tahminler yazan bilgi kaynakları da vardır. Hatta bilgi kaynakları, sözcük dizisi katmanında (yüksek doğruluk payıyla) yazılı sözcük dizilerine dayanarak, sözcük katmanına sözcüklerle ilgili tahminleri işleyebilir. Üst katmanda (başka bulgulara göre) var olan şeylerden yola çıkarak alt katmanda (ilk işlemede gözden kaçmış olsa bile) olması gerekenleri çıkarsama süreci, daha sonraki YZ araştırmalarında sık sık yeniden kendini gösteren bir kavramdır. Bildiğim

tainty", *Computing Surveys*, c. 12, no: 2, Haziran 1980.

21 HEARSAY'in, örnek bir cümleyi nasıl işlediğine dair ayrıntılı bir özet için bkz. *a.g.e.*



Resim 17.5. Karatahta mimarisi.

kadarıyla, son derece önemli bu YZ yeniliği ilk olarak HEARSAY-II sisteminde boy göstermiştir.

Karatahta mimarisinin mucitlerinden Raj Reddy'ye göre (Victor Lesser, Lee Erman ve Frederick Hayes-Roth'la birlikte),[22] Herbert Simon, Allen Newell'la birlikte üzerinde çalıştıkları üretim sistemi mimarisinin "işleyen bellek" bileşenini betimlemek amacıyla sık sık "karatahta" sözcüğünü kullanırmış. Bu üretim sistemi, EĞER-İSE kurallarını (bunlara ürün deniyordu) kullanıyordu ve bu kuralları, işleyen belleğin içeriği harekete geçirip içlerine yeni veriler işliyordu. Konuşma işlemeye ilişkin farklı bilgi kaynaklarının çeşitliliğini fark eden Reddy ve ekibi, üretim sistemi fikrini genelleştirip üretim kurallarını büyük programlara dönüştürerek, bunları "bilgi kaynakları" olarak isimlendirdi ve işler belleği geliştirip katmanlı Karatahta yapısını oluşturdu.

DARPA konuşma anlama projesinin sonunda HEARSAY-II, daha önce hiç karşılaşmadığı, cümle başına ortalama yedi sözcük düşen yirmi üç sözlü cümleyle sınandı ve bunların %81'ini sözcüğü sözcüğüne doğru tanıdı, ayrıca veritabanı aramasında yanıtlarının %91'i, sözcüğü sözcüğüne doğru tanınan cümlelerle aynı sonucu vermiştir. HEARSAY tasarımcıları, bu performansın, "1971'de DARPA programı için belirlenmiş... iddialı hedefleri tutturmaya yaklaştığını" söy-

22 Telefonla sohbet, 14 Ağustos 2008.

lemiştir. HEARSAY-II hedefe yaklaşmış olsa bile, elde ettiği sonuçlar HARPY'ninkiler kadar iyi değildi.

Her ne kadar Karatahta mimarisi günümüzün konuşma tanıma sistemlerinde artık kullanılmıyor olsa da, kimi YZ programları bu mimariyi benimsemiştir. (Bunları kitabın ilerleyen sayfalarında göreceğiz.) Russell ve Norvig'e göre, "Karatahta sistemleri günümüzün kullanıcı arayüzü mimarilerinin temelidir."[23]

17.3.3 SUR Programının Özeti ve Etkisi

CMU'nun HEARSAY-II ile HARPY'si 8 Eylül 1976'da CMU'da ve BBN'nin HWIM'si 10 Eylül'de BBN'de gösterildi. Bu projelerin özet raporunda MIT'den Dennis Klatt şöyle yazmıştı: "Bu üç sistemin yetenekleri arasında büyük fark olup olmadığı net değil. Ancak, ARPA'nın hedeflerini tutturan sadece HARPY'dir."[24]

HEARSAY-II'nin yaratıcıları, HARPY'nin üstün performansını üç etkene bağlamıştı: olası çözümleri daha geniş araması (dile getirilmiş olabilecek tüm cümlelerin önceden hesaplanmış ağı sayesinde), yakın sözcükler arasındaki geçiş görüngüleri hakkında daha geniş bir yerleşik bilgisi oluşu ve daha geniş şekilde sınaması, ayar yapması ve hata düzeltmesi.[25]

Bununla birlikte, kimi araştırmacılar ve DARPA idarecileri, testlerin yapılma şekli üzerine tartışmış ve hiçbir sistemin, SUR programının hedeflerini karşılamadığını ileri sürmüştür. Yine de DARPA, teklif edilen devam projesini desteklememe kararı aldı. Ancak bu program, konuşma anlamanın mantıklı bir teknik hedef olduğunu gösterip, konuşma işleme teknolojilerinde ilerlemeyi tetiklemiştir, özellikle de sistem örgütlenmesinde, sözdiziminde, anlambilimde ve ses işlemede. Ulusal Araştırma Konseyinin bir raporu, şu sonuca varmıştı: "Konuşma anlama araştırmalarına DARPA'nın sağladığı mali destek son derece önemli olmuştur... bu araştırmanın sonuçları, IBM ve BBN gibi oturmuş şirketlerin ve Nuance Communications (bir SRI girişimi) ile Dragon Systems gibi yeni şirketlerin ürünlerine girdi... Günümüz piyasasında önde gelen ticari konuşma tanıma programı, yani Dragon'un "Naturally Speaking" yazılımının [artık Nuance tarafından

23 Stuart Russell ve Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2. baskı, s. 580, Upper Saddle River, NJ: Prentice Hall, 2003.

24 Dennis H. Klatt, "Review of the ARPA Speech Understanding Project", *Journal of the Acoustical Society of America*, c. 62, no: 2, Aralık 1977, s. 1345-1366.

25 Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser ve D. Raj Reddy, *a.g.e.*

satılıyor] kökleri, doğrudan 1971 ile 1975 arasında SUR'un parçası olarak CMU'da yapılmış çalışmaya dayanıyor."[26]

17.4 KONUŞMA TANIMADA SONRAKİ ÇALIŞMALAR

Konuşma tanıma araştırmaları aynı zamanda, DARPA'nın SUR programıyla doğrudan ilişkili olan laboratuvarların dışında başka laboratuvarlarda da yürütüldü. Örneğin, New York eyaletinin Yorktown Heights bölgesindeki IBM Thomas J. Watson Araştırma Merkezi Bilgisayar Bilimleri Bölümü'nün Konuşma İşleme Ekibinden Frederick Jelinek'e, konuşma tanımada istatistiksel yöntemler (saklı Markov modelleri dahil) kullanımının ilk yandaşlarından olduğu söylenir.[27] SMM yaklaşımı sonunda tüm önde gelen konuşma tanıma şirketleri tarafından benimsenmiştir.

1984'te DARPA, "Stratejik Bilişim" programının parçası olarak konuşma tanıma çalışmalarını yeniden desteklemeye başladı (bu program ileriki bir bölümde betimlenecek). Programın katılımcıları arasında CMU, SRI, BBN, MIT, IBM ve Dragon Systems vardı. Birkaç sene içinde CMU'da geliştirilmiş sistemler arasında, örneğin, Kai-Fu Lee ile arkadaşlarının yarattığı SPHINX ve Alex Waibel ile arkadaşlarının yarattığı çok dilli konuşma ve çeviri sistemi JANUS bulunuyordu. (Bu ve başka sistemler, "Speech at CMU" internet sayfasında açık kaynak yazılımı olarak var: http://www.speech.cs.cmu.edu/. Bu sayfa aynı zamanda, başka konuşma tanıma laboratuvarlarının sayfalarına bağlantılar de içeriyor.)

James ve Janet Baker, CMU'daki DRAGON çalışmalarını temel alarak 1982'de Dragon Systems'i kurdular. 1997'de Dragon, kişisel bilgisayarlar için bir konuşma tanıma programı olan "Dragon Naturally Speaking"i piyasaya sürdü. Programın, 23.000 sözcüklük bir dağarcığı vardı.[28] Bunu, ViaVoice'le IBM ve Microsoft dahil başka şirketlerin konuşma tanıma yazılımları izledi.

26 *Funding a Revolution: Government Support for Computing Research*, Dokuzuncu Bölüm, Committee on Innovations in Computing and Communications: Lessons from History, Computer Science and Telecommunications Board, Commission on Physical Sciences, Mathematics, and Applications, National Research Council, Washington, DC: National Academy Press, 1999. Şu internet adresinde: http://books.nap.edu/openbook.php?record_id=6323& page=15.

27 Örneğin bkz. Frederick Jelinek, "Continuous Speech Recognition by Statistical Methods", *Proceedings of the IEEE*, c. 64, no: 4, Nisan 1976, s. 532-556.

28 Dragon NaturallySpeaking, günümüzde Nuance aracılığıyla elde edilebilir.

Sözlü cümlelerin metne dökülmesi günümüzde büyük oranda çözülmüş bir sorundur. Örneğin, yüksek nitelikli konuşma tanıma, günümüzde pek çok otomatik telefon yanıt sisteminde yaygınlıkla kullanılmaktadır. Bununla birlikte, örneğin, bilgisayar sistemleriyle genel diyaloglara olanak tanıyan genel dil konuşmalarını (ya da metinlerini) *anlama*, uzun vadeli bir araştırma sorunu olmayı sürdürüyor. Bu sorunla ilgili çalışmalardan bahsetmeyi ileriki bir bölümde sürdüreceğim.

18
DANIŞMA SİSTEMLERİ

18.1 SRI'NIN BİLGİSAYAR TABANLI DANIŞMANI

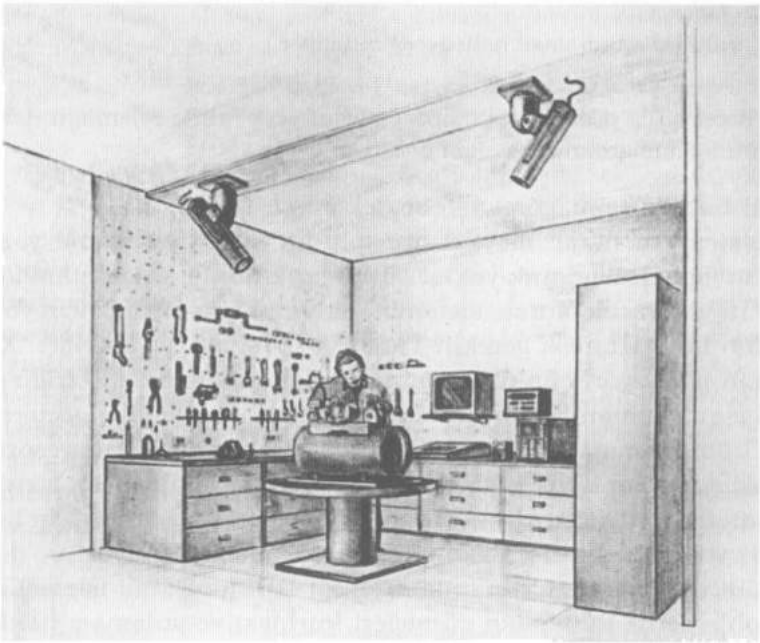
SRI'da çalışma arkadaşlarımla birlikte, bir taraftan DARPA'nın askerî geçerliliği olan uygulamalara ilgisini tatmin edip, öte yandan "Robot Shakey" projesi altında yapageldiğimiz planlama ve görme araştırmalarını sürdürme çareleri ararken, donanım bakımı, tamiri ve eğitimi sorunuyla yüz yüze geldik. Bu kalemlerin maliyetini düşürüp nadir bulunan uzmanlara gereksinimi azaltacak herhangi bir teknolojinin, ordu için son derece önemli olduğunu belirlemiştik. Dahası, bu gereksinimin, "salt daha fazla ve daha iyi kullanım kılavuzu yazmakla karşılanamayacağını, bize kalırsa gelişmiş bir bilgisayar sisteminin yaşamsal önem taşıdığını" da söyledik.

Bu sorun üzerine çalışmaya, bir tamirci çırağına küçük bir hava kompresörünü nasıl söküp montajlayacağı hakkında tavsiyeler verildiği bir ortam geliştirilerek başlanmasını önerdik.[1] Tavsiye (tamirci çırağı ihtiyaç duyduğunda), sıradaki adımda yapılacak şey hakkında fikir veren bir planlama ve gerçekleştirme sistemi tarafından yaratılır. Çırak, önceden tanımlanmış dar bir sözcük dağarcığından yararlanarak tavsiye isteyebilecekti; sistem bu isteği kusursuz bir konuşma tanıma donanımıyla çözümleyip konuşma modülüyle yanıt verecekti. Ayrıca sistem, işin ilerleyişini takip etmek ve çırağın işaret ettiği aksamı gözlemlemek için bir yapay görme sistemi de barındıracaktı. Önerilen düzenek Resim 18.1'de gösteriliyor.

Bu proje, tüm hedeflerine ulaşacak kadar sürmemiş olsa da, "Bilgisayar-Tabanlı Danışman" (CBC) sisteminin ilk sürümlerinden biriyle, önceden kısmen sökülmüş bir hava kompresörünü (Sears modeli 17209, 1/2 Yüksek Basınçlı kompresör) çırağın (proje ekibi üyelerinden Georgia Sutherland canlandırmıştı) montajlamasının istendiği bir gösterim yapıldı. Hem sistemin hem de çırağın soruları ve yanıtları sözlüydü. Sistemle girilen etkileşimlerden birinin örneği aşağıdadır (parantez içindeki yorumlarımla birlikte):[2]

1 Bkz. Nils J. Nilsson ve diğ., "Plan for a Computer-Based Consultant System", SRI AI Center, Technical Note 94, Mayıs 1974. (http://www.ai.sri.com/pubs/files/1298.pdf.)

2 Peter E. Hart, "Progress on a Computer Based Consultant", SRI AI Center Technical



Resim 18.1. Çırağa yardımcı olması için teklif edilen düzenek. (SRI International'ın izniyle)

Sistem: `Lütfen hava kompresörünü montajla.`
Çırak: `Nasıl?`

(Çırak tavsiye istiyor ve sistem, daha önce betimlemiş olduğum Sacerdoti'nin NOAH sistemini kullanarak hiyerarşik bir montaj planı kuruyor.)

S: `Pompayı kur.`
(Sistem, NOAH'ın yarattığı üst düzey plandaki ilk eylemi söylüyor.)

Ç: `Tamam.`
(Çırak, pompayı nasıl kuracağını biliyor ve bu işi yaparken daha fazla tavsiye istemediğini belirtiyor. CBC projesinin bu aşamasında sistem, görsel yoklama olmaksızın, çırağın ilk adımı tamamladığını varsayıyor.)

S: `Pompa yatağını yerleştir.`
(Sistem, üst düzey plandaki ikinci eylemden bahsediyor.)

Ç: `Nasıl?`
(Çırak, bu adımı nasıl başaracağını bilmediği için tavsiye istiyor.)

S: `Pompa yatağını pompaya iliştir.`
(Sistem, üst düzeydeki pompa yatağını yerleştirme eyleminin daha ayrıntılı adımlardan ilkini belirtiyor.)

Note 99, Ocak 1975, s. 23. (http://www.ai.sri.com/pubs/files/1389.pdf.)

Ç: `Tamam.`
(Çırak, bu adımı nasıl halledeceğini biliyor.)

S: `Pompa yatağını kasnak kayışı yuvasına bağla.`
(Sistem, üst düzeydeki pompa yatağını yerleştirme eyleminin daha ayrıntılı adımlardan sıradakini belirtiyor.)

Bu tür bir etkileşim, görev bitene dek sürer.

Sistem, montajın mevcut durumunu, NOAH sisteminin yarattığı yordam ağını kullanarak yoklar. Eğer çırak herhangi bir adımda "Neden?" diye soracak olursa, sistemin, bir yanıt oluşturmak için yordam ağından faydalanması gerekir. Yanıt da şöyle bir şey olabilir: "X işini yapman gerekiyor, çünkü y yapmakla ilgili eylemlerden biridir o" ya da "X işini yapman gerekiyor, zira bir sonraki z eylemine geçmen için önce bunu tamamlaman şart". Ayrıca çırak, hava kompresörünün ana aksamından birini yerine yerleştirirken yardım isterse, sistem lazer işaretleyicisiyle çırağa yardımcı olabilecekti. Bu örnek için hesaplanmış yordam ağının bir kısmı Resim 18.2'de gösteriliyor.

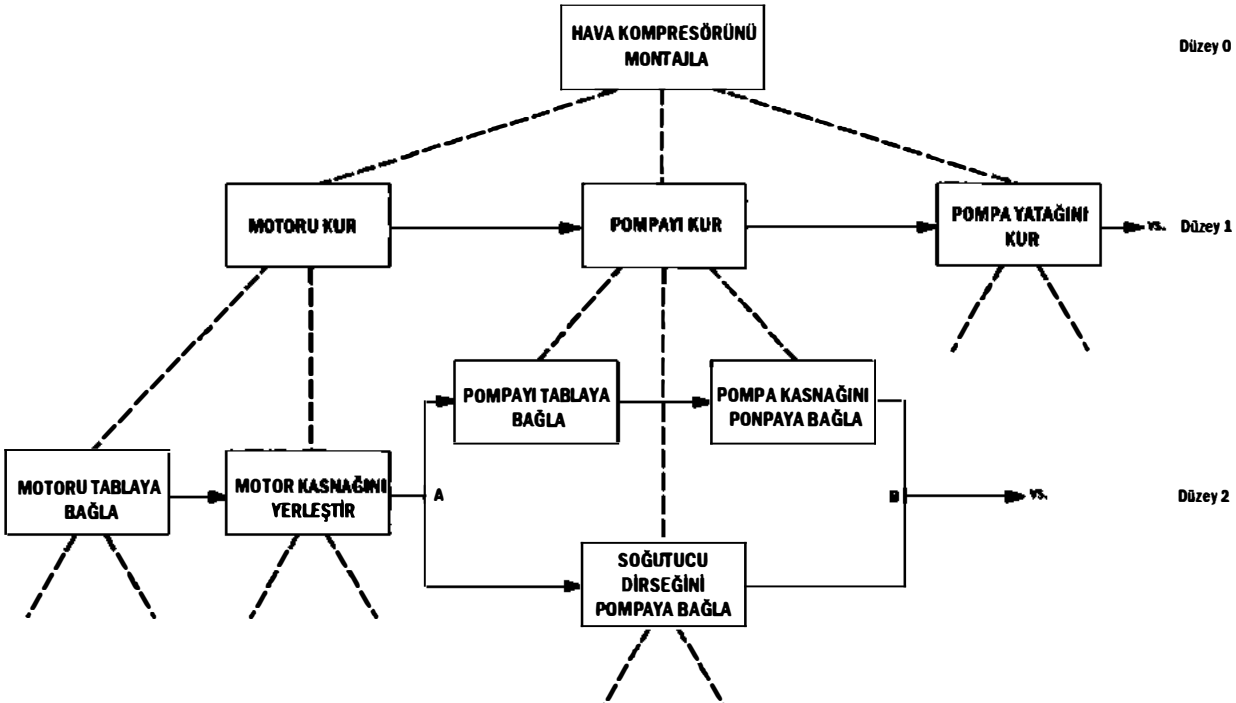
CBC projesi aynı zamanda, SRI'nın DDİ (doğal dil işleme) ekibine, sohbetlerde kullanılan cümleleri kurması ve anlaması hakkında geliştirdikleri fikirlerden bazılarını deneme fırsatı yaratmıştır. CBC projesinde çırak ve tavsiye veren kişi, belli bir görev, ismini koymak gerekirse, hava kompresörü üzerinde çalışmak hakkında konuşuyorlar. Görevin yapısı, NOAH'ın yarattığı yordam ağını modellediği haliyle, cümle anlama için faydalı olacak önemli ve edimsel bilgi temin eder. Bu bilgiden, TDUS (Task Dialog Understanding System) adlı sistem yararlanır; bu sistem, montaj işinde çırağa yol gösteren sistemin sözlü diyaloğundan daha karmaşık diyaloglarla başa çıkabilir.[3] TDUS, çırakla metin tabanlı sohbet yapılmasına olanak tanımak amacıyla NOAH planlama sistemini, (sözdizimsel, anlamsal, edimsel bileşenleri olan) doğal dil anlama sistemiyle bütünleştiriyordu.

Cümle anlamada görev yapısının oynadığı rolü açıklamak için, TDUS'la ilgili bir makaleden alınmış örnekten faydalanacağım.[4] Şu cümlelere göz atın:

Konuşmacı 1: John pompayı neden söktü?
Konuşmacı 2: *Onu* tamir etmek için *yaptı bunu.*

3 Ann E. Robinson, Douglas E. Appelt, Barbara J. Grosz, Gary G. Hendrix ve Jane J Robinson, "Interpreting Natural-Language Utterances in Dialogs About Tasks", SRI AI Center Technical Note 210, 15 Mart 1980. (http://www.ai.sri.com/pubs/files/709.pdf.)

4 *A.g.e.*, s. 11.



Resim 18.2 Hava kompresörü montajlamaya yarayan yordamsal ağın bir kısmı. (SRI International'ın izniyle kullanılmıştır.)

İkinci cümledeki vurgulu sözcüklerin atıfta bulunduğu nesneler yorumlanırken, ilk cümlenin belirlemiş olduğu görev bağlamına gönderme yapmak yardımcı olur. Gizli özne zaten John, "yaptı bunu" sökme işine atıfta bulunuyor, ve "onu" sözcüğü pompaya gönderme yapıyor. TDUS, diyaloğun değişen "bağlam"ından ve hedeflerinden epey yararlanır. TDUS'un yaratıcılarının yazdığı gibi:[5]

> Diyalog ilerledikçe, katılımcılar dikkat odaklarını sürekli değiştirir, böylece sözlerin üretilip yorumlandığı bağlam evrilir. Konuşmacı, onu dinleyen kişiye neye nasıl bakması gerektiği hakkında ipuçları verir; neye odaklanmak gerekiyor, nasıl odaklanmak gerekiyor, odaklanma ne kadar dar ya da geniş olmalı. Söylemin odaklanışı için bir gösterim, isim öbekleriyle atıfta bulunulan nesnelerin tanımlanmasında bu gösterimi kullanma yöntemleri ve odaklanıştaki kaymaları tespit edip gösterecek yöntemler geliştirdik.

("Söz", "konuşmacı" ve "dinleyici" sözcükleri bire bir anlamlarıyla anlaşılmamalı. TDUS, konuşma dilini değil yazı dilini işliyordu. DDİ araştırmalarında çoğunlukla bu sözcükler, kullanılan iletişim aracı ne olursa olsun, cümlelere, cümle kuruculara ve cümle alıcılara atıfta bulunmak maksadıyla genel anlamda kullanılmıştır.)

Harvard Üniversitesi profesörü olarak bu konu ve bu konunun DDİ'deki rolü üzerine çalışmayı sürdürmüş olan Barbara J. Grosz'un (1948- ; Resim 18.3) başlıca ilgi alanı bu odaklanma meselesiydi. Bağlamlar, hedefler ve odaklanılan noktalarla başa çıkma mekanizmalarının yanı sıra TDUS, İngilizcenin pek çok sözdizimi yapısını tanıyabilen DIAGRAM[6] adlı bir dilbilgisi, süreçleri ve hedefleri gösterip uslamlayan araçlar ve diyalog ilerlerken farklı bilgi tiplerinin nasıl etkileşime girdiğini betimleyen bir çerçeve de içeriyordu.

Birkaç paragraf önce anlattığıma benzer şekilde CBS sisteminin gösterimi, J. C. R. Licklider [1973'te IPTO başkanlığına dönmüştü] için 23 Nisan 1975'te SRI'da düzenlendi. Ziyaretiyle ilgili izlenimleri hakkında Licklider ileride şunu söyleyecekti:[7]

5 *A.g.e.*, s. 11.

6 Jane J. Robinson, "DIAGRAM", SRI AI Center Technical Note no: 205, 1980; Şubat 1980 tarihli SRI AI Merkezi Teknik Notu olarak şu internet adresinde: http://www.ai.sri.com/pubs/files/712.pdf.

7 J. C. R. Licklider, "The Early Years: Founding IPTO", *Expert Systems and Artificial Intelligence: Applications and Management* içinde, Thomas C. Bartee (ed.), Indianapolis, IN: Howard W. Sams & Co., 1988, s. 223.



Resim 18.3. Barbara J. Grosz (Fotoğraf, fotoğrafçı Tony Rinaldo'nun izniyle.)

> DARPA'ya ikinci kez geldiğimde, donanım bakımıyla uğraşan çok etkileyici YZ sistemleri görmüştüm. SRI'nın, bir pompanın sökülüp nasıl yeniden toplanacağını tarif eden bir programı vardı. Bu, hayli karışık bir cihaz falan değildi, ama bir pompanın tüm aksamından ve bunların nasıl çalıştığından anlayan bir bilgisayar görmek çok etkileyici olmuştu.

Licklider'in aşıladığı cesaret sayesinde, CBC projesine devam etme konusunda iyimserdik ve askerî cip motorunu tamir etmekle ilgili teşhisler yapıp tavsiyeler verecek bir sistem tasarlamaya başlamıştık. Ne yazık ki, DARPA IPTO'nun yeni icra müdürlerinden Albay David Russell bu projeyi kabul etmedi. Licklider'in 23 Nisan tarihli ziyaretinden birkaç gün önce SRI'ya uğradıktan sonra Russell, Licklider'e gönderdiği e-postada şunları yazmıştı:[8]

> SRI programıyla ilgili kayda değer bir endişe olduğunu kabul etmeliyim, özellikle YZ programı üzerindeki idari baskılardan ötürü. Nils'in üzerinde çalıştığı programın öngörülen planına baktığımda, cip için deneysel bir CBC geliştirme hedefiyle 3 senede 2,2 milyon dolar harcanacağını görüyorum... Bana kalırsa bunu kısa vadeli bir uygulama olarak savunmak mümkün değil...
>
> Belki zor olacak ama senden istediğim şey, hava kompresörü aşaması tamamlandığında CBC programını sona erdirmeyi cidden düşünüp SRI'yı daha savunma ağırlıklı uygulamalara yönlendirmen ya da çalışmalarını Ulusal Bilim Vakfı'na (NSF) geçirmelerini sağlaman. Bunun ay-

8 Bu e-postanın bir nüshası dosyamdadır.

kırı bir düşünce olduğunun farkındayım ama bence durum bu.

Bu yorumları Nils'le yüz yüze konuşmuş değilim, gerçi program sona erdirilse ne yapacağını sordum. Programı doğru anlamayarak olumsuz bir görüş oluşturmuş olabilirim, ayrıca programla ilgili senin görüşlerini duymaksızın SRI ekibini üzmek istemiyorum.

O sene içinde Russell, DARPA IPTO'nun yöneticisi olarak Licklider'in yerine geçip CBC projesini sonlandırdı. (Ancak TDUS projesi NSF desteğiyle sürecekti.) Ardından, SRI ekibine verilen DARPA desteği, veritabanlarının doğal dil arayüzlerine (bunu ileride anlatıyorum) ve fotoğraf yorumcularına yardımcı olacak "görüntü anlamaya" kaydırıldı. Bunun yerine bazılarımız, bilgisayar tabanlı danışma sistemleri üzerinde çalışmak için DARPA harici destekler aradık. Stanford Üniversitesi'nde, uzman sistemler diye adlandırılan süregiden çalışmalar, bizi bu yöne girmeye yüreklendirmişti.

18.2 UZMAN SİSTEMLER

18.2.1 MYCIN

Stanford'un HEURISTIC DENDRAL projesi, bilgisayarları, kimyaya ve spektroskopiye dair uzmanlık bilgisiyle donatmanın gücünü göstermişti. Bu projenin kıdemli üyeleri Feigenbaum, Lederberg ve Buchanan, benzer bir yaklaşımın tıp alanında da işe yarayacağına inanıyordu. 1970'lerin başında Buchanan, Stanford Tıp Fakültesi Klinik Farmakoloji Başkanı Stanley Cohen'la, Cohen'in MEDIPHOR adlı bilgisayarlı ilaç etkileşimi uyarı sistemi hakkında konuşmaya başladı. Aynı dönemde, Stanford Tıp Fakültesi öğrencilerinden Edward (Ted) Shortliffe (1947- ; Resim 18.4), Stanford'da YZ'yle ilgili ders alıp Cohen'in projesinde asistan olmuştu. Shortliffe, Buchanan ve Cohen hep birlikte, doktorlara bakteri enfeksiyonları ve tedavileri hakkında danışmanlık yapacak bir bilgisayar programı yaratma fikrini oluşturdular. Shortliffe, programa MYCIN adını vermişti, zira bu, antibakteriyel etken maddeleri isimlendirirken kullanılan yaygın bir son ektir. Bu tür bir program, bulaşıcı hastalık uzmanlarının tanı ve tedavi bilgilerini içerecekti.

MYCIN'i geliştirirken karşılaşılan ilk soru, uzmanlık bilgisinin nasıl temsil edileceğiydi. Shortliffe ve Buchanan, DENDRAL'de kullanılan "EĞER-İSE kuralları"na benzer bir şeyin uygun olacağını düşündüler. Belirli hastalık belirtilerine hangi hastalığın yol açabileceğinin tanısını koyarken, ayrıca tedaviyi reçetelendirirken, hekimler bir tür EĞER-



Resim 18.4. Bruce Buchanan (solda) ve Ted Shortliffe (sağda). (Fotoğraf, Ed Feigenbaum'un izniyle.)

İSE usavurumundan faydalanır: EĞER belirtiler şunlar şunlarsa, İSE nedeni büyük olasılıkla bunlar bunlardır. Bu tür usavurumun ardındaki bilgi, hem vaka deneyimlerine hem de hastalıklar hakkında bilimsel bilgiye dayanır. Programın gereksinim duyduğu EĞER-İSE bilgisinin, zaten böyle düşünen uzmanlarla konuşularak elde edilebileceklerine inanmışlardı.

İlginçtir, tıbbi meselelere dair EĞER-İSE usavurumunun uzun bir geçmişi var. J. H. Breasted'in,[9] ilkçağdan kalma bir Mısır papirüsündeki cerrahi bilgiler üzerine yazdığı kitabı kısmen özetleyen Robert H. Wilkins şunu der: "Geçmişi MÖ 17. yüzyıla uzanan Edwin Smith Cerrahi Papirüsü, bilinen tüm tıp papirüsleri içinde en eskilerden biridir."[10] (Söz konusu papirüsü, 1882'de Luksor şehrinde bir antikacı dükkânından Edwin Smith satın almıştı.) Wilkins, papirüsten bazı kurallar sıralar, bunlardan biri de şudur:

9 J. H. Breasted, *The Edwin Smith Surgical Papyrus,* iki cilt, Chicago: University of Chicago Press, 1980.

10 Wilkins'in makalesinin bir kopyası için bkz. http://www.neurosurgery.org/cybermuseum/pre20th/epayrus.html.

Vaka Otuz

Başlık: Hastanın boyun omurlarından birinde burkulmayla ilgili talimatlar.

Muayene: Eğer boyun omurlarından birinde burkulma olan bir adamı muayene ediyorsan, ona şunu söylemelisin: "İki omuzuna ve göğüs kafesine bak". Bu hareketi yaptığında, bu bakış ona acı veriyor olmalı.

Tanı: Hastaya şunu demelisin: "Boyun omurunda burkulma var. Bu rahatsızlığı tedavi edeceğim."

Tedavi: İlk gün boyuna taze et bağla. Ardından, hasta iyileşene dek boynuna her gün bal sür.

Yeni oluşan bu tanı ve tedavi sisteminin geliştirilmesine iki uzman daha katılmıştı: Stanford Bulaşıcı Hastalıklar Bölümü Başkanı Thomas Merigan ve bu bölümün doktorlarından Stanton Axline. Bu projenin tarihçesiyle ilgili özetlerinde,[11] Buchanan ve Shortliffe, programa MYCIN adını bulma şerefinin Axline'a ait olduğunu söylerler.

Bu ekip, Ekim 1973'te Ulusal Sağlık Enstitülerine (NIH) başarıyla sonuçlanacak bir proje başvurusu yaptı. Shortliffe, tıbbi araştırmalarını, MYCIN'i temel alacak Bilgisayar Bilimi doktora çalışmalarıyla birleştirmeye karar vermişti. Kullanmak istediği LISP sürümü (BBN-LISP, daha sonra INTERLISP olacaktı) Stanford'da olmadığı için, SRI YZ ekibinin PDP-10 bilgisayarını kullanmıştır.

Tıp uzmanlarından sağlanan EĞER-İSE kuralları çoğunlukla belirsizliklerle çevriliydi. Buchanan ve Shortliffe şunu belirtir: "Cohen ve Axline, bir gözlem kümesinin, ilintili sonuç üzerindeki etkilerini betimlerken "akla getiriyor" ya da "haklı çıkarıyor" gibi sözcükler kullanıyordu. Kurallarımızda olasılıksal ifadeleri ele almamız gerekeceği anlaşılmıştı..."

Shortliffe, MYCIN'in EĞER-İSE kurallarına yeterlik kazandırmak amacıyla olasılık kullanımının çeşitli yollarıyla boğuştuktan sonra, bir parça doğaçlama bir kavram olan "kesinlik çarpanları"nı kullanmaya karar verdi.[12]

11 Bruce G. Buchanan ve Edward H. Shortliffe (ed.), *Rule-Based Expert Systems: The MYCIN Experments of the Stanford Heuristic Programming Project,* Reading, MA: Addison-Wesley, 1984. Bu kitap artık basılmıyor ve şu internet adresinde: http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics/RuleBasedExpertSystems. Shortliffe'ın tezi kitap olarak da basılmıştır. Edward H. Shortliffe, *Computer-Based Medical Consultations: MYCIN,* New York: Elsevier, 1976.

12 Olasılık kuramlarına katı şekilde bağlı olmayan fikirler kullanmaya kalkışan başkaları da olmuştur. Bunlar arasında Arthur Dempster ve Glenn Shafer (bkz. Glenn Shafer, *A Mathematical Theory of Evidence,* Princeton, NJ: Princeton University

Örneğin, aşağıda MYCIN kurallarından birini görüyoruz (hem içsel LISP dilinde hem de konuşma diline çevirisinde):

```
KURAL036
ÖNERME: ($VE  (EŞİT BAĞLAM GRAM GRAMNEG)
              (EŞİT BAĞLAM MORPH ROD)
              (EŞİT BAĞLAM AIR ANAEROBIC)
EYLEM: (SONUÇ BAĞLAM KİMLİK BACTEROIDES TALLY 0.6)
EĞER:  1) Organizmanın gram boyaması gram eksiyse, ve
       2) Organizmanın morfolojisi çubuk şekilliyse, ve
       3) Organizma oksijensiz solunum yapıyorsa
İSE: Organizmanın kimliğinin bakteroit olabileceğini gösteren makul
       bulgular (0,6) var demektir.
```

Bu kuraldaki 0,6 rakamının anlamı, uzmanın bu sonuca "inanç derecesi" ya da "kesinliği"dir. Shortliffe, inanç derecesinin, olasılık değerlendirmeyle aynı olmadığını düşünüyordu; çünkü başka unsurların yanı sıra, Kural 036'yı temin eden uzmanların illâ, o organizmanın bakteroit *olmama* olasılığının 0,4 olduğunu düşünmediklerini belirtir. Özgün MYCIN sisteminde bu tür 200 kural vardı. 1978'e gelindiğinde kural sayısı 500'e çıkmıştı.

MYCIN'in kurallarına çoğunlukla geriye dönük uslamlama tarzında başvuruluyordu. Örneğin, "EĞER x1 ve x2 mevcutsa İSE *y* söz konusudur" biçimindeki bir kural, eğer sistemin genel hedefi *y* sonucuna ulaşmaksa kullanılır. Bu kuralın kullanımı, "İSE" kısımları ya x1 ya da x2 olan kuralların kullanımına yol açacaktır. Bir kurallar zincirinin sonunda, sistemin doktor kullanıcısından (ya da veritabanından), "EĞER" kısmı için bilgi sağlaması istenir. Dolayısıyla, eğer MYCIN belli bir organizmanın kimliğinin bakteroit olduğunu belirlemeye çalışıyorsa, KURAL 036'dan faydalanılıp doktora (ya da veritabanına), acaba organizmanın gram boyaması gram eksi miydi, vesaire diye sorulur.[13]

Press, 1976) ve "bulanık mantığı" geliştirmiş olan Lotfi Zadeh (pek çok kaynağımdan biri olan şu makaleye bkz. Lotfi A Zadeh, "A Fuzzy-Algorithmic Approach to the Definition of Complex or Imprecise Concepts", *International Journal of Man-Machine Studies*, c. 8, 1976, s. 249-291, şu internet adresinde: http://www-bisc.cs.berkeley.edu/ZadehFA-1976.pdf.) vardı. Bu seçeneklerden kitabın ilerleyen bölümlerinde bahsediyorum.

13 MYCIN kurallarının nasıl elde edilip kullanıldığına dair tam bir açıklama için bkz. Bruce G. Buchanan ve Edward H. Shortliffe, *a.g.e.*

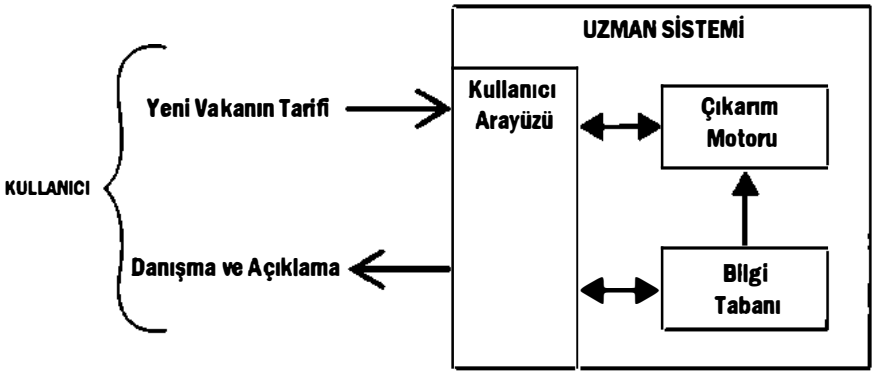
MYCIN, "bir "danışma sistemi" olarak biçimlendirilmişti; yani, belirli bir hasta hakkında bilgi sağlayan doktorla etkileşime giriyordu. Kuralların ve kural zincirlerinin kullanımı, sistemin kendi uslamlaması hakkında "açıklama" sunmasına olanak tanıyordu. Örneğin, kullanıcıya Kural 036'nın yönelttiği bir sorudan sonra, eğer kullanıcı, "Neden organizmanın morfolojisinin çubuk şekilli olup olmadığını sordun?" diye sorarsa, sistem (İngilizce olarak), "çünkü organizmanın kimliğinin bakteroit mi olduğunu belirlemeye çalışıyorum" gibi bir yanıt verir.

Peki, MYCIN, birincil görevi olan tedavi önermeyi nasıl yapıyor? Shortliffe ve çalışma arkadaşları, doktorlara, bazı vakalarda MYCIN'in önerdiği tedavilerle kendi tedavilerini karşılaştırmalarını istedikleri değerlendirme formları dağıtmıştı. Ana sonuçları şu oldu: "Değerlendirmeye katılanların çoğunluğu, MYCIN'in tedavilerinin %70'ini kabul edilebilir buldu". Bu arada, şunu da belirtmişlerdi: "Aslında tedavilerin %75'i, aynı ölçütlerle değerlendirilen Stanford akademisyenlerinin genelde ulaştığı mutabakat derecesinden daha iyiydi."[14]

MYCIN'in getirdiği yeniliklerden biri (örneğin DENDRAL'in aksine), kullandığı uslamlama sürecinin (kuralları kullanmanın), kendi tıbbi bilgisinden (kuralların kendisinden) oldukça ayrı oluşuydu. Dolayısıyla, programı iki parçaya ayırmak yaygın uygulama haline gelmiştir: kuralları uygulayan "çıkarsama motoru" ve kurallardan oluşan "bilgi tabanı". Prensipte, çıkarsama motoru değiştirilmeksizin yeni kurallar eklenebilir. Bu ayrım, Resim 18.5'te gösteriliyor. Söz konusu ayrım, çıkarsama motorunu değiştirmek zorunda kalmaksızın, sırf tıbbi bilginin yerine başka bir bilgi tabanı yerleştirerek, başka uygulamalar için uzman sistemler kurulabileceğini akla getiriyor. William van Melle, tam da bu işi yapan EMYCIN ("E", "empty" [boş] anlamında) adlı bir sistem geliştirmiştir.[15] Belirli bir alandaki uzmanlarla birlikte bir sistem tasarımcısı, tam da o alan için EĞER-İSE kuralları yaratmak maksadıyla EMYCIN'le etkileşimli biçimde çalışabilir. EMYCIN, bünyesindeki çıkarsama moturundan faydalanarak, danışma sırasında sistem kullanıcısına tavsiyeler vermek üzere bu kuralları kullanabilir. Nitekim EMYCIN, vergi planlama ve mekanik yapı çözümlemesi gibi çeşitli alanlar için farklı uzman sistemler oluşturmak üzere kullanılmıştır.

14 Bruce G. Buchanan ve Edward H. Shortliffe, *a.g.e.*, 30. ve 31. Bölümler.

15 EMYCIN şurada betimleniyor: Bruce G. Buchanan ve Edward H. Shortliffe, *a.g.e.*, 15. Bölüm. EMYCIN, van Melle'nin doktora tezinin konusuydu: William van Melle, "A Domain-Independent System That Aids in Constructing Knowledge-Based Consultation Programs", Stanford Üniversitesi Bilgisayar Bilimi Bölümü. Ayrıca bkz. Stanford Raporları no: STAN-CS-80-820 ve HPP-80-22, 1980.



Resim 18.5. MYCIN tarzı uzman sisteminin yapısı.

Araştırmacılar, MYCIN ile EMYCIN'in yararlandığı eminlik çarpanlarının ufak bir çeşitlemesinin, bunun yerine olasılık kulanmaya denk olduğunu kısa süre içinde keşfetmiştir. Olasılık kuramıyla olan bu bağlantı, ne MYCIN'in ne de EMYCIN'in kaçınabileceği sonuçları gündeme getirir. Özellikle, bu sistemlerin uslamlaması, olasılık kuramıyla sadece, kuralların nasıl kullanılacağı hakkında görece kısıtlı kimi varsayımlar altındayken tutarlılık sergiler. Russell ile Norvig'in belirttiği gibi, eğer bu varsayımlar karşılanmıyorsa, "eminlik çarpanlarının, bulguların aşırı hesaba katılması yüzünden müthiş yanlış bir inanış derecesi doğurmuştur. Kural kümesi büyüdükçe, kurallar arasında istenmeyen etkileşimler yaygınlaşmıştır ve uygulamacılar, daha fazla kural eklenirse başka kuralların eminlik çarpanlarının "ayarlanması" gerektiğini görmüştür."[16] İleriki bir bölümde göreceğimiz üzere, günümüzün yöntemleri daha gelişmiş olasılık teknikleri kullanmaktadır.

Yine de MYCIN'in ve çeşitli EMYCIN programlarının başarısı, çok daha fazla uzman sisteminin geliştirilmesine yol açmıştır; bu sistemlerden bazıları EMYCIN'i temel almışken bazıları da kendi özgül yaklaşımlarını benimsemişti. Allen Bewell'ın, Buchanan ve Shortliffe imzalı kitap için yazdığı Sunuş bölümünde belirttiği gibi: "MYCIN, bütün dünyaya yeni bir alanın açıldığını gösteren özgün bir uzman sistemidir... MYCIN, yaratılan bu yeni yolun ete kemiğe bürünmüş halidir. Dolayısıyla, bu sistemin tam kaydını ve gelişiminin içsel tarihini bir araya toplamak, YZ tarihçesinde önemli bir olayı kaydetmeye hizmet eder."[17]

16 Stuart Russell ve Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2. baskı, Upper Saddle River, NJ: Prentice Hall, 2003, s. 525.

17 Bruce G. Buchanan ve Edward H. Shortliffe, *a.g.e.*

18.2.2 PROSPECTOR

SRI'da Shortliffe'nin MYCIN çalışmasından esinlenmiş olan kimilerimiz, uzman sistemlerin tıp dışı uygulamalarını incelemeye başlamıştı. Bu alanlardan biri, "bütünleşik zararlı idaresi"ydi; bu alanda, tahıllar ve bunların zararlılarıyla ilgili bilgi, zararlı saldırısının etkilerini en az kimyasal kullanımıyla gidermeye çalışmak için kullanılabilirdi. Her ne kadar birtakım teklifler kaleme alınmış ve ABD Ziraat Bakanlığı'ndaki ve Çevre Koruma Dairesi'ndeki bilimciler bu konuya ilgi göstermiş olsa da, teklifler ödenek bulamayınca bu fikir terk edilmiştir.

Sonunda Peter Hart ve Richard Duda, olası "sert kayaç" maden yatakları hakkında sondajcılara tavsiye veren sistemlere odaklanmışlardı.[18] Hart, Stanford'un petrol mühendisliği profesörlerinden John Harbaugh ve Harbaugh'nun lisansüstü öğrencilerinden Alan Campbell'la daha önce konuşmuştu. (Alan Campbell, muhtemelen gezegenin en büyük çinko-kurşun yatağını keşfetmiş olan dünyaca ünlü merhum Neal Campbell'in oğluydu. Alan, gençliğinin büyük kısmını maden kamplarında geçirmişti.) Hart ve Duda, Campbell'in aracılık etmesi sayesinde, Stanford Yerbilim Fakültesi'nin eski dekanı ve sert kayaç maden yatakları uzmanı Charles Park'la tanıştı. Park, kurşun-çinko yatakları hakkındaki bilgiyi EĞER-İSE kuralları biçiminde kodlamada Hart ile Duda'ya yardımcı oldu. Stanford İktisadi Jeoloji Bölümü profesörlerinden Marco Einaudi'yle birlikte yapılan ek çalışmalar, ek kurallar ve kuralları örgütlemeye dair fikirler doğurdu. Sonunda ABD Jeolojik Tetkik Arama, maden yataklarına yönelik uzman sistemi PROSPECTOR'u ortaya çıkaracak projeye mali destek sağladı.[19]

PROSPECTOR programının yazılmasına ve tasarlanmasına epey insan katılmıştır. Bu uğraşın önderleri Duda ve Hart'tı. Bu projeye, çalışmalar bir parça ilerledikten sonra, CBC projesinin sürmeyeceğini DARPA'dan duyduğumda katıldım. Projeye katkı verenler arasında John Gaschnig (1950-1982), Kurt Konolige, René Reboh, John Reiter, Tore Risch ve Georgia Sutherland de vardı. MYCIN'in, geliştirilen teknoloji üzerindeki etkisi baskındı; "bunun başlıca sebebi, karara yol açacak bilgiyi gösterirken kurallardan faydalanması ve belirsizliklerle

18 İktisadi jeologlar maden yatakları ile maden cevherleri arasındaki ayrım yapar. *Cevher, kâr* getirecek biçimde çıkarılan madendir. Sert kaya madenleri arasında bakır, kurşun, çinko, vesaire vardır ama hidrokarbonlar yoktur.

19 PROSPECTOR ilk olarak şurada betimlenmiştir: Peter E. Hart, "Progress on a Computer-Based Consultant", *Proceedings of the International Joint Conference on Artificial Intelligence,* c. 2, 1975, s. 831-841.

başa çıkan örgün mekanizmalar barındırmasıydı."[20] Başka bir tıbbi tanı sistemi olan INTERNIST-1'in de önemli etkileri olmuştur; bu sistemi biraz sonra kısaca anlatacağım. Bu etkiler, taksonomi bilgisi kullanımıyla ve (sadece sorulan değil) gönüllü verilmiş bilgiyle başa çıkma kabiliyetiydi.

PROSPECTOR, çıkarımlar yapmak ve danışma sürecini yönlendirmek için kurallardan faydalanıyordu. Bu kurallara iki örnek şöyle:

> Kural 3: "Sülfitin üzerini kaplayan barit katmanı, olası bir devasa sülfit yatağına işarettir."

ve

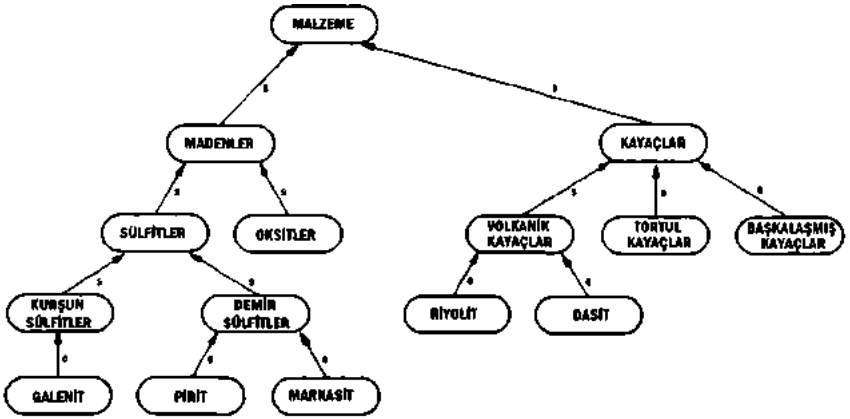
> Kural 22: "Billur şekilli kaya oyukları, sülfitin varlığına işarettir."

Kurallar, "bölmeli anlamsal ağlar" olarak kodlanmıştı; bu format ilk kez, doğal dil işleme sistemlerinin gereksinim duyduğu bilginin gösterimi amacıyla Gary Hendrix'in (1948-) Texas Üniversitesi'ndeki doktora tezinde kullanılmıştır.[21] Anlamsal ağlar aynı zamanda, PROSPECTOR'un yararlandığı taksonomi bilgisini gösterirken de kullanılıyordu. Bu tür bir ağın örneği Resim 18.6'da gösteriliyor. Sistemin kuralları, "çıkarsama ağı" denen yapıyla birbirine bağlanabilir. Kuroko tipi devasa bir sülfit yatağı hakkında mantık yürütmenin sadeleştirilmiş bir örneği Resim 18.7'de gösteriliyor. Kural 3'ün önermelerinden birinin belirlenmesinde Kural 22'nin nasıl yardımcı olduğuna dikkat edin. Ayrıca, galenit, sfalerit ya da kalkopirit olduğu bilindiğinde sülfitin var olduğu çıkarımını yapmak üzere taksonomiden faydalanıldığına da dikkat edin.

Ağda kural önermelerinden yapılan kural sonuç çıkarımları, olasılığa ve Bayes kuralına bağlıdır; "kesinlik çarpanı" gibi doğaçlama rakamlara değil. Jeoloji uzmanlarından, bir kural hakkındaki şüphelerini, tasarımcılara iki rakam vererek nicellemeleri istenmişti. Bu rakamlardan bir tanesi, önermeler doğru olduğu takdirde o sonucun

20 Richard O. Duda ve diğ., "Semantic Network Representations in Rule-Based Inference Systems", *Pattern-Directed Inference Systems* içinde, D. A. Waterman ve Frederick Hayes-Roth (ed.), Orlando, FL: Academic Press, Inc., 1978. Şu internet adresinde: http://www.ai.sri.com/pubs/files/751.pdf.

21 Gary G Hendrix, "Partitioned Networks for the Mathematical Modeling of Natural Language Semantics", doktora tezi, Texas Üniversitesi Bilgisayar Bilimi Bölümü, 1975. Kısa bir makale için bkz. Gary G. Hendrix, "Expanding the Utility of Semantic Networks Through Partitioning", *Proceedings of the Fourth International Conference on Artificial Intelligence*, 1975, s. 115-121. SRI YZ Merkezi Teknik Notu 105 olarak bu makale şu internet adresinde: http://www.ai.sn.com/pubs/files/1380.pdf.



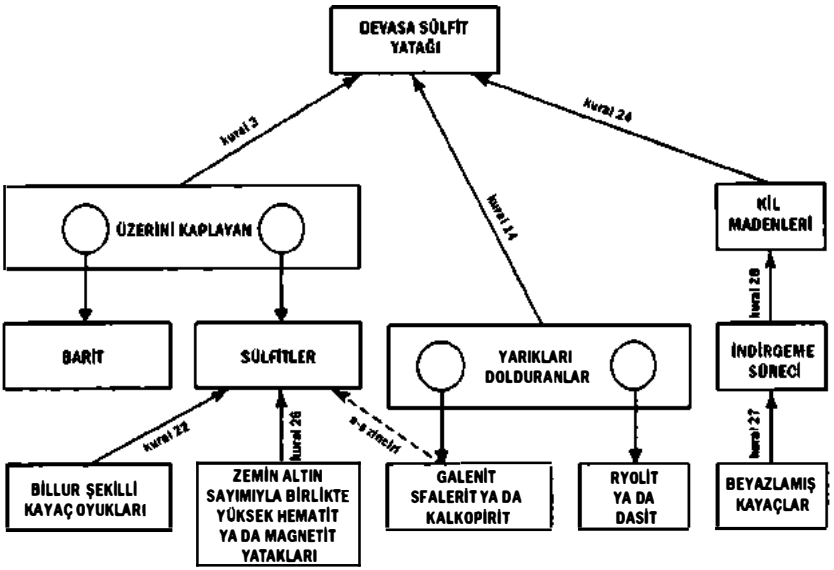
Resim 18.6. Kısmi bir jeoloji taksonomisi. (SRI International'ın izniyle kullanılmıştır.)

olasılığını yükselten etkendi. Öteki rakamsa, önermeler yanlış olduğu takdirde, o sonucun olasılığını düşüren etkendi. Önermelerin olasılığı uyarınca sonucun olasılığını belirlemek için, bu rakamlarla birlikte Bayes kuralı kullanılıyordu.[22] PROSPECTOR'un çıkarsama yöntemleri her ne kadar MYCIN'in yöntemlerine kıyasla gelişmiş olsa da, sadece belirli çıkarım-ağ türleri için geçerli olasılıksal sonuçlar veriyordu. Glenn Shafer ile Judea Pearl'ün açıkladıkları gibi: "Olasılıklar, EĞER-İSE kurallarına iliştirilmiş rakamlar gibi bir şeylerin peşine takılamaz. Olasılık hesaplarının sonuçları ancak, eğer bu hesaplamalar olasılık kuramı ilkelerine uymuşsa anlamlıdır."[23] Günümüzün uzman sistemleri, Bayes ağlarının daha genel çerçevesini kullanıyor; bu sistemler daha sonra anlatılacak.

Bir PROSPECTOR danışma oturumunda olağan biçemi, belirli bir konumu değerlendirmekle ilgilenen bir jeologla iletişim kuruluyordu. Jeolog, bazı bilgileri programa gönüllü sunabilir, bunlar da PROSPECTOR

22 PROSPECTOR'un çıkarım yöntemlerinin betimlenişi için bkz. Richard O. Duda, Peter E. Hart ve Nils J. Nilsson, "Subjective Bayesian Methods for Rule-Based Inference Systems", *Proceedings of the AFIPS National Computer Conference* içinde, c. 45, 1976, s. 1075-1082. Yeniden basıldığı yer. G. Shafer ve J. Pearl (ed.), *Readings in Uncertain Reasoning,* San Francisco: Morgan Kaufmann Publishers, 1990, s. 274-281. SRI YZ Merkezi Teknik Notu 124 olarak bir versiyonu şu internet adresinde: http://www.ai.sri.com/pubs/files/755.pdf.

23 Glenn Shafer ve Judea Pearl (ed.), *Readings in Uncertain Reasoning,* San Francisco: Morgan Kaufmann Publishers, 1990. Bu kitap artık basılmıyor ama bazı bölümleri şu internet adresinde: http://www.glennshafer.com/books/rur.html.



Resim 18.7. PROSPECTOR çıkarım ağının sadeleştirilmiş versiyonu. (SRI International'ın izniyle kullanılmıştır.)

kurallarından bazılarını devreye sokabilir. Sonra sistem, artık jeolog ne bulmak istiyorsa onun olasılığını değiştirmede en etkin olacak ek bilgileri hesaplar. Böylece PROSPECTOR, o bilgiyi (ve olasılığını) ortaya koyacak bir soru sorar. Bu süreç boyunca kullanıcı, sorulmadan da her an bilgi ekleyebilir.

PROSPECTOR, sorulmadan verilen bilgilerden de faydalanabildiği için, bu programın seyri, bir soru-yanıt danışma oturumunun parçası olmak zorunda değildir. Bunun yerine kullanıcı, "sahadaki bulgularla" ilgili verileri PROSPECTOR'a girebilir, program da böylece kendi sonuçlarını çıkarır. Bu bulgular bir veritabanından ya da daha iyisi, farklı maden türlerinin var olduğunun bilindiği bölgeleri gösteren yükselti haritalarından gelebilir. (Yaklaşık bu zamanlarda SRI'dan Kurt Konolige, PROSPECTOR ekibine katılıp PROSPECTOR'un harita verilerini girdi olarak kullanmasına olanak tanıyan bir program yazmıştı.)

PROSPECTOR'un harita verisi kullanımıyla ilgili en dramatik an, Washington eyaletindeki Tolman dağında porfiri molibden yatağının yerini başarıyla tespit etmesiydi.[24] Tolman dağında daha önce yapıl-

24 Alan N. Campbell, Victor F. Hollister, Richard O. Duda ve Peter E. Hart, "Recognition of a Hidden Mineral Deposit by an Artificial Intelligence Program", *Science*, c. 217,

mış keşiflerin sonuçları, olası molibden yataklarıyla ilgili önemli jeolojik verileri sıralayan haritalar oluşturmada kullanılmıştı. PROSPECTOR, öncelikle porfiri molibden yatakları uzmanı Victor F. Hollister ve Alan Campbell'den elde edilen kurallardan yararlanarak haritaları işledi. Bu işlemin sonucunda sistem, maden yataklarının göreli "elverişliliğini" gösteren başka bir harita çizdi. Girdi haritalarından bazılarının bilgisayar görüntüleri Resim 18.8'de gösteriliyor. Bu haritaların betimlediği şeyin coğrafi ayrıntılarını açıklayacak değilim ama Campbell ve Hollister gibi uzmanların önemli saydığı veri türlerini temsil ettiklerini söyleyebilirim.

PROSPECTOR, bu tür verilerden yola çıkarak, elverişlilik haritaları çiziyordu; bunlardan biri Resim 18.9'da gösteriliyor. Haritanın sağ kenarındaki ölçek (renklendirildiğinde), +5'ten (çok elverişli) -5'e (epey elverişsiz) kadar elverişliliği gösteriyor. Elverişli alanların en büyüğünde gerçekleştirilmiş kapsamlı kazı çalışmalarını temel alan bir maden şirketi, burada açık-kuyu tarzı bir maden kurmayı planlamıştı ("önerilen kuyu" etiketiyle gösterilen yüzey).

Bu sonuç değerlendirilirken dikkatli olunmalı. PROSPECTOR, daha önce sondaj yapılmamış bir yerde cevher yatağı bulmuş değildi. *Artificial Intelligence* dergisinin yazı işleri müdürüne yazdığımız mektupta belirttiğimiz gibi,[25]

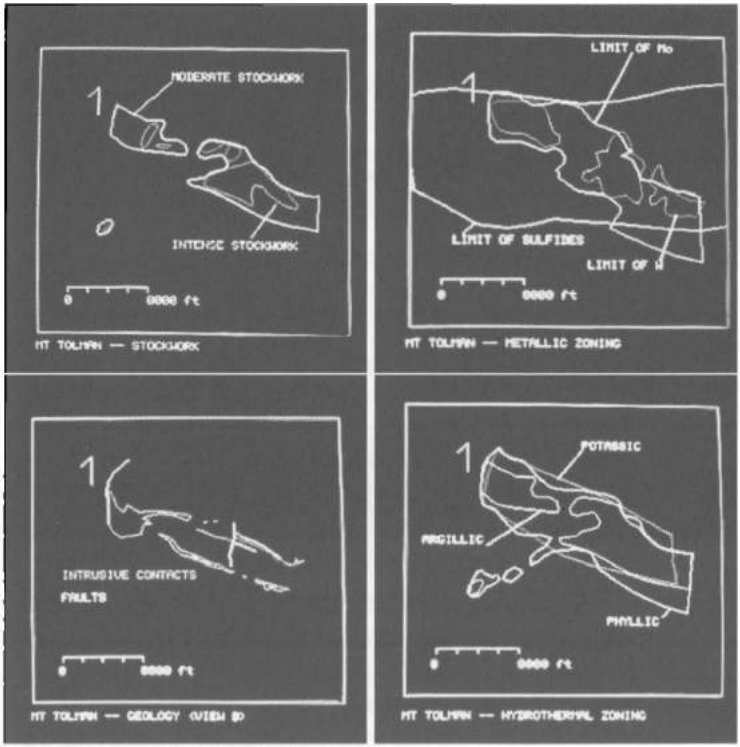
> Büyük bir maden şirketi, tek bölgede 200'ü aşkın sondaj kuyusu kazarak bir molibden cevheri zaten bulmuştu... ayrıca bilgilenmek için daha fazla kazı yapmaya niyetlendiklerini biliyorduk.
>
> ...
>
> [Bu ek kazılar], PROSPECTOR'un elverişlilik haritasıyla kayda değer uyum göstermişti; bu uyum, hem PROSPECTOR'un büyük ve önceden bilinmeyen bir cevher madeni bölgesi hakkında tahminini hem de PROSPECTOR'un boş arazilere dair tahminini doğruluyor.
>
> ...
>
> Ne yazık ki, maden sanayisindeki uzun süren iktisadi buhran koşulları bu bölgeyi madencilik için kârsız kılıyor... Dolayısıyla, PROSPECTOR'un başarısı şimdilik iktisadi değil bilimseldir.

Daha fazla ayrıntıyla ilgilenen okurlar, yukarıda atıfta bulunulan *Science* makalesine ve PROSPECTOR projesi sonuç raporuna bakabilir.[26]

no: 4563, 3 Eylül 1982, s. 927-929.

25 Richard O. Duda, Peter E. Hart ve Rene Reboh, yazı işleri müdürüne mektup, *Artificial Intelligence*, c. 26, 1985, s. 359-360.

26 Richard O. Duda, "The PROSPECTOR System for Mineral Exploration", Final Report



Resim 18.8. Tolma Dağı girdi haritalarından bazıları. (Fotoğraflar, Richard Duda'nın izniyle.)

PROSPECTOR'un bilgisayar kodu, ABD Jeolojik Tetkik Arama'ya verildi ve burada Richard B. McCammon, bu programın ardılı PROSPECTOR II'yi geliştirdi. McCammon, sistemini özetlerken şöyle yazar:[27]

> PROSPECTOR'un ardılı PROSPECTOR II, ABD Jeolojik Tetkik Arama'da geliştirildi. Şimdilik, bilgi tabanı 86 maden yatağı modeli ve 140'ı aşkın maden yatağı hakkında bilgi içeriyor. Sistemi kullanan jeolog, dakikalar içinde belli bir bölge için gözlemlediği verileri girebilir, değerlendirmeye

prepared for the Office of Resource Analysis, U.S. Geological Survey, Reston, VA 22090, Nisan 1980.

27 Richard B. McCammon, "PROSPECTOR II–An Expert System for Mineral Deposit Models", *International Journal of Rock Mechanics and Mining Sciences and Geomechanics Abstracts*, c. 33, no: 6, Eylül 1996, s. 267A-267A(1). Ayrıca bkz. Richard B McCammon, "PROSPECTOR II", *Proceedings of the Annual AI Systems in Government Conference* içinde, H. J. Antonisse, J. W. Benoit ve B. G. Silverman (ed.), Mart 1989, Washington, DC, s. 88-92.



Resim 18.9. Elverişlilik haritası. (Fotoğraf, Richard Duda'nın izniyle.)

alınacak yatak modellerini seçebilir, gözlemlenen verilere en fazla uyan modeller hakkında tavsiye alabilir ve belirli bir model söz konusu olduğunda, verilerden hangisinin açıklanabileceğini, hangi verilerin açıklanamayacağını ve modelin hangi kritik özelliklerinin verilerde gözlemlenmemiş olduğunu öğrenebilir.

18.2.3 Başka Uzman Sistemler

MYCIN ve PROSPECTOR çalışmalarını başka uzman sistemler izlemiştir. Bunların bazıları, tıpkı MYCIN gibi, tıbbi tanı ve tedavi amaçlıydı.[28] Bunlar arasından, bilgisayar bilimcileri Randolph A. Miller ile Harry Pople ve Pittsburgh Üniversitesi'nden doktor Jack D. Myers'ın birlikte yarattığı INTERNIST-1 programını ve Rutgers Üniversitesi'nden Casimir A. Kulikowski ile Sholom M. Weiss'ın yarattığı CASNET'i (Causal-ASsociational NETwork) anacağım.

INTERNIST-1 tanı programları dizisi, dahiliye uzmanlığını içeriyordu.[29] Bu bilginin bir kısmı, bir tür anlamsal ağda ya da hastalık durumları taksonomisinde (tıpta *nozoloji* denir) temsil ediliyordu. *New England Journal of Medicine* dergisinde çıkmış bir makalede Miller, Pople ve Myers, "bilimsel dergilerde yayımlanmış 19 klinopatolojik alıştırmada (Massachusetts Gene Hastanesinin Vaka Sicili) INTERNIST-1'in performansının, nitelik bakımından hastane klinik uzmanların-

28 Bkz. Peter Szolovits (ed.), *Artificial Intelligence in Medicine,* Boulder, CO: Westview Press 1982. Bkz. http://groups.csail.mit.edu/medg/ftp/psz/AIM82/ch0.html.

29 Harry E. Pople Jr., "Heuristic Methods for Imposing Structure on Ill-Structured Problems: The Structuring of Medical Diagnostics", *Artificial Intelligence in Medicine* içinde, 5. Bölüm, Peter Szolovits (ed.), Boulder, CO: Westview Press, 1982. Şu internet adresinde: http://groups.csail.mit.edu/medg/ftp/psz/AIM82/ch5.html.

kine benzediğini ama vaka uzmanlarınınkinden aşağı olduğunu" ifade ettiler. Bununla birlikte, "programın var olan hali [henüz] klinik uygulamalar için yeterli değil" sonucuna varmışlardı.[30] İleride, INTERNIST-1'de toplanan tanı bilgisinin büyük kısmı, internist sistemleri için tanı kararı destek sistemi olan QMR'de (Quick Medical Reference) istiflenecekti.[31] (Bunu satın alan First Databank, o zamandan sonra bu işi sürdürmemiştir.)

CASNET de ağlardan faydalanıyordu.[32] Bu ağlardaki "çıkarsama kuralları"; gözlemleri, patoloji-fizyoloji durumlarını, tanı durumlarını ve tedavi durumlarını birbirine bağlıyordu. Bunların öncelikli uygulama alanı glokom hastalığıydı; bu hastalık için, çıkarsama kurallarının temel alabileceği sağlam fiziksel modelleri vardı.

Carnegie Mellon Üniversitesi'nde John McDermott (1942-), Digital Equipment isimli şirketin VAX bilgisayar sistemlerinin sipariş ve biçimlenişine yardımcı olan XCON (eXpert CONfigurer) adlı kural tabanlı bir sistemin geliştirilmesine katkıda bulunmuştu. XCON, McDermott'un R1 adlı eski bir sisteminden baş vermişti.[33] R1 ve XCON, OPS5 adlı özel bir kural işleme dilinde yazılmıştı; bu dil, CMU'da Charles Forgy'nin (1949-) geliştirdiği OPS dil ailesine mensuptur.[34] (OPS'nin, *Official Production System*'in kısaltması olduğu söylenir.) OPS dilleri, EĞER-İSE kurallarını etkin biçimde birbirlerine bağlamak için Forgy'nin "Rete" algoritmasından faydalanıyordu.[35] XCON, kullanıma ilk kez 1980'de New Hampshire Salem'deki DEC tesisinde girdi.[36]

30 Randolph A. Miller ve diğ., "INTERNIST-1: An Experimental Computer-Based Diagnostic Consultant for General Internal Medicine", *New England Journal of Medicine*, c. 307, 19 Ağustos 1982, s. 468-76.

31 Randolph A. Miller ve diğ., "The INTERNIST-1/Quick Medical Reference Project-Status Report", *The Western Journal of Medicine*, c. 145, no: 6, 1986, s. 816-822. Şu internet adresinde: http://www.pubmedcentral.nih.gov/picrender.fcgi?artid=1307155&blobtype=pdf.

32 Casimir A. Kulikowski ve Sholom M. Weiss, "Representation of Expert Knowledge for Consultation: The CASNET and EXPERT Projects", *Artificial Intelligence in Medicine* içinde, 2. Bölüm, P. Szolovits (ed.), Boulder, CO: Westview Press, 1982. Şu internet adresinde: http://groups.csail.mit.edu/medg/ftp/psz/AIM82/ch2.html.

33 John McDermott, "R1: A Rule-Based Configurer of Computer Systems", *Artificial Intelligence*, c. 19, no: 1, 1980, s. 39-88.

34 Charles Forgy, "OPS5 User's Manual", Teknik Rapor CMU-CS-81-135, Carnegie Mellon Üniversitesi, 1981. Ayrıca bkz. Lee Brownston ve diğ., *Programming Expert Systems in OPS5*, Reading, MA: Addison-Wesley, 1985.

35 Charles Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", *Artificial Intelligence*, c. 19, 1982, s. 17-37.

36 Bkz. http://en.wikipedia.org/wiki/Xcon.

Kesin olmayan bilgileri işleme sorunundan XCON'da uzak durulmuştu, çünkü yeterli bilgi olmaksızın çözmesi gereken yapısal bir sorunla neredeyse hiç karşılaşılmıyordu. XCON'la ve DEC'deki benzer yapılandırma sistemleriyle ilgili bir makaleye göre,[37] 1989'a gelindiğinde bu sistemlerin toplam 17.500 kuralı vardı. Makale şunu da söylüyordu:

> ... Digital şirketine toplam getirisinin, yılda 40 milyon doları aştığı tahmin ediliyor.
>
> Biçimlendirme sistemlerinin kullanımı, müşteriye tam ve tutarlı yapılandırılmış sistemler yollanmasını sağlıyor. Eksik siparişler bu süreçten çıkamaz. Buna ek olarak, XCON, sistem performansını ideal hale getiren biçimlendirme de yaratıyor, böylece müşteriler, ürünlerimize dair hep en iyi görüşü ediniyor. Biçimlendirme sistemlerinin ortaya çıkışından önce, aynı aksamın farklı kardeşlerine ek olarak biçimlerini gönderirdik sık sık.

XCON'a ve bunun DEC'deki yavrularına ek olarak 1980'ler boyunca başka uzman sistemler de oluşturulmuş ve şirketler ile araştırma laboratuvarları tarafından kullanıma sokulmuştur. General Electric 1983 yılında DELTA'yı (Diesel Electric Locomotive Troubleshooting Aid) geliştirdi; bu prototip sistemin amacı, General Electric'in dizel-elektrikli lokomotiflerinin bakımında demiryolu çalışanlarına yardımcı olmaktı. Programın yaratıcıları, "lokomotifteki pek çok sorunu teşhis edebilir ve bakım elemanlarına onarım önerilerinde bulunabilir", diyordu. Bu sistem, "Kıdemli Saha Hizmeti Mühendisinin bilgisini kısmen temsil eden" 530 kural barındırıyordu.[38]

Kanada Ulusal Araştırma Konseyi mühendislerinin geliştirdiği JETA (Jet Engine Troubleshooting Assistant) ise başka bir örnekti. JETA hakkında yazılmış bir makaleye göre, bu sistem, "Kanada Hava Kuvvetlerinde kullanılan CF-5 eğitim avcı uçaklarındaki General Electric J85-CAN-15 jet motorunun arızalarını tespit etmek üzere geliştirilmiştir."[39] Jet motorları ve bunların olası arızaları ile belirtile-

37 Virginia E. Barker ve Dennis E. O'Connor, "Expert Systems for Configuration at Digital: XCON and Beyond", *Communications of the ACM*, c. 32, no: 3, Mart 1989, s. 298-318.

38 Piero P. Bonissone ve H. E. Johnson Jr., "DELTA: An Expert System for Diesel Electric Locomotive Repair", *Proceedings of the Joint Services Workshop on Artificial Intelligence in Maintenance*", Boulder, CO, Ekim 4-6, 1983, AD-A145349, Haziran 1984, s. 397-413 (Defense Technical Information Center Accession Number ADA145349).

39 Phillippe L. Davidson ve diğ., "Intelligent Troubleshooting of Complex Machinery", *Proceedings of the Third International Conference on Industrial Engineering Applications of Artificial Intelligence Expert Systems*, s. 16-22, Charleston, South Carolina, USA, 16-18 Temmuz 1990. Ayrıca bkz. M. Halasz ve diğ., "JETA: A Knowledge-Based Approach

ri hakkındaki bilgi, çerçeveler olarak kodlanmıştır. Kurallar yalnızca "çerçeve içine yerleştirilmiş özgül denetim işlevleri ve eşzamansız kullanıcı girdisi" için kullanılır.

Kredi analizine yarayan CCH-ES adlı uzman sistemi, Temmuz 1989'da Dun & Bradstreet'in (D&B) Kredi Takas (CCH) bölümünde kullanıma sokuldu. Bu sistem yaklaşık 800 kural içeriyordu; CCH müşterileri hizmet almak için aradığında ya da analistler dosyaları gözden geçirmek istediğinde çevrimiçi işlemler için kullanılıyordu. İlişkili veritabanlarında güncelleme yapıldığında toplu dosyalar işleme konuyordu. Bu sistem hakkında yazılmış bir makaleye göre, "Analistler CCH-ES sistemiyle yaklaşık %98,5 uyuşuyor... Bu sistem, D&B'de büyük başarı kaydetmiştir. CCH'ye, uzman düzeyinde tutarlı ve üst sınıf kredi analizi kararları veren otomatik bir kredi analisti uzman sistem kazandırmış oldu. Müşteriler bu sistemi birörnek övüyor."[40]

The Rise of the Expert Company başlıklı kitapta başka uzman sistemler de betimlenmiştir.[41] Bu kitabın Ek bölümünde Paul Harmon, 1980'lerin ortasından sonuna dek kullanıma girmiş 130'u aşkın uzman sistem sıralar; bunlar arasında şunlar vardır:

- *Grain Marketing Advisor*, tahıl mahsullerini pazarlama ya da depolama yolları bakımından çiftçilere yardımcı oluyordu.
- *ACE*, telefon operatörü şirketlere, telefon kablosu arızalarını gidermede yardım ediyordu.
- *IDEA*, tamircilere, Infotron IS4000 Yerel Alan Şebekesindeki arızaları tespit etmede yardımcı oluyordu.
- *Diag 8100*, Travelers Şirketinde IBM 8100 bilgisayarlarının sorunlarını ve kusurlarını teşhis etmede yardımcı oluyordu.
- *Intelligent Peripheral Troubleshooter*, Hewlett-Packard disk sürücülerinin arızalarını gidermede yardım ediyordu.
- *SNAP*, Infomart (Dallas'ta bir bilgisayar mağazası) müşterilerine, kişisel bilgisayar gereksinimlerini belirlemede yardım ediyordu.

to Aircraft Gas Turbine Engine Maintenance", *Journal of Applied Intelligence*, c. 2, 1992, s. 25-46.

40 Roger Jambor ve diğ., "The Credit Clearing House Expert System", *IAAI-91 Proceedings*, 1991, s. 255-269.

41 Edward Feigenbaum, Pamela McCorduck ve H. Penny Nii, *The Rise of the Expert Company: How Visionary Companies Are Using Artificial Intelligence to Achieve Higher Productivity and Profits*, New York: Times Books, 1988.

- *Pile Selection*, Kajima Construction Şirketinde tasarımcılara, binaların temeline çakılacak kazık malzemesinin seçiminde yardımcı oluyordu.
- *Exper TAX*, Coopers and Lybrand şirketinin müşterilerine, ABD'nin yeni vergi yasalarının uygulanışını gözden geçirmede yardımcı oluyordu
- *Dipmeter Advisor*, petrol kuyusu açarken karşılaşılan jeolojik oluşumların çözümlenmesine yardım ediyordu.

18.2.4 Uzman Şirketler

Bu uygulamaları geliştirip sahaya sürmek amacıyla yeni şirketler kuruldu ya da oturmuş şirketlerde yeni bölümler açıldı. Bunların ilki, Teknowledge'tı. Stanford akademisyen ve araştırmacılarından oluşan bir ekip, uzman sistemler pazarlayıp bu sistemler hakkında danışmanlık hizmeti verecek bu şirketi kurmuştu. Teknowledge, temel teknoloji olarak EMYCIN'den faydalanıyordu. Başka bir şirketse, sigorta taahhütleri ve kredi analizleri için uzman sistemler pazarlamak üzere Peter Hart ile Richard Duda'nın (PROSPECTOR araştırmacılarından bazılarıyla birlikte) kurduğu Syntelligence şirketiydi. Syntelligence'de, uzman sistemler, René Reboh ile Tore Risch'in geliştirmiş olduğu ve PROSPECTOR'da kullanılmış fikirleri temel alan SYNTEL dilinde yazılmıştı. Charles Forgy, CMU'dan ayrıldıktan sonra, "gelişkin kural-tabanlı araçlar geliştirip pazarlamak amacıyla" 1983'te Production System Technologies şirketini kurdu.[42] Bu dönemde kurulmuş şirketler arasında Aion Corporation, Helix Expert System, Ltd., Exsys, A.Ş., Inference Corporation ve IntelliCorp vardı.[43] Kendi uzman sistemi sürümlerini geliştirmek isteyen müşteriler için bu iş çok zor olmadığından (bunlar düşük maliyetli iş istasyonlarında ve kişisel bilgisayarlarda çalışabiliyordu), pek çok uzman sistem şirketi ya kepenkleri kapatmış, ya daha büyük şirketler tarafından satın alınmış ya da işlerini dönüştürüp ek ya da ilişkili hizmetler sunmaya başlamıştır.

1980 ve 1990'larda uzman sistemler furyası bir parça dindikten sonra, bazı geliştiriciler, "iş dünyası kurallarını" öğrenen ve kullanan sistemlere yoğunlaştı. Business Rules Group [İş Dünyası Kuralları Ekibi] adlı bir kuruluşa göre iş dünyası kuralı, "iş dünyasının kimi yönlerini tanımlayan ya da sınırlandıran bir yargıdır. İş dünyasının yapısını

42 http://www.pst.com/.

43 Yukarıda bahsedilen Harmon'un ek bölümü bazı şirketleri sıralıyor, tıpkı şu internet adresi gibi: http://dmoz.org/Computers/ArtificialIntelligence/Companies/.

açıklamayı ya da iş dünyasının davranışlarını denetim altına almayı ya da etkilemeyi hedefler."[44] Örneğin, bir iş dünyası kuralı şunu ileri sürebilir: "Bir ürün sayısı 200'ün altına düştüğünde, o ürünün imalatının yapılmasını iste." İş dünyası kuralları, EĞER-İSE yargısı biçimini alır; tıpkı uzman sistemi kuralları gibi. İş uygulamalarında, uzman sistem çıkarsama motorları başkalaşım geçirip iş kuralı motorlarına (BRE; business rule engine) dönüşür. Bu motorlar, ya iş uygulamalarıyla ilgili soruları yanıtlamak, komut vermek ya da uyarı göndermek gibi eylemlerde bulunmak üzere kullanılır.[45] Uzman sistem yazılımları temin eden insanların bazıları iş-kuralları yazılımlarına geçmişti. Örneğin 2002'de Charles Forgy, RulesPower, A.Ş.'yi kurdu; bu şirketin iş dünyası kurallarını idare etme sistemleri (BRMS), Rete algoritmasının eski sürümlerini kullanıyordu. (2005'te RulesPower, kimi varlıklarını analiz ve karar idaresi teknolojisiyle ilgili bir şirket olan Fair Isaac Şirketine sattı ve o zamandan beri ismi FICO'dur.)

44 Şu internet adresinden: http://www.businessrulesgroup.org/defnbrg.shtml.

45 İş dünyası kuralları konusunda beni aydınlattığı için Business Process Trends (www.bptrends.com) genel yayın yönetmeni Paul Harmon'a teşekkür ediyorum.

19
SORGULARI VE SİNYALLERİ ANLAMAK

19.1 DÜZENEK

Yaklaşık 1970'lerin ortasına dek DARPA idarecileri, bilgisayar araştırma programlarını, uygulamaları vurgulayacak şekilde betimleyerek (Savunma Sanayisi araştırmalarının askerî uğraşlarla ilintili olmasını şart koşan) Mansfield Ek Yasasının etkisini yumuşatmayı başarabilmişti. 1960'ların ikinci yarısı ve 1970'lerin ilk yarısında DARPA IPTO Müdürlüğü yapmış olan Larry Roberts şunu yazmıştı:[1]

> Mansfield Ek Yasası, DARPA'da kalışım sırasında başımıza özel bir sorun açmıştı. Bizi epey bir bürokrasi ve olan biteni farklı bir temelde savunmaya zorladı. Araştırmadan çok geliştirmeye ağırlık vermeye itmişti bizi, böylece savunabileceğimiz bir karışım elde edebilecektik. Ancak, Mansfield Ek Yasası yüzünden ekibimizde bir projeyi bırakmak zorunda kaldığımızı sanmıyorum. Bilgisayar bilimini savunmanın bir yolunu hep buluyorduk...
>
> Kongreye gönderilen YZ konulu resmî raporlar kuramsal tasalardan ziyade olası etkiler altı çizili bir biçimde yazılıyordu.

IPTO'da Roberts'in altında çalışan Cordell Green ise şunu yazmıştı:[2]

> Genel konuşmak gerekirse, YZ alanında ortaya çıkan ve iyi olduğunu düşündüğümüz her şey desteklenmişti...
>
> Görevlerimden biri YZ bütçesini savunmaktı ama bu iş çok da zor değildi... her türlü bilgisayar biliminin bir geçerliliği vardı, çünkü bu bilim dalının, enformasyon işleyen her büyük kuruluş üzerinde büyük etkisi olur ve Savunma Bakanlığı doğal olarak böyle bir kuruluştur... tüm bu araştırmaların canlı tutulması gerekiyordu, çünkü olası askerî geçerlilikleri vardı.

Ancak, 1970'lerin ortasına gelindiğinde, askerî bakımdan faydalı sistemleri üretme baskısı çok daha yoğunlaşmıştı. Eskiden görece gü-

1 Lawrence G. Roberts, "Expanding AI Research and Founding Arpanet", *Expert Systems and Artificial Intelligence: Applications and Management* içinde, Thomas C. Bartee (ed.), Indianapolis, IN: Howard W. Sams & Co., 1988, s. 229-230.

2 C. Cordell Green, "Al During IPTO's Middle Years", *a.g.e.*, s. 238-240.

dümsüz temel YZ araştırmalarını cömertçe destekleyen DARPA, bunun yerine "acil Savunma sorunlarını" çözmeye odaklanır olmuştu. Her ne kadar DARPA IPTO'nun o zamanki müdürü J. C. R. Licklider, temel YZ araştırmalarına epey sıcak baksa da, DARPA'nın üst yönetiminin tavrı bütünüyle farklıydı. Licklider, kendi YZ programını DARPA'nın "üst yönetimine" açıklamada zorluk yaşıyordu. 1970'lerin ilk yarısında DARPA müdürü Stephen Lukasik (Licklider'e göre[3]):

> YZ'nin ne yanındaydı, ne de karşısında. İşlerin iyi idare edilmesi taraftarıydı ve bazı YZ çalışmalarının iyi idare edilmediğini düşünüyordu... Belli kilometre taşlarına ulaşmadığı müddetçe bir teklifin teklif sayılamayacağı gibi sabit bir fikri vardı. Bana kalırsa, bir teklifin, içerdiği kilometre taşı oranında iyi olduğuna inanıyordu... Sanırım, YZ'den tiksiniyor değildi, fakat YZ'nin, araştırmacıların daha büyük, daha katı ve daha yapılandırılmış bir bürokraside yaşamayı öğrenmesini gerektirecek kadar önemli bir alan olduğu inancını edinmişti.

Projelerin idare ediliş şekli hakkında Lukasik'in görüşü, DARPA'nın desteklediği temel YZ araştırmalarını doğrudan etkiledi. Örneğin, SRI bilgisayar tabanlı danışman projesinin ilerlemesini anlatan ve Şubat 1975'te gönderdiğim "Üç Aylık İdare Raporu", Licklider'i benden, raporu bir "PERT Çizelgesi"nde belirli yollar boyunca ilerleyişi vurgulayacak şekilde düzenlememi istemeye itmişti. 3 Mart 1975 tarihli mektubunda bana şöyle yazmıştı: "İstediğim şey PERT Çizelgesi; böylece kaydedilen başarıları kırmızıyla işaretleyip şablonunun bütününde senin hangi noktaya geldiğini görebilirim... Sende böyle bir çizelge var mı? Varsa, lütfen bana bir kopyasını gönder. Yoksa, yapmaya ne dersin? Böyle bir çizelgenin ARPA'da bize çok faydası dokunurdu."[4]

Elbette, her ne kadar temel araştırmada çözülmeye uğraşılan problemler genel olarak betimlenebilse de, çözümlerin ne olacağı (önceden) betimlenemez. Aslında, öncü araştırma ilerledikçe, yeni sorunlar belirir; dolayısıyla tüm sorunlar bile önceden tarif edilemez. Var olan bir teknolojiyi belli uygulamalarda hayata geçirmek üzere yapılan ayrıntılı planlar, temel araştırma söz konusu olduğunda oluşturulamaz. Ne yazık ki, DARPA'nın yönetimi, temel araştırmaları başlatmayı ve idare etmeyi bilen insanlardan, teknoloji uygulamalarını idare etmeyi bilen insanlara geçiyordu.

3 J. C. R. Licklider, "The Early Years: Founding IPTO", *a.g.e.*, s. 225-226.

4 Licklider'in mektubu dosyamda duruyor.



Resim 19.1. George Heilmeier (Fotoğraf, DARPA'nın izniyle.)

Kısa vadeli ve katı şekilde idare edilen araştırmalara geçiş, 1975'te Stephen Lukasik'in yerine George Heilmeier'ın (Resim 19.1) DARPA Müdürü olmasıyla iyice belirginleşti. Heilmeier, ilk likit kristal ekranı icat etmiş araştırma ekibine başkanlık ettiği RCA'dan gelmişti. Licklider ileride, Heilmeier'ın, "YZ'yi likit kristal ekranları anladığı şekilde anlamak istediğini", yazacaktı.[5]

Heilmeier'ın IPTO'ya verdiği görevlerden biri, YZ araştırma programı (ve başka bilgisayar bilimi programları) için bir "yol haritası" (yani ayrıntılı bir plan) oluşturmaktı. Bu yol haritası, geçmiş başarıları özetlemeli, var olan teknolojinin askerî sorunlar bağlamında hangi alanlara uygulanabileceğine işaret etmeli ve yol boyunca ulaşılacak kilometre taşlarını göstermeliydi. DARPA yönetiminin bu "kılavuzluğu" Licklider'e büyük zorluklar çıkardı; bu zorluklardan bazılarını, YZ araştırmalarının kimi önderlerine 1975 yılının Nisan ayında yolladığı bir e-postada açıklamıştır. (Bu "Paskalya Mesajını" alanlar arasında ben de vardım.) Bazı ikazları şöyleydi:

> Bu Paskalya notunun amacı, ARPA'daki bir gelişmeyi size bildirmektir. Beni endişendiren bu gelişme sanırım sizleri de endişelendirecektir...
>
> ... ARPA'da ağır basan eğilim, askerî problemlerin sınırlı bağlamında araştırma yapmak ve kavram tanımlandığında kendine askerî "alıcı" bulamayacak araştırmalara hiç bulaşmamaktır...
>
> Yeni Müdür Geroge Heilmeier, IPTO'nun, üniversitelerdeki YZ uğraşlarını savunma sanayisi bakımından geçerliliği olan problemlere yönlendirmesi konusunda ciddi baskı yapıyor...

5 J. C. R. Licklider, *a.g.e.*, s. 226.

> ... ARPA'nın on yıldır temel araştırmaları görece üst düzeyde desteklemesi (bu işe 50 milyon doları aşkın para harcamıştır) yüzünden bu durumu anlamak güç; ayrıca yeni müdürün, hatta eskisine göre, "Ulusal savunma bağlamında bu çabalardan ne elde ettik?" sorusu doğal.

Licklider'in Paskalya notuna göre, Heilmeier'ın IPTO'dan Savunma Bakanlığı için yapmasını bekledikleri arasında şunlar vardı:

- Başka kodların ve gürültünün varlığında Mors alfabesiyle yazılan kodları okuyan bilgisayarlar,
- Bir konuşmada kilit sözcükleri tanıyıp tespit edecek bilgisayarlar,
- Savunma Bakanlığının "yazılım problemini" çözmek,
- Komuta ve idareye gerçek bir katkıda bulunmak,
- Sonar radarları konusunda iyi iş çıkarmak.

Heilmeier'ın listesindeki maddelerden biri konuşma işlemeyle ilgili olsa da, DARPA'nın müdürü olarak sebep olduğu ilk zaiyatlardan biri SUR programıydı. Bu program sırasında geliştirilmiş sistemlerin hiçbiri gerçek zamanlı olarak tepki veremiyor, yeterince büyük sözcük dağarcıklarıyla başa çıkamıyordu. Heilmeier, konuşma anlamanın hâlâ bir temel araştırma etkinliği olduğuna inanıyordu (büyük olasılıkla sağlam sebeplerden ötürü). Dolayısıyla, bu tür projeleri Ulusal Bilim Vakfı'nın (NSF) desteklemesi gerektiğini düşünüyordu; ve projenin devam etmesi için DARPA'ya sunulan teklifleri reddetmişti.

Ne yazık ki, Licklider'in kendi listesinde olan araştırma alanlarının çoğu (Paskalya notunda bunlardan da bahsedilmişti), açıkçası Heilmeier'in listesinde yoktu. (Licklider'in listesindeki maddelerin birinden bahsetmekten kendimi alamıyorum: "Yeterince eğitilmemiş bakım elemanlarına, karmaşık donanımların bakımında kılavuzluk edecek bir sistem geliştirmek.") Ancak, Heilmeier'ın sıraladığı maddelerden biri, hem DDİ (doğal dil işleme) hem de bilgisayarla görme çalışmalarına gerekçe oluşturacak kadar muğlaktı. Bu madde, "komuta ve idare"dir; bu etkinlik, komutanlara geçerli bilgiyi ulaştırıp sunmayı, böylece komutanların, askerî kuvvetleri etkinlikle denetlemesini içeriyordu.

DARPA program görevlilerinden Floyd Hollister ve Albay David Russell, DARPA yönetimini, metin tabanlı doğal dilin, büyük ve dağınık veritabanlarına işlenmesinin, komuta ve idare sistemleri için önemli bir bileşen olacağına ikna etmişti. Bu erişim teknolojisinin, "komuta ve idare test cihazı sistemleri" dedikleri şeye uygulanabilecek kadar ileri olduğunu savunmuşlardı. En sonunda Bill Woods ve BBN'deki

çalışma arkadaşları, ay taşları veritabanının doğal dil "ön yüzü" olan LUNAR'ın marifetlerini sergilemişti. İngilizce ya da başka bir doğal dil kullanarak bilgisayarlarla iletişim kurma sorunu üzerinde çalışmaya çoktan başlamış olan birtakım araştırmacılar da vardı. (Örneğin, 1977'de MIT'de düzenlenmiş olan Beşinci Uluslararası Ortak Yapay Zekâ Konferansında (IJCAI) DDİ (doğal dil işleme) konulu kırkı aşkın makale sunulmuştu ve ACM'nin (Association for Computing Machinery [Bilgi İşleyen Makineler Birliği]) *SIGART Newsletter* dergisi 1977 Şubat sayısında, "Doğal Dil Arayüzleri" üzerine süregiden araştırmalar hakkında 52 özet yayımlamıştı.) Bu bölümün geri kalanında, doğal dil kullanarak bilgisayarlarla iletişim kurma konusunda bu dönemde elde edilmiş bazı başarılardan bahsediyorum.

Komuta ve idarede ikinci bir önemli alan da havadan çekilmiş fotoğrafların çözümlenişini otomatik hale getirmekti. Bu fotoğraflarda yollar, köprüler, askerî donanım gibi askerî önem taşıyan hedefleri belirlemek, istihbarat analistlerinin genelde saatlerini alır. Bilgisayarla görme alanındaki araştırmacıların geliştirmiş olduğu teknikler, insan analistlere yardımcı olacak araçlar temin edebileceği için, DARPA'nın bilgisayarla görme araştırmalarını desteklemek için sağlam sebepleri vardı. DARPA, askerî fotoğraf ve ilintili görüntülerin otomatik ve yarı-otomatik yorumu ve incelenmesi için gerekli teknolojiyi geliştirmek amacıyla 1976'da "Görüntü Anlama" (GA) programını başlattı. Bu, en başta beş yıllık bir program olarak doğmuş olsa da, yirmi yılı aşkın süre devam etmiştir (daha kapsamlı hedeflerle birlikte). İleriki bir bölümde, bilgisayarla görme hakkında başka araştırmaların yanı sıra görüntü anlama çalışmasını özetleyeceğim.

Sonar hakkında bir şeyler yapmak da Heilmeier'in listesindeki maddelerden biriydi. Aslında Licklider, Paskalya notunda şunu belirtmişti: "[Heilmeier'ın] başlıca sihirli çözüm alanlarından biri sualtı ses ve sonardır ve IPTO, HASP projesine (Ed Feigenbaum'un YZ yaklaşımı) "ortak olma" yolunda ilerliyor". HASP'ı ve DARPA'nın bu projeye nasıl "ortak olduğunu" bölümün sonuna doğru anlatacağım.

19.2 BİLGİSAYAR SİSTEMLERİNE DOĞAL DİL ERİŞİMLERİ

19.2.1 LIFER

SRI'da Gary Hendrix (Resim 19.2), LIFER (Language Interface Facility with Elliptical and Recursive Features) adlı bir sistem geliştiriyordu; INTERLISP dilinde programlanan bu sistemin amacı, veritabanlarının



Resim 19.2. Gary Hendrix (Fotoğraf, Gary Hendrix'in izniyle.)

ve başka yazılımların doğal dil "ön yüzlerini" hızlıca geliştirmekti. LIFER, teknik uzmanlığı olmayan kullanıcının, bir veritabanı sistemiyle ya da başka yazılımlarla doğal dil (örneğin İngilizce) aracılığıyla etkileşime girmesine olanak tanıyordu. LIFER bünyesindeki çözümleyici öğe, cümleleri ve soruları bu dile çevirerek yazılımla uygun etkileşime girilmesini sağlıyordu. LIFER'da, eliptik (yani eksik) girdilerle başa çıkan, imla hatalarını düzelten, acemilere dilin kapsamını açıklamalarla genişleme imkânı tanıyan mekanizmalar vardı.

LIFER'ın ilginç bir özelliği, kullandığı dilin, "örüntüler" bağlamında tanımlanması ve bu örüntülerin, uygulamalar alanında anlamsal kavramlardan faydalanmasıydı. Örneğin, bu tür bir örüntü şöyle olabilir:

```
WHAT IS THE <ATTRIBUTE> OF <PERSON> [Kişinin nitelikleri Nedir]
```

Burada WHAT, IS, THE, OF sözcükleri, İngilizce herhangi bir sorguda bulunabilecek gerçek sözcüklerdir ve <ATTRIBUTE> (nitelikler), <PERSON> (kişi) sözcükleriyse, önceden tanımlanmış kümelerde herhangi bir sözcüğe uyacak "joker"lerdir. <ATTRIBUTE>, YAŞ; AĞIRLIK; BOY gibi sözcüklere uyacak biçimde tanımlanabilir. <PERSON> ise JOHN, SUSAN, TOM gibi isimlere uyabilir. O halde bu örüntü, şöyle bir cümleyi tanıyacaktır:

```
WHAT IS THE HEIGHT OF SUSAN [Susan'ın Boyu Kaçtır]
```

"Dilbilgisi" tanımlamanın bu yöntemi, örneğin C <= İÖ YÖ gibi olağan sözdizimsel öbek yapısı kurallarına ters düşer. Daha önce bahsettiğim gibi, uygulama alanındaki kavramları temel alan dilbilgilerine "anlamsal dilbilgisi" denir.

LIFER, girdi cümlesini çözümlemek için (önceki bölümde anlattığıma benzer) basit bir genişletilmiş geçiş ağı kullanıyordu. Dilbilgisi tarafından tanımlanmış her örüntü, geçiş ağında olası bir "yola" denk

geliyordu. Girdi cümlesi, bu cümle o yollardan biriyle eşleştirilerek çözümlenir ve bu eşleştirmede <ATTRIBUTE> gibi bir jokerden faydalanıldığı bir kenara not edilir. Girilen yola ve bu yoldaki jokerlerin değerlerine bağlı olarak, uygun veritabanı sorgulamasını yapacak ya da uygun bir komutun gereklerini yerine getirecek yazılım otomatik olarak yaratılır.[6] 1982'de Hendrix, SRI'dan ayrılıp Symantec'i kurdu; bu şirket, LIFER gibi anlamsal dilbilgilerini temel alan bir doğal dil soru-yanıtlama sistemi geliştirip pazarlamayı planlamıştı. [Belki doğal dil işleme alanı (ya da hedeflenen pazar) buna pek hazır değildi, çünkü Symantec daha sonra bilgisayar güvenliği ve antivirüs yazılımı pazarlamaya geçti.]

SRI'da LIFER, çoklu ve dağınık veritabanlarına ulaşmaya yarayan "LADDER" adlı sistemin doğal dil bileşeni olarak kullanılmıştı.[7] LADDER (Language Access to Distributed Data with Error Recovery), İngilizce sorguyu, çok basit bir veritabanı örgütlenmesi benimsemiş olan varsayımsal bir veritabanı sorgusuna çeviriyordu. IDA (Intelligent Data Access) adlı bir sistem sayesinde bu varsayımsal sorgu, veritabanının gerçek örgütlenişini hesaba katan bir dizi gerçek veritabanı sorgusuna dönüştürülür. Aynı zamanda bu sistem, çok etkin sorgular üretip veritabanı içeriğinde herhangi bir hatalı güncellemeyi belirlemek üzere sözdizimsel ve anlamsal bilgiyi de hesaba katıyordu. (Stanford Üniversitesi ile SRI'nın DARPA tarafından desteklenen KBMS (Knowledge Based Management System) adlı ortak programında, IDA'ya benzer sistemler üzerine araştırmalar yapılmıştır.)

6 LIFER'in teknik ayrıntıları için bkz. Gary G. Hendrix, "LIFER: A Natural Language Interface Facility", SRI YZ Merkezi Teknik Notu 135, Aralık 1976 (şu internet adresinde: http://www.ai.sri.com/pubs/files/1414.pdf); Gary G. Hendrix, "The LIFER Manual: A Guide to Building Practical Natural Language Interfaces", SRI YZ Merkezi Teknik Notu 138, Şubat 1977 (şu internet adresinde: http://www.ai.sri.com/pubs/files/749.pdf); ve Gary G. Hendrix, "Human Engineering for Applied Natural Language Processing", *Proceedings of the 5th IJCAI*, 1977, s. 183-191 (ayrıca SRI YZ Merkezi Teknik Notu 139 olarak şu internet adresinde: http://www.ai.sri.com/pubs/files/748.pdf).

7 Earl D. Sacerdoti, "Language Access to Distributed Data with Error Recovery", *Proceedings of the 5th IJCAI*, 1977, s. 196-202, ve SRI YZ Merkezi Teknik Notu 140 olarak Şubat 1977'de basılmıştır (şu internet adresinde: http://www.ai.sri.com/pubs/files/747.pdf); Earl D. Sacerdoti, "A LADDER User's Guide (Revised)", SRI YZ Merkezi Teknik Notu 163R, Mart 1980 (şu internet adresinde: http //www.ai.sri.com/pi.ibs/files/735.pdf); ve Gary G. Hendrix ve diğ., "Developing a Natural Language Interface to Complex Data", *ACM Transactions on Database Systems*, c. 3, no: 2, Haziran 1978, s. 105-147 (SRI Merkezi Teknik Notu 152, Ağustos 1977; şu internet adresinde: http://www.ai.sri.com_pubs/files/741.pdf).

DARPA'nın askerî uygulamalara odaklanışıyla tutarlı olarak, LADDER, gemi boyutları, tipleri, konumları, vesaire hakkındaki bilgiden faydalanarak gemilerle ilgili soruları yanıtlayabiliyordu. LADDER'ın ilk sürümlerinden biriyle gerçekleştirilmiş örnek soru-cevaplar Resim 19.3'te gösteriliyor. Sistemin yazım hatalarını düzelttiğine, eksikli sorularla başa çıkabildiğine, kısaltmaları kabul ettiğine dikkat edin.[8]

19.2.2 CHAT-80

1979 ile 1982 arasında Fernando Pereira (1952-) ve David H. D. Warren (yaklaşık 1950-), Pereira'nın Edinburgh Üniversitesi'ndeki doktora tezinin bir parçası olarak CHAT-80 adlı bir sistem geliştirdiler. CHAT-80, coğrafi olgular hakkında görece karmaşık, İngilizce soruları yanıtlayabiliyordu.

Pereira'nın tezine bakılırsa,[9] CHAT-80 çalışmaları, "Colmeraurer'in önceki doğal dil çalışmalarını netleştirip geliştirme gayretleriyle" başladı. CHAT-80, aslen Alain Colmeraurer'in geliştirmiş olduğu mantık tabanlı programlama dili PROLOG'la yazılmıştı. Aslında, CHAT-80'in kullandığı dilbilgisi, PROLOG diliyle ifade edilmiş mantıksal formüllerden oluşuyordu. Örneğin,

```
cümle(c(İÖ, YÖ), C0,C) :- isim_öbeği (İÖ, N, C0,C1),
yüklem_öbeği(YÖ, N, C1,C)
```

CHAT-80'in dilinde bunun anlamı şudur: "Eğer C0 ile C1 noktaları arasında N sayıda isim öbeği (tekil ya da çoğul) ve C1 ile C noktaları arasında N sayıda yüklem öbeği varsa, C0 ile C noktaları arasında (sözcük) zincirli bir cümle vardır". PROLOG koşul tümceleriyle tanımlanmış bu tür dilbilgilerine *Kesin Koşullu Dilbilgileri* (KKD) denir. CHAT-80, İngilizce cümleleri çözümlemek için bu tür koşul tümcelerinden faydalanır. Asıl çözümlemeyi, bu koşul tümcelerinden oluşan PROLOG programı halleder.

CHAT-80'de, İngilizce bir sorgunun anlamının (yani anlambiliminin) hesaplanmasına, sorgunun sözdizimsel yapısı (PROLOG programının

8 LADDER'in daha ileri bir sürümüyle daha kapsamlı bir etkileşim için şu yayının Ek A kısmına bkz. Earl D. Sacerdoti, "A LADDER User's Guide (Revised)", SRI YZ Merkezi Teknik Notu 163R, Mart 1980.

9 Fernando Pereira, "Logic for Natural Language Analysis", doktora tezi, Edinburgh Üniversitesi, 1982. Bu tezin hafifçe değiştirilmiş bir versiyonu SRI YZ Merkezi Teknik Notu 275 olarak yayımlanmıştır ve şu internet adresindedir: http://www.ai.sri.com/pubs/files/669.pdf.

```
-What is the speed of the Kitty Hawk
PARSED!
((SPEED 35 KNOTS))

-Of the Ethan Allen
TRYING ELLIPSIS:  WHAT IS THE SPEED OF THE ETHAN ALLEN
((SPEED 30 KNOTS))

-Displacement
TRYING ELLIPSIS:  WHAT IS THE DISPLACEMENT OF THE ETHAN ALLEN
((STANDARD-DISPLACEMENT 6900 HUNDRED-TONS))

-length of the fastest Soviet sub
TRYING ELLIPSIS:  WHAT IS THE LENGTH OF THE FASTEST SOVIET SUB
((LENGTH 285 FEET / SPEED 30 KNOTS))

-Who owns the KIEV
     OWNS <== (assumed spelling error)
PARSED!
((COUNTRY USSR))

-who owns the JFK
TRYING ELLIPSIS:  ELLIPSIS HAS FAILED
THE PARSER DOES NOT EXPECT THE WORD "JFK" TO FOLLOW "WHO OWNS THE"
OPTIONS FOR NEXT WORD OR META-SYMBOL ARE:
<SHIP-NAME>

-Define JFK to be like Kennedy
PARSED!
 . [JFK is now a synonym for KENNEDY, which is a ship name]
 .
-REDO -2          [that is, parse WHO OWNS THE JFK]
PARSED!
((COUNTRY USA))

-? BUILT LAFAYETTE
TRYING ELLIPSIS:  ELLIPSIS HAS FAILED
 . [error message omitted]
 .
-Let "? built Lafayette" be a paraphrase of "who built the Lafayette"
PARSED!
 .
 .
-? built Lafayette
PARSED!
((BUILDER GENERAL.DYNAMICS))

-owns longest nuclear submarine
TRYING ELLIPSIS:  ? OWNS LONGEST NUCLEAR SUBMARINE
((COUNTRY USSR / LENGTH 426 FEET))
```

Resim 19.3. LADDER'la örnek soru-cevaplar. (SRI International'ın izniyle kullanılmıştır.)

hesapladığı haliyle) kılavuzluk eder ve bu hesap mantıksal formül olarak ifade edilir. Ardından bu formül, baştaki soruyu yanıtlamak için gereken, tekil veritabanı sorgularına dönüştürülür.[10] (CHAT-80'in çalışan bir sürümünün nasıl elde edildiği hakkında bilgi için bkz. http://www.cis.upenn.edu/~pereira/oldies.html.)

10 Bu görece teknik süreçlerin ayrıntılarına ilgi duyan okurlar Pereria'nın tezine ya da şu yayına bakabilir: David H. D. Warren ve Fernando Pereira, "An Efficient Easily Adaptable System for Interpreting Natural Language Queries", *Computational Linguistics*, c. 8 , no: 3-4, Temmuz-Aralık 1982, s. 110-122.

Aşağıda, CHAT-80'in yanıtlayabildiği sorgulardan birkaç örnek var (Pereria'nın tezinde beşinci bölümden):

```
S: Yukarı Volta'nın [Burkina Faso] başkenti neresidir?
Y: Vagadugu.
S: Londra hangi ülkenin başkentidir?
Y: Birleşik Krallık.
S: Hem Afrika ülkelerine hem Asya ülkelerine kıyısı olan okyanus
   hangisidir?
Y: Hint Okyanusu.
S: Baltık denizine kıyısı olan ülkelerin başkentleri nedir?
Y: danimarka: kopenhag, doğu almanya: doğu berlin, finlandiya:
   helsinki, polonya: varşova, sovyetler birliği: moskova, isveç:
   stokholm, batı almanya: bonn.
S: Ekvator çizgisinin güneyinde kalan Okyanusya hariç ülkelerin
   toplam yüzölçümü kaçtır?
Y: 10.228 milkare.
S: Hiçbir ülkesi, nüfusu 1 milyonu aşan ikiden fazla şehir içermeyen
   kıtalar hangileridir?
Y: afrika, antartika, okyanusya.
S: Akdenize kıyısı olan hangi ülke, nüfusu hindistan'ın nüfusunu
   aşan bir ülkenin komşusudur?
Y: türkiye.
```

Her ne kadar, bu örnekler görece etkileyici bir performansa işaret etse de, CHAT-80'nin yetenekleri, kısıtlı sözcük dağarcığıyla ve dilbilgisiyle sınırlanmıştır. Bu kısıtlamalar, Pereira'nın tezinde ayrıntılarıyla betimlenmiştir.

19.2.3 Taşınır Doğal Dil Sorgu Sistemleri

Anlattığım üzere CHAT-80, coğrafi olgular hakkında veritabanını sorgulamak üzere hayata geçirilmiş bir sistemdi. Bununla birlikte, tasarımının büyük kısmı coğrafyaya özgü olmadığı için, başka veritabanlarını ele alacak şekilde kolaylıkla değiştirilebilirdi. CHAT-80, farklı veritabanı çeşitlerinin doğal dil kullanıcılarına hizmet etmek üzere uyarlanabilmesi anlamında "taşınır" olan birkaç sorgu sisteminden biriydi. Caltech'de geliştirilmiş ASK,[11] SDC'de geliştirilmiş EUFID,[12]

11 Bozena H. Thompson ve Frederick B. Thompson, "Introducing ASK, A Simple Knowledgeable System", *Conference on Applied Natural Language Processing*, 1983, s. 17-24. Şu internet adresinde: http://ucrel.lancs.ac.uk/acl/A/A83/A83-1003.pdf.

12 Marjorie Templeton ve John Burger, "Problems in Natural Language Interface to

BBN'de geliştirilmiş IRUS,[13] Duke Üniversitesi'nde geliştirilmiş LDC-1,[14] Bell Laboratuvarları'nda geliştirilmiş NLP-DBAP,[15] SRI'da geliştirilmiş TEAM[16] bu tür sistemlerdendi.

Öbürlerine göre TEAM hakkında daha fazla şey bildiğim için, sınıfının temsilcisi olarak bununla ilgili birkaç söz edeceğim. TEAM (Transportable English Database Access Medium), DARPA tarafından desteklenmişti, ve veritabanı yöneticisinden veritabanı hakkında bilgi elde etmek ve o veritabanına uygun İngilizce altkümesiyle sorulmuş soruları yorumlayıp yanıtlamak üzere tasarlanmıştı. Öbür taşınır sistemler gibi TEAM de öyle kurulmuştu ki bu sistemi yeni veritabanına ve ilgili konulara uyarlamak için gereken bilgi, bu veritabanının doğal dil arayüzleri hakkında hiçbir şey bilmesi gerekmeyen bir uzmandan elde edilebiliyordu.

TEAM işleyişini açıklamak amacıyla, sistemin tasarımcıları, coğrafi verilerle ilgili dört "dosya"dan (ya da "bağıntı") meydana gelen bir veritabanı kullandılar. Bu dosyaların kısmi sürümleri Resim 19.4'te gösteriliyor. TEAM'in, "Her kıtanın en yüksek zirvesini göster" sorusunu yanıtlarken faydalandığı bazı adımları izleyeceğim.

TEAM, İngilizce sorguyu mantıksal bir ifadeye dönüştürmek amacıyla DIALOGIC[17] adlı bir altsistem kullanıyordu. DIALOGIC içinde DIAMOND'ı[18] temel alan bir altsistem, DIAGRAM dilbilgisini kullanarak

DBMS with Examples from EUFID", *Proceedings of the First Conference on Applied Natural Language Processing*, 1983, s. 3-16. Şu internet adresinde: http://www.aclweb.org/anthology-new/A/A83/A83-1002.pdf.

13 Madeleine Bates ve Robert J. Bobrow, "A Transportable Natural Language Interface", *Proceedings of the 6th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1983, s. 81-86.

14 Bruce Ballard, John C. Lusth ve Nancy L. Tinkham, "LDC-1: A Transportable, Knowledge-Based Natural Language Processor for Office Environments", *ACM Transactions on Information Systems*, c. 2, no: 1, Ocak 1984, s. 1-25.

15 Jerrold M. Ginsparg, "A Robust Portable Natural Language Data Base Interface", *Conference on Applied Natural Language Processing*, 1983, s. 25-30. Şu internet adresinde: http://ucrel.lancs.ac.uk/acl/A/A83/A83-1004.pdf.

16 Barbara J. Grosz ve diğ., "TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces", *Artificial Intelligence*, c. 32, no: 2, Mayıs 1987, s. 173-243. 20 Ekim 1986 tarihli SRI Teknik Notu 356R olarak şu internet adresinde: http://www.ai.sri.com/pubs/files/601.pdf.

17 Barbara Grosz ve diğ., "DIALOGIC: A Core Natural-Language Processing System;" *Proceedings of Ninth International Conference on Computational Linguistics*, 1982, s. 95-100. Şu internet adresinde: http://www.aclweb.org/anthology-new/C/C82/C82-1015.pdf.

18 DIAMOND'ı SRI'da William Paxton geliştirmiştir ve şu yayında betimleniyor: Ann

WORLDC

NAME	CONTINENT	CAPITAL	AREA	POP
Afghanistan	Asia	Kabul	260,000	17,450,000
Albania	Europe	Tirana	11,100	2,620,000
Algeria	Africa	Algiers	919,951	18,510,000

BCITY

NAME	COUNTRY	POP
Brussels	Belgium	1,050,787
Buenos Aires	Argentina	8,925,000
Canberra	Australia	210,600

CONT

NAME	HEMI	AREA	POP
Africa	S	11,500,000	41,200,000
Antarctica	S	5,000,000	500
Asia	N	16,990,000	2,366,000,000

PEAK

NAME	COUNTRY	HEIGHT	VOL
Anocagua	Argentina	23,080	N
Annapurna	Nepal	26,504	N
Chimborazo	Ecuador	20,702	Y

Resim 19.4. Bir TEAM veribankasında kullanılan dosyalar. (SRI International'ın izniyle kullanılmıştır.)

sözdizimsel çözümleme yapıyordu.[19] En yüksek puanlı ayrıştırma ağacı Resim 19.5'te gösteriliyor.

Bu ayrıştırma ağacını ve veritabanında kullanılmış kavramlar hakkındaki bilgiyi temel alan anlamsal bir çözümleme sistemi, sorguyu şu aşağıdaki mantıksal ifadeye dönüştürüyordu (daha iyi anlaşılsın diye burada konuşma diline yakın bir biçimde sunuluyor):

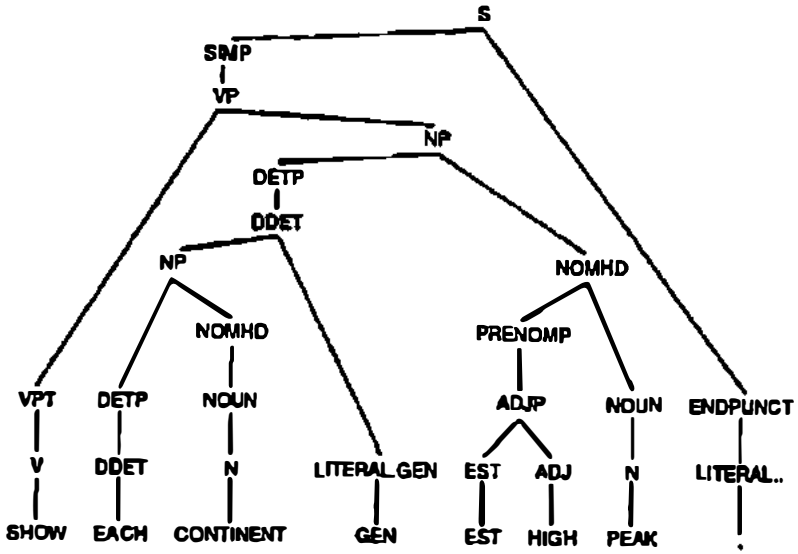
```
FOR EVERY CONTINENT
    WHAT IS EACH PEAK
    SUCH THAT THE PEAK IS THE HIGHEST PEAK SUCH THAT
        THE CONTINENT IS CONTINENT OF THE PEAK?

HER KITA İÇİN
    HER ZİRVE NEDİR
    ÖYLE Kİ BU ZİRVE EN YÜKSEK ZİRVE OLSUN ÖYLE Kİ
        KITA BU ZİRVENİN KITASI OLSUN?
```

Ardından TEAM, veritabanının yapısı hakkındaki bilgisini ve mantıksal ifadesinin bileşenlerinin veritabanındaki dosyalarla nasıl ilintili olduğuna dair bilgisini, güncel veritabanı sorgusunu oluşturup bir yanıt ortaya çıkarmak üzere kullanır.

E. Robinson ve diğ., "Interpreting Natural-Language Utterances in Dialogs About Tasks", YZ Merkezi Teknik Notu 210, SRI International, Mart 1980. Şu internet adresinde: http://www.ai.sri.com/pubs/files/709.pdf.

19 Jane J. Robinson, "DIAGRAM: A Grammar for Dialogs", *Communications of the ACM*, c. 25, no: 1, Ocak 1982, s. 27-47. Şubat 1980 tarihli SRI YZ Merkezi Teknik Notu olarak şu internet adresinde: http://www.ai.sri.com/pubs/files/712.pdf.



Resim 19.5. İngilizce "Her kıtanın en yüksek zirvesini göster" sorusu için ayrıştırma ağacı. (SRI International'ın izniyle kullanılmıştır.)

19.3 HASP/SIAP

1972'de, Larry Roberts henüz IPTO müdürüyken, Stanfordlu Ed Feigenbaum'dan, DENDRAL'da onca başarıyla kullanılmış YZ fikirlerini, gizli hidrofon dizilerinden elde edilmiş akustik verilerden faydalanarak okyanusta gemileri ve denizaltıları tespit ve takip etme sorununa uygulama hakkında kafa yormasını istedi.

Hidrofon dizilerinin topladığı akustik verilerin bir kısmı, gemilerin dönen şaftlarından, pervanelerinden ve güvertede ileri geri hareket eden makinelerden gelir. Farklı gemiler, kendilerine özgü niteliklerini yansıtan temel frekanslar ve armoniyle ses çıkarır. Bu tür verileri inceleyen insan uzmanlar, okyanus seslerinin sonogram görüntüsüne bakıp ses tayfını kayıtlı ses kaynaklarıyla eşleştirerek, var olabilecek (eğer varsa) gemileri tespit etmeye ve yerini belirlemeye çalışır. Bu kararlar verilirken çoğunlukla, sinyallerde olmayan ve başka sensörlerden gelen raporlar, istihbarat raporları ve gemilerin ve yaygın deniz rotalarının özgün nitelikleri hakkındaki genel bilgiyi de kullanmak gerekir.

Bu çözümleme sorunu bazı etkenler yüzünden iyice güçleşir:[20]

> Uzaktaki gemilerden gelen zemin gürültüsü, fırtına kaynaklı ve biyolojik gürültülerle karışır. Sensörlere ulaşan sesler gece-gündüz ve mevsim döngülerine göre değişir. Farklı yollar üzerinden gelen ses enerjisi aniden gelmez olur ya da deniz taşıtları ışınımının ancak bir kısmı ulaşır. Tek kaynaktan çıkan ses, pek çok farklı doğrultudan geliyormuş gibi görünebilir. Alıcının nitelikleri de, farklı kaynaklardan gelen sesin karışmasına yol açıp tek kaynaktan geliyormuş gibi görünmesine yol açabilir. Son olarak, en çok ilgilenilen denizaltı hedefleri epey sessiz ve gizli ilerler.

DARPA tarafından desteklenen ve bu sorunu ele alan çalışmalar, 1973'te Systems Control Technology, A.Ş.'de (SCI) başladı. Palo Alto kökenli bu şirketin söz konusu alanda uzmanlığı vardı ve gizli askerî projeler üzerinde çalışabiliyordu. (SCI'yı ileride British Petroleum satın alacaktı.) Feigenbaum ve SCI'daki çalışma arkadaşları, DENDRAL'ın "yarat ve sına" stratejisinin, okyanusları tarama sorununda işe yaramayacağını kısa süre içinde fark etti, çünkü belirli tarama verileri için aday gemi konumları, ayrıca rotaları oluşturabilecek bir "geçerli hareket yaratıcısı" falan yoktu. Ancak, bu çözümleme sorununun, HEARSAY-II'nin (gürültüdeki sinyallerle başa çıkmada iyi olduğu ispatlanmış bir sistemdi) Karatahta mimarisinde kullanılmış olanlara benzer düzeylere bölünebileceğini fark eden ekip, benzer bir yaklaşımın, ele aldıkları sorunda da işe yarayacağını düşündü. Böylece ekip, Karatahta modelini temel alarak HASP (Heuristic Adaptive Surveillance Program) adlı bir sistem geliştirdi. Gerçek okyanus verilerini işleyecek sonraki bir çalışma, SCI'da 1976'da SIAP'la (Surveillance Integration Automation Program) birlikte başladı. HASP/SIAP sisteminin kısa bir açıklamasını sunup ardından nasıl iş gördüğünü özetleyeceğim.

Karatahtanın üst düzeyi bir "durum tahtası"ydı; programın oluşturup güncellediği, var olan okyanus durumunun bir modeliydi bu. Okyanusta olduğu varsayılan tüm gemileri belirli bir istatistiksel güvenirlik düzeyiyle tanımlıyordu.

Durum tahtasının hemen altında, varsayılan deniz taşıtlarını tek tek içeren düzey bulunuyordu. Her deniz taşıtı, sınıfıyla, yeriyle, anlık hızı, rotası ve hedefiyle ilgili bilgi eşliğinde ve bir güvenirlik değeriyle birlikte gösteriliyordu. Deniz taşıtı düzeyinin altında, varsayılan ses

20 H. Penny Nii, Edward A. Feigenbaum, John J. Anton ve A. J. Rockmore, "Signal-to-Symbol Transformation: HASP/SIAP Case Study", *AI Magazine*, c. 3, no: 2, 1982, s. 23-35.

kaynaklarının düzeyi vardı: konumlarıyla ve güvenirlik değerleriyle birlikte motorlar, şaftlar, pervaneler, vb. Akustik verilerden çıkarılmış tayf özellikleri en alt düzeydeydi.

Bu düzeyler, eğer belirli unsurların belli bir düzeyde olduğundan şüpheleniliyorsa, o zaman öbür unsurların başka bir düzeyde olduğu çıkarsanabileceği sonucuna varabilen (ya da bu unsurlar zaten o düzeyde varsa, güvenirlik değerlerinin ayarlanabileceği) bilgi kaynakları (BK) aracılığıyla birbirine bağlanıyordu. Tıpkı HEARSAY-II'deki gibi, bu bağlantılar birden fazla düzeyi kapsayabiliyor, ayrıca bir düzeyin altında, üstünde, içinde çıkarımlar yapabiliyordu. Bir BK'nin çıkarımı, başka bir BK'nin ek bir çıkarım yapmasına yol açabilirdi ve bu iş zincirleme olarak, geçerli tüm bilgiler kullanılana dek sürüyordu. Bu usul sayesinde, yeni bilgi uyarlanabiliyor ve olası gelecek olaylarla ilgili beklentiler tanımlanabiliyordu.

BK türlerinden biri EĞER-İSE kurallarından oluşmuştu. (Başka türlerden de faydalanılmıştır.) Örneğin, kaynak düzeyinde iş gören bir EĞER-İSE kuralı (kolay anlaşılsın diye konuşma diline çevrilmiştir) şöyledir:

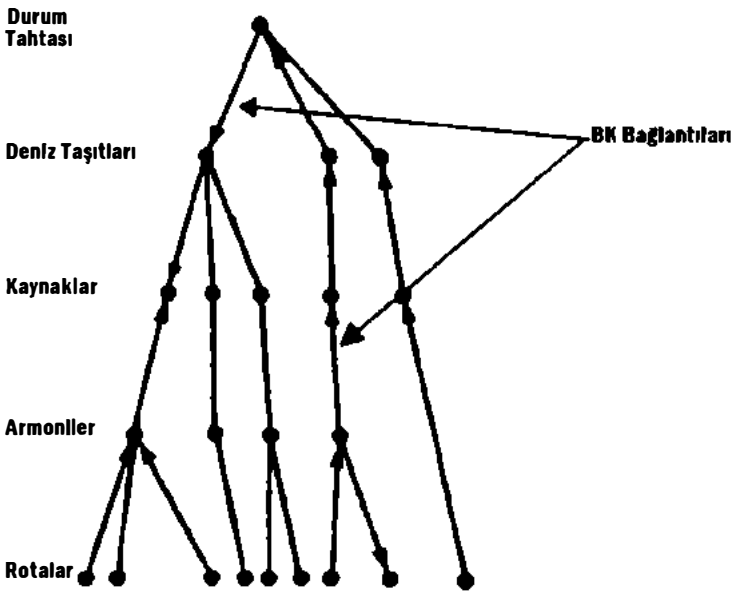
EĞER: Yakın geçmişteki sönümlenme yüzünden bir kaynak kaybedilmiş ve benzer bir kaynak başka bir frekansta devreye girmiş, ve iki kaynağın yeri görece yakınsa,

İSE: Güvenirlik değeri 3 ile bunlar aynı kaynaktır.

HASP/SIAP'ın birkaç bilgi kaynağı türü vardı ve bunların her biri, bir şekilde ilgili düzeye (ya da düzeylere) uygundu. Bazı BK'ler, yaygın ulaşım rotaları, sensörlerin konumu ve bilinen manevra alanları gibi çevreyle ilgili bilgiyi temel alıyordu. Deniz taşıtları ve deniz taşıtı tipleri, hızları, aksamları, akustik nitelikleri, merkez üsleri, vesaire hakkında bilgileri temel alan BK'ler de vardı. Çeşitli düzeylere denk düşen bilgileri ele alan BK'lere ek olarak, başka BK'lerin nasıl kullanılacağı hakkında bilgi içeren "meta" BK'ler de vardı.

Çeşitli düzeylerdeki bilgiler arasında köprü kuran BK'lerin bu fiilleri, ağ olarak gösterilebilir, örneğin Resim 19.6'da şema olarak örnek gösterilen ağ gibi. Çözümleme oturumunun sonunda, tüm BK'ler katılım şansı bulup etkileri sona erdiğinde, ortaya çıkan ağa, var olan okyanus durumunun "en iyi anlık hipotezi" (EGH) denir. Belirli bir HASP/SIAP işleminin EGH'sinin nasıl tarif edileceğine dair kısmi bir örnek şöyledir (konuşma diline çevrilmiş haliyle):[21]

21 *A.g.e.*, s. 28.



Resim 19.6. Farklı düzeylerdeki verileri birbirine bağlayan ağ yapısı. (Çizim şu yayından alınmıştır: H. Penny Nii, Edward A. Feigenbaum, John J. Anton ve A. J. Rockmore, "Signal-to-Symbol Transformation: HASP/SIAP Case Study", *AI Magazine*, sayı 3, no. 2, s. 26, Resim 2, Resim 2, © 1982, Association for the Advancement of Artificial Intelligence. İzinli kullanılmıştır.)

37.3 enlem ve 123.1 boylam civarında 2 gün, 4 saat, 55 dakika konumuna yerleşmiş Taşıt-1'in sınıfı Kiraz, Zambak, Lale ya da Gelinciktir. Taşıt-1 için farklı armoni kümelerinin desteklediği iki ayrı akustik kaynağı tespit edilmiştir. Kaynak-1, Kiraz ya da Gelincik sınıfı taşıtların şaftından ya da pervanesinden kaynaklanıyor olabilir. Kaynak-5 için de benzer kaynak olasılıkları vardır. Bu iki kaynak arasında olduğu bilinen mekanik oranın olasılığından ötürü, iki kaynak birlikte Taşıt-1'e benziyor.

MITRE şirketi, HASP/SIAP sisteminin performansını, iki uzman sonar analistinin performansıyla karşılaştıran bazı deneyler yapmıştı. Bu deneylerden birinde MITRE şu sonuca varmıştı: "HASP/SIAP'ın, okyanustan elde edilmiş verilerle iyi performans verdiği gösterilmiştir... Bu kısıtlı okyanus sahnesinde program önemsiz verilerden etkilenmemiş ve uzman analistlerinkine benzer sonuçlar vermiştir". Başka bir deneydeyse şu sonuca varılmıştır: "HASP/SIAP, okyanus sahnesini ikinci analistten daha iyi ve en az birinci analist kadar anlamıştır... Program, tek akustik sensörden daha fazlası olduğunda etkinlikle çalışabiliyor. SIAP, bir okyanus sahnesini üç saati aşkın süre boyunca

sınıflandırmıştır, bu da değişim geçiren okyanus durumunda SIAP'ın veriminin inandırıcı olduğunu gösteriyor". Üçüncü deneyse şu görüşü doğurmuştu: "SIAP programının, insan analiste göre çok daha fazla temas belirlemesi dışında, okyanus sahnesi hakkında betimlemeleri çok benzeşiyor". Dahası, "SIAP, bilgisayar kaynakları kullanımında artışlar olmaksızın gitgide daha zor okyanus sahnelerinde deniz taşıtı sınıflandırma işlemini gerçekleştirebilir."[22]

Daha önce bahsedildiği üzere, Karatahta modeli başka alanlarda da uygulanmıştı. Örnekler arasında protein kristalografi çözümlemeleri,[23] görüntü anlama[24] ve diyalog anlama[25] var. İlginçtir ki, Karatahta mimarisinin, teknolojinin ötesinde de etkileri olmuştur. Bilişsel psikolog Donald Norman, HEARSAY-II'nin, kuramsal psikoloji için bir fikir pınarı olduğunu söyleyip kendisine "genel bilişsel işleme sürecinin biçimiyle ilgili sezgiler" kazandırdığını eklemiştir.[26] Ayrıca, ileriki bir bölümde bahsedeceğim gibi, bazı neokorteks modelleri, Karatahta sistemlerinin hem biçimine, hem de mekanizmalarına benzeyen etkileşimli katmanlar içerir.

22 *A.g.e.*, s. 34.

23 Robert S. Engelmore ve H. Penny Nii, "A Knowledge-Based System for the Interpretation of Protein X-Ray Crystallographic Data". Stanford Bilgisayar Bilimi Bölümü Teknik Raporu CS-TR-77-589, 1977; şu internet adresinde: ftp://reports.stanford.edu/pub/cstr/reports/cs/tr/77/589/CS-TR-77-589.pdf.[

24 A. R. Hanson ve E. M. Riseman, "VISIONS: A Computer System for Interpreting Scenes", *Computer Vision Systems* içinde, A. Hanson ve E. Riseman (ed.), New York: Academic Press, 1978, s. 303-353.

25 W. C. Mann, "Design for Dialogue Comprehension", *Proceedings of the 17th Annual Meeting of the Association of Computational Linguistics* içinde, La Jolla, CA. Ağustos 1979, s. 83-84; bkz. http://ucrel.lancs.ac.uk/acl/P/P79/P79-1020.pdf.

26 Donald A. Norman, "Copycat Science or Does the Mind Really Work by Table Look-up?", *Perception and Production of Fluent Speech* içinde, R. Cole (ed.), Bölüm 12, Hillsdale, NJ: Lawrence Erlbaum Associates, Inc., 1980.

20
BİLGİSAYARLA GÖRMEDE İLERLEME

Yaklaşık 1970'den başlayarak bilgisayarla görme araştırmaları, YZ'nin oldukça gelişmiş bir alt uzmanlık alanına dönüşüp (sadece birkaçını saymak gerekirse) doğal dil işleme, robotik, bilgi gösterimi ve uslamlama gibi uzmanlaşmış alanların arasına katıldı. Bu bölümde, o dönem boyunca bilgisayarla görmedeki önemli ilerlemeleri betimleyeceğim. Bu ilerlemelerin bazıları, hava keşif, kartografi, robotik, tıp, belge çözümleme ve gözetleme gibi alanlarda belli uygulamaların peşinden koşarken kaydedilmiştir.[1]

20.1 ÇİZGİLERİ BELİRLEMENİN ÖTESİ

Önceki bir bölümde, görüntü kalitesini yükseltip görüntülerde kenar çizgilerini ve kenarları belirleyen bazı süzme tekniklerini anlatmıştım. Fakat, görüntülerin elde edildiği koşullar hakkındaki özel bilgi ve o görüntüde olması olası nesnelerin özellikleri hakkındaki genel bilgi kullanılarak bir görüntünün özellikleri belirlenirken çok daha fazlası yapılabilir.

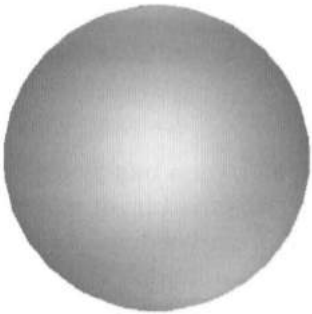
20.1.1 Gölgeden Şekil Bulmak

"Öze dönüş" denen harekette, araştırmacılar, yüzeylerden yansıyan ışık hakkında fizik ve geometri bilgisi kullanılarak iki boyutlu tek görüntüden bir görüntünün üç boyutlu özelliklerinin nasıl anlaşılacağını araştırmaya başlamıştı. Bu araştırmanın önderlerinden biri Berthold K. P. Horn'du (1943- ; Resim 20.1). Horn'un MIT'deki doktora tezi, bir nesnenin gölgesinden şeklini belirleyen matematiksel yöntemler ortaya koydu.[2] Tıpkı insanların, bir dairenin uygun şekilde gölgelenmiş

1 Bilgisayarla görme araştırmalarının kapsamlı bir listesi için CVonline İnternet sitesine bkz. http://homepages.inf.ed.ac.uk/rbf/CVonline/applic.htm

2 Berthold K. P. Horn, "Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View", MIT Elektrik Mühendisliği Bölümü doktora tezi, MIT YZ Laboratuvarı Teknik Raporu 232, Kasım 1970; bkz. http://people.csail.mit.edu/bkph/AIM/AITR-232-OCR-OPT.pdf. Horn, tezinde, görüntü parlaklığıyla ilgili çalışmaları için Thomas Rindfleisch'ın Ay topografyası araştırmalarına (1966) atıfta bulunur.





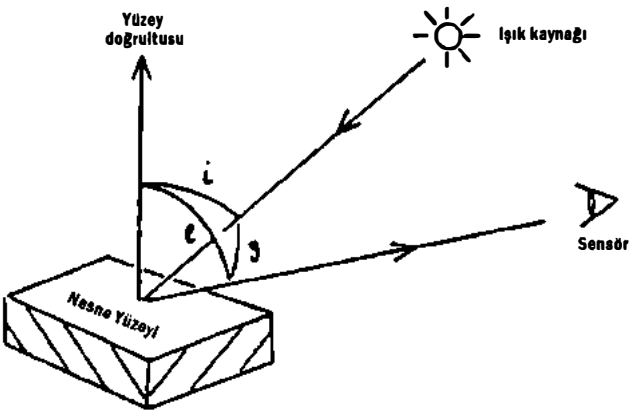
Resim 20.1. Berthold Horn (solda) ve gölgeli daire (sağda). (Fotoğraf, Berthold Horn'un izniyle.)

görüntüsünden aslında onun küre olduğunu algılayışı gibi, bilgisayarla görme sistemi de aynı işi yapabilir. Bilgisayara bu işi yaptırmak için Horn, yüzeylerin yansıma özelliklerine ve görüntüleme sürecinin geometrisine ilişkin bilgilerden yararlanmıştı.

Horn'un tekniğinin temel ilkeleri Resim 20.2'yle açıklanabilir: Yüzeyin son derece küçük bir parçasını, yüzeyden uzanan dik doğrultuya göre *i* açısıyla bir ışık kaynağı aydınlatıyor. Diyelim ki bir ışık sensörü (örneğin TV kamerası), ışık kaynağının doğrultusuna göre *g* açısıyla ve yüzeyin doğrultusuna göre *e* açısıyla yüzeyden yansıyan ışığı topluyor. Bu yüzey parçasından toplanan ışığın miktarı bu üç açıya, aydınlatmanın miktarına ve yüzeyin yansıtma özelliklerine bağlıdır. (Horn, bunun "mat" bir yüzey olduğunu varsaymıştı.) Toplanan ışığın miktarı süreksizliklerden böyle değiştiği için, şekil "gölgeli" görünür. Belirli koşullarda ve epeyce bir matematiksel işlemle, eğer öteki nicelikler de biliniyorsa, yüzeyin doğrultusu hesaplanabilir. Böylece, çok ama pek çok küçük yüzey parçasının doğrultusu bilindiğinde, yüzeyin genel şekli hesaplanabilir (yüzeyin görece pürüzsüz ve ani yoksun olduğu varsayılarak).

Horn, günümüzde MIT'de bilgisayar bilimi ve elektrik mühendisliği profesörüdür, ve bilgisayarla görmeyle ilintili konularda çalışmayı sürdürmektedir. Tezi, "gölgeden şekil bulma" alanında bir etkinlik furyası doğurmuştu.[3] Kimileri, gölgeden şekil bulma fikrinin kapsa-

3 Bu sorunun günümüzde ele alınışına dair bkz. Emmanuel Prados ve Olivier Faugeras, "Shape from Shading", *Handbook of Mathematical Models in Computer Vision* içinde, N. Paragios, Y. Chen ve O. Faugeras (ed.), New York: Springer-Verlag, 2006, s. 375-388; bkz. http://perception.inrialpes.fr/Publications/2006/PF06a/chapter-prados-faugeras.pdf.



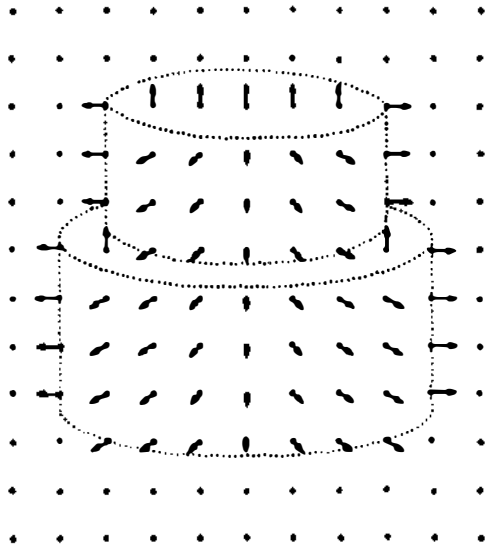
Resim 20.2. Gelen ışık yüzeyin küçük bir bölümünden yansıyor. (Çizim, Berthold Horn'un izniyle kullanılmıştır.)

mını genişletip gölge dışında başka şeylere dayanarak şekil hesaplamaya kalkıştı, örneğin çoklu görüntü (stereo), hareket, doku ve dış hatlar gibi. Böylece, önümüzdeki birkaç sayfada göreceğimiz üzere, nesnelerin şeklini belirlemenin ötesinde önemli işler yapılmıştır.

20.1.2 2½-B Taslak

Bakan kişi, üç boyutlu bir görüntünün ancak iki boyutlu imgesini görse de, David Marr (Horn'un fikirlerini geliştirerek), izleyicin yine de, görüntünün üzerindeki gölgelerden ve derinliği hakkında başka ipuçlarından yola çıkarak görüntünün, örneğin yüzey şekilleri, başka şekillerin önünü kapayan şekiller, düz yüzeyler arasında ani değişiklikler ve derinliği hakkında başka bilgiler gibi kimi üç boyutlu özelliklerini çıkarsayabileceği (dolayısıyla algılayabileceği) tespitinde bulunmuştu. Marr, bu özelliklerin gösterimine "2½-B taslak" (iki buçuk boyutlu) demişti (çünkü *tam* olarak üç boyutlu değildir). Marr'ın (kitabında betimlediği[4]) görme kuramına göre, lekelerden ve kenar çizgilerinden oluşan asal taslak (bkz. s. 183) yaratıldıktan sonra, görsel işlemede bir sonraki adım, bu 2½-B taslağı çizmekti. Resim 20.3'te örnek bir taslak gösteriliyor; bu resimde, yüzeyden dik çıkan oklar, elde edildikleri görüntünün asal taslağı üstüne oturtulmuş haldedir.

4 David Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, San Francisco: W. H. Freeman and Co., 1982.



Resim 20.3 2½-B taslak. (David Marr ve H. K. Nishihara, "Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes", *Proceedings of the Royal Society of London*, B Serisi, Biyolojik Bilimler, sayı 200, no. 1140, 23 Şubat 1978, s. 274.)

Son olarak, Marr'göre, 2½-B taslaktaki bilgi, nesne şekilleri hakkında saklanan bilgiyle birlikte, görüntüde belli nesnelerin yerini belirlemek ve böylece sahnenin 3-B modelini oluşturmak üzere kullanılabilirdi. Marr'ın bu süreç hakkında söylediklerini kısaca anlatacağım.

20.1.3 Yapı-İçi Görüntüler

SRI'da iki araştırmacı, Jay Martin Tenenbaum (1943- ; Resim 20.4) ve Harry Barlow (kısa süre önce Edinburgh'tan gelmişti), 2½-B taslak yaratmada kullanılanlara oldukça benzer görüntü işleme teknikleri geliştirmişti.[5] Görüntünün her pikselindeki yoğunluk değerinin, çeşitli etkenlerin karışık olarak birleşmesiyle belirlendiğini söylemişlerdi; bu etkenler arasında, çevresel ışıklandırmanın özellikleri ile sahnedeki görüntülerin yansıtma ve geometri özellikleri vardı. Bu etkenleri

5 Harry G. Barrow ve Jay Martin Tenenbaum, "Recovering Intrinsic Scene Characteristics from Images", *Computer Vision Systems* içinde, A. Hanson ve E. Riseman (ed.), New York: Academic Press, 1978, s. 3-26. Şu internet adreslerinde: http://web.mit.edu/cocosci/Papers/Barrow-Tenenbaum78.pdf ve http://www.ai.sri.com/pubs/files/737.pdf.

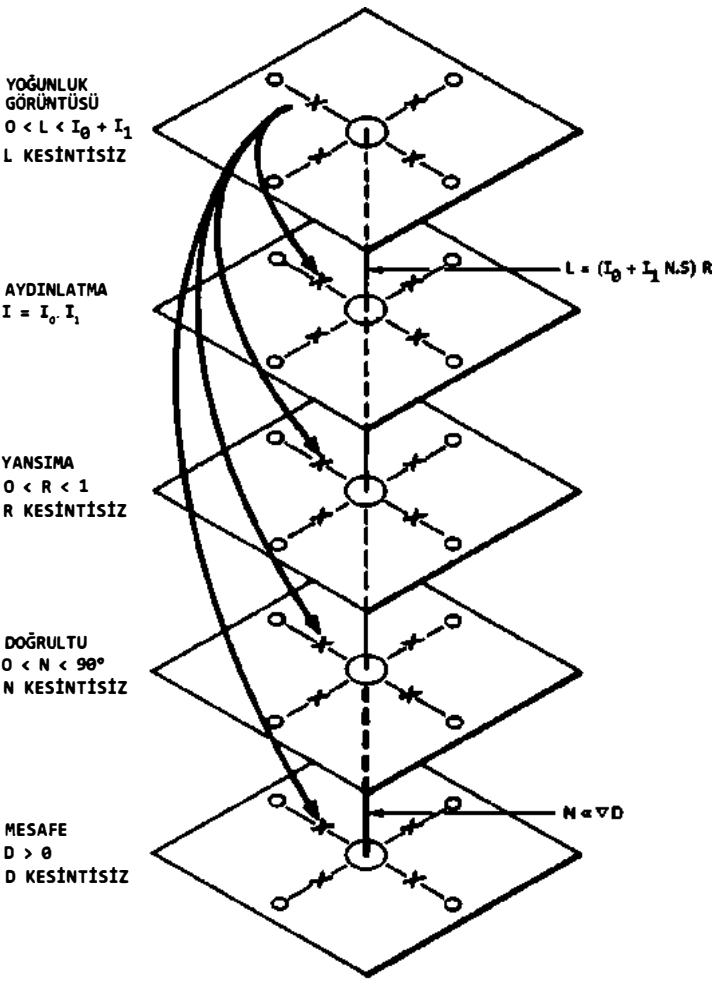


Resim 20.4. Jay Martin Tenenbaum (solda) ve Harry Barlow (sağda). (Fotoğraflar, J. Martin Tenenbaum'un ve Harry Barlow'un izniyle.)

çözümleyerek görüntü hakkında önemli üç boyut bilgisi elde edilebileceğini düşünmüşlerdi.

Barrow ve Tenenbaum, bu (hepsi yoğunluğu etkileyen) etkenlerden her birinin, "yapı-içi görüntüler" (*intrinsic images*) dedikleri hayalî şekillerle temsil edilebileceğini ileri sürdü. Bu şekiller, görüntünün bir yansımasını oluşturan ve yoğunluk görüntüsüyle üst üste oturan bir "piksel" ızgarasından oluşuyordu. Örneğin, yapı-içi görüntülerden biri aydınlatma görüntüsüydü. Bu görüntü, yansıtılan sahnedeki piksellerin üstüne düşen ışık miktarıyla değeri belirlenen piksellerden oluşuyordu. Elbette bu değerler biliniyor değildi, ama Barrow ve Tenenbaum, bu değerlerin yoğunluk görüntüsünden ve öteki yapı-içi görüntülerden yola çıkılarak tahmin edilebileceğini ileri sürdü.

Örnek olarak, Resim 20.5'te bir dizi yapı-içi görüntü gösteriyorum. Yoğunluk değerlerinin asıl görüntüsü en üstte yer alıyor. O görüntüde bir pikselin bilinen değeri, alttaki yapı-içi görüntülerin piksellerinin bilinmeyen değerlerine bağlıdır. Aslında, yapı-içi ya da asıl olsun, tüm görüntülerdeki piksel değerleri karşılıklı olarak bağımlıdır. Resimdeki oklar bu durumu yansıtıyor. (Yukarı giden oklar da olmalı.) Bazı görüntülerdeki piksellerin değeri temel alınarak, ötekilerin değeri, bilinen fiziksel ilişkilerden, görüntüler arasındaki kısıtlamalardan ve başka makul varsayımlardan yararlanarak hesaplanabilir. Sırası gelince bu değerler de başka değerlerin hesaplanmasına olanak tanır.



Resim 20.5. Yapı-içi görüntüler. (J. Martin Tenenbaum'un ve Harry Barlow'un izniyle kullanılmıştır.)

İşin özünde bu hesaplamalar, piksel değerlerini yapı-içi görüntüler kümesi boyunca "yayar" (Karatahta mimarisinde düzeylerin, öbür düzeyleri etkileyişi gibi). Tenenbaum ile Barrow'un daha sonra yöntemlerini özetlerken belirttiği gibi: "Bu oluşturma sürecini, bir dizi etkileşimli paralel ve yerel hesaplama olarak düşündük, tıpkı eşzamanlı denklemlerden oluşan bir sistemin, aşama dizilerini ileri besleme yerine gevşek bırakarak çözmesi gibi."[6] Barrow ve Tenenbaum, yapı-içi

6 Harry G. Barrow ve Jay Martin Tenenbaum, "Retrospective on 'Interpreting Line

görüntülerle ilgili kimi fikirlerinden, çizimleri üç boyutlu yüzey olarak yorumlama sorunu üzerine çalışırken de faydalanmıştır.[7]

Barrow ve Tenenbaum, çalışmalarının sadece bilgisayarla görme alanında değil, aynı zamanda insanların görme süreçleri için potansiyel bir "ön-bilişsel" model olarak da faydalı olmasını hedeflemişti. Fakat 1993'te, çalışmaları hakkında "geçmişe dönük bir incelemede" şöyle yazmışlardır:[8]

> Hesaplamalı görme alanının olgunluğuna ve sinirsel sistemlerdeki hızlı gelişmelere rağmen, görsel algıyı anlama hedefine yaklaşmak için önümüzde gidecek daha çok yol var. Bu hedefe varmak için, sinirbilim, sinir ağları, deneysel psikoloji, hesaplamalı görme dahil pek çok alandan öğrendiklerimizden faydalanmalıyız.

20.2 GÖRÜNTÜDE NESNELERİN BULUNMASI

20.2.1 Sahneler Hakkında Uslamlama

Gölgeden şekil bulma ve sahnelerde derinlik bilgisi çıkarma yöntemleri geliştirilmeden önce, pek çok araştırmacı, sahnede nesne bulma yöntemleri üzerinde çalışmıştı. Bu tekniklerin pek çoğunu 9.3 numaralı başlıkta betimlemiştim.

1970'lerin ilk yarısında Thomas Garvey, büro görüntülerinde masa, sandalye, çöp kutusu gibi nesnelerin yerini belirleyen bir sistem üzerine Stanford'da doktora yapmıştı.[9] Tezinin özet kısmında Garvey'in yazdığı gibi:

> Bu sistem, oda sahnesi görüntülerinde belli nesnelerin yerini bulmak için bir plan oluştururken, nesnelerin görüntülerini, aralarındaki ilişkileri ve eldeki sensörler hakkındaki bilgileri kullanır.

Drawings as Three-Dimensional Surfaces,'" *Artificial Intelligence,* c. 59, no: 1-2, 1993, s. 71-80.

7 Harry G. Barrow ve Jay Martin Tenenbaum, "Interpreting Line Drawings as Three-Dimensional Surfaces," *Artificial Intelligence,* c. 17, 1981, s. 75-116. Şu internet adresinde: http://web.mit.edu/cocosci/Papers/Barrow-Tenenbaum81.pdf.

8 Harry G. Barrow ve Jay Martin Tenenbaum, "Retrospective on 'Interpreting Line Drawings as Three-Dimensional Surfaces'", *Artificial Intelligence,* c. 59, no: 1-2, 1993, s. 71-80.

9 Thomas D. Garvey, "Perceptual Strategies for Purposive Vision", Stanford Üniversitesi doktora tezi, Eylül 1976'da SRI International YZ Merkezi Teknik Notu 117 olarak basılmıştır. Özeti şu internet adresinde: http://www.ai.sri.com/pub_list/759.

Bununla ilintili bir çalışmada Barrow ve Tenenbaum, MSYS adlı bir sistem geliştirdi; bu sistem, "sahnenin tutarlı bir açıklamasında anlaşana dek bilgi kaynaklarının yarıştığı ve işbirliği yaptığı"[10] sahneler hakkında uslamlama yapıyordu. MSYS, büro sahnesi görüntülerini çözümleyip, verilmiş aday yorumlar ve bunların olasılıkları içinden, görüntüdeki alanlar (masa üstü, sandalyenin arkası, zemin, kapı vs.) için en olası yorumu bulmaya uğraşıyordu. Alanlar arasındaki ilişkileri bilen MSYS (örneğin, "sandalye arkası genelde sandalye oturağına komşudur" gibi), alan hakkında yorumların toplamda en olası kümesini bulmaya çalışıyordu.

MSYS'nin gözden geçirdiği bir görüntüye ilişkin örnek, Resim 20.6'da gösteriliyor. Bu görüntüdeki kimi alanlar, olası yorumlarla belirlenip etiketlenmiştir.

Barrow ile Tenenbaum'un yazdığı gibi, MSYS'nin uslamlaması şöyle ilerler:

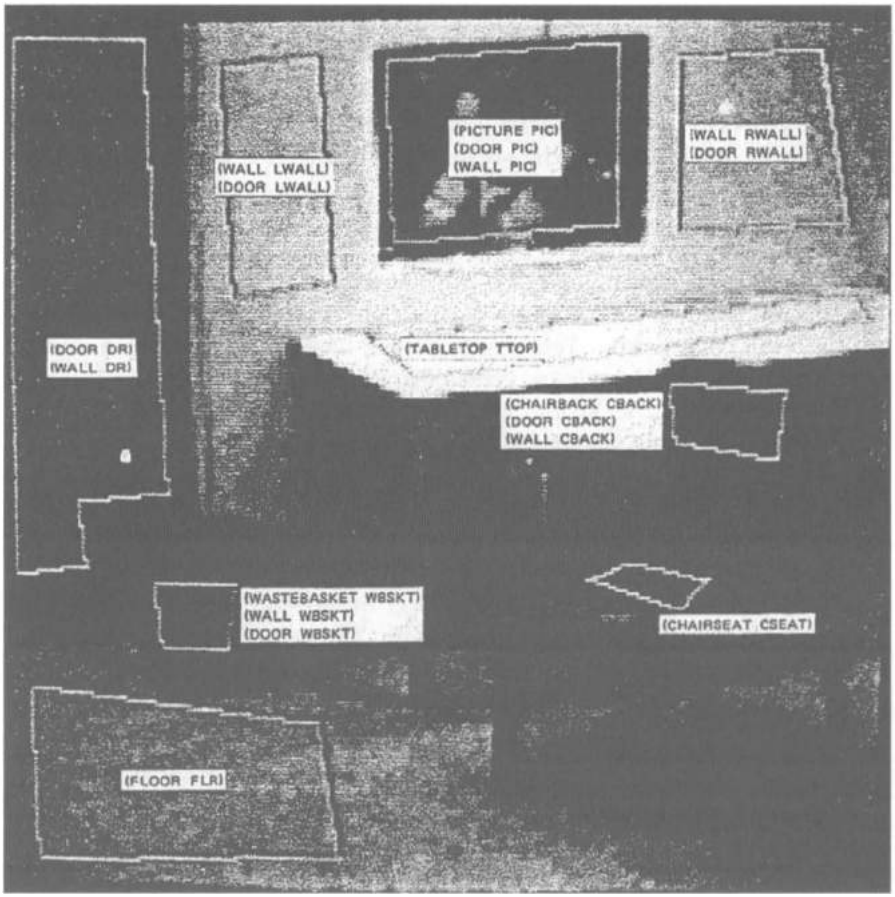
> PIC, WBSKT ve CBACK etiketli alanlar WALL [DUVAR] ya da DOOR [KAPI] olamaz, çünkü bunların parlaklığı, hemen üzerlerindeki görüntünün üst kenarının parlaklığından azdır; bu durum, duvarların ve kapıların parlaklığına dair bilgiye ters düşüyor. Bu nedenle PIC alanı PICTURE [TABLO] olmalı; WBSKT, WASTEBASKET [ÇÖP KUTUSU] olmalı ve CBACK, CHAIRBACK [SANDALYE ARKASI] olmalı.
>
> LWALL ve RWALL etiketli alanlar o halde WALL olmak zorunda, çünkü bunlar PIC alanına komşu ve DOOR, PICTURE'a komşu olamaz.
>
> DR etiketli bölge WALL olamaz, çünkü WALL etiketli tüm alanların parlaklığı aynı olmak zorunda. Dolayısıyla, DR alanı DOOR olmalı.

20.2.2 Şablon ve Modellerin Kullanılması

Nesne tanıma hakkında ilk çalışmaların büyük kısmı, görüntülerle eşleştirilebilen nesne "şablon" kullanımını temel almıştır. Martin A. Fischler ve Robert A. Elschlager, daha güçlü eşleştirme tekniklerine olanak tanıyan "esnek şablon" kullanarak bu fikri geliştirdi. Yüz gibi nesneleri ya da belli arazi özelliklerini, bu tür nesneler barındıran fotoğraflarda bulmak amacıyla o şablonlardan faydalanmışlardır.[11] Bu

10 Harry G. Barrow ve J. Martin Tenenbaum, "MSYS: A System for Reasoning about Scenes", SRI International YZ Merkezi Teknik Notu 121, Nisan 1976. Şu internet adresinde: http://www.ai.sri.com/pubs/files/757.pdf.

11 Martin A. Fischler ve Robert A. Elschlager, "The Representation and Matching of Pictorial Structures", *IEEE Transactions on Computers*, c. C-22. no: 1, Ocak 1973, s. 67-92.



Resim 20.6. Bazı alanların belirlenip etiketlendiği bir MSYS sahnesi. (Çizim, SRI International'ın izniyle kullanılmıştır.)

süreç, aranan nesne hakkında genel bir gösterim barındırmaya ve o gösterimi, fotoğrafla eşleştirme sürecine dayanıyordu. Nesnelerin gösterimleri, nesneyi pek çok ilkel parçaya bölüp "bu 'ilkel parçaların' söz konusu nesne için karşılaması gereken uzamsal ilişkilerin uygun bir yelpazesini belirlemeyi" temel alıyordu. Fotoğrafta var olan nesne için, "ilkel parçalarının (en azından önemli bir altkümesinin) var olmasını ve aynı zamanda birbirleriyle belirli bir uzamsal ilişki içinde de var olmalarını gerektirir..." Fischler ile Elschlager'in işaret ettiği gibi, parçalardan bazılarının var olmasını belirleme, nesnenin bütününün varlığını belirlemeye bağlıdır ve nesneyi belirleme, parçaları belirleme-

ye bağlıdır. Makalelerin başlıca katkısı, bu döngüsellikle başa çıkmak için dinamik programlama tarzında bir yöntem geliştirmiş olmasıydı.

Daha önce, David Marr'ın, asal taslak ve 2½-B taslak yaratma süreçleri üzerine çalışmalarını betimlemiştim. Bunlar, Marr'ın görme kuramında ilk iki aşamaydı. Marr, bu aşamaların, görüntüde olması olası nesnelerin şekilleri hakkında özgül bilgi olmaksızın önemli şekil bilgisini ortaya çıkarabileceğini savunuyordu. Şöyle yazmıştır:[12]

> İlk görsel işlemlerin çoğu, bir atın mı adamın mı yoksa ağacın mı parçası olduklarına bakmaksızın görünür yüzeyler hakkında doğrudan bilgi çekip çıkarmaktı... Temel alınabilecek başka hangi bilgilerin dikkate alınması sorununa gelince, genel bilgi yeterli olmalı; ilk görsel işlemlerde bu genel bilgi, genel kısıtlar ve yüzeylerin üç boyutlu uzamda birlikte var olduğu gerçeğinin geometrik sonuçlarından oluşur.

Marr, şekillerle ilgili özgül bilgiden üçüncü aşamada faydalanılması gerektiğini ileri sürdü. Bu aşamada süreç, nesnelerin üç boyutlu modellerini kullanır. Büyük modelin alt parçalara ayrıştırıldığı ve alt parçaların daha alt parçalara ayrıştırıldığı bir model hiyerarşisi kullanmayı önerdi. Örneğin, insan şekli Resim 20.7'deki gibi modellenebilir. Her kutu, üç boyutlu bir modele ve bunun alt modeline karşılık gelir. Kutunun sol tarafında eksen yönelimli model duruyor; sağ tarafta, bu modelin, alt modellerle nasıl gösterildiği görülebilir. (Eksenlerin doğrultusu, görüntünün uygun parçalarına göre ayarlanabilir.)

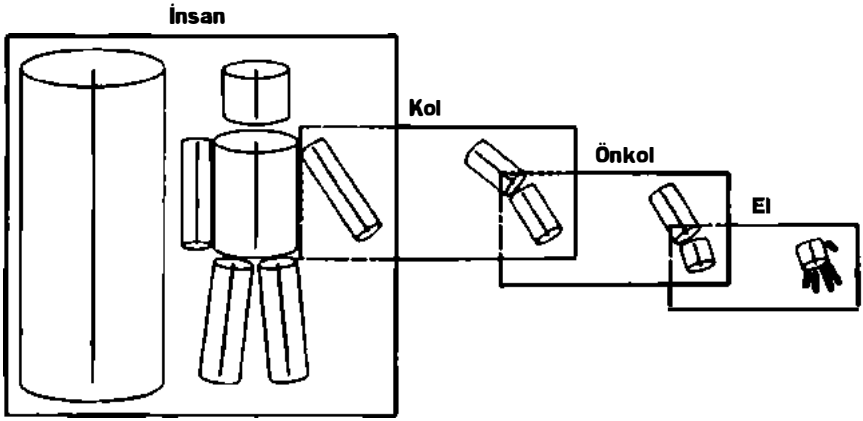
Bu üçüncü aşamada, bu tür modelleri şekil bilgisiyle ve 2½-B taslağın içerdiği başka üç boyut bilgileriyle karşılaştırmak, görüntüdeki nesnelerin belirlenip yerlerinin bulunmasını kolaylaştırır. Marr'a göre görme, "görüntülerden yola çıkıp dünyada neyin, nerede olduğunu keşfetme sürecidir."[13]

Marr, nesne parçalarının modeli olarak silindir kullanmayı öneren ilk kişi değildi. 1971 tarihli bir IEEE konferansı makalesinde Thomas O. Binford (1936-), "genelleştirilmiş silindir" fikrini tanıtmıştı (zaman zaman "genelleştirilmiş koni" de denir).[14] Daha sonraki bir makale bunları şöyle tanımlar: "Genelleştirilmiş koni, düzlemsel bir kesitle, uzamsal eğim omurgasıyla ve bir tarama kuralıyla tanımlanır. [İllâ dairesel olmayan bir] kesitin, [omurga denen bir eksen boyunca]

12 David Marr, *a.g.e.*, s. 272-4.

13 David Marr, *a.g.e.*, s. 23-60.

14 Thomas O. Binford, "Visual Perception by Computer", *Proceedings on System and Control*, Miami FL, Aralık 1971.



Resim 20.7. Marr'ın 3-B model hiyerarşilerinden bir örnek. (David Marr, *Vision* içinde, San Francisco: W. H. Freeman and Co., 1982, s. 306.)

ilerlerken taradığı, omurgaya göre sabit bir açıda tutulan ve tarama kuralına göre dönüştürülen bir hacmi temsil eder."[15]

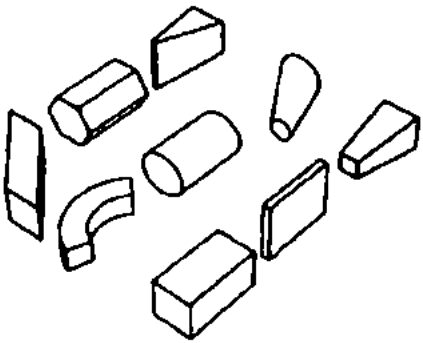
Binford'un yanında, sahnelerdeki nesneleri belirlemeye yarayan modellerden faydalanan Stanfordlu doktora öğrencileri çalışıyordu. Bunların içinden Gerald J. Agin,[16] Ramakant Nevatia[17] ve Rodney A. Brooks'un (1954-)[18] adlarını anabilirim; tüm bu isimler, "model tabanlı görme" olarak bilinen alana katkıda bulunmuştur. (Brooks ileride farklı konular üzerinde çalışacağı MIT'de profesör olacaktı. Sonraki çalışmaları ileride anlatılacak.)

15 Rodney A. Brooks, "Symbolic Reasoning among 3-D Models and 2-D Images", *Artificial Intelligence*, c. 17, no: 1-3, s. 285-348.

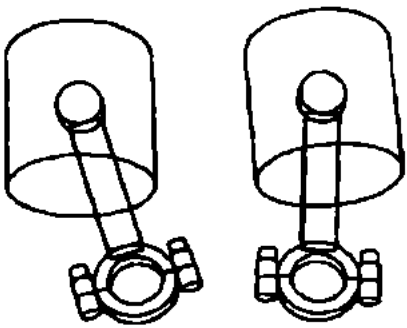
16 Gerald J. Agin, "Representation and Description of Curved Objects", Stanford Üniversitesi, doktora tezi, Stanford Yapay Zekâ Projesi Raporu AIM-173 (Ekim 1972) olarak basılmıştır. Ayrıca bkz. Gerald J. Agin ve Thomas O. Binford, "Computer Descriptions of Curved Objects", *Proceedings of the Third International Joint Conference on Artificial. Intelligence*, Ağustos 1973, s. 629-640; daha sonra şurada basılmıştır: Gerald J. Agin ve Thomas O. Binford, "Computer Descriptions of Curved Objects", *IEEE Transactions on Computers*, c. 25, no: 4, Nisan 1976.

17 Ramakant Nevatia, "Structured Descriptions of Complex Curved Objects for Recognition and Visual Memory", Stanford Üniversitesi Elektrik Mühendisliği Bölümü doktora tezi, Stanford YZ Laboratuvarı Raporu (Ekim 1974) olarak yayımlanmıştır.

18 Rodney A. Brooks, "Symbolic Reasoning among 3-D Models and 2-D Images", Stanford Üniversitesi Bilgisayar Bilimi Bölümü doktora tezi, 1981, Stanford BB Bölümü Raporu STAN-CS-81-861 olarak yayımlanmıştır. Ayrıca şurada basılmıştır: Rodney A. Brooks, *a.g.e.*



Resim 20.8. İlkel genel koniler ve genelleştirilmiş konilerden yola çıkılarak inşa edilmiş piston modelleri. (Rodney A. Brooks, "Symbolic Reasoning among 3-D Models and 2-D Images", *Artificial Intelligence*, sayı 17, no. 1-3, s. 285-348.)



Brooks'un ACRONYM sistemi,[19] farklı nesne türlerini modellemek amacıyla genelleştirilmiş konilerden faydalanıyordu. ACRONYM, bu modelleri, görüntülerdeki nesneleri belirleyip yerini bulmak için kullanıyordu. Modellerin ve model nesnelerin yapı taşları olarak kullanılabilecek genelleştirilmiş koni tipleri hakkında örnekler Resim 20.8'de gösteriliyor.

Görmenin ne işe yaradığıyla ilgili başka fikirler, Marr'ın ve görme sayesinde bütünlüklü görüntüler inşa etmek isteyenlerin görüşleriyle rekabet ediyordu. Kimileri, özellikle de robotik alanında çalışanlar, görme işleminin amacının, eylemi yönlendirecek şeyleri algılamak olduğunu ileri sürüyordu. Shakey'deki görme altprogramlarının pek çoğu, eylem programları içine yerleştirilmişti. Maryland Üniversitesi'nde Profesör Yiannis Aloimonos, bu "amaçlı" ya da "etkileşimli" yaklaşımı sa-

19 Bu sistemi ilk bildirinler: Rodney A. Brooks, Russell Greiner ve Thomas O. Binford, "The ACRONYM Model-Based Vision System", *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Tokyo, 1979, s. 105-113. Daha sonra gözden geçirilmiş versiyonu Brooks'un *Artificial Intelligence* dergisinde çıkmış makalesinde bildirilmiştir.

vunan araştırmacılardan biriydi. Görmenin hedefinin eylem olduğunu ileri sürer. Görme, "eylemle birlikte gözden geçirildiğinde kolaylaşır" der. Şöyle açıklar: "Sistemin oluşturması gereken uzay-zaman tarifleri genel amaçlı değil amaca yöneliktir. Bunun anlamı, o tanımların, yön bulmayla, elle yapılan işlerle ve tanımayla ilintili görevler gibi kısıtlı görev kümelerinde geçerli olmasıdır."[20] Marr'ın katkıda bulunmak istediği sinirbilim camiasından Patricia S. Churchland, V. S. Ramachandran ve Terrence J. Sejnowski ileride şöyle yazacaklardı: "Görme yetisi ne işe yarar? Üç boyutlu dünyanın eksiksiz bir içsel yaratımı gerçekten gerekli mi? Bu sorulara verilen biyolojik ve hesaplamalı yanıtları, saf görmeden oldukça farklı görme kavramları doğurmuştur [Marr'ın da savunduğu gibi]. Etkileşimli görme... eylemlere rehberlik edilmesinde görmenin ortağı olarak başka duyu sistemlerini de içerir."[21]

Her durumda, modeller, bilgisayarla görmede hâlâ önemli rol oynuyor. (Bununla birlikte, önde gelen görme araştırmacılarından biri bana, "model tabanlı görmenin kalıntısı sıfıra yakın"[22] demişti, başka biriyse çok amaçlı ve bilim-tabanlı-çözümleme yöntemleri yerine "günümüzde çoğu robotik sistemin görme için kestirme programlar kullandığını" söylemişti.[23])

20.3 DARPA'NIN GÖRÜNTÜ ANLAMA PROGRAMI

ABD'deki bilgisayarla görme araştırmalarının büyük kısmını DARPA ödenekleriyle destekliyordu, fakat görme araştırmacıları arasında (her zamanki gibi) desteğin sürmeyeceği endişeleri vardı. Tenenbaum, 1974'te katıldığı ve bilgisayarla görme araştırmalarının geleceğinin tartışıldığı bir DARPA toplantısını anlatır. DARPA destekli görme araştırmalarını izleyen yetkili, Hava Binbaşı David L. Carlstrom da toplantıdaymış ve sahadaki çeşitli çalışmaların uyum içinde hareket etmesini istiyormuş. DARPA bu alandaki çalışmaları yıllardır desteklediği için, Carlstrom, DARPA'nın yeni bir işe kalkıştığını göstermek

20 Onun internet sitesinden: http://www.cfar.umd.edu/yiannis/.

21 Patricia S. Churchland, V. S. Ramachandran ve Terrence J. Sejnowski, "A Critique of Pure Vision", *Large-Scale Neuronal Theories of the Brain* içinde, Christof Koch ve Joel L. Davis (ed.), Cambridge, MA: MIT Press, 1994, s. 23-65. Şu internet adresinde: philosophy.ucsd.edu/faculty/pschurchland/papers/kochdavis94critiqueofpurevision.pdf.

22 Martin A. Fischler, kişisel iletişim, 1 Ağustos, 2007.

23 Jay Martin Tenenbaum, kişisel iletişim, 31 Temmuz, 2007.

amacıyla yeni bir isme ihtiyaç duyuyormuş. Tenenbaum bana, yeni girişimin adının "görüntü anlama programı" olmasını Carlstrom'a tavsiye ettiğini anlatmıştı.[24] (Konuşma anlama alanında zaten süregiden bir DARPA destekli araştırma olduğunu anımsayın, dolayısıyla bu ifade "DARPA-dostu" bir sözdü.")

1976'da DARPA, Görüntü Anlama (GA) programını başlattı. Bu program, o alanda çalışan başlıca araştırma programlarından ve hem üniversitelerde hem şirketlerde faaliyet gösteren "ekip"lerden oluşan büyük bir uğraş haline geldi. Buna tek başına katılan laboratuvarlar arasında MIT, Stanford, Rochester Üniversitesi, SRI ve Honeywell laboratuvarları vardı. Üniversite/sanayi ekipleri ise USC-Hughes Araştırma Laboratuvarları, Maryland Üniversitesi-Westinghouse A.Ş., Purdue Üniversitesi-Honeywell A.Ş. ve CMU-Control Data Şirketiydi.

Düzenli çalıştaylarda, kaydedilen ilerlemeler bildiriliyordu. 1977'de düzenlenmiş bir çalıştayın toplantı raporu, programın hedefini ifade ediyordu: "Görüntü Anlama Programı, askerî fotoğrafların ve bununla ilintili görüntülerin otomatik ve yarı-otomatik yorumu ve çözümlemesi için gereken teknolojiyi geliştirmek üzere planlanmış beş yıllık bir araştırma uğraşıdır."[25] DARPA'nın GA programı için son hedefi, o toplatı raporunun kapağındaki çizimde gayet açık ifade edilmiştir (Resim 20.9).

Çizimin ima ettiği gibi komutanlar, bilgisayarla görme sistemlerinin, fotoğrafları çözümleyip fotoğrafın önemli bileşenleri ve aralarındaki ilişkiler hakkında yazılı bir betimleme oluşturmasını istiyordu.

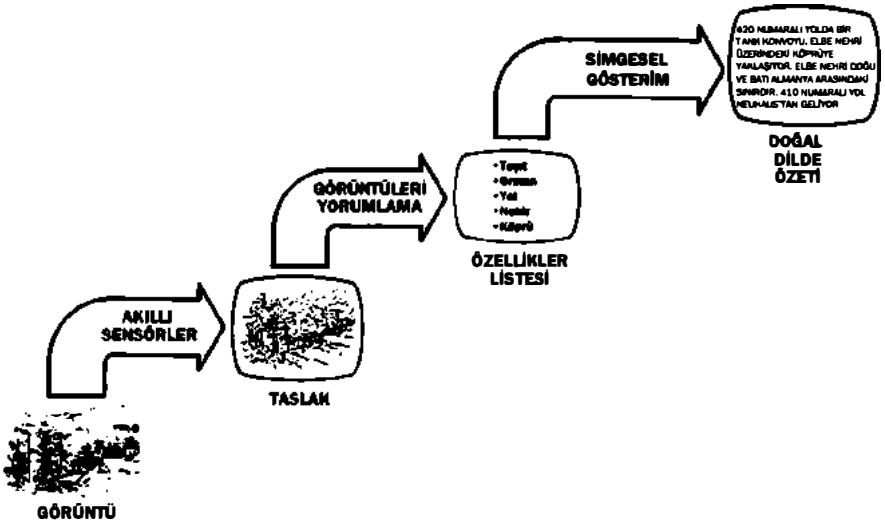
2½-B eskiz, yapı-içi görüntüler, genelleştirilmiş silindirler ve ACRONYM gibi daha önce anlatmış olduğum bazı bilgisayarla görme araştırmaları, GA programı tarafından desteklenmişti. Gelgelelim, DARPA'nın hedefleri ile bilgisayarla görme araştırmaları yapanlar arasında daima gerilim olmuştur. DARPA, bu programın, "sahada yetkin" sistemler üretmesini istiyordu. J. C. R. Licklider, Mart 1975'te düzenlenmiş ön hazırlık nitelikli GA çalıştayında bu noktayı vurgulamıştı:[26]

> Beş yıllık dönemin sonunda geliştirilen teknoloji o hale gelmiş olmalı ki, Savunma Bakanlığı birimleri, özgül sorunlar karşısında bu teknolojinin nasıl uygulanacağı hakkında yaptığımız muazzam araştırmalara gereksinim duymaksızın kendi özgül sorunları için bundan faydalanabilmeli.

24 Kişisel iletişim, 31 Temmuz 2007.

25 Lee S. Bauman (ed.), *Proceedings: Image Understanding Workshop*, Science Applications A.Ş, Inc., Rapor no: SAI-78-549-WA, Nisan 1977.

26 Alıntı şu yayının önsözünden: *Proceedings: Image Understanding Workshop*, Science Applications A.Ş., Mayıs 1978.



Resim 20.9. GA programının hedeflerinin çizimi. (SAIC'in izniyle kullanılmıştır.)

Bu nedenle, beş yıllık dönemin sonunda program, önemli Savunma Bakanlığı problemlerinin çözüldüğünü göstermeli.

DARPA'da görevli Hava Binbaşı Larry Druffel, 1978'de GA programının önderliği oldu. Kasım 1978'de şu tavsiyeyi verdi: "Dikkatli yaklaşım, Savunma Bakanlığı birimlerine aktarılacak oranda olgun teknikleri sağlamlaştırmak olacaktır."[27] 1979'a gelindiğinde, programın hedefleri genişlemiş, kartografiyi ve haritalandırmayı da içine almıştı. DARPA ile Savunma Bakanlığı Harita Birimi (SHB) arasında belirlenen "anlama genelgesi"nde (MOU), DARPA/SHB "test cihazlarının" geliştirilmesi sayesinde otomatik haritalandırma çalışmalarının desteklenmesi kararlaştırıldı. Kasım 1979'da Druffel şöyle yazacaktı:[28]

> Haritalandırmanın, grafik çiziminin, yerölçümü işlevlerinin otomasyonu sayesinde GA teknolojilerinin olgunluğunu değerlendirmeye alacak bir gösterim sisteminin planları ilerliyor. Bu sistem, bir taraftan belirli kartografik fotoğraf yorumlama işlevlerine odaklanırken, görüntü okuma camiasının bütününe, kendilerine özgü problemleri bakımından, Görüntü Anlama yöntemlerinin gelecekteki uygulamalarını değerlendirme fırsatı sunmalı.

27 *Proceedings: Image Understanding Workshop*, Önsöz, Science Applications, Kasım 1978.
28 *Proceedings: Image Understanding Workshop*, Önsöz, Science Applications, Kasım 1979.

Bu "beş yıllık" program 1981'de sona ermedi. DARPA'nın kanatları altında ve Donanma Komutanı Rohn Ohlander, Hava Yarbayı Robert L. Simpson Jr. ve diğerleri önderliğinde yaklaşık 2001'e dek sürdü. 1985'te Simpson, bu programın başarılarından bazılarını özetlemiştir:[29]

> Aslında 1975'te Yarbay David Carlstron'un sayesinde beş yıllık bir program olarak doğan GA çalışmalarının ilk yıllarında, bilgisayarla görmeyi genelde "görüntü işleme" denen şeyden farklılaştırmaya başlayan altdüzey görme teknikleri ve bilgi-tabanlı sistemler için güçlü bir temel döşemiş oldu. 1970'lerin ikinci yarısında ve 1980'lerin başında, Yarbay Larry Druffel'in yönetimi altında bu program, ACRONYM gibi model tabanlı görme sistemlerinin geliştirilmesine ve DARPA/DMA görüntü anlama test cihazı gibi daha anlamlı kavramsal gösterimlerde GA tekniklerinin gösterimine şahit oldu. Bu gösterimler ve bunların gelecekteki askerî kullanım potansiyelleri, GA programının, en başta biçilmiş beş yıllık ömrünün ötesine geçmesini sağlamıştır. Komutan Ron Ohlander yönetimi altında GA teknolojisi olgunlaşarak öyle bir noktaya ulaştı ki DARPA Stratejik Bilişim Programı, özerk kara taşıtı gibi büyük bir başvuruyu haklı çıkarabiliyor.

Ohlander'in dediği gibi, GA programı, öngörülen beş yıllık ömrünün ötesine geçmiştir. Henüz 1984'te, DARPA'nın bu çalışmalar için 4 milyon doları aşkın para harcadığı söylenmişti.[30] Potansiyel başvurulardan biri robot-denetimli askerî taşıtlar için bilgisayarla görmeydi; DARPA'nın "Stratejik Bilişim" programının bileşenlerinden biriydi bu. Bu uygulamayı ve başkalarını ileriki bölümlerde daha ayrıntılı anlatacağım.

Yapay zekânın gitgide büyüyen bir alt uzmanlık alanı olarak, bilgisayarla görme makaleleri bu konu başlığına adanmış yeni dergilerde belirmeye başladı; bunlar arasında *Computer Vision and Image Understanding* ve *IEEE Transactions on Pattern Analysis and Machine Intelligence* dergileri vardır. Bu alanı işleyen ders kitapları arasında, *Pattern Classification and Scene Analysis*[31] [Örüntü Sınıflandırma ve Görüntü Çözümleme] adlı ve iki de *Computer Vision*[32] [Bilgisayarla Görme] başlıklı kitabı sayabiliriz.

29 *Proceedings: Image Understanding Workshop,* Önsöz, Science Applications, Aralık 1985.

30 Alex Roland, Philip Shiman'la birlikte, *Strategic Computing: DARPA and the Quest for Machine Intelligence*, Cambridge, MA: MIT Press, 2002, s. 220.

31 Richard O. Duda ve Peter E. Hart, *Pattern Classification and Scene Analysis,* New York: John Wiley and Sons, Inc., 1973.

32 Michael Brady, *Computer Vision,* Amsterdam: North-Holland Publishing Co, 1981; Dana H. Ballard ve C. M. Brown, *Computer Vision*, New York: Prentice Hall, Inc 1982.

21
YÜKSELİŞ DÖNEMİ

Her ne kadar Mansfield Ek Yasası ve Lighthill Raporu 1970'ler boyunca temel YZ araştırmalarına zorluklar yaşatmış olsa da, önemli uygulamalar bulunacağına yönelik umutlar, hem devletten hem de sanayiden ödenekler gelmesine yol açtı. Heyecan, özellikle de uzman sistemler için duyulan heyecan, 1980'lerin ortasında zirve noktasına çıkacaktı.

1975 ile 1985 arasında kalan kabaca on yıllık dönemi, YZ'nin "yükseliş dönemi" sayıyorum. Bu yükselişin ardından bir tasarruf dönemi gelmiş olsa bile, başarıları pek çok ve önemliydi. 1980'de, tüm o yıllık konferanslarıyla, çalıştaylarıyla ve sempozyumlarıyla birlikte Amerikan Yapay Zekâ Birliği'nin (AAAI; günümüzde Yapay Zekânın İlerlemesi Birliği deniyor) kuruluşuna tanıklık edildi. (Resim 21.1, bu dönemdeki pek çok ticari fuardan bir manzara gösteriyor.) Ulusal ve bölgesel başka YZ teşkilatları da kuruldu. Başlangıcını, 1960'ların ikinci yarısındaki birkaç araştırma kurumuna borçlu olan Arpanet, zaman içinde evrilip, bütün dünyada bilgisayarları birbirine bağlayan İnternet'e dönüştü.

LISP programlama dilinin çeşitli sürümleri birleşip INTERLISP'i oluşturdu, bu da YZ'nin hem araştırmalarında hem de uygulamalarında baskın dil olmayı sürdürdü (gerçi Avrupa, Kanada ve Japonya'da PROLOG rağbet gören bir rakipti). MIT araştırmacıları ve öğrencileri, LISP programlarını etkin şekilde çalıştıran, iş istasyonu tarzında bilgisayarlar tasarlayıp bunlara Lisp makineleri dediler. Lisp Machines A.Ş. ve Symbolics, bu makineleri üretip satan iki şirkettir. Başlarda başarının tadını çıkarmışlardır ama yavaş yavaş yerlerini başka çalışma istasyonu tedarikçileri aldı.[1]

Uzman sistem şirketleri ve Lisp makinesi şirketleri furyasına pek çok başka YZ şirketi de katılmıştır. Örneğin, 1978'de Earl Sacerdoti ve Charles Rosen, robot görme sistemlerini pazarlamak amacıyla Machine Intelligence Şirketini kurdular. Daha önce IBM'e ilk bilgisayarını geliştirmesinde yardımcı olmuş Cuthbert Hurd (1911-1996) ve David Warren, PROLOG sistemleri pazarlayan Quintus A.Ş.'yi 1984'te kurdu.

1 Yeri gelmişken, "iwhois" internet sitesi (http://www.iwhois.com/oldest/), kayıtlı en eski ".com" sayfasının sahibi olarak Symbolics'i sayıyor (15 Mart 1985'te kaydedilmiş).



Resim 21.1. 1980'lerde düzenlenmiş ticari bir AAAI fuarından bir sahne. (Fotoğraf: Bruce B. Buchanan, "Some Recollections about the Early Days of AAAI", *AI Magazine*, sayı 26, no. 4, s. 14, © 2005 Association for the Advancement of Artificial Intelligence. İzinli kullanılmıştır.)

Berkeley Üniversitesi lisansüstü öğrencilerinden Fritz Kunze, 1984'te kurduğu Franz A.Ş.'yle, LISP programlama dilinin bir sürümü olan FranzLISP'i pazarlıyordu.[2] YZ konferanslarıyla ilişkili ticari fuarlarda yapılan gösterişli tanıtımlar bütün bu alana heyecan aşılıyordu. Derneğin kuruluşundan kısa süre sonra AAAI'nın üye sayısı 5.000'den, zirve noktası olan 16.241'e 1987'de ulaştı. (O yükselişten sonra AAAI üyeliği duraklayıp yaklaşık 5.000'e inmiştir.) YZ'nin kendileri için ne yapabileceğini merak eden bu yeni üyelerin çoğu, sanayi ve devlet kuruluşlarından geliyordu. Hem AAAI'da hem de IJCAI'da çeşitli YZ konu başlıkları hakkında verilen kurslar, bu yeni ve önemli alanı öğrenmek isteyen sanayi kökenli insanlardan epey rağbet görüyordu.

1980'lerin ilk yarısı boyunca, SRI'nın çeşiltli bölümlerindeki çalışma arkadaşlarım, özellikle de Savunma Bakanlığı projelerinde çalışanlar, o zamanlar yöneticisi olduğum SRI YZ Merkezi'nden yardım almaya hevesliydi. Bana kalırsa, asıl istedikleri şey, daha cazip hale gelsin diye, devlet kuruluşlarına sundukları proje tekliflerine "bir tutam YZ serpmemizdi."

2 Bkz. http://www.franz.com/about/company.history.lhtml.

Bilim yazarı George Johnson, artan bu ilgiyi 1984'te bildirirken şöyle yazmıştır:[3]

> TIMM [adlı bir uzman sistemin] broşürü "daha iyi bir beyin inşa ettik" diyor. Zeki Makine Modeli: "Uzman sistemler; bekleme süresi, personel gereksinimi ve uzman darlığı yüzünden yaşanan tıkanmaları azaltıyor. Ayrıca uzman sistemler hasta olmaz, istifa etmez, erken emekliliğe ayrılmaz". IBM, Xerox, Texas Instruments ve Digital Equipment Corporation gibi başka şirketler, duyurularında daha tutucu davranmış belli ki. Fakat çeşitli malları pazarlayan satış elemanlarının güçlü sesleri [1984'te AAAI sergi salonunda], zaman zaman karnaval çığırtkanlarının ya da yeni bir çağı haber veren peygamberlerin seslerine benziyor.

Bu yükseliş, Japonya'nın "Beşinci Nesil Bilgisayar Sistemleri" projesiyle sürdü. Sırası gelince bu proje, DARPA'yı "Stratejik Bilişim Girişimi"ni kurmaya teşvik etmiştir. O proje aynı zamanda, Avrupa'da benzer araştırma çalışmalarının oluşturulmasını (Birleşik Krallık'taki ALVEY Projesi ya da Avrupa'nın ESPRIT programı gibi), ayrıca bilgisayar donanımındaki gelişmeleri daha da ileri taşımak için Amerikan sanayisinin bir konsorsiyum kurmasını kışkırttı. YZ'nin kimi zorlukları ve kazanımları, vaat ettikleriyle karşılaştırıldığında, 1980'lerin ikinci yarısında bu yükselişin sonunu getirmiştir; bazılarının "YZ kışı" dediği döneme girilmiştir. Tüm bu konu başlıklarını ilerideki bölümlerde anlatacağım.

3 George Johnson, "Thinking about Thinking", *APF Reporter*, c. 8, no: 1, 1984. Bkz. http://www.aliciapatterson.org/APF0801/Johnson/Johnson.html.

BEŞİNCİ KISIM

"YENİ NESİL" PROJELER

22
JAPONLAR HEYECAN YARATIYOR

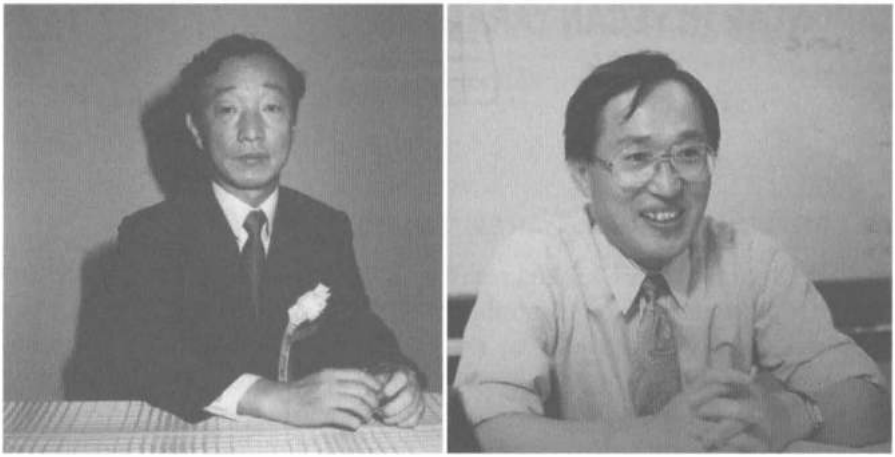
22.1 BEŞİNCİ NESİL BİLGİSAYAR SİSTEMLERİ PROJESİ

1982'de Japonya'nın Uluslararası Ticaret ve Sanayi Bakanlığı (UTSB), "Beşinci Nesil Bilgisayar Sistemleri" (BNBS) adını verdiği sistemi geliştirmek amacıyla devletin ve sanayinin ortaklaşa giriştiği bir proje başlattı. Bu projenin amacı, büyük veri ve bilgi tabanlarından YZ çıkarımları yapabilecek ve doğal dil aracılığıyla insanlarla iletişim kurabilecek bilgisayarlar üretmekti. Bu projeyle ilgili raporlardan birinin belirttiği üzere: "Bu sistemlerin, çıkarıma ve bilgi tabanlı işlevlere dayalı ileri değerlendirme becerileri ve zeki bir arayüz işlevi aracılığıyla esnek etkileşim becerileri olması bekleniyor"du.[1]

"Beşinci Nesil" sözü, bilgisayar teknolojisinin önceki "nesillerinin" ötesine geçecek çarpıcı ilerlemeyi vurgulamak amacıyla kullanılmıştı. İkinci Dünya Savaşı sırasında ve sonrasında geliştirilmiş *ilk nesil*, vakumlu tüplerden faydalanmıştı. 1959 civarında vakumlu tüplerin yerini transistörler alıp *ikinci nesle* can vermiş oldu, gerçi transistörler, kendilerinden önceki vakumlu tüpler gibi, birbirlerine ve devrenin öbür öğelerine hâlâ bakır tellerle bağlıydı. 1960'lar sırasında transistörler ve başka öğeler, tek katmanlı silikon çipler üzerinde imal edilir oldu ve bilgisayarı oluşturan birkaç çip, birbirlerine kablolarla bağlanıyordu. Bu küçük ölçekli entegrasyon (SSI) teknolojisini kullanan bilgisayarlar, *üçüncü nesli* oluşturmuştur. 1970'lerin ikinci yarısında, "çok büyük ölçekli entegrasyon" (VLSI) teknolojisi kullanılarak tek çipin üzerine koca koca mikroişlemciler yerleştirilebiliyordu; *dördüncü nesil* işte buydu. Japonların *beşinci nesli* ise, gelişmiş yazılımının yanı sıra, "ultra büyük-ölçekli entegrasyon" (ULSI) teknolojisini kullanan paralel işlemciler içerecekti.

UTSB, bilgisayar bilimcilerinin "iş istasyonu" demeye başladığı biçimde bir prototip makine geliştirmeyi planlamıştı; bu makine, paralel çalışıp çoklu veri ve bilgi tabanına erişebilen birkaç işlemciden oluşacaktı. Mantığa dayalı bir bilgisayar programlama dili olan PROLOG, sistemin "makine dili" olacaktı, çünkü Japonlar, bu dilin doğal dil

1 "Research Report on Fifth Generation Computer Systems Project", ICOT İlerleme Raporu, Mart 1983.

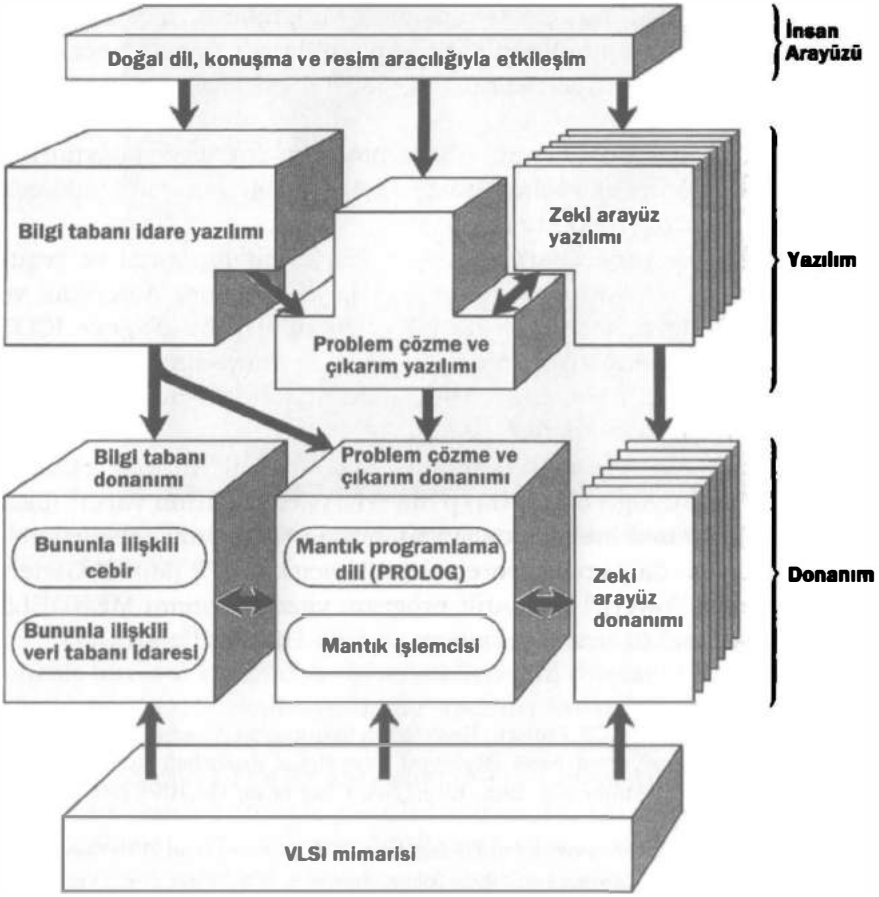


Resim 22.1. Kazuhiro Fuchi (solda) ve Koichi Furukawa (sağda). (Fuchi'nin fotoğrafı Tohru Koyama'nın izniyle. Furukava'nın fotoğrafı, Koiçi Furukava'nın izniyle.)

işlemeye, uzman işi uslamlamaya ve akıllarından geçirdikleri başka YZ uygulamalarına uygun olacağını düşünmüştü. PROLOG ifadelerinin çalıştırılması mantıksal çıkarsama içeriyordu, böylece makinenin performansı, saniye başına yaptığı mantıksal çıkarımla (SBMÇ) ölçülecekti. 1980'lerin ilk yarısında bilgisayarlar yaklaşık 100.000 SBMÇ yapabiliyordu. Japonlar, bu hızı en az bin katına çıkarabileceklerini düşünmüştü. Projenin ilerleyen safhalarında, PROLOG'u aynı anda birden fazla işlemciyi çalıştırmak üzere uyarlama zorlukları yüzünden, çoklu işlemci birimi çalıştırabilen yeni bir mantık tabanlı dil olan GHC (Guarded Horn Clauses) geliştirildi.

UTSB, BNBS çalışmaları için, "Yeni Nesil Bilgisayar Teknolojisi Enstitüsü" (ICOT) adlı bir enstitü kurdu. Bu enstitüde Bay Kazuhiro Fuchi'nin (1936-2006; Resim 22.1) başkanlık ettiği Araştırma Merkezi, prototip sistemi geliştirmek için gerek duyulan temel araştırmaları yürütecekti. Bu merkezin birinci yıl ilerleme raporuna göre,[2] "Araştırma Merkezi, Elektroteknik Laboratuvarlarından (ETL), Nippon Telefon'dan ve Kamu Telgraf Şirketinden (NTT) gelen kırk üst düzey araştırmacı ve sekiz bilgisayar imalatçısıyla açılmıştı". Projenin planı on yıllıktı: başlangıç araştırmalarına üç yıl, ara alt sistemleri kurmaya dört yıl, prototipi tamamlamaya son üç yıl ayrılmıştı. 1993'te proje, BNBS teknolojisini yaymak için iki sene uzatıldı.

2 *A.g.e.*



Resim 22.2. Beşinci nesil sistem mimarisi. (Çizim, Ed Feigenbaum'un izniyle kullanılmıştır.)

Japon bilgisayar bilimcisi Koichi Furukawa (1942- ; Resim 22.1), ICOT'un, beşinci nesil makinelerinde temel dil olarak PROLOG'u kullanma kararında etkili olmuştur. Furukawa, 1970'lerde SRI'da geçirdiği bir yıl esnasında, Harry Barlow'dan ve başkalarından PROLOG'u öğrenmişti. Bu dilden etkilenen Furukawa, Japonya'ya dönerken yanında, Alain Colmeraurer'nin bu dil için yazdığı çeviri programıyla (FORTRAN'da yazılmıştı) birlikte PROLOG'u da getirmişti. İleride ICOT'a katılıp müdür yardımcısı olacaktı. (Şimdi Keio Üniversitesi'nde fahri profesördür.)

Beşinci nesil sistem planının mimarisi Resim 22.2'de gösteriliyor. Bilgi tabanı, çıkarsama ve ara yüz işlevlerini gerçekleştirecek çeşitli

donanım modülleri, ileri çip teknolojileri kullanılarak hayata geçirilecekti. Donanım, buna uygun yazılım modülleriyle denetlenecekti ve sistemle etkileşim kurmak, konuşma, doğal dil ve resimler üzerinden olacaktı.

Bay Shunichi Uchida'nın, BNBS projesini özetleyen slaytlarına göre,[3] projenin on yıllık toplam bütçesi 54,2 milyar yen, yani yaklaşık 380 milyon dolardı (1990 kuruna göre).

Bu dönemde proje, paralel işleme, bilgisayar mimarisi ve çeşitli YZ sistemleri geliştirmede ilerlemeler kaydetti. Kimi Amerikalı ve Avrupalı ziyaretçiler (özellikle PROLOG uzmanları), bu projeye ICOT konuğu olarak katılmıştır. İşin aslı, Japonlar projeye uluslararası katılımı davet etmişti. 1984, 1988, 1992 yıllarında Tokyo'da uluslararası konferanslar düzenlenmiştir.[4]

ICOT, bir dizi "paralel çıkarsama makinesi" (PIM) inşa etmişti. Bunlardan en büyüğü olan PIM/p'nin 512 işlemci birimi vardı[5] (bkz. Resim 22.3). Bu makinelerde çalışacak birkaç YZ sistemi geliştirilmişti. Bunlar arasında paralel teorem ispatlayıcısı MGTP (Model Generation Theorem Power),[6] otomatik program yazma sistemi MENDELS ZONE,[7] hukuksal uslamlama uzman sistemi HELIC-II[8] vardı.

3 Shunichi Uchida, "FGCS Project: Knowledge Information Processing by Highly Parallel Processing", Yeni Nesil Bilgisayar Teknolojisi Enstitüsü (ICOT), Tokyo, Japonya, tarih belirtilmemiş. Bkz. http://www.icot.or.jp/ARCHIVE/PICS/OHP/Uchil-FGohpE.pdf.

4 ICOT Personeli (ed.) *Proceedings of the International Conference on Fifth Generation Computer Systems,* Haziran 1-5, 1992, Tokyo, Japonya: IOS Press, 1992; Yeni Nesil Bilgisayar Teknolojisi Enstitüsü (hazırlayan: ICOT), *Proceedings of the International Conference on Fifth Generation Computer Systems,* 28 Kasım-2 Aralık, 1988, 3 cilt, Tokyo, Japonya: OHMSHA, Ltd. ve Berlin: Springer-Verlag, 1988; Institute for New Generation Computer Technology (ICOT, hazırlayan), *Proceedings of the International Conference on Fifth Generation Computer Systems,* 6-9 Kasım, 1984, Tokyo, Japonya: OHMSHA, Ltd. ve Amsterdam: North-Holland, 1984.

5 Çeşitli PIM'leri betimleyen bir internet sitesi için bkz. http://www.icot.or.jp/ARCHIVE/Museum/MACHINE/pim-spec-E.html.

6 Örneğin bkz. Ryuzo Hasegawa, Miyuki Koshimura ve Hiroshi Fujita, "MGTP: A Parallel Theorem Prover Based on Lazy Model Generation", *Automated Deduction–CADE-11,* Bilgisayar Bilimi seminer notları, 11. Uluslararası Otomatik Çıkarım Konferansı Raporu, Berlin/Heidelberg: Springer-Verlag, 1992.

7 Örneğin bkz. Shinichi Honiden, Akihiko Ohsuga ve Naoshi Uchihira, "MENDELS ZONE: A Parallel Program Development System Based on Formal Specifications", *Information and Software Technology,* c. 38, no: 3, Mart 1996, s. 181-189.

8 Örneğin bkz. Katsumi Nitta ve diğ., "HELIC-II: Legal Reasoning System on the Parallel Inference Machine", *New Generation Computing,* c. 11, no: 3-4, Temmuz



Resim 22.3 PIM/p paralel bilgisayar sistemi. (Fotoğraf: http://www.icot.or.jp/ARCHIVE/Museum/MACHINE/pim-spec-E.html.)

Pek çok gözlemci, BNBS projesinden çıkan sonuçların çoğunun, ancak tarihsel bakımdan ilgi görebileceğini düşünür. Projede geliştirilmiş yazılım, önemli uygulama alanları bulamamıştır. Ticari iş istasyonlarının (hatta kişisel bilgisayarların) hızında ve gücündeki iyileşmeler, bunları PIM'ler karşısında üstün kılmıştı. Paralel işlemenin faydalarından bütünüyle yararlanmanın, bu tür hesaplamalara açık özel problemler dışında zor olduğu anlaşılmıştır. 1980'lerin ikinci yarısı ve 1990'larda grafikli kullanıcı arayüzlerinin (GUI; graphical user interface) geliştirilmesi, insan-bilgisayar etkileşimi için kabul edilebilir yöntemler sağlamıştı; böylece, YZ-bağımlı doğal dil ve konuşma anlama sistemlerine duyulan gereksinim (en azından o dönem için) azalmıştı. Bu projenin miraslarından biri, *New Generation Computing* [Yeni Nesil Hesaplama] dergisi olmuştur; bu derginin genel yayın müdürlüğünü bir zamanlar Koichi Furukawa yapmıştı. BNBS projesinin "müze" nitelikli internet sitesi şu adrestedir: http://www.icot.or.jp/ARCHIVE/HomePage-E.html. Bu internet sayfası, birtakım ICOT yayınına, yazılımına ve başka bilgilere linkler içeriyor.

Yazılışına birtakım bilgili kişilerin katkıda bulunduğu 1993 tarihli bir makale, proje hakkında bilgi verir.[9] ICOT'ta belli bir zaman

1993, s. 423-448.

9 Kazuhiro Fuchi ve diğ., "Launching the New Era", *Communications of the ACM*, c.

geçirmiş olan Evan Tick, makaleye yaptığı katkıda şöyle diyordu:[10]

> ... BNBS projesinin, simge işleme sahasındaki akademik gelişmelere, özellikle mantıksal programlamanın hayata geçirilmesine ve kuramına, kısıtlı ve eşzamanlı dillere, çıkarımsal ve hedef yönelimli veritabanlarına yaptığı katkıya çok saygı duyuyorum. Benim ilgilendiğim paralel mantıksal programlama, mimari ve uygulama alanlarında ICOT'un büyük katkıları olmuştur, fakat belki de ileri teknoloji aktarımı ile temel araştırmanın karışık takvimi kötü bir fikirdi.
>
> Bu temel araştırma aynı zamanda, teorem kanıtlamadan biyolojik hesaplamalara varana dek çeşitli alanlarda bir dizi güçlü uygulamalar doğurdu. Geniş ölçekte bakılacak olursa, ALVEY, ECRC, ESPRIT, INRIA ve MCC gibi benzer uluslararası projeler doğuran bir araştırma olması bağlamında başarılıdır. Bu örgütlenmeler birbirlerinden bir şeyler öğrendi ve temel araştırmalardaki akademik yetkinlikleri sayesinde daha geniş kapsamlı başarılar yakaladılar. Bu anlamda, bilgisayar bilimi camiası "beşinci nesil" çabasına çok şey borçludur.

Proje boyunca kaydedilmiş ve kaydedilememiş gelişmelerden ayrı olarak, 1980 ve 1981'de bu projeyle ilgili duyurular, ABD ve Avrupa'da benzer projelerin başlatılmasını teşvik etti. 1980 güzünde birkaç bilgisayar bilimcisi arasında elden ele dolaşmış "Beşinci Nesil Bilgisayarlar Hakkında Ön Rapor" adlı bir belge, bu projeyle ilgili haberleri yaymıştır. Ayrıca, BNBS projesini duyurmak amacıyla Ekim 1981'de Tokyo'da uluslararası bir konferans düzenlenmiştir.[11]

22.2 JAPON PROJESİNİN KİMİ ETKİLERİ

22.2.1 Mikroelektronik ve Bilgisayar Teknolojisi Ortaklığı

UTSB'nin beşinci nesil bilgisayar sistemi hakkında planlarını duyurması ve ICOT'un kurulması, ABD ve Avrupa'da uyarı çanlarının çalınmasına yol açtı. Tüketicilere yönelik elektronik piyasasında ve imalat alanında Japon egemenliğinin gitgide arttığının çok iyi farkında olan Amerikan bilgisayar sanayisi, bilgisayar teknolojisindeki dünya önderliğinin aşınmasından endişelenmişti.

36. no: 3, Mart 1993, s. 49-100.

10 *A.g.e.*, s. 99.

11 Örneğin bkz. T. Motooka ve diğ., "Challenge for Knowledge Information Processing Systems (Preliminary Report on FGCS)", *Proceedings of the International Conference on FGCS*, JIPDEC, 1981, s. 1-85.

Control Data şirketinin kurucusu William Norris, Florida'nın Orlando kentinde 1982 yılının Şubat ayında bilgisayar sanayisinin yöneticilerini, bir araştırma-geliştirme konsorsiyumu kurulması hakkında konuşmak üzere bir araya getirdi. Bu konsorsiyumun hedefi, üye şirketlerin en sonunda ürünlerinde kullanabileceği teknolojiler geliştirmek olacaktı. Bu toplantı, kâr amacı gütmeyen Mikroelektronik ve Bilgisayar Teknolojisi Ortaklığının (MCC) 1982'nin sonunda ve 1983'ün başında Texas'ın Austin kentinde kurulmasını getirdi. Ulusal Güvenlik Dairesinin eski müdürü ve CIA'nın müdür yardımcısı Amiral Bobby Ray Inman, bu kurumun ilk Başkanı, Kürsü Başkanı ve İcra Kurulu Başkanı seçildi. Konsorsiyuma ilk katılan şirketler arasında Digital Equipment Corporation, Harris, Control Data, Sperry-Univac, RCA, NCR, Honeywell, National Semiconductor, Advanced Micro Devices ve Motorola vardı. Bunlara ileride Microsoft, Boeing, GE, Lockheed, Martin Marietta, Westinghouse, 3M, Rockwell ve Kodak gibi başka şirketler de eklenecekti.

Yıllık bütçenin, ödenek ve araştırma personeli katkısı yapacak üye şirketlerin sayısına bağlı olarak 50 ile 100 milyon dolar arasında olması planlanmıştı. MCC, başlangıcında ana araştırma alanlarına odaklanmıştı, yani ileri bilgisayar mimarileri, yazılım teknolojisi, mikroelektronik paketleme, VLSI devrelerinin bilgisayar-destekli tasarımı gibi. YZ araştırması, mimari araştırmalarının bir parçası olarak yürütülecekti (son söz hakkına sahip olan Woodrow Bledsoe'nun yönlendirmesiyle).

Her ne kadar üye şirketler, MCC-destekli kimi yeniliklerden yararlanmış olsa da, 1987'den Inman'ın ayrılışından sonra MCC düşüşe geçti. O dönemde BNBS, artık tehdit olarak algılanmıyordu ve pek çok üye şirket, mali zorluklar yaşıyordu. Ayrıca İnternet ve kişisel bilgisayarların patlayan büyümesi ve gücü, MCC'de yapılan işleri gölgede bırakır olmuştu. Çalışan sayısı, 1985'te zirveye çıktığı 400'ten, yönetim kurulunun konsorsiyumu dağıtmayı oyladığı Haziran 2000'de 58'e düşmüştü.[12]

22.2.2 Alvey Programı

1982 yılının Mart ayında Britanya hükümeti, "bilgi teknolojisi (IT) alanında ortak bir araştırma programının ölçeği hakkında danışmanlık yapıp tavsiyeler vermesi amacıyla" bir komite kurdu. Komiteye,

12 MCC'nin yaklaşık ilk on yılının tarihçesi için bkz. David V. Gibson ve Everett M. Rogers, *R & D Collaboration on Trial: The Microelectronics and Computer Technology Corporation*, Cambridge, MA: Harvard Business School Press, 1994.

British Telecommunications'ın kıdemli genel müdürü John Alvey başkanlık edecekti. O yıl içinde, "İleri Bilgi Teknolojisi için bir Program" başlığıyla yayımlanan raporunda bu komite, Japonya'nın BNBS projesinin "büyük bir rekabet tehdidi" olarak görüldüğünü belirtmiş, ABD'nin öngörülen tepkisinin, "Britanya sanayisi için buna eş belki daha da büyük bir rekabet doğuracağını" eklemişti.[13] Raporda şu tavsiye edilmişti: "Sanayi, akademi işkolu ve öbür araştırma kuruluşları arasında, devletin arka çıkmasıyla kurulacak işbirliği sayesinde Britanya'nın IT teknik gücünü seferber edecek beş yıllık bir program oluşturulmalı. Hedef, IT dünya pazarında Britanya'ya gelecekte rekabet gücü kazandırması bakımından yaşamsal önem taşıyan çekirdek teknolojilerde, Britanya'ya güçlü bir yetenek edindirmek olmalı". Destek için belirlenen dört ana teknik alan, "Yazılım Mühendisliği, İnsan Makina Arayüzleri (MMI), Zeki Bilgi Tabanlı Sistemler (IKBS) ve Çok Büyük Ölçekli Entegrasyon (VLSI)" olacaktı. Tavsiye edilen bütçe 350 milyon dolardı; bu paranın üçte ikisini devletin, geri kalanını sanayinin vermesi öngörülmüştü.

Britanya hükümeti, komitenin raporunu 1983'te kabul etti ve komitenin tavsiyelerini hayata geçirmek amacıyla "Alvey programını" başlattı. Ticaret ve Sanayi Bakanlığında (TSB), bu programın eşgüdümünü sağlaması amacıyla, Bilim ve Mühendislik Araştırma Konseyi (SERC; Science and Engineering Research Council) Genel Sekreteri Brian Oakley'in başkanlık ettiği yeni bir yönetim kurulu oluşturuldu. Parasal destek ve ödenekleri, TSB, Savunma Bakanlığı (SB), SERC ve sanayi sağlayacaktı. Alvey programı, birtakım başarılarının yanı sıra, Britanya'da YZ araştırmalarının canlanmasına da yardımcı oldu. Oakley'e göre, "eğer 1970'lerin başındaki Lighthill Raporu YZ topluluğu için cennetten kovulmak anlamına gelmişse, 1980'lerin başındaki Alvey Raporu da cennete yeniden kabul edilmek anlamına gelmiştir."[14]

Bu program, 1987'de yaklaşık 45 milyon dolarlık ödenek zirve noktasına ulaştı ve 1991'e dek sürdü. Bu programın, hem akademi hem de sanayide araştırma-geliştirme çabalarının kapsamını genişleterek Britanya'nın bilgisayar bilimi camiasına güç verdiği söylenir. Brian Oakley ve Kenneth Owen, 1990'da yayımlanmış, programla ilgili mükemmel özetlerinde, YZ, paralel mimari, VLSI, entegre devre CAD,

13 Bu alıntıların yapıldığı komite raporu şu internet adresinde: http://www.chiltoncomputing.org.uk/inf/literature/reports/alvey-report/p001.htm.

14 Brian W. Oakley, "Intelligent Knowledge-Based Systems – AI in the U.K.", şu yayında: Ray Kurzweil, The Age of Intelligent Machines, Cambridge, MA: MIT Press, 1990. Şu internet adresinde: http://www.kurzweilai.net/articles/art0308.html?printable=1.

yazılım mühendisliği ve konuşma teknolojisi alanlarında Alvey'in katkılarını betimler.[15]

22.2.3 ESPRIT

1983'te Avrupa Ekonomik Topluluğu (Avrupa Birliği'nin öncülü), ESPRIT (European Strategic Program of Research in Information Technology [Bilgi Teknolojisi Avrupa Stratejik Araştırma Programı]) programını başlattı. ESPRIT'in YZ yönleri hakkında makale yazmış olan Luc Steels ile Brice Lepape'ye göre, bunun amacı, "Avrupa Topluluğunun (AT) bütününde sanayi, araştırma kuruluşları ve akademik kurumlar arasında ulus ötesi ortak bir araştırma teşvik etmekti."[16] Aynı zamanda bu, Japonya'daki BNBS programına Avrupalıların yanıtıydı.

ESPRIT, mikroelektronik, enformasyon işleme sistemleri (yazılım ve ileri enformasyon işleme dahil) ve uygulamalar (bilgisayarlı imalat ve ofis sistemleri dahil) adı altındaki üç ana kategoride araştırmaları desteklemek üzere kurulmuştu. Çoğu YZ araştırmasının destek bulacağı bilgi işleme kategorisiyse, bilgi mühendisliği, ileri mimariler (simgesel işlemeye yönelik bilgisayar mimarileri dahil) ve ileri sistem arayüzleri (konuşma, görüntü ve çoklu sensör uygulamaları) alanlarına bölünmüştü.

ESPRIT projesinin on yıl sürmesi, ESPRIT-I ve ESPRIT-II olarak iki aşamaya ayrılması öngörülmüştü. (Daha sonra üçüncü bir aşama da eklenmiştir.) ESPRIT-I'in ilk bütçesi 1,5 milyar ECU'ydu. (1999 yılının Ocak ayında ECU'nun yerini avro almıştır: bir ECU = bir avro.) Ödenekleri, AT ve proje katılımcıları eşit oranda sağlayacaktı. ESPRIT-II'nin bütçesiyse, ESPRIT-I bütçesinin iki katından fazlaydı. Luc Steels ile Brice Lepape'ye göre, 1993 yılına gelindiğinde bu programın, "tüm AT ve Avrupa Serbest Ticaret Anlaşması ülkelerinde ESPRIT projeleri üzerinde çalışan yaklaşık 1.500 kurumdan gelen 6.000'i aşkın bilimci ve mühendis vardı."

ESPRIT'in mali destek sağladığı projeler, program müdürlerince yukarıdan yönetilmek yerine, bireysel araştırmacıların ve kurumların verdiği önerilere göre şekilleniyordu. Bu proje önerilerini, dağınık bir uzmanlar ekibi gözden geçiriyordu. Bu program, "ulus ötesi işbirliği ağlarını", sanayi etkinliklerini ve kısa vadeli kazançları vurgulayan

15 Brian Oakley ve Kenneth Owen, *Alvey: Britain's Strategic Computing Initiative*, Cambridge, MA: MIT Press, 1990.

16 Luc Steels ve Brice Lepape, "Knowledge Engineering in ESPRIT", *IEEE Expert*, c. 8, no: 4, Ağustos 1993, s. 4-10.

teklifleri teşvik ediyordu. ESPRIT, Britanya'da kimi araştırmaları destekleyerek Alvey'le işbirliği de yapmıştır.[17]

ESPRIT, YZ bağlantılı bazı projeleri destekledi. Bunlar arasında, çeşitli bilgi tabanlı sistemler, mantıksal programlama ortamları, doğal dil çözümleme sistemleri, bilgi elde eden ve öğrenen sistemler için önerilmiş projeler vardı. Bir örnek olarak, "Yapay Öğrenme Alet Kutusunu" (YÖA) anabilirim. Bu, yapay öğrenme tekniklerinden oluşan bir paketti ve geliştiriciler, belli görevlere uygun algoritmaları seçip birleştirebiliyordu. Bu sistemin geliştirilmesinde ortak çalışan ekipler Fransa, Birleşik Krallık, Almanya, Yunanistan ve Portekiz merkezliydi. Steels ile Lepape'nin ortak makalesi, ESPRIT'in desteklemiş olduğu YZ çalışmalarının tam bir özetini sunuyor.[18] ESPRIT'in kaydettiği gelişmeler, sanayinin YZ gönülsüzlüğünü kırmaya kısmen yardımcı olmuştur.

YZ'deki ulusal çalışmalar konusunu ele alırken, Alman Yapay Zekâ Araştırma Merkezi'ni de anacağım (DFKI: Deutsches Forschungszentrum für Künstliche Intelligenz). 1988'de kurulmuş olan bu merkez YZ'nin tüm alanlarında araştırma yürütmeyi sürdürüyor. Bununla ilgili daha fazla bilgiyi internet sayfasında bulabilirsiniz: http://www.dfki.de/web/welcome?set_language=en&cl=en.

ABD'de ise, Alvey ve ESPRIT'e benzer bir DARPA programı 1980'lerin ilk yarısında başlatıldı. Bu program kısmen, JAPON BNBS projesine bir yanıttı; fakat aynı zamanda, iletişim teknolojisinde ve bilgisayar donanımı ile yazılımı alanlarında süregiden teknik gelişmelerden yararlanacak büyük bir programın zamanının geldiği gözlemine de çok şey borçludur. DARPA'nın bu programını bir sonraki bölümde anlatacağım.

17 Bu metinde, ESPRIT'le ilgili bol bol internet sayfası var. Bkz. http://cordis.europa.eu/esprit/home.html.

18 Luc Steels ve Brice Lepape, *a.g.e.*

23
DARPA'NIN STRATEJİK HESAPLAMA PROGRAMI

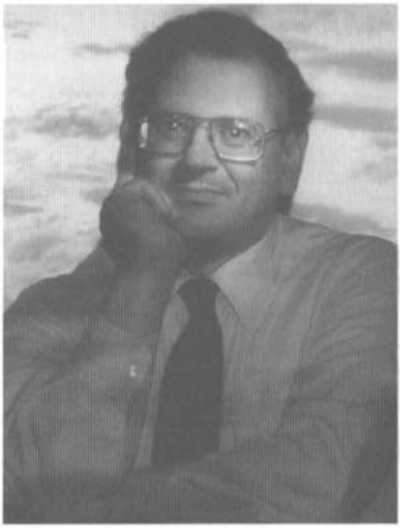
23.1 STRATEJİK HESAPLAMA PLANI

1980'lerin ilk yarısında uzman sistemler, ayrıca görüntü ve konuşma anlama ve doğal dil işleme gibi YZ teknolojileri çok umut vaat ediyordu. Aynı zamanda iletişim teknolojisi, bilgisayar ağları ve mimarilerinde, bilgisayar bellek ve işleme teknolojilerinde çarpıcı ilerlemeler de vardı. 1979'da DARPA'nın Bilgi İşleme Teknikleri Bürosunun (IPTO) genel müdürlüğüne getirilen Robert Kahn (1938- ; Resim 23.1), DARPA'nın, çok daha güçlü bilgisayar sistemleri yaratmak üzere tüm bu alanlardaki çabaları bütünleştirecek büyük bir araştırma-geliştirme programına hamilik etmesi gerektiğini düşünmeye başlamıştı. Aynı zamanda, Japonların BNBS programının, bilgisayar teknolojisinde ABD önderliğini tehdit etmesinden endişe duyuluyordu. Arkaplanda bu etkenlerin varlığıyla harekete geçen Kahn, "Stratejik Hesaplama" (SH) [Strategic Computing] olarak adlandırılacak programı planlamaya koyuldu.

Kahn, 1972'nin ikinci yarısında DARPA IPTO'ya yönetici olarak girmeden önce MIT'de profesör, BBN'de mühendisti. Burada, DARPA'nın ağ programını başlatıp yürüterek, günümüzün İnternetinin ilk sürümünü oluşturmak üzere Arpanet'le Packet Radio ve Packet Satellite ağlarını birleştirmişti. O zamanlar Stanford'da olan Vinton Cerf'le işbirliği yaparak, İnternetin ve "iletim denetimi protokolü"nün (İDP) [Transmission Control Protocol (TCP)] temel mimarisi olacak şeyi geliştirdiler. (İDP ileride birimlere ayrıştırılıp İDP/İP [TCP/IP] olacaktı; İP'nin anlamı, İnternet Protokolüdür.) Cerf 1976'da DARPA'ya katıldı ve 1982'ye dek orada ağ programını yönetti. Kahn ve Cerf, ortak çalışmalarından ötürü 2004 yılında ACM Turing Ödülünü paylaştı.[1]

Kahn, YZ'nin, özellikle de uzman sistemlerin SB'de önemli rol oynayabileceğini düşünüyordu. Unutmayalım ki 1970'lerin ortasında DARPA'nın YZ'ye verdiği destek, George Heilmeier'ın DARPA genel müdürüyken sergilediği tutum sırasında aksamıştı. Büyük zaiyatlardan biri, konuşma anlama programıydı. SH programı, YZ araştırmalarını

1 İnternetin evrimi hakkında daha fazlasını öğrenmek için bkz. Robert E. Kahn ve Vinton G Cerf, "What Is the Internet (And What Makes it Work)" şu internet adresinde: www.cnri.reston.va.us/what_is_internet.htm1.



Resim 23.1. Robert E. Kahn (Fotoğraf, Robert E. Kahn'ın izniyle.)

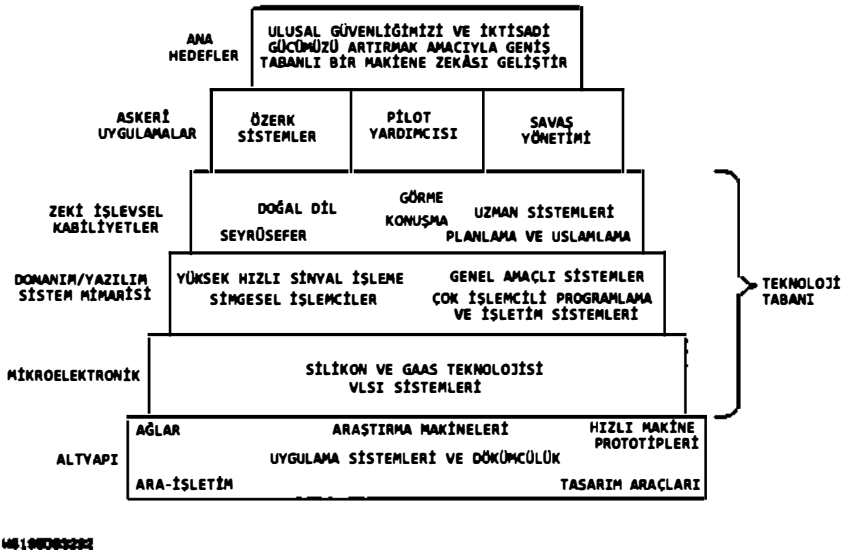
canlandırabilirdi, fakat Kahn'ın görüşü uyarınca daha önemlisi, umut vaat eden YZ tekniklerinin üniversiteden çıkıp gerçek uygulamalara aktarılmasını kolaylaştırabilirdi. SH programının tarihi hakkında epey araştırma yaparak yazdığı kitabında Alex Roland, bunu şu şekilde belirtiyor:[2]

> 1983'te [uzman sistemlerin yükselişinden sonra] Robert Kahn ve SH mimarları, YZ'den yararlanmanın zamanı geldiğine inanıyordu. YZ, sonunda laboratuvardan çıkıp gerçek dünyaya giriyordu... YZ, SH'nin temel bileşeni olacaktı; uzman sistemlerse en önemli öğe olacaktı. Bu sistemler, makinelerin "düşünmesine" olanak tanıyacaktı.

Kahn, SH programını, geliştirilecek ilintili teknolojilerin bir piramidi olarak görüyordu. Tabanda, hızlı tasarım olanakları ve gerekli donanım uygulamaları gibi yetkinleştirici teknolojiler vardı. Bunun üzerinde, YZ'nin öne çıktığı donanım ve yazılım teknolojileri bulunuyordu. Bunların hepsi, robot taşıtlar ve askerî savaş yönetimi yardımcıları gibi belli askerî sistemlerde bir araya gelecekti. Bu piramidin (pek çok) sürümlerinden biri Resim 23.2'de gösteriliyor.

SH programı, bu piramitteki tüm teknolojiler için araştırma-geliştirme çabalarını destekleyecek, eşgüdümleyecek ve idare edecekti.

2 Alex Roland (Philip Shiman'la birlikte), *Strategic Computing: DARPA and the Quest for Machine Intelligence, 1983-1993*, Cambridge, MA: MIT Press, 2002, s. 191-192.



Resim 23.2. SH programının yapısı ve hedefi. (Çizim, DARPA'nın izniyle kullanılmıştır.)

Milyar dolarlık bir program olacaktı bu, yani o zamana dek ABD devletinin üstlendiği en büyük bilgisayar araştırma-geliştirme programıydı.

Kahn'ın patronu, 1981 yılının Temmuz ayında DARPA genel müdürü olmuş Robert Cooper'dı. Cooper, Kahn'ın SH programı hakkında fikirlerinden heyecan duyuyordu, gerçi araştırma stratejisi ve programın tarifi konularında Kahn'dan ayrılıyordu. Alex Roland'ın yazdığı gibi:[3]

> Kahn, bu teknoloji tabanını kurduğunuzda güzel şeyler olur diyordu. Cooper, pilot yardımcısı projesine odaklandığınızda teknoloji tabanı bunu izler diyordu. Paradigmalardan biri, teknolojiyi arkadan ittirmek, öbürü ise teknolojiyi çekiş gücü olarak kullanmaktı. Biri yukarıya, öteki aşağıya... Bunlar arasındaki gerilim, tarihinin büyük kısmında SH'de sıkıntı yaratmıştır.

SH programını, boyutundan ötürü, DARPA'nın bütçesini denetleyen Kongre komitelerine "satmak" gerekiyordu. Cooper, bunu Kongreye satma yolunun, askeriyeye özgü uygulamaları vurgulamaktan (aslında vaat etmekten) geçtiğini biliyordu. Cooper, kongreyi ikna etme konusunda haklıydı ama en sonunda teknolojiyi çekiş gücü olarak kullanma stratejisi o kadar işe yaramamıştır.

3 *A.g.e.*, s. 71.

SH programının gerekliliğine Kongrenin ikna edilmesini kolaylaştıran etkenlerden biri, Japonların BNBS programıydı. Roland'a göre, "Kongreyi, Reagan yönetiminden de bilgisayar camiasından da [Kahn ve Cooper dahil] çok Japonların Beşinci Nesil programı etkilemiştir."[4] Edward Feigenbaum ile Pamela McCorduck'un Japon projesi hakkında kitap yayımlamış olması,[5] bu endişeleri pekiştirmişti. Kitaplarının önsözünde şunu soruyorlardı: "Bu önemli mücadeleye girecek miyiz? Girmeyeceksek, ulusumuzu, ilk büyük sanayi sonrası tarım toplumu saysak yeridir". Şu uyarıda da bulunmuşlardı:[6]

> ... iktisadi güvenliğimiz bir yana, ulusal çıkarımız, Japon projesini görmezden gelmemize olanak tanımıyor. Bilgi işleme, ABD'de yıllık 88 milyar dolarlık bir sanayidir ve bunun kaybı felaket olur. Onlarca yıldır dünyanın bir numarası olan bu Amerikan sanayisinde oluşacak kusur, ölümcül bir iktisadi yara demektir... Genelde savaşları üstün teknoloji kazanır, bu savaş ister askerî, ister ticari, ister kültürel olsun.

Haziran 1983'te Feigenbaum, Bilim, Uzay ve Teknoloji Meclis Komitesi önünde ifade verdi. Roland'a göre Feigenbaum komiteye şunu demiştir: "Akıl yürüten makineler çağı kaçınılmaz bir gelişme olacak... Bilişimin bariz kaderidir bu."[7]

Kahn, planın nasıl çerçeveleneceği konusunda Cooper'ın görüşüne boyun eğmeye ikna edilmişti ve plan en sonunda Ekim 1983'te kâğıda döküldü.[8] SH'yi destekleyecek ödeneklerin, 1984 mali yılında başlayacak çalışmalar için 50 milyon dolar düzeyinde olması kararlaştırılmıştı. (Kongre üyelerinden biri, DARPA'nın "bundan çok daha fazlasını" harcamasını bile tavsiye etmişti.) 1983'ten 1993'e dek on yıllık dönemde DARPA, SH'ye bir milyar doları aşkın para harcamış-

4 *A.g.e.*, s. 91.

5 Edward Feigenbaum ve Pamela McCorduck, *The Fifth Generation: Japan's Computer Challenge to the World,* Boston, MA: Addison-Wesley Longman Publishing Co., Inc., 1983. Aynı yazarların aynı başlıkla bir makalesi şurada basılmıştır: *Creative Computing,* c. 10, no: 8, Ağustos 1984, s. 103, ve şu internet adresinde: http://www.atarimagazines.com/creative/v10n8/103_The_fifth_generation_Jap.php.

6 Edward Feigenbaum ve Pamela McCorduck, *a.g.e.*, s. 19-20.

7 Alex Roland, *a.g.e.*, s. 91-92. Roland şuraya atıfta bulunuyor: U.S. Congress, House, Committee on Science, Space, and Technology, *Japanese Technological Advances and Possible U.S. Responses Using Research Joint Ventures. Hearings,* 98. Kongre, 1. Oturum, 29-30 Haziran 1983, s. 116-143, s. 119'da.

8 "Strategic Computing: New-Generation Computing Technology: A Strategic Plan for Its Development and Applications to Critical Problems in Defense", Defense Advanced Research Projects Agency, Arlington, Virginia, 28 Ekim 1983.

tır.[9] Plan, başlıca iki itici gücü desteklemeyi öngörüyordu, yani özgül uygulamalar yaratacak büyük projeler ve o uygulamalar için gerekli olacak "teknoloji tabanını" geliştirecek temel araştırmalar. Önümüzdeki kısımlarda bu iki itici gücün de kimi yönlerini anlatıyorum.

23.2 ANA PROJELER

Programın başında üç ana uygulama seçilmişti. Üç askerî kurumu da programa "kazandırmak" amacıyla bu plana "Pilot Yardımcısı" (Hava Kuvvetleri için), uçak gemisi "Çarpışma İdare Sistemi" (Deniz Kuvvetleri için) ve "Özerk Kara Taşıtı" (Kara Kuvvetleri için) denmişti. Ancak, hepsini ortak hizmet desteği vermeye yüreklendirmek amacıyla, plan şu uyarıyı da içeriyordu:[10]

> Örneğin, kara taşıtı yerine özerk denizaltı taşıtıyla uğraşmak belki yeğlenebilir ve çarpışma idare sistemi, donanma uygulamalarından ziyade kara çarpışmaları için daha uygun bulunabilir.

Mark Stefik'in, bu programa dair mükemmel incelemesinde ve değerlendirmesinde yazmış olduğu gibi, "DARPA tüm kuvvet komutanlıklarına, bu uygulamaların kimi özelliklerinin her an kulvar değiştirebileceğini, dolayısıyla planın bütününü satın almalarının daha iyi olacağını söylüyordu."[11] Doğrusu, düşman hatlarının ardına akıllı bomba fırlatan "özerk hava taşıtı" gibi ek uygulamalar ileride hayata geçirilecekti.

23.2.1 Pilot Yardımcısı

Şubat 1986'da başlatılan Pilot Yardımcısı (PY) [Pilot's Associate] programı, savaş uçağı subayına yardım edecek etkileşimli bir bilgisayar sistemi geliştirmeyi hedeflemişti. Sistemle etkileşim, grafikli kullanıcı arayüzü, ses tanıma (gürültülü ve zor koşullarda çalışabilen) ve ses sentezi üzerinden olacaktı. Görev planlarını hazırlayıp gözden geçirecek, pilota hedefler hakkında tavsiyeler verecek, tehditlerin ciddiyetini değerlendirecek, uçaksavar füzelerine karşı kaçış manevralarında pilota yardımcı olacak (pilotu bayıltacak kadar hızlı manevralarda uçağı

9 Alex Roland, *a.g.e.*, s. 319.

10 "Strategic Computing: New-Generation Computing Technology: A Strategic Plan for Its Development and Applications to Critical Problems in Defense", Defense Advanced Research Projects Agency, Arlington, Virginia, 28 Ekim 1983.

11 Mark Stefik, "Strategic Computing at DARPA: An Overview and Assessment", *Communications of the ACM*, c. 28, no: 7, s. 690-704, Temmuz 1985.

otomatik uçurmak gibi) ve sıradan işleri üstlenecekti. Pilotun vereceği girdilere ek olarak bu sistem, seyir sistemi yardımlarından ve birtakım sensörlerden bilgi alacaktı. Tavsiyeler ve kararlar, işbirliği yapan uzman sistemlerini, otomatik planlama sistemlerini ve plan gerçekleştirme sistemlerini temel alacaktı.

1983 DARPA stratejik planının, 1989'a gelmeden ulaşılacağını öngördüğü teknik yetenekler arasında, 10.000-kurallı, gerçek-zamanlı uzman sistemler, saniyede 10^8 poligonlu canlı ekranlar, 200-sözcüklü, yüksek gürültülü ortamlarda konuşmacıdan bağımsız biçimde konuşma tanıma ve 1.000 sözcüklük dağarcığıyla konuşma çıktı sistemi vardı.[12] Elbette, uçağa uygun küçük boyutlu donanım da gerekecekti.

Bu programı, Ohio eyaletinin Dayton şehrindeki Wright-Patterson Hava Kuvvetleri Üssü'nün Hava-Elektronik Laboratuvarı (ileride Hava Kuvvetleri Araştırma Laboratuvarının bir parçası olacaktı) yönetmiştir.[13] (Beş adayın katıldığı) Ön keşif çalışmalarından sonra Lockheed Aeronautical Systems (ileride Lockheed Martin şirketinin parçası olacaktı) ve McDonnell Douglas (ileride Boeing'in parçası olacaktı) ekipleri, 1986'dan 1992'nin ortasına dek sürecek sözleşmelerle ödüllendirildi. DARPA'nın idare ettiği projelerde standart hale geldiği üzere, sistemin çalışan bir sunuşunun yapılması gerekiyordu. Bu projeyi anlatan web sitelerinden birine göre (en son 2004'te güncellenmiştir), Lockheed'in programı, "... kusursuz yapay zekâ ve destek karar sistemlerini hava çarpışmalarının karmaşık ve hızla değişen dünyasına... olağanüstü başarıyla uyarlamasından ötürü", Amerikan Havacılık ve Uzay Enstitüsü (AIAA) Sayısal Elektronik Ödülüne layık görülmüştür.[14] Aynı site, şunu da ileri sürer:

> 1991'de, teknik bakımdan PY, zamanının en ileri, işe yarar ve gerçek zamanlı zeki sistemiydi ve dünyada üstüne yoktu. Bizler [Lockheed'te], gerçek zamanda iş gören altı uzman sistemi, gerçekçi (fazlasıyla gerçekçi olduğunu söyleyenler de vardı) bir çarpışma benzetim programına başarıyla yerleştirmiştik. Bu sistemin her bileşenine yüklenmiş bilgi, günümüzün savaş uçaklarının işleyişine açıkça uygulanabilen gerçekçi bir çarpışma deneyimiydi.

12 "Strategic Computing: New-Generation Computing Technology: A Strategic Plan for Its Development and Applications to Critical Problems in Defense", Defense Advanced Research Projects Agency, Arlington, Virginia, 28 Ekim 1983.

13 Wright Patterson Hava Kuvvetleri Üssünden insanların programa dair betimlemesi ve incelemesi için bkz. Sheila B. Banks ve Carl S. Lizza, "Pilot's Associate: A Cooperative, Knowledge-Based System Application", *IEEE Expert*, c. 6, no: 3, 1991, s. 18-29.

14 http://www.dms489.com/PA/PA_index.html.

Başarılı gösterisine rağmen Savunma Bakanlığı, bu teknolojiyi avcı uçaklarına doğrudan yerleştirmeye yanaşmadı. Az önce bahsettiğimiz web sitelerinden birinin belirttiği gibi, "teknolojiyi piyasaya sürme işi PY çalışmasını üstlenenlere düşmüştü."[15] Pilot Yardımcısı programının miraslarından biri, Kara Kuvvetleri helikopterlerinde kullanım için geliştirilmiş Pervaneli Araç Pilotu Yardımcısı sistemidir. Hava trafiği idaresini içeren olası bir uygulama asla gerçekleşmemiştir, gerçi "[NASA araştırmalarının] sonuçları, PY teknolojisinin, işletim masraflarında hem havayollarına hem de Hava Trafik Kontrol merkezine büyük tasarruflar yapmayı mümkün kılması ve ek güvenlik sunması yanında, kara ve deniz üzerindeki hava sahasında serbest uçuşa olanak tanıyabileceğini göstermeye yetmişti."[16]

23.2.2 Savaş Yönetimi Sistemleri

1984'te DARPA, Filo Karargâhı Savaş Yönetimi Programına (FKÇİİ) [Fleet Command Center Battle Management Program] ödenek sağlamaya başladı. Analytics adlı şirket, FKÇİİ'nin mühendislik işlerini yüklenmişti. Bu şirket, programın idaresine, gelişmelerin test edilmesine ve konfigürasyonun idaresine destek veriyordu. Bu programı betimleyip değerlendirmeye alan 1990 tarihli bir makalede, Analytics mühendislerinden Rin Saunders, programın amacının, "ABD Büyük Okyanus Filosunun başkomutanına (BOFBK) Büyük Okyanus ve Hint Okyanusu bölgesindeki yaklaşık 300 geminin idaresini planlayıp gözlemede yardım edecek" bir sistem yaratmak olduğunu yazmıştı.[17] Saunders, bu sistemin, uzman sistemlerini işlevsel kullanıma dahil etme bakımından Stratejik Bilişim programları arasında en başarılısı olduğunu ve "kullanıcı topluluğu içinde en büyük görünürlüğe ve katılıma sahip olduğunu" söylüyordu.

FKÇİİ'de uzman sistemlerin büyük rol oynaması planlanmıştı. DARPA stratejik planı, gerçek zamanın beş katı sürede iş gören ve epey karmaşık bağlamlar içinde saniyede 10.000 kural işleyen sistemler öngörmüştü.[18] FKÇİİ, birbirleriyle yerel ağ üzerinden iletişim

15 http://www.dms489.com/PA/PA_index.html.

16 http://www.dms489.com/PA/PA_index.html.

17 Rin Saunders, "The Fleet Command Center Battle Management Project: Lessons Learned", *Proceedings of the IEEE Conference on Managing Expert System Program and Projects*, Eylül 1990, s. 51-60.

18 "Strategic Computing: New-Generation Computing Technology: A Strategic Plan for Its Development and Applications to Critical Problems in Defense", Ek I Tablo I.3, Defense Advanced Research Projects Agency, Arlington, Virginia, 28 Ekim 1983.

kuran iki büyük uzman sisteminden oluşuyordu. Bunlardan biri olan "Güç Gereksinimleri Uzman Sistemi (Force Requirements Expert System [FRESH])", filodaki gemilerin o anki konumlarını ve teyakkuz durumlarını izleyip belirgin değişiklikler olduğunda uyarmak üzere tasarlanmıştı. Texas Instruments'in geliştirmiş olduğu FRESH'in, örneğin o anki planla devam etmek, onarımları hızlandırmak ya da hazır olmayan gemilerin yerine başkalarını göndermek gibi, yapılması gereken işler hakkında önerilerde bulunması bekleniyordu.

Saunders şunu ileri sürmüştü:[19]

> Şu an için FRESH, on iki BFOBK kurmay koduna günlük uyarı özetleri temin ediyor. FRESH'e ve BFOBK kurmaylarına çözsünler diye aynı görevin verildiği denemede FRESH'in uzmanlarla aynı hükümlere vardığı kanıtlanmıştır. BFOBK kurmaylarının tahminine göre FRESH, eskiden iki saat süren gözleme işlerini bir dakikada bitirip 120 kat zaman tasarrufu sağlıyor. Etkisiz kalmış bir geminin yerine yenisini atama gibi planlama görevlerinde zaman tasarrufu bire dört yüzdür.

Başka bir uzman sistem olan "Yetenek Değerlendirme Uzman Sistemi" [Capabilities Assessment Expert System] (CASES), ABD ve düşman kuvvetleri hakkındaki bilgiyi kullanıp varsayımsal bir çarpışmada iki tarafın neler yapacağını tahmin etmeye çalışıyordu. Bu sistemi BBN geliştirmiştir. Saunders'e göre:[20]

> CASES, uçak gemisi filolarının görev sahasını irdelemek; hücum denizaltılarını konuşlandırma stratejilerini değerlendirmek; çıkarma öncesi keşfin ve nükleer denizaltıların düşman denizaltılarına erken zaiyat verdirmesinin etkilerini öngörmek ve tahmin edilen düşman yeteneğindeki ve havadaki değişikliklere göre farklı saldırı starejilerinin bedel ve kazançları hakkında içgörü sağlamak amacıyla kullanılmıştır.

İki uzman sistem de, doğal dil anlama ve oluşturma becerilerinden çok faydalanmıştır. Bu sistemler, Symbolics Lisp makinelerinde çalışıyordu ve ticari uzman sistem kabuklarını (*shells*) kullanılarak yazılmışlardı. CASES'daki çarpışma benzetimleri, Encore paralel işlemcilerle çalıştırılıyordu. FRESH'in ve CASES'in son prototipleri BFOBK'ye 1990'da gönderildi. Fakat prototip aşaması bitip de DARPA ödeneklerinin sonuna gelindiğinde, Deniz Kuvvetleri bu sistemlerle devam etmeme kararı aldı.

19 Rin Saunders, *a.g.e.*, s. 53.
20 *A.g.e.*, s. 53.

Şimdilerde Computer Sciences şirketinin teknik müdürü olan Saunders, bana FKÇİİ'yle ilgili bazı anılarını anlatmıştı. Bu anılar, 1990'da yazdıklarıyla bir parça çelişiyor. E-postayla bana şunu yazmıştı:[21]

> FKÇİİ'nin hedefleri, 1980'lerin teknik imkânlarını aşıyordu.
>
> ...
>
> Deniz Kuvvetlerinin FRESH'i rafa kaldırma kararının sebebi, bunu elde tutmayı zorlayan bir neden olmayışıydı. Filo planlamacılarının uzman görüşleriyle aynı sonuçlara, günler yerine saatler içinde ulaşıyordu. Ancak planlamacılar emekliye ayrılmak falan istemiyordu ve deniz savaşında onların planlama hızı gayet yeterliydi.
>
> CASES'e gelince, Filo, benzetim aleti olsun istiyordu. Fakat benzetimleri, farazi senaryoları değerlendirmek üzere zekice düzenleme hedefi asla harekete geçirilmemiştir.
>
> ...
>
> Bana kalırsa, DARPA/ISTO içinde, FRESH ve CASES teknolojileriyle ilgili araştırma gündeminin, çalışma olanaklarındaki değişikliklere uydurulmadığı gitgide anlaşılıyordu... Ayrıca, Kongre DARPA'ya, ödeneklerini o zamanlar en öne çıkan konu olan denizaltı karşıtı silahlara yönlendirmesi için baskı yapıyordu. Üstelik DARPA ile Deniz Kuvvetleri arasındaki ilişkiler daima gerilimliydi, kopmaya yakındı. Deniz Kuvvetleri, programın denetimini ele geçirmek için DARPA'yla sürekli boğuşuyordu, çünkü kısa vadeli kazanç elde etmek amacıyla programı kendi çıkarları uyarınca denetleyip yeniden tanımlamak istiyordu. En sonunda, herkes, eşyaları toplayıp eve dönme vaktinin geldiğine karar verdi.

23.2.3 Özerk Taşıtlar

DARPA'nın SH programı şemsiyesi altında mali destek verilmiş üçüncü büyük uygulama projesi, 1984 yılının Ağustos ayında başlatılmış olan "Özerk Kara Taşıtı" (Autonomous Land Vehicle: ALV) projesiydi. Martin Marietta (ileride Lockheed ile birleşip Lockheed Martin olacaktı), bu "projenin tümleştiricisi" olarak seçildi ve bu şirkete kırk iki aylık bir süre için 10,6 milyon dolar verildi.[22] SRI, Carnegie Mellon Üniversitesi, Maryland Üniversitesi, Hughes Araştırma Laboratuvarları, Advanced Decision Systems ve Michigan Çevresel Araştırmalar Enstitüsü (ERIM), tamamlayıcı parçaları ve araştırma yardımı sağladı. ABD Kara Kuvvetleri Mühendislik Topografya Laboratuvarı, bu çalışmanın eşgüdümlenmesine yardım etti. Projenin hedefi, Kara Kuvvetlerinin uzun vadeli "stratejik görüşüyle" aynı çizgideydi; ordu,

21 E-postayla yazışma, 12 Aralık 2007.
22 Alex Roland, *a.g.e.*, s. 222.

lojistik ve malzeme sevkinde, arama-kurtarma görevlerinde, hatta çarpışmada özerk taşıtlar kullanmak istiyordu.

Douglas W. Gage'in yazmış olduğu bir makalede ALV projesiyle ilgili epey bilgi bulmak mümkün. Makalenin özetini yorumlamak yerine doğrudan alıntı yapayım:[23]

> ALV, Standard Manufacturing'in, asfalt yolda 45 mil/saat, arazide 18 mil/saat hıza çıkabilen, sekiz tekerlekli, hidrostatik-sürüşlü basit-arazi taşıtı üzerine inşa edilmişti. ALV, tozsuz ve klimalı ortamda altı raf dolusu elektronik donanım taşıyabiliyor, 12 kW'lık dizel yedek güç birimlerinden enerji sağlayabiliyordu. İlk sensör takımı, renkli video kameradan ve 1-2 saniye aralıklarla 64 ile 256 arası piksel görüntü gönderen, Michigan Çevresel Araştırmalar Enstitüsü (ERIM) kökenli bir lazer tarayıcıdan oluşuyordu. Video ve menzil verilerini işleme modülleri, aracın önündeki manzaranın bir modelini oluşturmak amacıyla kullanılan yol-kenarı bilgisini üretiyordu. Üst düzey uslamlama işlemini, hedef takipçisi ve yön bulma modülleri yürütüyordu; bu modüller, gidilmesi gereken yolu, taşıta yön veren pilot modülüne aktarıyordu.

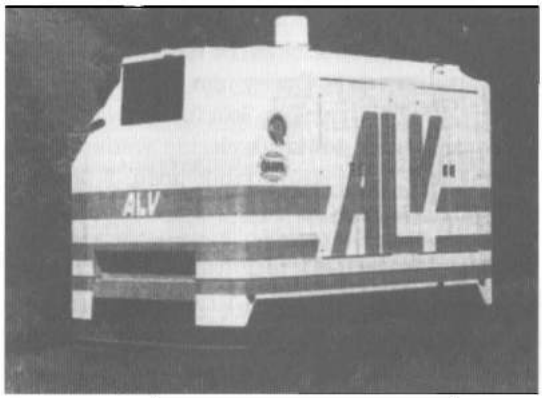
ALV'nin bir fotoğrafı ve sistem yapısı Resim 23.3'te gösteriliyor.

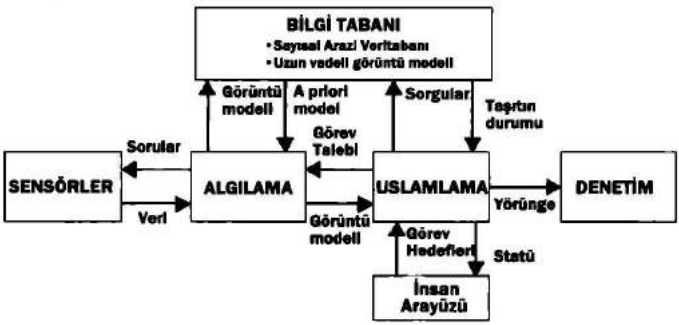
ALV, yollarda ve arazide yönlendirilmeden kendi başına giden, çevresi hakkında kendisini bilgilendirecek görme programları, ayrıca planlama programları ve rotalarını denetleyecek uzman sistemler kullanan askerî taşıtların öncüsü olacaktı. Bu araçların, başka taşıtlar, taşlar, ağaçlar, hendekler, su engelleri, dik ya da çamurlu arazi gibi tehlikelerden kaçınmaları gerekiyordu. Ayrıca, hemen çevrelerindeki yol işaretlerini ve belirgin nesneleri tespit edebilmeliydiler. Tüm hesaplamalar taşıt içinde yapılmalı, dolayısıyla taşıtlar, kendilerine yetebilmeliydi.

DARPA Stratejik Bilişim Planı, ALV için kimi belli kilometre taşları belirlemişti.[24] 1985'te, "yoldan gitme gösterisi" düzenlenecekti; bu testte ALV'nin, önceden belirlenmiş 20 km'lik bir rotayı 10 km/saat hızla aşması gerekiyordu. 1986'da, yol genişliğinden çok daha dar ve "100 metreden az aralıklarla yerleştirilmiş sabit, çok yüzlü nesneler" kullanılarak "engellerden kaçma gösterimi" düzenlenecekti. 1990'da

23 Douglas W. Gage, "UGV HISTORY 101: A Brief History of Unmanned Ground Vehicle (UGV) Development Efforts", *Unmanned Systems Magazine, Special Issue on Unmanned Ground Vehicles*, c. 13, no: 3, Yaz 1995.

24 "Strategic Computing: New-Generation Computing Technology: A Strategic Plan for Its Development and Applications to Critical Problems in Defense", Defense Advanced Research Projects Agency, Arlington, Virginia, 28 Ekim 1983. Bu paragrafta tarif edilen kilometre taşları ve başka kilometre taşları da Ek 1'deki Tablo 1.1'den alınmıştır.





Resim 23.3. Martin Marietta'nın ALV'si (üstte) ve sistem yapılanması (altta). (ALV fotoğrafı DARPA'nın izniyle; diyagram: R. Terry Dunlay, "Obstacle Avoidance Perception Processing for the Autonomous Land Vehicle", Proceedings of the IEEE Robotics and Automation Conference, Los Alamitos, CA: CS Press, 1988, s. 912-917.)

ve 1991'de, 90 km/saate varan hızlarla başka taşıtların da olduğu yollarda "yol ve arazi gösterimi" yapılması planlanmıştı.

İnsansız taşıtlar araştırmalarındaki ilerlemeyi değerlendiren bir Ulusal Araştırma Konseyi raporuna göre, "ALV 1985'te 1 km'lik yolu 3 km/saat hızla aştı... 1986'da, 4 km'lik yolu 10 km/saat hızla aştı. ALV 1987'de en yüksek hızına, 20 km/saate ulaştı... ayrıca yola konulmuş engellerden kaçmak için lazer tarayıcıdan faydalanıyordu."[25] Aynı rapor şöyle sürüyor:

> 1987 yılının Ağustos ayında ALV, sensör verilerine dayanarak ilk özerk arazi seyrini gerçekleştirdi. Bu ve yaklaşık bir yıl içindeki izleyen denemelerde ALV, birkaç kilometrelik yollarda, tek başına durup çıkıntı

25 National Research Council Staff, *Technology Development for Army Unmanned Ground Vehicles*, Washington, DC: National Academies Press, 2003, s. 152-153.

> yapan çeşitli engel türleri arasında yolunu buldu. Arazide dik eğimler (bazıları 15 dereceyi aşan), hendekler ve oluklar, büyük meşe makileri, çalılar ve kaya çıkıntıları vardı. Deney için insan yapımı nesneler kullanılmıştır. Kesin tespit edilebilecek en küçük engellerin yüksekliği 2 ila 3 ayaktı. Taşıt, yeri geldiğinde takım üyelerine yaklaşıp onları tespit edebiliyor, çarpmamak için manevra yapabiliyordu; 3,5 ile 5 km/saat arası hıza ulaşmış, yolların yaklaşık %75'ini tamamlamıştı.

ALV için görme, en zorlu sorunu oluşturuyordu. Kayaları, ağaçları, yolları ve hendekleri tanımak, 1980'lerin hem bilgisayarla görme algoritmalarının hem de bilgisayar işlem hızlarının sınırlarını zorluyordu. Roland'a göre, "DARPA, ALV görme sisteminin, saniyede 10 ile 100 milyar arası komut hızına gereksinim duyduğunu tahmin ediyordu, oysa 1983'te bu hız saniyede ancak 30 ile 40 milyon arası talimattı."[26] ALV'nin zamanında insansız taşıtlardaki algı sistemlerinin, taşıtın bilgisayar gücünün yaklaşık %85'ini kullandığı tahmin ediliyor.[27]

Roland, ALV görme sisteminin performansını yorumlarken şöyle yazmıştır:[28]

> Görme sisteminin, ışığın niteliği, güneşin konumu, gölgeler vesaire gibi çevresel şartlara karşı epey hassas olduğu anlaşılmıştı. Sistem, aydan aya, günden güne, hatta testten teste farklı çalışıyordu. Bazen yol kenarını doğru şekilde belirlerken, bazen de belirleyemiyordu. Sistem, yolun asfaltını yol sırtlarındaki topraktan kolaylıkla ayırt edebiliyordu, fakat çevresinde manevra yapan ağır taşıtlar yüzünden asfalta sıçrayan toprak ALV'yi yanıltıyordu. Güz mevsiminde, gökyüzünde daha alçakta duran güneş, bizzat asfalt yoldaki cilalı taşlardan harika biçimde yansıyor, taşıtın aklını karıştıran ışıltılı yansımalar oluşturuyordu. Ağaç gölgeleri sorun teşkil ediyordu, tıpkı sert Colorado havasının ve sekiz tonluk taşıtların sık sık zorunlu kıldığı yol onarımlarından kaynaklanan asfalt yamaları gibi.

DARPA, ALV programını Nisan 1988'de iptal etti, fakat özerk taşıtlara gösterilen ilgi, araştırma laboratuvarları, sanayi ve başka devlet kurumlarında sürdü. 1990'lar boyunca Savunma Bakanlığının desteklediği programlar arasında, uzaktan yönetilen taşıtlar için DEMO-I ve özerk taşıtlar için DEMO-II vardı. Ana yüklenici olarak yine Martin Marietta'nın Havacılık Bölümü seçilmişti. Alt sistem sağlayan eş-yükleniciler arasında CMU, Hughes Research Laboratuvarları,

26 Alex Roland, *a.g.e.*

27 National Research Council Staff, *Technology Development for Army Unmanned Ground Vehicles*, Washington, DC: National Academies Press, 2003, s. 152-153.

28 Alex Roland, *a.g.e.*

Advanced Decision Systems, SRI, Teleos, JPL, ayrıca Massachusetts ve Michigan Üniversiteleri vardı. Gösterimler için seçilen taşıt bir HMMWV'ydi (High-Mobility Multipurpose Wheeled Vehicle, yani humvee). Nitekim 1990'ların ortasında bir dizi gösteri düzenlendi.[29] Carnegie Mellon Üniversitesi, "Terragator" ve "Navlabs" serisi dahil bir dizi önemli insansız taşıt geliştirmişti. (Navlabs'tan ileride daha fazla bahsedeceğim.) ALV programı tüm hedeflerine ulaşamamış olsa da, özerk taşıtlar çağını başlattığı söylenebilir. Her geçen seneyle birlikte bu taşıtlar daha becerikli ve "özerk" oluyor.

23.3 YZ TEKNOLOJİ TABANI

SH programının ana odağı, yukarıda bahsedilen bir dizi uygulama olsa da, bu uygulamaların peşinden başarıyla gidebilmek için, dayandıkları teknolojide birtakım ilerlemeler gerekiyordu. Robert Cooper'ın SH hakkındaki görüşüne göre, bu uygulamalar, görme, robotik, uzman sistemler, konuşma tanıma ve doğal dil işleme teknolojisini "ileri çekecekti". Dolayısıyla, uygulamalarla ilgili ana sözleşmelere ek olarak bazı sözleşmeler, teknoloji geliştirme hedefliydi. Elbette, teknoloji geliştiricilerinin, bu teknolojileri ileri çeken uygulamalarla yakın ilişki içinde olması bekleniyordu.

23.3.1 Bilgisayarla Görme

Ron Ohlander ve (daha sonra) Robert Simpson Jr., SH günleri sırasında YZ araştırmalarından sorumlu DARPA IPTO yöneticileriydi. Daha önce "görüntü anlama" konusunda başlamış çalışmayı sürdüren DARPA, bilgisayarla görmenin dört ana alanına odaklanmıştı, yani görsel modelleme ve tanıma, dinamik görüntü ve devinim çözümlemesi,

29 DEMO-I ve DEMO-II teknolojisinin özetleri ve gösterimleri için bkz. Douglas W. Gage, "UGV HISTORY 101: A Brief History of Unmanned Ground Vehicle (UGV) Development Efforts", *Unmanned Systems Magazine, Special Issue on Unmanned Ground Vehicles,* c. 13, no: 3, Yaz 1995, ve National Research Council Staff, *Technology Development for Army Unmanned Ground Vehicles*, Washington, DC: National Academies Press, 2003, s. 148. İkinci kitap şu yayını tavsiye ediyor: Oscar Firschein ve Thomas Strat (ed.), *Reconnaissance, Surveillance, and Target Acquisition for the Unmanned Ground Vehicle: Providing Surveillance "Eyes" for an Autonomous Vehicle,* San Francisco, CA: Morgan Kaufmann Publishers, 1997.

görme-tabanlı engelden kaçınma ve yol bulma, paralel hesaplama mimarileri kullanan görme algoritmalarını gerçeklenmesi.[30]

Chuck Thorpe, Takeo Kanade ve CMU'da başka araştırmacılar, özellikle ALV'nin gereksinim duyduğu görme sistemlerine yoğunlaşmıştı.[31] Stratejik Bilişimi destekleyen öteki önemli bilgisayarla görme araştırmaları, Azriel Rosenfeld yönetiminde Maryland Üniversitesi'nde, Alan Hanson ile Edward Riseman yönetiminde Massachusetts Üniversitesi'nde, Martin Fischler yönetiminde SRI'da ve sanayinin laboratuvarlarında yapılıyordu.

Bilgisayarla görme teknolojisi, robotik alanındaki uygulamaların yanı sıra, kartografi ve fotoğraf yorumlama alanlarında uygulanmıştır. İlk fotoğraf-keşif sistemlerinden biri, CIA'in destekleyip Hughes Araştırma Laboratuvarları'nın geliştirdiği SCORPIUS'tu (Strategic Computing Object-Directed Reconnaissance Parallel-Processing Image-Understanding System). Bu sistemin, hava ve uydu fotoğraflarını tarayıp gemi, bina, uçak ve istenen başka nesneleri belirlemesi bekleniyordu. SRI'dan Martin Fischler'e göre bu proje, BBN'de geliştirilmiş yeni bir paralel-işlemli bilgisayar ("Butterfly Multiprocessor") kullanmayla ilişkili altyapı sorunlarına boğulmuştu.[32]

SH programının ardından hayata geçirilen başka fotoğraf-keşif projeleri daha başarılı olmuştur. DARPA ve CIA'in ortaklaşa desteklediği RADIUS (Research And Development for Image Understanding Systems), bu projelerden biriydi. SRI'da RADIUS projesi kapsamında geliştirilmiş görüntü işleme sistemleri, üç boyutlu arazi modellerine yer görüntüsü yerleştirebiliyordu (örneğin Yosemite Vadisinde "uçuş" benzetimi gibi). Aynı zamanda bu sistemler, fotoğraflarda binaların ve başka nesnelerin yerini belirlemede yardımcı oluyor, böylece fotoğrafı yorumlayan insanın işini kolaylaştırıyordu.[33]

30 Örneğin bkz. Robert L. Simpson Jr., "Computer Vision: An Overview", Konuk Editörün Sunuşu, *IEEE Expert*, s. 11-15, Ağustost 1991.

31 Örneğin bkz. Takeo Kanade, Chuck Thorpe ve William Whittaker, "Autonomous Land Vehicle Project at CMU", *Proceedings of the 1986 ACM Computer Conference*, Şubat 1986, s. 71-80 ve Yoshimasa Goto ve Anthony Stentz, "Mobile Robot Navigation: The CMU System", *IEEE Expert*, 1987, s. 44-54. (İkinci makale şu internet adresinde: http://www.ri.cmu.edu/pub_files/pub3/goto_y-1987_l/goto_y_1987_l.pdf.)

32 Kişisel iletişim, 15 Kasım 2007.

33 Bkz. Thomas M. Strat ve Oscar Firschein, *RADIUS: Image Understanding for Imagery Intelligence*, San Francisco: Morgan Kaufmann Publishers, 1997. Ayrıca bkz. RADIUS internet sitesi: http://www.ai.sri.com/~radius/.

23.3.2 Konuşma Tanıma ve Doğal Dil İşleme

Pilot Yardımcısı ve Savaş Yönetimi projeleri, bilgisayar sistemlerinin, sözlü talepleri ya da komutları anlama kabiliyetine bağlıydı. Pilot Yardımcısı projesi, konuşma tanıma özelliğine gereksinim duyuyordu ve Savaş Yönetimi projelerine, konuşma tanıma ve metin anlama özellikleri gerekliydi. Bu yüzden, o teknolojileri ileri taşımak amacıyla DARPA, bazı temel araştırma projelerine mali destek vermişti. DARPA'nın, kendi bünyesindeki konuşma anlama araştırma programını 1976'da kestiğini anımsayın, fakat bu sahada, özellikle CMU'da, MIT'de ve IBM'de yaşanan gelişmeler, bu geri çekilişi haklı çıkarıyor.

Texas Instruments, sekiz ayrı konuşma tanıma araştırma projesini Pilot Yardımcısı projesi için bütünleştirmişti. CMU ise Savaş Yönetimi projeleri için dokuz projenin çalışmalarını bütünleştirmişti. Roland'a göre, SH programının sonuna gelindiğinde, konuşma tanıma sistemleri, "orta düzeyde arka plan gürültüsü olan ve konuşmacıya az sıkıntı çıkaran bir ortamda herhangi biri tarafından söylenen 10.000 doğal dil sözcüğünü tanıyabiliyordu."[34]

Daha sonra CMU'da, Kai-Fu Lee ve arkadaşları, DARPA desteğiyle birlikte, konuşmacıdan bağımsız konuşma tanıma sistemi SPHINX'i geliştirdi.[35] SPHINX, tanımayı kolaylaştırması amacıyla, saklı Markov modellerinden ve sözcük zincirleri olabilirlikleri hakkında istatistiksel bilgiden faydalanıyordu.

Konuşma tanıma çalışmaları, bazı sanayi ve üniversite araştırma laboratuvarlarında sürüyor. Bir örnek vermek gerekirse, CMU'da SPHINX projesi, "konuşmadan faydalanan uygulamalar yaratmak isteyen herkese temel bir teknoloji düzeyi sağlayacak, makul oranda olgun [açık-kaynak] ve dünya çapında bir dizi konuşma bileşenlerini" oluşturmuştur. Bununla birlikte, projenin SourceForge'daki[36] internet sayfası şu uyarıda bulunuyor: "SPHINX, son ürün değildir. Belirli düzeyde uzmanlığı olan kişiler, burada sunulan SPHINX sürümleriyle harika sonuçlar elde edebilir, ama toy kullanıcının mutlaka yardıma ihtiyacı olacaktır. Başka bir deyişle, bu yazılım, konuşma alanında

34 Alex Roland, *a.g.e.* Roland, bu bilgi için, MIT'de konuşma araştırmacısı ve profesörü olan Victor Zue'ye atıfta bulunuyor.

35 Kai-Fu Lee, Hsiao-Wuen Hon ve Raj Reddy, "An Overview of the SPHINX Speech Recognition System", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, c. 38, no: 1, Ocak 1990. Şu internet adresinde: http://www.ricmu.edu/pub_files/pub2/lee_k_f_1990_1/lee_k_f.1990_1.pdf.

36 htrp://cmusphinx.sourceforge.net/html/cmusphinx.php.

deneyimi olmayan kullanıcılara değil, uzman kullanıcılara göredir". Bilgisayar hızı ve belleğindeki ilerlemeler sayesinde, yüksek kaliteli, gerçek zamanlı ve ticari olarak pazarlanabilir bazı konuşma sistemlerini makul masraflarla üretmek mümkün olmuştur.

SH programı aynı zamanda, DARPA'nın "yeni nesil sistemleri" dediği şeyin parçası olarak, metin anlamaya yönelik temel araştırmaları da desteklemiştir. 1984'te yedi yüklenici DARPA'yla sözleşme yaptı.[37] BBN Laboratuvarları, USC/Information Sciences Enstitüsü, Pennsylvania Üniversitesi ve Massachusetts Üniversitesi, klavyeyle yazılan sorulara yanıt verebilen doğal dil arayüzleri üzerinde çalışacaktı. New York Üniversitesi, Systems Development Corporation (ileride Unisys'in bünyesine girecekti) ve SRI International, askerî iletilerde serbest biçimli metinleri anlama üzerine çalışacaktı. BBN ve USC-ISI, aslında BBN'nin IRUS sistemini temel alan arayüz çalışmasını eşgüdümleyecekti; bu çalışma ileride büyüyüp JANUS olacaktı.[38] New York Üniversitesi ve SDC, metin anlama çalışmalarını birlikte yürütüp PROTEUS ve PUNDIT sistemlerini geliştirecekti.[39]

Roland'a göre bu çalışmanın sonuçları, "çok umut vaat ediyor, ilk SH planının belirlediği kilometre taşlarını aşıyordu... asıl planın belirlediği ölçü birimleri, hata hızında ve sözcük dağarcığı boyutunda aşılmıştı."[40] Roland şu ayrıntıları da verir:[41]

> ... BBN'nin IRUS sistemi, 4.500 sözcüklük dağarcığıyla ve 1.200 teknik kavramıyla birlikte, 1986'da Savaş Yönetimi Programının test cihazlarına kurulduğunda olumlu bir tepki olmuştu. 1987'ye gelindiğinde IRUS, artık, muğlak ve kafa karıştırıcı soruları anlamak üzere tasarlanmış JANUS'a dönüştürülüyordu. 1987 güzünde JANUS, Hava-Karacılık Savaş Yönetimi Programında başarıyla sunuldu.

37 37 *Proceedings of the Workshop on Strategic Computing Natural Language*, Robert Simpson'ın Önsözü, Morristown, NJ: Association for Computational Linguistics, 1986.

38 IRUS/JANUS çalışmasının (ayrıca BBN'deki ilintili çalışmaların) kısa bir tarihçesi için bkz. Ralph Weischedel, "Natural-Language Understanding at BBN", *IEEE Annals of the History of Computing*, Ocak-Mart 2006, s. 46-55. Bu projeye dair bir BBN raporu şu internet adresinde: http://www.aclweb.org/anthology-new/H/H86/H86-1001.pdf.

39 Şu çevrimiçi belgeye bakınız: Ralph Grishman ve Lynette Hirschman, "PROTEUS and PUNDIT: Research in Text Understanding", şu internet adresinde: http://www.aclweb.org/anthology-new/H/H86/H86-1002.pdf.

40 Alex Roland, *a.g.e.*, s. 212.

41 *A.g.e.*, s. 212.

Bu arada, 1986'da PROTEUS sisteminin ilk sürümü, donanım aksaklıklarıyla ilgili donanma zaiyat raporlarını başarıyla işlemeye başlamıştı.

23.3.3 Uzman Sistemleri

1970'lerin ikinci yarısı ve 1980'lerin ilk yarısında uzman sistemler çok umut vaat ederek ortaya çıktığı için, Stratejik Bilişim programında bunlara baş rollerden biri verilmişti. Bu sistemler, SH uygulamalarına "zekâ" katacak uslamlama etkenleri olacaktı. DARPA zeki sistemler yöneticisi Ronald Ohlander, 1984'te uzman sistemler için yüklenici-ler aramaya koyuldu. Ohlander, DARPA'ya gönderilen elli proje teklifi içinde altısına mali destek verilmesini tavsiye etti. En sonunda bu listeye yedinci bir proje de eklendi. Roland'ın bildirdiği gibi, Stanford, yeni uzman sistemler mimarisi üzerinde; BBN, gerekli bilgiyi uzman sistemine ulaştırma sorunu üzerinde; Ohio Devlet Üniversitesi, uzman sistemlerin kendi sonuçlarını açıklama yöntemleri üzerinde; ve Massachusetts Üniversitesi ile General Electric, muğlak bilgi üzerinden uslamlama teknikleri üzerinde çalışacaktı.[42]

Öbür proje yüklenicileri, hem Edward Feigenbaum hem de başka Stanford araştırmacıları tarafından kurulmuş yeni şirketler olacaktı. Bunlardan biri olan IntelliCorp (en başta IntelliGenetics olarak kurulmuştu), EMYCIN'den esinlenilmiş KEE (Knowledge Engineering Environment [Bilgi Mühendisliği Ortamı]) adlı bir uzman sistemi kabuğu zaten yaratmıştı.[43] IntelliCorp, yeni bilgi eklendiğinde ya da silindiğinde bilgi tabanının tutarlılığını sürdürme yeteneği, bilgi tabanındaki özgül bilgiler bunu sağlamasa bile makul çıkarımlar yapma yeteneği, plan ve takvim yapma gibi yetenekler dahil bazı özellikleri güncelleyerek KEE'nin kapsamını genişletmeyi önermişti. Ayrıca bilgi elde etme, kullanıcı arayüzü oluşturma ve nesneleri hiyerarşik olarak betimleme olanakları da olacaktı. Intellicorp 1984'te 1.286.781 dolar değerinde bir DARPA sözleşmesiyle ödüllendirildi.[44] En sonunda KEE, OPUS adlı sisteme evrimleşecekti.[45]

Yeni kurulmuş başka bir şirket, Teknowledge A.Ş., uzman sistemleri kurarken kullanılabilecek ABE adlı bir uzman sistem seti ya-

42 *A.g.e.*, s. 195.

43 T. P. Kehler ve G. D. Clemenson, "KEE – The Knowledge Engineering Environment for Industry", *Systems and Software*, c. 3, no: 1, Ocak 1984, s. 212-224.

44 Alex Roland, *a.g.e.* s. 198.

45 Richard Fikes ve diğ., "OPUS: A New Generation Knowledge Engineering Environment", Faz 1 Nihai Rapor, IntelliCorp, Mountain View, CA. 1987.

ratmayı önermişti. ABE yepyeni bir sistem olacaktı, dolayısıyla DARPA için bir tür kumar sayılırdı, fakat şirkette kadrolu ya da danışman olarak bir sürü YZ uzmanı çalışıyordu.[46] Teknowledge'e vaatlerini yerine getirmesi için 1.813.260 dolarlık DARPA sözleşmesi sunuldu.[47]

Hem KEE hem de ABE, SH uygulamaları proje yüklenicileri tarafından kullanılmıştır; fakat ikisi de, uzman sistemlerden beklenen zekâyı yaratamamıştır. Gelgelelim, bu beklentiler olasılıkla gerçekçi değildi. Uzman sistem uygulamaları öncülerinden Frederick Hayes-Roth ile Neil Jacobstein, bilgi-tabanlı sistemler (BTS) hakkında 1994 tarihli değerlendirmelerinde şunu yazmışlardır: "... her ne kadar uzman sistem ve BTS teknolojilerinin şu anki neslinin, insan benzeri sağlam ve genel bir zekâ yaratma umudu hiç olmamışsa da, bunu geliştiremediği için pek çok insan hayal kırıklığı yaşamıştır."[48] Yine de Hayes-Roth ve Jacobstein, aşağıdaki şu yargıya varırlar:[49]

> BTS, son on yıllık sanayi deneyimi üzerinde kayda değer oranda etkili olmuştur; çoğunlukla hızı, kaliteyi ya da fiyat performans oranını katbekat artırmıştır. Bu sistemler en büyük 500 şirketten küçük girişim firmalarına, askerî kurumlara, devlet kuruluşlarına, sağlık hizmetlerine ve eğitim kurumlarına varana dek her büyük kuruma nüfuz etmiştir. BTS uygulamaları, müzikten tıbba kadar fiilen her insan girişiminde bulunuyor artık.

DART (Dynamic Analysis and Replanning Tool) da bir BTS'ydi ve bu sistemin kökleri, SH programı sırasında yapılan YZ araştırmalarına kadar uzanır. BBN'de geliştirilmiş olan bu sistem, 1990 İran Körfezi Savaşı sırasında Avrupa'dan Suudi Arabistan'a donanım ve personel taşınmasına yardımcı olmuştur. İşin doğrusu, o zaman DARPA genel müdürü olan Victor Reis'ın, "DART zaman çizelgeleme uygulaması, DARPA'nın 30 yıldır YZ araştırmalarına yaptığı tüm yatırımı birkaç ayda geri ödedi" dediği söylenir.[50]

46 Lee D. Erman, Jay S. Lark ve Frederick Hayes-Roth, "ABE: An Environment for Engineering Intelligent Systems", *IEEE Transactions on Software Engineering*, c. 14, no: 12, Aralık 1988, s. 1758-1770.

47 Alex Roland, *a.g.e.*, s. 201.

48 Frederick Hayes-Roth ve Neil Jacobstein, "The State of Knowledge-Based Systems", *Communications of the ACM*, c. 37, no: 3, Mart 1994, s. 36.

49 *A.g.e.*, s. 36.

50 Sara Reese Hedberg, "DART: Revolutionizing Logistics Planning", *IEEE Intelligent Systems*, Mayıs/Haziran 2002, s. 81.

23.4 DEĞERLENDİRME

Peki, SH programı hakkında genel olarak ne söylenebilir? Askerî hedeflerine ulaşabileceğini düşünenler de bu hedeflere ulaşmamasında kusur bulanlar da bu programı eleştirmiştir. İlk kesim arasında yer alan Toplumsal Sorumluluk Sahibi Bilişim Profesyonelleri (CPSR) birliği, yayımladıkları mektuplarından birinde birtakım endişeler taşıyordu:[51]

1. SHİ [Stratejik Hesaplama İnisiyatifi] yeni özgül silah sistemleri, robot tanklar gibi özerk taşıtlar, savaş pilotu "yardımcısı" ve savaş uçağı filosu "savaş yönetimi sistemi" vaat ediyor. Endişemiz şu ki bilgisayar araştırmalarıyla ilgili proje teklifleri, genel bilimsel meziyetlerinden ziyade bu özgül uygulamalardaki geçerlilikleri bakımından değerlendirilecek.
2. SHİ, kestirilemez koşullar altında karmaşık askerî sistemlerin işleyişini denetlemek amacıyla yapay "zekâ" kullanımını teşvik ediyor. Endişemiz şudur ki, özellikle risk yüksek olduğunda, son derece belirsiz durumlar, yapay zekânın uygulanması bakımından kesinlikle yanlış ortamlardır.
3. SHİ, ülke savunmasıyla ilgili olduğu düşünülen sorunların çözümü olarak bilgisayar teknolojisinin askerî uygulamalarını destekliyor. Endişemiz şudur ki, geçmişte yeni silah teknolojilerinde üstünlük kurma girişimleri, güvenliğimizi artıracak yerde, sonu görünmeyen bir silahlanma yarışını hızlandırmıştır.

   Saydığımız son endişe en ağırıdır. Son tahlilde, ne ulusal güvenliğe giden yolun askerî üstünlükten geçtiğine, ne de bu üstünlüğün bilgisayar kullanımıyla elde edileceğine inanıyoruz.

Çoğu eleştirmen, SH programında, SH uygulamalarıyla yeni YZ teknolojileri doğurma hedefine ulaşıldığından kuşku duyar. SH uygulamalarının öngördüğü türde yetenekleri geliştirmek, birtakım YZ icatları gerektiriyordu ve icat için gerekli atmosfer, sıkı şekilde belirlenmiş kilometre taşı gösterimlerine nüfuz etmemiştir. Aksine, Roland'ın da belirttiği gibi, "uygulamalar, gösterim tarihlerine yetişmek için doğaçlama, satışa yönelik çözümler uydurmuştur."[52] Dahası, vitrine sürülen sistemler, yani ALV, Pilot Yardımcısı ve savaş yönetimi sistemleri, umulan askerî müşteriler tarafından hemen "satın alınmadı". Yine de

51 *The CPSR Newsletter*, c. 2, no: 2, Bahar 1984. Ayrıca bkz. S. M. Ornstein, B. C. Smith ve L. A. Suchman, "Strategic Computing: An Assessment", *Bulletin of the Atomic Scientists*, c. 40, no: 10, Aralık 1984, s. 11-15.

52 Alex Roland, *a.g.e.*, s. 243.

bu sistemler, çok daha fazla performans gösteren benzer sistemlerin öncüsüdür.

"YZ teknoloji tabanına" SH'nin verdiği cömert destek, genel olarak YZ araştırmalarını beslemiştir, gerçi bu araştırmalar, SH uygulamalarına eklenecek sonuçlar ortaya çıkaramamıştır. SH programının sonuna doğru kısa bir düşüşten sonra YZ araştırmaları, hem üniversitelerde hem de sanayide sürekli palazlanmıştır ve önemli yeni beceriler üretmeyi sürdürmektedir.

SH programı, icatları uygulamalarla birleştirme sorununa ek olarak, kurumsal sorunların ve bütçe kısıtlamalarının da ceremesini çekmiştir. DARPA çalışanlarıyla asker müşterileri arasında sık sık sürtüşme yaşanırdı. Dahası, Kahn ve Cooper dahil, SH programını planlamış pek çok kişi, 1985'in sonlarına doğru DARPA'yı terk etti. 1986 baharında DARPA, Bilgi İşleme Teknikleri Dairesini Mühendislik Uygulamaları Dairesiyle birleştirdi ve adını Bilgi Sistemleri Teknoloji Dairesi (ISTO) koydu. Art arda ISTO müdürleri, hem bütçe dalgalanmalarına katlanmak zorunda kalmış hem de bu dalgalanmalara neden olmuştur. Bazı YZ yaklaşımlarından şüphe duyan Jacob Schwartz, Eylül 1987'de ISTO müdürü oldu. Kimi YZ programlarını hemen iptal edip bazılarını da yenilemedi. Derken 1991'de, ISTO ikiye ayrılıp SISTO'yu (Yazılım ve Zeki Sistemler Teknolojisi Dairesi) ve CSTO'yu (Bilgisayar Sistemleri Teknolojisi Dairesi) oluşturdu, böylece temel araştırmaları uygulamalarla eşleştirme çabasını pratikte bitirmiş oldu.

SH programı yavaş yavaş gözden yitmiştir. Roland'a göre 1989'dan sonra bu programın adı kamu belgelerinde ya da raporlarında anılmıyor. DARPA bütçesinden 1993'te atıldı ve en sonunda yerine, Yüksek Performanslı Hesaplama (High Performance Computing [HPC]) dahil başka programlar geçti.[53] Her ne kadar programın kendisi ortadan kaybolmuş olsa da, öteki "yeni nesil" projelerin başarıları gibi bunun da başarıları pek çoktur. 1980'lerde kaydedilmiş ilerlemeler, çeşit çeşit gerçek dünya uygulamalarının altından kalkan bir teknoloji olarak yapay zekânın rüştünü ispatladı. İleriki bir bölümde anlatacağım gibi, birçok yeni YZ teknolojisi 1980'lerde ve 1990'larda icat edilmiştir ve bunların en azından bazılarına yeni nesil projeler neden olup ortaya çıkardı. Bununla birlikte, zaman sırasına aşağı yukarı uyan tarihçeme devam etmeden önce, alanın kıyılarında ve bizzat içinde fokurdayan anlaşmazlıklardan bahsetmek için şimdilik konunun dışına çıkacağım.

53 *A.g.e.*, s. 285.

ALTINCI KISIM

PERDE ARASI

24
HIZ TÜMSEKLERİ

Yapay zekânın ilk günlerinden itibaren kötümserler hep var olmuştur. Alan Turing, 1950 tarihli makalesinde bu kesimlerin itirazlarını öngörmüştü (onlara yanıt da vermişti). Bu bölümde, YZ etrafında dönen, Turing'in öngöremediği kimi tartışmalar dahil bazı anlaşmazlıkları anlatacağım. Ayrıca, bu alanın karşısına çıkan birtakım dişli zorlukları betimleyeceğim. 1980'lerin ortalarında bu zorluklar yüzünden bazıları, o ana dek kaydedilmiş ilerlemeleri görece hiçe sayıp gelecekteki ilerlemeler konusunda karamsarlığa kapıldı. Örneğin, filozof Hilary Putnam, *Dædalus* adlı derginin 1988'de YZ için özel sayı çıkarma gereksinimini duymasına şaşırıp şunu yazmıştır:[1] "Bunu *şimdi* mesele yapmanın ne gereği var? Neden tüm bir *Dædalus* sayısı buna ayrılıyor? Neden YZ bir şeyler başarana dek bekleyip *o zaman* bu alan için sayı basmıyoruz ki?"

Alanın dışından gelen sataşmalar ve hayal kırıklığı ifadeleri, kimilerinin "YZ kışı" dediği şeyin başlamasını kolaylaştırmıştır.

24.1 ÇEŞİTLİ GÖZLEMCİLERİN GÖRÜŞLERİ

24.1.1 Zihin Makine Değildir

Filozof Alan Ross Anderson, makalelerden derlediği *Minds and Machines* başlıklı kitabına yazdığı Sunuş bölümünde, zihin makine mi değil mi meselesi hakkında iki aşırı uç görüşü nakleder:[2]

> (1) İnsanoğlunun yalnızca çok incelikli bir düzenekten ibaret olduğunu ve bizde "zihnin" olmasının, bu düzeneğin *çok* incelikli olmasından kaynaklandığını söyleyebiliriz, ya da
>
> (2) *herhangi* bir makinenin insan aklının ürünü olduğunu (prensipte kürekten öte olmadığını) ve *bizde* zihin olsa da, bize özgü bu özelliği çocuğumuzdan başka hiçbir şeye aktaramayacağımızı söyleyebiliriz: yani hiçbir makine, salt insana özgü olan bu özelliği edinemez.

1 Hilary Putnam, "Much Ado about Not Very Much", *Dædalus* (Yapay Zekâ Özel Sayısı), Kış 1988, s. 269-281.

2 Alan Ross Anderson (ed.), *Minds and Machines*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1964.

Çoğu YZ araştırmacısı büyük olasılıkla bu iki görüşten ilkine katılır. Kesinlikle ben de böyle düşünüyorum (gerçi ben olsam "yalnızca" sözcüğünü kullanmazdım). Marvin Minsky, "Zihin etten bir makinedir" dediğinde bu görüşü güçlü şekilde benimsemiş olduğunu gösterdi. Ancak, kimi filozoflar ikinci görüşü savunur. Bunların en göze çarpanı herhalde, Britanyalı filozof John R. Lucas'tır (1929-).

Lucas, "Zihinler, Makineler ve Gödel"[3] başlıklı denemesinde savını, Kurt Gödel'in, hem tutarlı olan hem de (en azından) cebir işlemleri yapabilen herhangi bir mekanik sistemin kanıtlamayacağı birtakım doğru önermelerin olduğunun ispatına dayandırır. Lucas, insanların (en azından bazı insanların) bu önermelerin doğru olduğunu "görebildiğini", oysa makinelerin bu önermeleri kanıtlayamayacağını kabul eder. Lucas'ın savındaki hatalara işaret eden insanlar çıkmıştır[4] ve Lucas, *The Freedom of the Will* [İrade Özgürlüğü] adlı kitabında bu itirazların en azından bir kısmına yanıt verdiğini öne sürüyor.[5] 1990'da Brighton'daki Turing Konferansında okunmuş makalesinde Lucas, şunu söyleyerek savını bir parça zayıflatmış gibi duruyor:[6]

> Ortaya koyduğum sav iki katlı. Zihinlerin doğal olarak makinelerden daha iyi olduğunu söyleyen basit ve ucuz bir kanıt ileri sürmüyorum; bunun yerine, makul bir makineci tezin ileri sürebileceklerini çürütecek bir şema oluşturuyorum... Dolayısıyla, esasen savımın iki kısmı ilk olarak ayakları yere basan bir eksi savdır, belirli bir iddia ortaya koyan makineciye hitap eder ve ona, geçerli olduğunu kabul etmek zorunda olduğu vasıtalarla, iddiasının makul olmadığını ispatlar; ikincisi uçarı bir artı savdır, zeki insanlara, izleyicilere, ayrıca makineciliğin belirli sürümlerini benimsemiş makinecilere hitap eder ve bu yönde bir savın, düşünülebilecek başka makinecilik biçimleriyle de başa çıkabileceğini söyler.

Bana kalırsa insanlar da, makinelere uygulanan her türlü Gödel kısıtlamasına tabidir; ama bu inancımın sebebi, insanların makine olduğuna inanmamdır. Lucas kendi bakış açısını savunmayı sürdürüyor,

3 John R. Lucas, "Minds, Machines and Gödel", *Philosophy*, c. XXXVI, 1961, s. 112-127; yeniden baskısı: Alan Ross Anderson (ed.), *Minds and Machines*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1964, s. 43-59; şu internet adresinde: http://users.ox.ac.uk/~jrlucas/Godel/mmg.html.

4 Bizzat Lucas, şu internet sitesinde bazı kaynaklar sıralıyor: http://users.ox.ac.uk/~jrlucas/Godel/referenc.html.

5 John R. Lucas, *The Freedom of the Will*, Oxford: Oxford University Press, 1970. Lucas, kitabın artık basılmadığını ve Oxford Üniversitesi Yayınevi'yle temas kurarak istenebileceğini internet sitesinde duyuruyor.

6 Şu internet adresinde: http://users.ox.ac.uk/~jrlucas/Godel/brighton.html.

çünkü (bence) insanların makine olmadığını düşünmek istiyor. Her koşulda bu tartışma bir parça kısır, çünkü YZ'nin uygulamada yapabileceklerini gerçekten kısıtlıyor değil. Özgün makalesinde Lucas bile şunu kabul etmiştir: "Örneğin supaplardan ve rölelerden oluşan çok ama çok karmaşık sistemler yapabiliriz ve bunlar, o şekilde programlamamış olsak da, bizim hata ya da rasgele diye niteleyemeyeceğimiz zekice şeyler yapabilir; fakat bu durumda, makine olmaktan çıkmışlar demektir". (Bu noktada, "makine"yi yeniden tanımlayarak tartışmayı kazanmaya uğraşıyor.)

Mühendis Mortimer Taube (1910-1965) de insanların makine olmadığına inanıyordu. 1961'de yazdığı *Computers and Common Sense: The Myth of Thinking Machines* [Bilgisayarlar ve Sağduyu: Düşünen Makineler Efsanesi] başlıklı kitabında,[7] bilgisayarların akıl yürütmesine, insan dillerini çevirmesine ve öğrenmesine yönelik çabalara ateş püskürür. Ancak, bilgisayarların yapamayacağını söylediği pek çok şey şimdiden başarılmıştır.

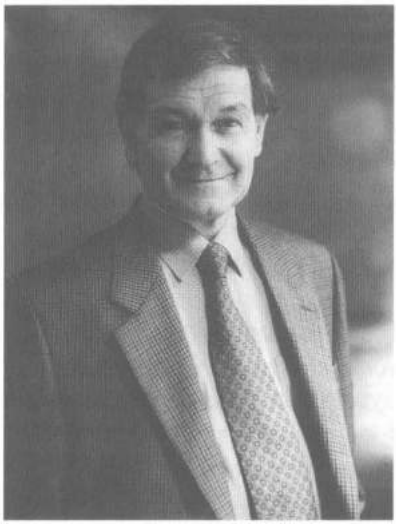
24.1.2 *Zihin Bilgisayar Değildir*

A. Yeni Bir Fizik Gerekli

İnsanlar makine olsa da, bilgisayarlar, en azından o günün transistörlü ve sıradan elektromanyetik donanımlı bilgisayarları, kendi işleyiş şekilleriyle zekâ sergileyemez savını ileri sürenler de olmuştur.

Britanyalı fizikçi Sir Roger Penrose (1931- ; Resim 24.1), bilgisayarların tabi olduğu sınırlara dair Lucas'ın Gödelci savlarına ikna olmuştu. (Penrose, kuantum fiziği, görelilik kuramı, evrenin yapısı ve "Penrose mozaikleri" çalışmalarıyla ünlüdür.) Lucas gibi o da, bilgisayarların asla bilinçli olmayacağına ve bir insan zekâsının tüm kapsamına ulaşamayacaklarına inanıyordu. Fakat Penrose bu kısıtlamaların yalnızca fiziğin o an için bilinen yasalarını temel almış makineler için geçerli olduğunu düşünüyordu. Gödel'in kısıtlamalarını savuşturmak için (Penrose, beynin bu işi başardığına inanıyordu), yeni bir fizik türüne başvurulması gerektiğini ileri sürüyordu; bu fizik, "doğru kuantum kütleçekimi" dediği şeyi içermeliydi. Ne yazık ki doğru kuantum kütleçekimi, artık bu her neyse, henüz keşfedilmiş değil (belki de icat edilmiş değil, demeliyim).

7 Mortimer Taube, *Computers and Commonsense: The Myth of Thinking Machines*, New York: Columbia University Press, 1961.



Resim 24.1. Sör Roger Penrose.

Penrose bu fikirleri (cazip fizik bilgileriyle birlikte) iki kitapta sunmuştur: *The Emperor's New Mind*[8] (İmparatorun Yeni Zihni) ve *Shadows of the Mind: A Search for the Missing Science of Consciousness*[9] (Zihnin Gölgeleri: Kayıp Bilinç Bilimini Aramak). Birçok kişi gibi ben de, YZ'nin tüm iddialarını hayata geçirmek için yeni bir fizik gerektiğinden kuşku duyuyorum. (Fakat, bunları gerçekleştirmiş değiliz elbette.) Penrose, "Bir Gölgeden Kuşku Duymanın Ötesinde: Zihin Gölgeleri Hakkında Yorumlara Yanıt" başlıklı makalesinde, görüşlerine yönelik eleştirilerin bazılarına yanıt vermeye çalışır.[10]

B. Amaçlılık Gerekli

Amerikalı filozof John Searle (1932- ; Resim 24.2), bildiğimiz haliyle bilişim süreçlerinde, biz insanlarda var olan bir şeyin eksik olduğunu ileri sürer; onun ve başka filozofların "amaçlılık" dediği şeydir bu. Amaçlılık, nesnelere ve nesne özelliklerine "anlam" iliştirmeyle ilintilidir. Searle'nin

8 Roger Penrose, *The Emperor's New Mind: Concerning Computers, Minds and the Laws of Physics*, New York: Random House, Inc., 1989.

9 Roger Penrose, *Shadows of the Mind: A Search for the Missing Science of Consciousness*, Oxford: Oxford University Press, 1994.

10 Roger Penrose, "Beyond the Doubting of a Shadow: A Reply to Commentaries on Shadows of the Mind", *PSYCHE*, c. 2, Ocak 1996, s. 23. Şu internet adresinde (eleştirmenlerin makalelerine linklerle birlikte): http://psyche.cs.monash.edu.au/v2/psyche-2-23-penrose.html.



Resim 24.2. John Searle. (Fotoğraf, John Searle'ün izniyle.)

tanımı şöyle: "Amaçlılık... belirli zihinsel durumları dünyadaki nesnelere ya da koşullara yönlendiren zihinsel özelliktir". Örneğin, Searle'e göre, "inançlar, arzular ve niyetler amaçlı durumlardır."[11] Dolayısıyla, der, "John uzun boyludur" sözünü bilgisayarda örneğin "G33(K077)" gibi bir mantıksal ifadeyle göstermek mümkünse de, bilgisayarın, G33 ifadesinin "gerçek dünyada" "uzun boy" özelliğine atıfta bulunduğunu ya da K077'nin "gerçek dünyada" John'a karşılık geldiğini bilmesinin yolu yok. Kısacası, bilişim süreçleri "hakkındalık" özelliğinden yoksundur, diyor: Yararlandıkları simgelerin ne hakkında olduğunu bilmezler. Bunun aksine, insanlar sözcük kullandığında, bu sözcüklerin ne hakkında olduğunu bilir.

Searle, "anlama" hakkında önerdiği düşünce deneyinden ötürü YZ araştırmacıları ve filozoflar arasında ünlüdür. Buna "Çince Odası" deneyi denir.[12] Searle, deney düzeneğini şu şekilde kurar:

> Diyelim ki bir odada kilitli kaldım ve elime Çince yazılı kocaman bir tomar tutuşturdular. Diyelim ki yazılı ya da sözlü Çince bilmiyorum (gerçekten de bilmem), hatta Çince yazıyı Japonca yazıdan ya da anlamsız karalamalardan ayırıp tanıyabilir miyim ondan da emin değilim.

Searle, bu deneyi "öykü anlamaya" dair YZ araştırmaları bakımından geçerli kılmak amacıyla, bu odada Çince simgelerle yazılmış iki tomar

11 Bkz. John R. Searle, "Minds, Brains, and Programs", *Behavioral and Brain Sciences*, c. 3, no: 3, 1980, s. 417-457. Şu internet adresinde: http://www.bbsonline.org/documents/a/00/00/04/84/bbs00000484-00/bbs.searle2.html.

12 *A.g.e.*

olduğunu ve bunların, Searle'ün bilmediği bir öykü ve bu tür öyküler hakkında genel bilgi içerdiğini varsayar. Odada ayrıca İngilizce yazılmış kurallar da vardır; bu kurallar, Çince yazı karakterleri kümelerini işlemeyi ve bu tür işleme sonucunda Çince karakterler yaratmayı tanımlar.

Derken bu odaya Çince simgelerle yazılmış üçüncü bir tomar gelir. Searle'ün belirttiği gibi, odada şöyle kurallar vardır:

> Bu kurallar, belirli Çince simgelere, üçüncü tomarda bana sunulan belirli şekillere karşılık olarak belirli şekiller vermeyi anlatıyor. Tanımadığım ve tüm bu simgeleri bana veren insanlar, ilk iki tomara öykü ve bunun arkaplan bilgisi derken, üçüncü tomara "soru kümesi" diyor. Dahası, üçüncü tomara yanıt olarak onlara sunduğum simgelere "soruların yanıtı" diyorlar ve bana İngilizce verdikleri kural takımına "program" diyorlar... Diyelim ki bir süre sonra Çince simgeleri işlerken komutları uygulamada çok başarılı hale geliyorum, programcılar da programları yazmada öyle iyi hale geliyorlar ki dışarıdan bakıldığında, yani kilitli kaldığım odanın dışından bakıldığında, sorulara verdiğim yanıtlar anadili Çince olanların yanıtlarından kesinlikle ayırt edilemiyor. Yanıtlarıma bakan hiç kimse, tek sözcük Çince konuşmadığımı anlayamıyor.

Temelde Searle'ün sorusu şu: "(Searle, kurallar ve Çince simge tomarlarını içeren) odanın Çinceyi "anladığı" söylenebilir mi?" Searle, yanıtın "hayır" olduğunu söyler, çünkü odada olup bitenler, simgelerin ne olduğunu anlamaksızın "örgün bir simge işleme"dir. Searle'ün sözcükleriyle söylemek gerekirse:

> Bizzat örgün simge yönlendirimlerinin amaçlılığı olmadığı için, bunlar oldukça anlamsızdır; *simge* yönlendirimi bile değildir bunlar, çünkü hiçbir şeyi simgelemezler. Dilbilim jargonuyla konuşmak gerekirse, bunların sözdizimi vardır ama anlambilimi yoktur. Bilgisayarların barındırıyormuş gibi göründüğü amaçlılık sadece bu cihazları programlayanların ve bunları kullananların, yani girdiyi gönderenlerin ve çıktıyı yorumlayanların zihnindedir.

Searle, Çince odasının, anlamayı taklit ettiğini kabul ederken, taklitler ve "gerçek" arasındaki ayrımı da ortaya koyar. Şöyle yazmıştır:

> Beşinci düzeyden bir yangının bilgisayar benzetiminin bütün mahalleyi yakıp kül edeceğini ya da sağanak yağış benzetimi yüzünden hepimizin su altında kalacağını kimse düşünmez. O halde anlamayla ilgili bir bilgisayar benzetiminin neden gerçekten bir şeyleri anladığı sanılsın ki?... Benzetim için tek ihtiyacınız, doğru girdi ile çıktı ve girdiyi çıktıya dönüştüren ara bir programdır. Bilgisayarın yaptığı her iş için elinde olan şey budur. Benzetimi kopyayla karıştırmak, ister acı, ister aşk, bilişsel

yetiler, yangınlar ya da sağanak yağmurlar söz konusu olsun, yine aynı hataya düşmek demektir.

Searle'ün Çince Odası bana, Herb Simon'ın, Mantık Kuramcısı [Logic Theorist (LT)] programının işleyişini taklit eden deneyini hatırlatıyor. Kitabın 3.2 numaralı başlığından, LT'nin el ürünü bir benzetimle başladığını anımsarsınız: Simon'ın çocukları bilişim öğeleri olarak kullanılırken, programın önerme değişkenlerini içeren kayıtlar olarak not kartları yazılmış ve saklanmıştı. Büyük olasılıkla çocuklar önermeli mantık konusunda cahildi, yine de Simon'dan, çocuklardan ve not kartlarından oluşan bu düzenek, teorem kanıtlayabilmişti. Açıkçası, teorem kanıtının "benzetimini yapmak", teoremi gerçekten kanıtlamakla aşağı yukarı aynı şeydir; tıpkı toplama işlemini taklit etmenin, toplama yapmakla aynı olması gibi. Peki, anlamayı taklit etme, anlamak kadar gerçek olabilir mi?

Searle'ün savlarına verilebilecek çeşitli yanıtlar var ve yanıt verenler bakımından hiç kıtlık yaşanmamıştır! Searle, makalesinde, olası pek çok yanıtı öngörmüş ve Searle'ün özgün makalesinin yanında yirmi sekiz itiraz da yayımlanmıştır. Bu konuda benim aklımdan geçenler şöyle:

Diyelim ki, İngilizce sorulmuş bir soruyu anlama sürecinde Searle'ün beyninin içine bakabiliyoruz. Soruyu anlarken ve bir yanıt oluşturup nakledeken, muhtemelen trilyonlarca sinaps, milyarlarca sinir hücresini ketlemekle ya da tetiklemekle meşgul olur. Herhangi bir sinapsın ya da bağlandığı sinir hücresinin, İngilizce anladığını söyleyemeyiz. "Anlama" süreci, sinir hücrelerinin işleyişini çözümlemeye yarayan ayrıntı düzeyinde anlamlı olan bir kavram değildir. Benzer biçimde, bilgisayarla (ya da Simon'ın çocuklarıyla) teorem ispatlama süreci, tek tek transistörler (ya da not kartlarını tutan çocuklar) düzeyinde anlamlı olan bir kavram değildir. İster beynin ister bilgisayarların (ya da başka bir şeyin) görüngülerini açıklarken, açıklama düzeyine uyan kavramlardan faydalanırız. "Anlama" kavramı, sinir hücresi düzeyinde değil "kişinin bütünü düzeyinde" gözlenen zihinsel etkinliklere uygulamayı faydalı bulduğumuz bir kavramdır. Benzer biçimde, oda, Searle, kurallar ve Çince yazı karakterleri bir araya geldiğinde oluşan bu yapının, Çince anladığını söylemek bence yerinde olacaktır.

Fakat anlam ve amaçlılık hakkında ne demeliyiz? Bilgisayar belleğine G33(K077) yazdığımızda bunun bir "anlamı" var mı? Bilgisayar belleğinde başka ne olduğuna göre değişir, özellikle de bellekte G33 ve K077 simgeleriyle ilişkili ne olduğuna göre. Simgeler ve bunlar arasındaki bağlantılar bir ağ oluşturur ve anlamları içeren şey ağın bütü-

nüdür. M. Ross Quillan'ın 1966 tarihli tezinde yanıtlamaya kalkıştığı soruyu anımsayın: "Sözcüklerin 'anlamlarını' saklamaya ne tür bir gösterimsel biçem olanak tanır ki bu anlamların, insanlarınkine benzer şekilde kullanımı mümkün olsun?" Belki de, 6.3 numaralı başlıkta yazdığım bir paragrafı burada tekrarlamak yerinde olacak:

> Quillian'a göre bir terimin anlamı, ağdaki yeri ve öteki terimlerle arasındaki bağlantı üzerinden gösterilir. Aynı fikirden sözlüklerde de faydalanılır; sözlükte bir sözcüğün anlamı, bu sözcüğün başka sözcüklerle ilişkisinden bahsedilerek verilir. O sözcüklerin anlamları da daha başka sözcüklerle ilişkileri üzerinden tanımlanır. O halde sözlükleri, sözcüklerle bağlantılı sözcüklerden oluşan büyük birer anlamsal ağ sayabiliriz.

Bazı durumlarda, ağdaki simgelerle dünyadaki gerçek nesneler arasında, bilgisayarın duyu ve devinim olanakları aracılığıyla bir bağ kurmak da gereklidir. Newell ve Simon, fiziksel simge sistemi hipoteziyle (PSSH) ilgili makalelerinde bu gereksinimi öngörmüştü. Bu hipotezin iddiasına göre fiziksel bir simge sisteminde (örneğin bilgisayarda), zeki davranışlar için gerekli ve yeterli koşullar vardır. Newell ve Simon şöyle yazmışlardır:[13]

> Fiziksel simge sistemi, zamanla evrimleşen bir simge yapıları derlemesi üreten bir makinedir. Bu tür bir sistem, nesneler dünyasında bizzat bu simgesel ifadelerden daha kapsamlıdır.

Bu "nesneler dünyası" bakımından, fiziksel simge sistemi, "niteleme" kabiliyetini içerir (örgün simge yönlendirimi için anlamına ek olarak). Newell ile Simon'un tanımı şöyle: "[Simgelerden meydana gelen] bir ifade, eğer belirli bir ifade için bu sistem ya *bizzat nesneyi etkiliyorsa ya da nesneye bağlı olarak hareket ediyorsa*, o nesneyi niteleyebilir (vurgular bana ait).

Bana kalırsa, bilgisayar bilimindeki kimi temel görüşler konusunda Searle'ün kafası karışmış. Her ne kadar bilgisayarların, insanların elinden gelen zeki davranışların tüm yönlerini sergileyebildiği deneysel olarak belirlenmemiş olsa da (simge yönlendirimi ve çevresine bağlı olmak gibi), Searle'ün düşünce deneyinin bu olasılık üzerine şüphe düşürdüğüne inanmıyorum.

13 Allen Newell ve Herbert A. Simon, "Computer Science as Empirical Inquiry: Symbols and Search", *Communications of the ACM*. c. 19, no: 3, Mart 1976, s. 113-126. Bkz. http://www.rci.rutgers.edu/~cfs/472_html/ALSEARCH/PSS/PSSH1.html.

Searle'ün kendisi belli türde fiziksel sistemlerin zeki olup bir şeyleri anlayabileceğine inanır. İnsanların böyle bir sistem olduğunu düşünür. Şöyle yazıyor:

> "Makineler düşünebilir mi?"
> Yanıt açıkça evettir. Bizler tam da öyle makineleriz.
> "Evet, fakat, insan eseri yapay bir makine düşünebilir mi?"
> Sinir hücreleriyle, aksonlarıyla ve dendritleriyle, diğer her şeyiyle birlikte bizimkine yeterince benzer tam bir sinir sistemi barındıran bir makineyi yapay olarak üretmenin olası olduğunu varsayarsak, bu soruya da verilecek yanıt açıkça evettir.

Yine de Searle, sinir hücrelerinden oluşan beynin, transistörlerden oluşan bilgisayarlardan ne farkı olup da ilkine amaçlılık bahşederken öbürüne bahşetmediği hakkında hiç ipucu vermiyor. Searle, makinelerin düşünmesi için, beyninkine eş "içsel nedensel güç" barındırmak zorunda olduklarını iddia eder. Bu içsel nedensel güçlerin ne olabileceğini ise söylemez.

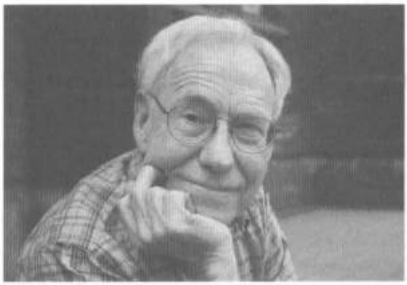
C. Güçlü ve Zayıf YZ

Searle'ün makalesinde, "güçlü YZ" ve "zayıf YZ" tanımları sunulur; bu kavramlar, iki tür YZ uğraşı arasındaki ayrımı ortaya koymak bakımından faydalıdır. Güçlü YZ, doğru düzgün programlanmış bir bilgisayarın, zihin yerine geçebileceği ve en az insanlar kadar düşünebileceği iddiasıyla ilişkilidir. Güçlü YZ'ye ulaşmak, pek çok yapay zekâ araştırmacısının son amacıdır. Searle'ün makalesi, güçlü YZ'nin imkânsız olduğunu göstermeye kalkışıyor. Bununla birlikte, zayıf (ya da "ihtiyatlı") YZ uygulamacıları, zihin hakkındaki hipotezleri ifade edip sınayarak, programlardan, zihni araştırmanın aleti olarak faydalanır. Aynı zamanda zayıf YZ, insanın zihinsel etkinliklerinin aynısını yaratmak yerine bu etkinliklere yardımcı olacak programlar yaratma çabasıyla ilişkilidir. Zayıf YZ çoktan başarılı olmuştur (olmayı da sürdürüyor), oysa güçlü YZ arayışı hiç şüphesiz daha uzun süre devam edecek.

D. "Küresel Süreçler" Gerekli

Günümüzde UC Berkeley'de felsefe profesörü olan Hubert L. Dreyfus (Resim 24.3), meslek yaşantısına MIT'de felsefe öğreterek başladı.[14] YZ girişimiyle ilk orada karşılaştı, ayrıca 1960'ların ilk yarısında kardeşi

14 Dreyfus'la meslek yaşantısı üzerine internet üzerinden yapılmış söyleşi için bkz. http://globetrotter.berkeley.edu/people5/Dreyfus/dreyfus-con0.html.



Resim 24.3. Hubert Dreyfus. (Fotoğrafın telif hakkı: Sijmen Hendriks. Sijmen Hendriks'in izniyle kullanılmıştır.)

Stuart'la birlikte Herb Simon'un bir konuşmasına katılmışlardı. YZ'yle ilgili kimi noktalar, ayrıca bu konuşmada duydukları Dreyfus kardeşlerin içine dert oldu. O zamanlar, California Santa Monica'daki RAND şirketi, bilgisayarcıların yanında bir de filozof bulundurmanın iyi fikir olduğunu düşünüyordu. RAND'da çalışan bir yöneylem araştırma uzmanı olarak Stuart, kardeşi Hubert'i tavsiye etti. Böylece Hubert 1961 yazını, RAND'da danışman sıfatıyla YZ üzerinde çalışarak geçirdi. O yazın ardından kısa süre sonra Hubert, "Simya ve Yapay Zekâ" başlıklı bir RAND makalesi kaleme aldı; bu makalede, başkaları yanında, YZ araştırmalarının hedeflerinin simyanınkiler kadar ulaşılmaz olduğu sonucuna vardı.[15]

Dreyfus, makalesinde YZ'nin ilerleyişini dört alana ayırmıştı: oyun oynama, problem çözme (teorem kanıtlama dahil), dil çevirisi, örüntü tanıma. Şöyle yazmıştır:

> Genel örüntü şöyle şekilleniyor: Başta basit görevlerde gösterilen kolay becerilere dayalı çarpıcı başarılar ya da karmaşık görevlerde düşük nitelikli çalışmalar, ardından verimin düşüşü, gerçeklerle yüzleşme ve bazı durumlarda karamsarlık.

Gelernter'in geometri teoremi kanıtlama makinesinin tipik bir örnek olduğunu ileri sürer: "En başta "hayret verici" başarı ve ardından aynı oranda hayret verici başarısızlık için bundan daha çarpıcı örnek yok". İlerleme kaydedildiği hakkında YZ iddiasına yanıt verirken şöyle yazmıştır: "[İlerlemenin] bu tanımına göre, ağaca tırmanan ilk kişi, Ay'a uçma yönünde gerçek bir ilerleme kaydettiğini söyleyebilir."

Dreyfus'a göre bu durgunluğun nedenlerinden biri, YZ araştırmalarının, düşüncenin basit ve kati işlemler kümesi olarak çözümlene-

15 Hubert L. Dreyfus, "Alchemy and Artificial Intelligence", RAND makalesi P-3244, The RAND Corporation, Santa Monica, CA, Aralık 1965. Şu internet adresinde: http://www.rand.org/pubs/papers/2006/P3244.pdf.

bileceği (sonlu bir veri kümesine kurallar uygulamak gibi) varsayımına dayanmasıdır. Bunun yerine "düşünme, küresel süreçler içerir ve bunlar, ayrık adımların bir dizisi, hatta paralel kümesi olarak anlaşılamaz" iddiasında bulunmuştur. Bu küresel süreçler üç şekilde kendini gösterir. İlki, "sınır bilinçtir". Beyin, "gündelik deneyimlere özgü ucu açık sonsuz bilgiye" bu bilinci kullanarak erişir. Sınır bilinç sayesinde insanlar ayrıntıları ve büyük resmi eş zamanlı olarak gözden geçirebilir. Önemli olanı önemsiz olandan ayırt ederken, insan düşüncesinde başka bir küresel süreç iş başındadır. Üçüncüsü, muğlaklığı azaltmaya olanak tanıyan "küresel bağlam"dır. Bu yeteneklerin birleşimi, Dreyfus'un "aşikâr kümelenme" dediği şeye olanak tanır; beyin, örneğin insan yüzleri gibi karmaşık örüntüleri tanırken bundan faydalanır. Dreyfus, bilgisayar programlarının, zeki davranışlar için gerekli olan bu küresel süreçleri kullanamayacağını iddia etmişti.

Dreyfus, beynin bilgi işleme biçiminin bilgisayarınkinden bütünüyle farklı olduğunu belirtmişti. Beyinde bilginin, "tıpkı ağ içinde en kısa yol problemini çözen analog direnç [bir tür analog bilgisayar] gibi küresel olarak işlendiğini", yazmıştır. Dahası, "zeki davranışı olası kılmada beden önemli rol oynar", demiştir. YZ'deki ilerlemeler bakımından "bedenleşmenin" önemini vurgulayan başkaları da oldu. Bu konu hakkında benim de kısaca söyleyeceklerim olacak.

Dreyfus, gelecek hakkında (1960'ların başından göründüğü haliyle) şunları yazmıştı:

> Yeni ve hızlı makinelerin, daha iyi programlama dillerinin ve daha zeki sezgisel yöntemlerin, ön cepheyi ne ölçüde ileri taşıyacağını ancak deneyler belirleyebilir. Yine de, ele aldığımız alanlardaki çarpıcı yavaşlama ve öngörüleri yerine getirmedeki genel başarısızlık, sınırın yakın olabileceğini akla getiriyor.

Dreyfus'un YZ hakkındaki yorumları, makinelerle insan düzeyinde yapay zekâya ulaşmanın imkânsız olduğunu düşündüğü anlamına gelmez; filozof John Haugeland'ın "güzel eski usul yapay zekâ" (ETYZ) dediği[16] yöntemlerle, yani simgesel olarak gösterilen ayrı olgu ve kural kümeleri ile sezgisel arama kullanan yöntemlerle bu hedefe ulaşmanın imkânsız olduğunu düşünüyordu sadece (hâlâ da öyle düşünüyor). Prensipte, "tıpkı insan beyni gibi işlev gören bir alet inşa eder ya da taklit edersek, zeki davranışları taklit edebileceğimizi" kabul

16 John Haugeland, *Artificial Intelligence: The Very Idea*, Cambridge, MA: MIT Press, 1985.

ediyordu. Fakat bu işin uygulamada başarılamayacağını düşünüyordu. "Beyindeki fiziksel süreçleri betimleyen denklemleri bilmiyoruz ve bilseydik bile, en basit tepkiyi tanımlayan denklemlerin çözümü, bu işi imkânsız kılacak kadar zaman alırdı". Makalesinin özeti şu yargıyla biter: "Yapay zekâda belirgin gelişmeler için... bütünüyle farklı bir bilgisayar türü beklenmeli. Bu bilgisayarın var olan bir örneği, çok az anladığımız insan beynidir."

Dreyfus'un RAND makalesinde geçen başlıca fikirler, bazı kitaplarında ve makalelerinde sunulup kapsamı genişletilmiştir.[17] Pamela McCorduck'un *Machines Who Think* (Düşünen Makineler) başlıklı kitabı,[18] Dreyfus hakkında mükemmel bir bölüm içeriyor; bu bölümde Dreyfus'un savlarının ayrıntıları veriliyor ve YZ bilimcileriyle arasındaki kavgalı ilişkiler etraflıca anlatılıyor. McCorduck bu konuları gayet güzel anlattığı için, Dreyfus'un son birkaç makalesinde betimlediği "bedenleşme" ihtiyacı hakkındaki fikirlerine yoğunlaşacağım.

E. "Var Olmak" Gerekli

Bana kalırsa Dreyfus'un asıl dediği şey şu: İnsan zekâsı, insanların kurallarla yönetilmesinden değil "dünyadaki varoluşlarından" türüyor. YZ programlarında kural kullanımı (tıpkı insanlardaki gibi), yeterli davranışlara olanak tanısa da uzman davranışlarını mümkün kılmaz. Dreyfus'un 2005'te yaptığı bir konuşmadan bazı alıntılara bakalım:[19]

> ... örgün talimatlarımıza kurallarla başlarız; fakat bu kurallar, bizler beceri kazandıkça daha esnek tepkilere yol verir... Gerçek görüngüler gösteriyor ki uzman olmak için, kurallara, gerçeklikten kopuk biçimde uymaktan çıkıp daha ayrıntılı ve duruma özgü hareket etmeye geçmeliyiz.

17 Örneğin bkz. Hubert L. Dreyfus, *What Computers Can't Do: A Critique of Artificial Reason,* New York: Harper& Row, 1972 (2. baskı, 1979). Gözden geçirilmiş baskısı: *What Computers Still Can't Do: A Critique of Artificial Reason*, MIT Press, 1992. Hubert L. Dreyfus ve Stuart E. Dreyfus, *Mind Over Machine: The Power of Human Intuition and Expertise in the Era of Computer*, New York: Free Press, 1986.

18 Pamela McCorduck, *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*, San Francisco: W. H. Freeman and Co., 1979.

19 Hubert L. Dreyfus, "Overcoming the Myth of the Mental: How Philosphers Can Profit from the Phenomology of Everyday Expertise", Presidential Address, *Proceedings and Addresses of the American Philosophical Association,* c. 79, no: 2, Kasım 2005. Şu internet adresinde: http://socrates.berkeley.edu/~hdreyfus/pdf/Dreyfus%20APA%20Address%20%202010.22.05%20.pdf.

> Genel olarak, uzman, kendi eylemlerine karar verirken ya da gerekçelendirirken, kuralları ya da standartları temel almak yerine, o anki somut duruma anında tepki verir.
>
> ...
>
> Bilgi mühendisleri denen kişilerin uzmanlardan temin ettiği kurallara dayanan "uzman sistemler", en iyi haliyle yeterliydi. Öyle görünüyor ki uzmanlar, YZ araştırmacılarının varsaydığı gibi hatırlamadıkları kuralları kullanmak yerine, artık kullanmadıkları kuralları hatırlamak zorunda kalıyor. İşin doğrusu, uzmanlar hiçbir kurala uymuyor da olabilir.

Dreyfus'a göre, yeterli davranıştan uzman davranışına geçmek, dünyaya gömülmüş bir beden sayesinde "var olmayı" gerektiriyor. İnsanlar gibi bedenleşmiş etmenler, "dünyanın öyle içindedirler ki sonsuza dek sürecek, her şeyi formül haline getirme işinden kaçınabilirler" (YZ programları buna boşuna uğraşır).[20] Dreyfus'un bedenleşme ihtiyacına dair bu görüşü, "görüngübilim" denen felsefe ekolünün bir dalını temel alıyor. Dreyfus bana, Martin Heidegger'in (1889-1976) "temel varoluş usulümüz olarak insanlarla ve nesnelerle gerçek yaşamdaki bağlılığımızı" vurgulayan varoluşsal görüngübilimin, kendi ETYZ eleştirisinin zeminini oluşturduğunu yazmıştı.[21] YZ'nin başarılı olması için şuna gerek duyduğunu ileri sürer:[22]

> ... kendimize özgü gömülü ve bedenleşmiş olma yöntemimizin modeline gerek var, öyle ki deneyimlediklerimiz, bu yöntem sayesinde bizim için önemlidir. Yani, programıza bir beden modeli eklemek zorundayız ve bu modelin tıpkı bizim gibi gereksinimleri, arzuları, mutlulukları, acıları, hareket tarzı, kültürel arkaplanı, vesairesi olmalı.

Bedenleşmeyi savunan başkalarına göre, zeki etmenin ihtiyaç duyduğu "hesaplamaların" bazıları, bedeni ile çevresinin arasındaki dinamik etkileşim sayesinde başarılabilir. Örneğin Rolf Pfeifer, Max Lungarella ve Fumiya Iida şöyle yazmışlardır: "Bedenleşmiş bir bakış

20 Hubert L. Dreyfus, *What Computers Still Can't Do: A Critique of Artificial Reason*, s. 255, Cambridge, MA: MIT Press, 1992.

21 E-postayla haberleşme, 9 Ağustos 2007. Dreyfus'un Heidegger'e dair yorumları için bkz. Hubert L. Dreyfus, *Being-in-the-World: A Commentary on Heidegger's Being and Time, Division I*, Cambridge, MA: MIT Press. 1991.

22 Hubert L. Dreyfus, "Why Heideggerian AI Failed and How Fixing it Would Require Making it More Heideggerian", (APA'nın 2006 Barwise ödülünü almakla bağlantılı olarak yazılmış bir makale), *Philosophical Pschology*, s. 20, no: 2, 2007, s. 247-248; yeniden baskısı: Michael Wheeler (ed.), *The Mechanization of Mind*, Cambridge MA: MIT Press, 2008.

açısı, denetimi ve işlemeyi etmenin (*agent*) tüm yönlerine (merkezî sinir sistemi, kas-iskelet sisteminin maddi nitelikleri, duyu morfolojisi, çevreyle etkileşim) dağıttığı için, robotiğin karşısına çıkan zorluklarla başa çıkacak alternatif bir yol sunar. Klasik yaklaşımda denetçinin gerçekleştirdiği görevler, artık öz-örgütlenme sürecindeki morfoloji ve maddeler tarafından kısmen devralınıyor..."[23]

Ancak bir beden gerekli olsa bile, bunun biçimi, ilintili YZ sisteminin ne amaçla kullanıldığına bağlı olarak değişecektir. Kurmaca robot HAL 9000'in bedeni, denetlediği koca uzay gemisiydi. Robot Shakey'nin vücudu, gereksinimlerine gayet uygundu. Eğer kullanıcılarla tüm internet sayfalarının içeriği hakkında diyaloğa giren bir "sohbet Google'ı" geliştirilecek olsaydı, bunun bedeni bütün internet ve internete erişimi sağlayan altprogramlar olurdu.

24.1.3 *Beyin ile Bilgisayar Arasındaki Farklılıklar*

Dreyfus'a ek olarak bazı YZ eleştirmenleri, "beynin bilgisayar olmadığına", dolayısıyla beynin yapabildiklerini bilgisayar kullanarak yapmaya çalışan insanların başarısızlığa mahkûm olduğuna işaret etmiştir. Bu eleştirmenler çoğunlukla şu aşağıdaki gibi ayrımları öne çıkarır:

- Bilgisayarların belki yüzlerce işlemci birimi varken beyinde trilyonlarca işlemci bulunur.
- Bilgisayar saniyede milyarlarca işlem yaparken beyin sadece binlerce işlem yapar.
- Bilgisayar çökebilir ama beyin, hataya dayanıklıdır.
- Bilgisayar ikili sinyal kullanırken beyin analog sinyal kullanır.
- Bilgisayar ancak programcısının dediklerini hayata geçirir, oysa beyin yaratıcıdır.
- Bilgisayar sıralı işlemler yaparken beyin büyük oranda paralel işlemler yapar.
- Beyin "mantıksal" olmakla kısıtlıdır, oysa beyin "sezgisel" olabilir.
- Bilgisayar programlanırken beyin öğrenir.

Bu ayrımların pek çoğunun artık geçerli olmayışı bir yana,[24] kıyasla-

23 Rolf Pfeifer, Max Lungarella ve Fumiya Iida, "Self-Organization, Embodiment, and Biologically Inspired Robotics", *Science*, c. 318, no: 5853, 16 Kasım 2007, s. 1088-1093.

24 Örneğin 2003'te yazılmış bir makale şunu iddia ediyor: "Google'ın mimarisi, hataya dayanıklı yazılımlı 15.000'i aşkın ticari sınıf PC kümesinden oluşur." Kuşkusuz,

malar, "beynin" ne anlama geldiğiyle "bilgisayarın" ne anlama geldiğine göre değişir. Eğer beyin anlayışımız, sinir hücresi bileşenleri, pek çok sayıda aksonları, dendritleri ve sinaps bağlantıları bağlamında kalıyorsa ve eğer bilgisayar anlayışımız, hepsi de transistörlü devreler tarafından gerçekleştirilen "von-Neumann-tarzı" sıralı işlemler (bilgi bitlerinin okunması, işlenmesi ve yazılması) bağlamıyla kısıtlıysa, o zaman elbette beyin *o* tür bir bilgisayar değildir.

Ancak, "bilişim"i, sadece alt düzey bir von Neumann-tarzı tanıma atıfta bulunarak anlamayız. Pek çok betimleme düzeyinden herhangi birine göre anlayabiliriz bunu. Örneğin bilişim, aynı anda etkin olan çok sayıda "bilgi kaynağının", Karatahtadaki karmaşık simgeleri eş zamanda olmayan biçimde okuması, dönüştürmesi, yazması bağlamında ya da Pandemonium-tarzı ağda yer alan simge işleyici ve sinir ağı cinlerinin derlemesi bağlamında anlaşılabilir. Belki de beynin işleyişine dair yavaş yavaş gelişen anlayışımız, en sonunda yararlı hesaplama modelleri doğuracak. "Hesaplama"nın ne olabileceği hakkında fikirler gitgide genişliyor, dolayısıyla beynin bilgisayar olmadığını iddia edenler, beynin hangi tür bilgisayar olmadığını belirtirken daha net olmalı. (Ne de olsa, eğer Lucas gibi kimi insanlar, makinenin ne olabileceğini kısıtlıyorsa, başkalarının da bilgisayarın ne olabileceği hakkındaki tanımı genişletmesi hakçadır.)

24.1.4 Peki Yapmalı Mıyız?

YZ'nin yapamadıklarını temel alan eleştirilerin yanı sıra, *yapmaması gerekenleri* temel alan eleştiriler de var. YZ'nin "yapmaması gereken" şeyler olduğunu savunan kimi insanlar, öğretme, rehberlik etme ve hukuksal yargıda bulunma gibi doğal olarak insan-merkezli işleri makinelerin yapmaya kalkışmasının uygunsuzluğundan bahseder. Daha önce adını andığımız Toplumsal Sorumluluk Sahibi Bilgisayar Profesyonelleri (CPSR) birliği gibi başkaları, YZ teknolojisinin (ya da başka bir teknolojinin) silahlarda ya da keşif görevlerinde, ya da deneyime dayalı insan yargısını gerektiren işlerde kullanıldığını görmek istemez. Buna ek olarak, 19. yüzyıl Britanyasının Makine Düşmanları gibi, makinelerin insanın yerini alıp işsizlik ve iktisadi yoksullaşma yaratacağından endişe duyan insanlar da var. Son olarak, YZ'nin ve

Google günümüzde ağa bağlanmış çok daha fazla sayıda bilgisayar kullanmaktadır. Bkz. Luiz Andre Barroso, Jeffrey Dean, ve Urs Hölzle, "Web Search for a Planet: The Google Ouster Architecture", *IEEE Micro,* Mart-Nisan 2003. Şu internet adresinde: http://labs.google.com/papers/ googlecluster-ieee.pdf.



Resim 24.4. Joseph Weizenbaum. (Fotoğraf, MIT Müzesinin izniyle.)

bilgisayar teknolojilerinin, insanları insanlıktan çıkarmasından, kişiler arası temasları azaltmasından ve insan olmanın anlamını değiştirmesinden kaygı duyanlar da var.

2.3.3 numaralı başlıkta bahsettiğim ELIZA programını yazmış olan Joseph Weizenbaum (1923-2008; Resim 24.4), insanlara bırakılması gerektiğini düşündüğü sorumlulukların bilgisayarlara verilmesinin tehlikeleri hakkında yazmış, konuşmalar yapmıştır. Kimilerine göre Weizenbaum'un bu endişesinin itici gücü, ELIZA'yla yapılan sohbetleri bazı insanların gerçek insan sohbeti sanmasının yarattığı şaşkınlık ve sarsıntıdır. Weizenbaum, *Computer Power and Human Reason: From Judgement to Calculation* (Bilgisayar Gücü ve İnsan Aklı: Yargıdan Hesaplamaya) başlıklı kitabında,[25] şunu ileri sürer: "İnsan ile makine arasında fark var ve ... bilgisayarların belirli işleri o işlerin altından kalkabilmesinden bağımsız olarak, yapmaması *gerekir*."

Weizenbaum kitabında, kişinin büyüdüğü, yaşadığı ve çalıştığı kültürel ortamın önemine vurgu yapar. Hiçbir makine, insan tarzı bir geçmiş deneyimlemez (ya da olsa deneyimleyemezdi), dolayısıyla hiçbir makineye, birçok şeyin yanı sıra böylesi bir geçmişin doğurduğu şefkati ve bilgeliği de gerektiren kararları alma ya da tavsiyeleri verme olanağı tanınmamalı. "Bu düşünce ve eylem alanlarındaki" deneyimsizliğin aynı zamanda "insanların birbirleriyle ilişkileri, makineler ve makine-

25 Joseph Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation*, San Francisco: W. H. Freeman and Co., 1976.

insan ilişkileri" için de geçerli olduğunu söyleyerek değindiği konunun altını çizmiş olur.[26] Dolayısıyla Weizenbaum'un, nasıl bir makinenin hukuksal kararlar alması uygunsuzsa, Amerika'da yetişmiş bir insanın Japonya'da hukuksal kararlar almasının da öyle uygunsuz olacağını düşündüğünü varsayıyorum. Dahası, makineye insan benzeri bir beden ve duyu düzeneği verilerek gerekli altyapıyı edinmesini sağlama fikriyle alay eder. Şöyle yazmıştır: "Yapay zekâ çalışmalarını şevklendiren en derin ve en görkemli fikir... insan modelinde bir makine, kendi çocukluğu olmuş, çocuk gibi dil öğrenmiş, kendi organlarıyla dünyayı duyumsayarak bilgi edinmiş ve en sonunda insan düşüncesinin tüm alanlarına göz atmış bir robot inşa etmektir."[27]

Weizenbaum, 1936'da ailesiyle birlikte Nazi Almanyası'ndan kaçmıştı. Bu deneyim, onun keskin toplumsal sorumluluk duygusunu daha da bilemiştir. Örneğin şöyle yazmıştı:[28]

> "Bir yargıç (ya da psikiyatr), bilgisayara anlatamayacağımız ne biliyor olabilir?" sorusu, canavarca bir iğrençliktir. Bu sorunun, hastalıklı oluşunu ortaya çıkarmak amacıyla da olsa, basılıp yayılması çağımızın deliliğinin bir işaretidir.
>
> ...
>
> [Önemli konulara] "yapabilir mi, edebilir mi" tarzı sorularla karar verilemez. Bilgisayar uygulamalarına dair sınırlar, en sonunda zorunluluklar bağlamında ifade edilmelidir. En temel içgörü olarak beliren şey, bilgisayarları bilge kılmanın yöntemini bilmediğimiz için, bilgelik gerektiren görevleri bilgisayarlara vermeme zorunluluğudur.

Her ne kadar Weizenbaum, makinelerin "yetileriyle" ilgili konularda lafı bir parça dolandırsa da, makinelerin "yapmaması gerekenler" kadar "yapamadığı şeyler" olduğuna inandığını sanıyorum. Çünkü, eğer makineler, insanların "şefkatle ve bilgelikle" verdiği hükümleri gerçekten verebilse, bu işi neden *yapmasınlar ki? Herhangi* bir teknolojiyi toplumsal karşıtı amaçlarla (savaş gibi) kullanma kaygısına ek olarak, bence gerçek tehlike, makineleri zamanından önce kullanmaktır: Henüz yeterliliğe ulaşmadıkları halde belli bir görevi başarabileceklerini düşünmektir asıl tehlike.

Makinelerin her şeyi "devralacağı" beklentisinden irkilenlerden biri de hekim, biyolog ve deneme yazarı Lewis Thomas'dı (1913-1993).

26 *A.g.e.*, s. 223-224.
27 *A.g.e.*, s. 202-203.
28 *A.g.e.*, s. 227.

The New England Journal of Medicine adlı dergi için kaleme aldığı ünlü köşe yazılarından birinde, "Biyoloji Bekçisinin Notları" başlıklı metinde şöyle yazar:[29]

> İnsan türünün geleceği hakkında tüm fikirlerin en can sıkıcısı yapay zekâ kavramıdır. En sonunda insanların evrimi aşarak kendilerinden daha zeki olmasını sağlayacak karmaşıklığa ve ustalığa sahip bilgisayarlar imal etme hırsı ve bu cihazların insanları ya da daha sonra makineleri geliştirme yolunda insanların yerine geçecek olması, bana derinden yanlış, hatta belki şeytani geliyor.
>
> ...
>
> Yapay zekâ çalışanlarının bahsettiği işte şöyle bir şey: Geçmişe bakıp geleceği hakkında doğru öngörülerde bulunabilen ve geleceği istediği gibi değiştirme amacıyla kusursuz planlar oluşturabilen, ayrıca en korkuncu da, belli bir amaç güden mekanik bir beyin.
>
> ...
>
> Bana kalırsa bu kesinlikle iğrenç bir beklenti ve bunun gerçekten kaçınılmaz olduğunu düşünseydim, her günümü buna karşı koymakla geçirirdim.

Genelde bilgisayarların ve özelde zeki makinelerin uygunsuz kullanımlarının tehlikeleri hakkında uyarıda bulunmuş başka yazarlar da olsa, bunların sadece birinden, "yeni-Makine Düşmanı" olduğunu itiraf eden bir kişiden bahsedeceğim. Önde gelen yazarlardan ve toplumsal düşünürlerden Theodore Roszak (1933-2011), kanımca okunmaya değer biridir. *The Cult of Information* (Enformasyon Kültü) başlıklı kitabında,[30] "enformasyona" ve "enformasyon işlemeye" sevdalı ve gitgide büyüyen bir kültün, yozlaştırıcı kültürel etkileri olduğunu ileri sürmüştü, "özellikle de iş, gençleri eğitmeye geldiğinde". Roszak, "bilgisayarların yapabileceklerine ve yapamayacaklarına, yapması ve yapmaması gerekenler hakkında makul ve dengeli görüş benimsemiş tüm ciddi enformasyon teknolojisi araştırmacılarının ve kullanıcılarının destekçisi" olduğunu yazmıştı; fakat "bir enformasyon gizeminin yaratılması, veri, bilgi, yargı, imgelem, içgörü ve bilgelik arasındaki temel düşünsel ayrımları imkânsız kılmıştır", iddiasında bulunur.[31]

29 Lewis Thomas, "Notes of a Biology Watcher: On Artificial Intelligence", *The New England Journal of Medicine*, c. 302, no: 9, 28 Şubat 1980, s. 506vd.

30 Theodore Roszak, *The Cult of Information: A Neo-Luddite Treatise on High-Tech, Artificial Intelligence, and the True Art of Thinking*, 2. baskı, Berkeley, CA University of California Press, 1994.

31 *A.g.e.*, s. xviii-xix.

Enformasyon işleme ile düşünme arasında "yaşamsal bir ayrım" olduğunu ileri sürerek şöyle yazar:[32]

> [Bilgisayarın] veri saklama becerisi, aşağı yukarı insanlarda bellek dediğimiz şeye denk geldiği için, ayrıca mantıksal yöntemlere uyma becerisi az çok insanlarda uslamlama dediğimiz şeye karşılık geldiği için, [enformasyon] kültünün pek çok üyesi, bilgisayarların yaptığı işin aşağı yukarı düşünmek dediğimiz şeye denk geldiği sonucuna varmıştır.

Ancak Roszak, bilgisayarların gerçekten "düşünemeyeceği" sonucuna varır. Ancak, bilgisayarların düşünebileceğine inanan (ya da hileyle buna kandırılmış kişilerin), salt "veri işlemeyi" değil düşünmeyi de gerektiren işler için uygunsuz biçimde bilgisayarları kullanmasının tehlikesinden bahseder. Bence Roszak'ın burada haklı bir kaygısı var; YZ, ona yüklemeye çalıştığımız görevlerin tümünü başaracak noktaya henüz ulaşmış değil.

Fakat Roszak şöyle de yazmıştır:[33]

> Veri işleme ve yordamlarla düşünme içeren kısıtlı işlevsel uygulamalar hariç, bilgisayarlar asla zihne eş olamayacak ya da onun yerini alamayacak. Bu olasılık prensipte elenmiştir, çünkü bu çabanın temelini döşeyen metafizik varsayımlar yanlıştır.

Bu noktada onunla aynı fikirde değilim. Beynin bir tür makine olduğunu, dolayısıyla onu anlayıp tıpkı onun gibi işleyen bir makine yapabileceğimizi söyleyen varsayım dışında, YZ alanında hiçbir "metafizik varsayım" bilmiyorum. Dahası, bu metafizik varsayımın yanlış olduğunu gösteren güvenilir bir bulguya da rastlamış değilim.

Roszak, bilgisayarların zihnin yerini tutmayacağını düşünse de, şu tehlike konusunda endişelenmekten geri durmaz: "Zihni ve kullanımlarını, makinelerin taklit edebileceği biçimde yeniden tanımlamak olasıdır. O zaman, zihinsel etkinliğin çıtasını daha alt düzeye çeken mekanik bir karikatürümüz peydahlanmış demektir."[34]

Roszak'ın en azından YZ hakkında söylecek iki iyi şeyi de var; bir tanesi YZ'nin olumsuz bir sonucudur, öbürüyse olumlu bir katkısıdır. Olumsuz sonuç hakkında şunu der:[35]

32 *A.g.e.*, s. xiv.
33 *A.g.e.*, s. 232.
34 *A.g.e.*, s. 232.
35 *A.g.e.*, s. xxiv.

> YZ'nin tüm biçimlerinde ironik ama epey değerli bir nitelik vardır. İnsan zekâsını taklit etme ya da aşma çabası, insan zihninin kafamızdan bile geçirmediğimiz inceliklerini ve çelişkilerini gün ışığına çıkarıyor. YZ, kahramanca başarısızlıklarıyla bize gerçek zekânın aslında ne kadar tuhaf olduğunu öğretiyor.

Olumlu tarafa gelince şu yorumu yapar:[36]

> Bununla birlikte, YZ alanında kayda değer ilerleme kaydedilmiştir... Çoğunlukla, uzmanlara kendi çalışmaları hakkında soru soran bilgisayar programcıları, örgün olarak tanımlanmış yöntemler, varsayımlar, değerler belirleyebilir. Sonuçta ortaya Uzman Sistem çıkar, yani YZ'nin birkaç kullanışlı uygulamasından biri. Edward Feigenbaum'a göre bu tür sistemler, makine zekâsının bir sonraki çağına geçiş kapısıdır; buna veri işleme yerine "bilgi işleme" der. "Bilgi" sözcüğüyle ne kastediyorsa etsin, bu sözcüğünün, bir zamanlar bu alanda hüküm süren yaklaşımdan daha karmaşık bir düşünce yaklaşımını gösterdiği kesindir.

24.1.5 Başka Görüşler

1981 yılının Ocak ayında, planladığım konuşma için insanların YZ hakkındaki görüşlerini toplarken, bilgisayar biliminin ve ilişkili disiplerin kimi önderlerine yazıp, YZ'nin başarıları, zayıflıkları, beklentileri hakkında neler düşündüklerini sordum. Sorularıma bazı yanıtlar aldım ve birkaç yorumdan alıntılar sunacağım.[37]

Bilgisayar bilimcisi (ve Stanford'dan çalışma arkadaşım) Donald Knuth şöyle yazmıştı:

> Şimdiye dek YZ'nin, gerçekte "düşünme" gerektiren her şeyi başarmış olması ama insanların ve hayvanların "düşünmeksizin" becerdiği şeylerin çoğunda başarısızlığı ilgimi çekiyor; bir şekilde bu iş çok daha zor! Bence, YZ programları oluşturulurken elde edilen bilgi, programların kullanımından daha önemli...

Hem yapay çeviriyle ilgili ALPAC raporuyla (Kısım 7.2) hem de konuşma anlama hakkındaki olumsuz yorumlarıyla bağlantılı olarak bahsetmiş olduğum John R. Pierce, bana yazdığı çok kısa mektupta şunları ifade etmişti:

36 *A.g.e.*, s. 232.

37 Tüm bu yanıtlar dosyalarımda duruyor.

> Yapay zekâya gelince, "Yapay Zekâ gerçek aptallıktır" sloganını ben icat ettim sanırım.
>
> ...
>
> Yapay zekâya kızıyorum, çünkü bilgisayarlara haksızlık olduğunu düşünüyorum; fakat yapay zekâ çalışanları LISP'i yaptı ki bu gayet güzel bir şey.

Bu mektup, ne bu sloganı[38] ne de YZ'nin neden "bilgisayarlara haksızlık" olduğunu etraflıca açıklıyor.

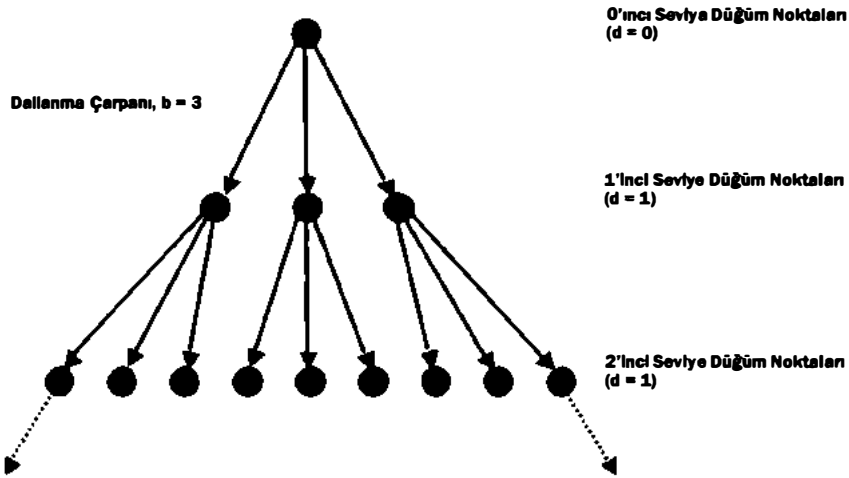
Hollandalı bilgisayar bilimcisi Edsger W. Dijkstra (1930-2002), bilgisayar biliminde yaptığı çok sayıda icatla ünlüdür ki grafiklerdeki en kısa (ya da en masrafsız) yolu bulan algoritma da bunlara dahildir. Aynı zamanda, "yapısal programlama" [*structured programming*] denen şeyi de savunuyordu; bu yöntem, program yazmanın (ve anlamanın) verimini çok artırmıştır. Mektubuma yanıt olarak şöyle yazmıştır (içten bir yaklaşımla ve çok güzel bir elyazısıyla):

> "Fakat bizler sadece simge kurcalayan makineleriz, öyle değil mi?" diyen yapay zekâ aydınları için, "Hiçbir kör, görme yetisinden onlar kadar yoksun değil!" denebilir sadece. Bu benzetme öyle sığ ki, buna ancak ortaçağın düşünme şekli yakışır.

YZ'nin "abartılı iddiaları"yla ilgili kaygılara ek olarak, pek çok bilgisayar bilimcisi, YZ'nin, özenli bilimsel standartlar benimsemeyen bir tür "yan etkinlik" ve şarlatanların yuvası olduğunu düşünüyordu. SRI'da bir iş için 1961'de ilk kez mülakata girdiğimde, oradaki bir araştırmacı, sinir ağları üzerine çalışmaya karşı beni uyarmıştı. Bu tür araştırmaların, henüz "olgunlaşmadığını" ve oraya katılırsam itibarımın zedelenebileceğini söylemişti.

Bana kalırsa, "saygınlık" tasasının, bazı YZ araştırmacıları üzerinde aptallaştırıcı etkisi olmuştur. Bunlar, "YZ gösterişli oluşundan ötürü eleştiriliyordu; artık somut ilerleme kaydettiğimize göre, saygınlığımızı kaybetme tehlikesi kalmadı" bile demiştir. Bu tutuculuğun sonucunda, (insan düşüncesine yardımcı olmaya adanmış YZ çeşidi), "zayıf YZ" araştırmalarına yoğunlaşılırken (insan düzeyinde zekâyı mekanikleştirmeye uğraşan YZ çeşidi) "güçlü YZ" araştırmalarından uzaklaşıldı. Bu durum fazlasıyla kötü, çünkü her ne kadar zayıf YZ'nin hedeflerinin önemli ve değerli olduğunu düşünsem de, insan beyninin yeteneklerini taklit eden bir insan eseri inşa etmek muazzam

38 Çevrimiçi ansiklopedi Wikipedia, Pierce maddesinde bu slogandan bahseder: http://en.wikipedia.org/wiki/John_R._Pierce.



Resim 24.5. Bir arama ağacı.

bir bilimsel başarı olacaktır; bu riske girmeye değer ve hiç de "iğrenç", "şeytani", "sinsi" ya da "prensipte imkânsız" bir girişim değildir bu.

24.2 ÖLÇEK SORUNLARI

24.2.1 Birleşimsel Patlama

Arama, yapay zekâda belirgin bir önem taşıdığı için, arama problemlerinin ne kadar zorlu olabileceğini belirtmek önemlidir. Tipik bir arama problemi, çoğunlukla düğüm noktalarından bir "ağaç" büyütmek olarak belirtilir, örneğin Arthur Samuel'in 5.4 numaralı başlıkta gösterilen dama ağacı ya da 5.1 numaralı başlıkta gösterilen kaydırmalı blok (sekizli bulmaca) arama ağacı gibi. Örneğin, eğer arama ağacında her düğüm noktasının olası üç "çocuk" düğüm noktası varsa (yani "dallanma çarpanı" üçse), ağacın üst kısmı Resim 24.5'teki gibi görünür.

Ağacın "birinci seviyesi"nde üç düğüm noktası varken, ikinci seviyesinde dokuz tane var vs. Genel olarak, dallanma çarpanı b olan bir ağacın d seviyesinde b^d adet düğüm noktası olur (yani b, d kere kendisiyle çarpılır). Eğer bir arama süreci en son adıma kadar dallanma çarpanı b'yle bütün bir ağaç oluşturup d seviyesinde tüm düğüm noktalarını barındırıyorsa, o zaman bu arama işlemenin üreteceği toplam düğüm sayısı, $\frac{b}{(b-1)}(b^d - 1)$ olarak hesaplanır. Lise cebir bilgisini hâlâ unutmamış olan okurlar, bu ifadelerin, d'nin "üstel" fonksiyon-

ları olduğunu fark edecektir. Arama ağacındaki düğüm noktalarının sayısı, ağacın derinliğinin üstel bir fonksiyonu olduğu için, arama işlemine, üstel işlem denir.

Eğer bir program, belli bir hedefe ulaşmak için, dallanma çarpanı üç, en derin seviyesi 10 olan bir ağacı araştırmak zorundaysa, 88.572 düğüm noktası yaratacak demektir. Bunun gibi rakamlar 1960'lar ve 1970'lerin bilgisayarlarının hünerleri kapsamında kalıyordu, dolayısıyla o bilgisayarlar, YZ'nin kimi basit "oyuncak problemlerini" çözebiliyordu; fakat daha gerçekçi problemlerin dallanma çarpanı çok daha yüksektir ve hedefler çok daha alt katlardadır. Örneğin, dallanma çarpanı 10, derinliği 20 kat olan bir ağacı (orta zorluktaki arama problemlerine karşılık gelen bir ağaç) araştırmak, 10^{20}'yi aşan sayıda düğüm noktası yaratmayı gerektirir ki bu oldukça imkânsız bir iştir. (10^{20}, birin ardında 20 adet sıfır eklemek anlamına geliyor, yani 100 kere milyar kere milyar.)

Bu tür arama işlemlerinin zorluğunun iki yüzü var: hesaplama zamanı ve bellek alanı. Bir an için sadece hesaplama zamanını ele alalım: Saniyede milyar tane düğüm noktası yaratabilsek bile (bu belki ancak düşüncede olası), yukarıda bahsettiğimiz dallanma çarpanı ve derinliği olan ağacı oluşturmak yine de 100 milyar saniye sürerdi (3.000 yıldan fazla).

Bellek alanına gelince, günümüzün kişisel bilgisayarlarında bile epey saklama alanı var; 100 gigabaytlık (100 milyar bayt) bellek olağandır. Tek düğüm noktasının yaklaşık bir bayt gerektirdiğini varsayarsak, alçakgönüllü boyutlu bir ağaç için bile bu tür yüz milyar bilgisayara denk saklama alanı gerekli olurdu.

Aramanın üstel doğasının oluşu, problemin boyutu büyüdükçe (dallanma çarpanıyla ya da arama ağacının derinliğiyle ölçülür), bunu çözmek için gerekli hesaplama zorluğunun çok hızlı artması anlamına gelir; yani, "birleşimsel patlama" denen şeyi yaratır.

Elbette, ilk YZ araştırmacıları bile birleşimsel patlamaları biliyordu. Sezgisel yöntemle ilgilenmelerinin sebebi budur. Ne yazık ki, sezgisel yöntem, arama işleminin üstel niteliğini değiştiremez; en iyi olasılıkla, dallanma çarpanını düşürür. O "patlama" yine de olur, ama o kadar hızlı değil. Örneğin, dallanma çarpanını 10'dan 4'e düşürüp yine 20 seviyeli bir arama yapmak, 4^{20}'yi aşkın, yani kabaca bir trilyon düğüm noktası üretmeyi gerektirir.

YZ eleştirmenleri, YZ'nin başarılarına ve beklentilerine dair karamsar değerlendirmelerinde bu soruna odaklanmıştı. Örneğin, Sir James Lighthill (16. Bölüm'de bahsettiğim raporunda) şöyle yazmıştı:

"YZ'nin yarattığı hayal kırıklıklarının genel sebeplerinden biri, 'birleşimsel patlamanın' sonuçlarını fark etmemektir". Daha önce bahsettiğim gibi, Lighthill'in raporu, Britanya'da YZ araştırmalarına ödenek bulmanın zorlaşmasına sebep olmuştur.

24.2.2 Karmaşıklık Kuramı

Problem çözmeye dair bilişimsel zorlukları kendine dert edenler sadece YZ araştırmacıları değildir. Bilgisayar biliminin "karmaşıklık kuramı" adlı dalı, farklı problem türlerini çözmek için farklı program türlerinin (en kötü ihtimalle) ne kadar zaman ve bellek alanı harcayacağı konusunu ele alır. Birçok şeyin yanı sıra şunu da sorarlar: "Problemin *boyutu*, bunu çözmek için gereken zamanı ve alanı nasıl etkiliyor?"

Yine birtakım örneklere göz atalım. Bir rakam listesi içinde en büyük rakamı bulmaya çalışan bilgisayar programının harcayacağı zaman, bu problemin boyutuyla, yani listedeki madde sayısıyla orantılıdır. (En kötü ihtimal, en büyük rakamın listedeki son madde olmasıdır; o durumda program listedeki her maddeyi incelemek zorunda kalır.) Bu tür bir programın, "doğrusal zaman" harcadığı söylenir. Benzer biçimde, belli bir madde belli bir listede var mı yok mu diye bakan bir bilgisayar programı da doğrusal zaman harcar. (En kötü ihtimal yine, maddenin listede son sırada olmasıdır.) İki örnekte de, listenin boyutunu iki katına çıkaracak olsak, gerekli zamanı ikiye katlamamız gerekir.

Bir isim listesini sıralamak, örneğin bunları alfabe sırasına göre dizmek, daha zor bir problemdir. Liste sıralayan programlar, harcadıkları zamana ve programlama karmaşıklıklarına göre değişir. Makul oranda basit sıralama programları, listedeki madde sayısının karesiyle orantılı olarak zaman harcar. Yani bu programların, 100 maddelik listeyi sıralaması için harcadığı zaman, sadece 50 maddelik listeyi sıralarken harcadıkları zamanın dört katıdır. (Öğe sayısını ikiyle çarpmak, sıralama zamanını ikinin karesi kadar, yani dört katı artırır. Aslında sıralama işi daha hızlı yapılabilir. Listeleri, boyut çarpı boyutun logaritmasıyla doğru orantılı sürede sıralayabilen programlar var.) Problemin boyutuyla, karesiyle, kübüyle ya da (en kötü ihtimalle) boyutunun başka "üstleriyle" orantılı olarak süre harcayan programların, "polinom zaman" içinde çalıştığı söylenir.

Bizzat problemler ise, bunları çözebilen en az karmaşık programların karmaşıklığına göre derecelendirilebilir. Örneğin, bir problemi doğrusal zamanda çözen bir program varsa, ve hiçbir program bundan

hızlı çözemiyorsa, o durumda bu problemin "doğrusal karmaşıklığı" olduğu söylenir. Belli bir maddenin belli bir listeye ait olup olmadığını bulmak, doğrusal karmaşıklığı olan bir problemdir. Polinom zamanda çözülebilen problemleri P sınıfı diye adlandırıyoruz. Bu sınıfın üyeleri arasında, ortak bölenin en büyüğünü hesaplamak ve bir sayının asal olup olmadığını belirlemek var.

Ne yazık ki, daha önce bahsettiğim gibi, pek çok YZ programında kullanılan arama yöntemleri, problemin boyutunun üstel katlarına denk gelen çalışma süresi gerektirir. Örneğin, dallanma çarpanı b ve derinliği d olan bir arama ağacı, b^d'yle orantılı olarak zaman harcar. Önceki örneğime dönecek olursam, dallanma çarpanı 3, derinliği 10 olan ağacın 88.572 adet düğüm noktası yaratılmasını gerektirdiğini görmüştük. Fakat derinliği iki katına çıkarıp 20 yapmak, dört çarpı 88.572 düğüm noktasını aramayı değil (arama zamanı, derinliğin karesiyle orantılı olsa böyle olurdu), 88.572 çarpı 88.572, yani neredeyse sekiz milyar düğüm noktası aramayı gerektirir. Üstel karmaşıklık, polinom karmaşıklığından çok ama çok daha berbattır!

Amerikalı bilgisayar bilimcisi Stephen Cook (1939-) [ayrıca eski Rus, şimdi Amerikalı bilgisayar bilimcisi Leonid Levin de (1948)], problemlerin ve problem çözümünde kullanılan programların karmaşıklığına dair anlayışımıza büyük katkılarda bulunmuştur. Özellikle, Cook (ve ondan bağımsız olarak Levin), belirlenimci olmayan polinom (NP) problemler sınıfını tanımlamıştır. Bu problem sınıfında, aday çözümler polinom zamanda sınanabilir (ama ille de çözülecek demek değildir bu). Örneğin, kaydırmalı blok bulmacasını çözmek üzere önerilen hamle dizisinin, bulmacayı gerçekten çözüp çözmediği polinom zamanda *sınanabilir*, fakat (bilindiği kadarıyla) çözüm *bulmak* için üstel zaman gerekir. Dolayısıyla kaydırmalı blok problemi, pek çok YZ programıyla birlikte, NP sınıfına mensuptur.[39]

NP sınıfındaki problemleri çözecek polinom programları var olabilir mi, olamaz mı bilinmiyor. Varsa, NP, P'ye eşit olur. Şimdiye dek, NP sınıfındaki problemleri çözen programlar (en kötü olasılıkla) üstel zamana gereksinim duymuştur. Acaba NP, P'ye eşit midir problemi, bilgisayar biliminde çözülmeden kalmış en ünlü problemlerden biridir. Pek çok insan NP'nin P'ye eşit olmadığını, çünkü öyle olsa şimdiye dek bunu bulmuş olacağımızı düşünüyor. (Clay Matematik Enstitü-

39 39 Kaydırmalı blok bulmacayla ilgili kanıt için bkz. Daniel Ratner ve Manfred Warmuth, "Finding a Shortest Solution for the N*N-extension of the 15-puzzle Is Intractable", *Journal of Symbolic Computing*, c. 10, 1990, s. 111-137.



Resim 24.6. Jacob (Jak) Schwartz. (Fotoğraf, Diana Schwartz'ın izniyle kullanılmıştır.)

sü, P-NP meselesinin çözümü için 1.000.000 dolar ödül koydu; yani bunlar eşit mi, değil mi gösterene bu para verilecek.)[40]

24.2.3 Aklı Başında Bir Değerlendirme

Karmaşıklık kuramında elde edilen bu sonuçlar yüzünden, bazı insanlar yapay zekâdan beklentiler konusunda şüpheye düşmüştür. En etkili ve bana kalırsa en zeki değerlendirmelerden birini, matematikçi ve bilgisayar bilimcisi Jacob T. Schwartz (1930-2009; Resim 24.6) kaleme almıştı. DARPA'daki göreviyle bağlantılı olarak bu isimden daha önce bahsetmiştim. Schwartz, "Yapay Zekânın Sınırları" başlıklı makalesinde,[41] insan beyninin sıradışı güçlerinin, devasa hesaplama ve saklama yeteneğini kullanma becerisinden doğduğunu yazar: "Bu yetenekleri sayesinde beyin, görece düzensiz biçimde sunulan bilgiyi düzenleyip, içsel olarak düzenli yapılar oluşturur ve incelikli, tutarlı olan simgesel ve gerçek dünya eylemleri bu yapıları temel alabilir". Schwartz'ın iddiasına göre, beyne rakip olmak için YZ'nin ihtiyaç duyduğu ve ulaşmak için boşuna çaba sarf ettiği şey, "bir tür bilgisayar eylemine doğrudan rehberlik edecek tutarlı yapılardır... bu yapılar,

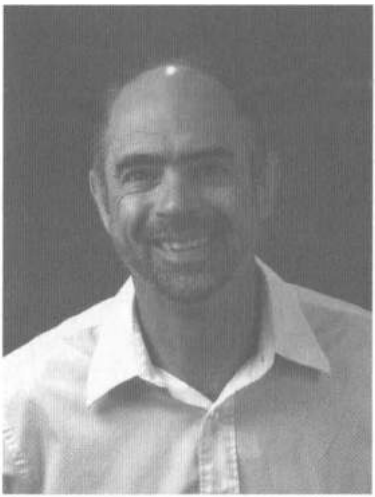
40 Bkz. http://www.claymath.org/prizeproblems/index.htm

41 Jacob Schwartz, "Limits of Artificial Intelligence", *Encyclopedia of Artificial Intelligence*, c. 1 içinde, Stuart C Shapiro ve David Eckroth (ed.), New York: John Wiley and Sons, Inc. 1987, s. 488-503.

görece düzensiz, parça parça girdilerden otomatik olarak yaratılmalıdır". Bu tür "düzenli yapıları" yaratma kabiliyeti, muazzam bir ilerleme anlamına gelir, çünkü bilgisayarlar, "*bütünüyle yapılandırılmış bilgi* kabul ettiği, aldığı, yararlandığı" problemleri çözme konusunda "insanları zaten epey aşmıştır". Schwartz şöyle yazmıştı: "Bilgisayarları ayrıntılı programlama ihtiyacının dayattığı temel engel aşılabilirse, bilgisayarlar, dünyanın tüm kütüphanelerindeki bilgileri yalayıp yutar ve bu bilgileri, insanı aşan bir etkinlikle kullanabilir". (Elbette günümüzde, kütüphanelerin ötesinde tüm bilgiler ve yanlış bilgiler Dünya Çapında Ağ'da var.)

YZ araştırmacılarının, parça parça girdilerden otomatik olarak bir yapı üretme amacıyla faydalandığı yöntemler, sezgisel aramayı temel alıyordu; ya mantıksal çıkarım zincirlerini ya da ağaçlardaki (genelde ise grafiklerdeki) yolları aramak söz konusuydu. Mantıksal tümdengelimden (başka amaçların yanı sıra), soruları yanıtlamak, uzman tavsiyesi vermek, program ve plan yaratmak amacıyla yararlanılmıştır. Grafiklerde yol arama yaklaşımıysa, geometri teoremleri kanıtlama, cümleleri çözümleme, robot planları üretme amacıyla kullanılmıştır. Tüm bunlar, yapılandırılmamış girdilerden (Schwartz'ın terimini kullanmak gerekirse) bir *yapı* üretmenin örnekleridir. Ancak tüm bu "başarılar", ya önemsiz ve ufak problemler üzerinden ya da katı şekilde sınırlandırılmış problemler üzerinden elde edilmiştir. Ne yazık ki, Schwartz'ın doğru olarak belirttiği gibi, bugüne dek "dağınık ve parça parça girdilerden genel olarak faydalı simgesel yapılar" yaratma çabalarının hepsi istisnasız birleşimsel patlama tarafından mağlup edilmiştir. Schwartz, Dreyfus'un şu suçlamasını tekrarlayarak durumu özetler: "... günümüze dek, YZ araştırmalarının tarihçesi, belirli alanlarda çok kısıtlı başarılardan meydana gelmiştir; bunların ardından, ilk başarıların akla getirdiği başarıların genel hedeflere ulaşamayışı gelmiştir..."

Schwartz'ın YZ hakkındaki görüşlerinin, birtakım sonuçlar doğuran etkileri olmuştur, çünkü, 23.4 numaralı başlıkta bahsettiğim gibi, 1987'den 1989'a kadar DARPA ISTO'nun genel müdürlüğünü yapmış, YZ araştırmalarına birtakım kesintiler getirmiştir (1987'de iptal edilen araştırma projelerimden biri dahil). Schwartz, her ne kadar genelde YZ çalışmalarına önem vermese de, "klasik bilim ve algoritma tekniklerinin uygulandığı YZ alanlarının, sadece daha az odaklanılmış yaklaşımların işe yaradığı daha derin problemlerle uğraşan alanlara kıyasla daha hızlı ilerlemesinin beklenebileceğini" yazmıştır. Örnek olarak, şu probleme atıfta bulunmuştu: "Belli şekillerde engeller barındıran bir ortamda bilinen şekilli en az bir nesnenin, engellerle ya da



Resim 24.7. Richard Korf. (Fotoğraf, Richard Koch'un izniyle.)

öbür hareketli nesnelerle çarpışmaksızın belirli bir noktadan başka bir noktaya hareket etmesini sağlamak."

Her ne kadar karmaşıklık kuramının bu sonuçları, YZ'nin hızlı ilerleyişinin yoluna çıkan "hız tümseklerinden" biri olsa da, YZ araştırmacıları, birleşimsel patlama probleminin etrafından dolanmanın çeşitli yollarını bulmuştur. Örneğin, karmaşıklık sonuçlarının en kötü ihtimalle performansını temel alışına ve en kötü olasılığın ele alındığı durumlara kıyasla çözümlerin daha hızlı bulunabileceğine işaret ederler.

Los Angeles California Üniversitesi YZ araştırmacılarından Richard Korf'un (1956- ; Resim 24.7) çalışmalarından bahsedeceğim. Korf, son derece zorlu problemleri ele alan çalışmalarıyla bilinir; arama yöntemlerinde yeni fikirler keşfetmek için kaydırmalı blok bulmacasını ve başka bulmacaları laboratuvar "meyve sineği" olarak kullanır çoğunlukla.

Kaydırmalı blok bulmacası örneğini, sezgisel arama işlemini açıklamak amacıyla kullandığımı hatırlarsınız. Benim verdiğim örnek 3×3 dizilişle sekiz kareden oluşuyordu; klasik hali, 4×4 dizilişle on beş kareden oluşur. Daha büyük halleri de kafada canlandırılabilir, örneğin 5×5 dizilişle yirmi dört kare gibi. 4×4 bulmaca, sezgisel arama yöntemi için zaten oldukça zorlu bir problem oluşturuyor. Aslında Jack Schwartz, arama süreçlerini ölçeklendirme zorlukları hakkındaki yorumunda şöyle yazmıştır: "... [3×3 bulmacanın] durum grafiği, 9! yani 362.880 düğüm noktasından meydana geliyor, dolayısıyla basit bir problem için bile, kaba güçle grafikte arama, külfetli hale gelir. Buna karşılık gelen 4×4 bulmacada, durum uzamı 16! yani 10^{13}'ü aşkın dü-

ğüm noktası içerir, dolayısıyla bu arama uygulanamaz haldedir". [Doğrusu Schwartz, iki örnekte de hedefi 2 katıyla ıskalamıştı (aslında hiç önemli değil). 1879'dan beri herhangi bir başlangıç dizilişinden sonra, olası tüm dizilişlerin ancak yarısına ulaşabileceğiniz bilinir.[42] Dolayısıyla, sekiz kareli sürümde, durum grafiğinin bütünü, 9!/2 = 181.440 boyutunda iki ayrı grafikten oluşur.]

Ancak Korf, 4×4 bulmaca örneklerini çözmek amacıyla sezgisel arama programları yazmakla kalmamış, 1996 tarihli bir makalenin özetinde, öbür yazarla birlikte şunu kaleme almıştır:[43]

> Yirmi Dörtlü Bulmacanın, yani gayet iyi bilinen kaydırmalı blok bulmacasının 5×5 sürümünün tüm rasgele örnekleri için ilk optimal çözümü bulduk. Bu probleme yaptığımız yeni katkı, daha güçlü ve makul bir sezgisel fonksiyondur.
>
> ...
>
> Sezgisel arama problemlerinin ölçeği artırıldıkça daha güçlü sezgisel fonksiyonların hem gerekli hem hesaplı olduğunu gözlemledik.

5×5 bulmacanın arama uzamı, 4×4 bulmacanınkinden yaklaşık trilyon kat büyüktür. Bunun 7.756×10^{24} civarında düğüm noktası bulunur.

Eğer "en iyi" çözümlerde ısrarcı olmak yerine "iyi" ya da yaklaşık çözümlerle yetinilse arama işlemi kolaylaşır. İleriki bir bölümde, büyük arama problemlerini ele alma konusunda kaydedilmiş çarpıcı ilerlemeleri anlatacağım; bu ilerleme, hız tümseklerini bir parça düzlemiştir.

Ancak, karmaşıklık kuramının sonuçları etrafından dolanmanın bu yollarına rağmen, YZ çalışanları başka yetersizlikleri de fark etmeye başladı; şimdi bu konuya değiniyorum.

24.3 KABUL EDİLMİŞ YETERSİZLİKLER

YZ araştırmacıları, pratik önemi olan problemlerle karşı karşıya geldikçe, bazı zorlukları bizzat kabul etmek zorunda kalmıştır. Bunlar, kimi uygulama alanlarında ortaya çıkmıştı. Sadece birkaçından bahsedeceğim.

42 W. W. Johnson ve W. E. Story, "Notes on the 15 Puzzle", *American Journal of Mathematics*, c. 2, 1879, s. 397-404. Bu alıntı için Richard Korf'a teşekkür ediyorum.

43 Richard E. Korf ve L. A. Taylor, "Finding Optimal Solutions to the Twenty-Four Puzzle", *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, Menlo Park, CA: AAAI Press and Cambridge, MA: MIT Press. Ağustos 1996, s. 1202-1207.

Örneğin, çözümü zor matematik teoremlerini kanıtlama girişimlerinde, teorem kanıtlayıcı programlar, ara sonuçları saklamaya yarayan alanı çabucak tüketiyordu. Fakat insanlar (en azından bazıları), teorem kanıtlayabiliyor. Peki, bilgisayarların kullanmadığı hangi yöntemlerden yararlanıyorlar acaba? Muhtemelen matematikçiler başarılarında sezginin, yargının, deneyimin, matematikte uzmanlaşmanın, vesairenin önemli olduğunu söyleyecektir. Şimdiye kadar, bilgisayarlara bu yetileri kazandırmanın zor olduğu anlaşılmıştır.

MAC HACK VI ve CHESS 4.6 gibi gayet güzel satranç oynayan iki örneğe rağmen, oyun oynama konusunda bilgisayarlar, dünya şampiyonlarını yenmekten uzaktı. Aslında, 1978'de Toronto'da düzenlenmiş Kanada Ulusal Sergisinde, Northwestern Üniversitesi'nden David Levy, Bilgisayar Satrancı Şampiyonu unvanlı CHESS 4.7'yi mağlup etmiş, böylece on sene önce John McCarthy ve Donald Mischie'yle girdiği bahsi kazanmıştı. Levy'nin belirttiği gibi: "Bilgisayar önünde net galibiyet almayı başardım; üç oyunu ben, bir oyunu o almış, biri de berabere bitmişti (altıncı oyunu oynamaya gerek kalmamıştı), ve bu maçla birlikte tutuştuğumuz bahsi de kazanmış oldum."[44]

Şampiyon olma niteliğini açıklayan şey ne? Satranç şampiyonlarının, bilgisayarlara kıyasla daha fazla satranç konumuna bakması olası değil. Bununla birlikte, muhtemelen oyun ağacının en önemli dallarında çok daha ileriyi görebiliyorlar. Deneyimleri sayesinde, incelemeye değer aday konumların potansiyelini değerlendirip değersiz konumları iredelemeyi bırakıyorlar. Belki de satranç hamlelerini tek tek gözden geçiren satranç programlarına kıyasla daha stratejik düşünüyorlar.

İnsanlar için herhalde satrançtan da zorlu başka bir oyun Go'dur; 4.000 yılı aşkın süre önce Çin'de doğmuş olan bu oyun, günümüzde Asya ülkelerinde büyük rağbet görüyor. Go oyununda iki oyuncu, masa üstüne çizilmiş yatay ve dikey çizgilerden meydana gelen 19×19'luk ızgaranın kesişim noktalarına siyah ve beyaz taşları sırayla yerleştirir. Oyunun kurallarını burada anlatacak değilim; tek söyleyeceğim, oyuncu taşı yerleştirirken (en azından oyunun erken aşamalarında), neredeyse 361 (19×19) konumdan birine karar vermek durumundadır. En güçlü arama işlemleri için bile, her düğüm noktasının yaklaşık 361 ardılının olduğu bir ağaçta arama yapmak söz konusu olamaz. İnsan oyuncular başka stratejiler kullanıyor olmalı ve bu stratejiler her neyse, YZ araştırmacıları bunları hâlâ bilmiyor. Pek çok YZ araştırmacısına

44 David Levy, *Robots Unlimited: Life in a Virtual Age*, Wellesley, MA: A. K. Peters, Ltd., 2006, s. 84.

göre Go oyununda gösterilen performans, satrançta gösterilen performansa kıyasla YZ'nin daha iyi bir ölçüsüdür. Son gelişmelerin bir kısmını bu kitabın ilerleyen sayfalarında anlatıyorum.

Her ne kadar uzman sistemler, tıp, yerbilim, kimya ve başka sınırlı alanlardaki özgül sorunlar hakkında yarar sağlayacak biçimde (hatta iktisadi yarar sağlayarak) akıl yürütse de, bu sistemlerin "kırılgan" olduğu söylenebilir; yani, kendi alanlarının dışındaki problemlerle, hatta kurallarında belirtilmemiş bilgiye ihtiyaç duyuyorlarsa kendi uzmanlık alanlarındaki problemlerle karşılaştıklarında çökerler. Bir şeyi bilmiyorlarsa bilmiyorlar demektir, dolayısıyla insan uzmanların daha iyi iş çıkardığı durumlarda yanlış yanıtlar verebilirler. John McCarthy, tıp uzman sistemi MYCIN'le bir etkileşiminde, farazi bir hasta hakkında bilgi girerken, hastanın erkek olduğunu ama amniyosentez geçirdiğini yazmış. MYCIN tüm bu bilgileri şikâyetsiz kabul etmiş! Erkek hastaların gebe kalamayışı, MYCIN'e verilmesi gereken "uzman bilgisi"nin parçası sayılmamış demek ki.

Uzman sistemlerin kırılgan oluşunun sebeplerinden biri, "sağduyudan" yoksun olmalarıdır.[45] İnsanlar, eğitim ve mesleki deneyim aracılığıyla elde edebilecekleri uzman bilgisine ek olarak, bol bol genel bilgiye de sahiptir. Örneğin, sadece kadınların gebe kalabileceğini, şemsiyenin güneşe ve yağmura karşı koruduğunu, belirli kuşların göç ettiğini, marketlerden gıda satın alınabileceğini ve milyonlarca ama milyonlarca olguyu bilirler. Michigan Üniversitesi YZ araştırmacısı ve profesörü Benjamin Kuipers (daha önce Austin'deki Texas Üniversitesi'ndeydi), sağduyuyu şu şekilde tanımlamıştı: "Sağduyu bilgisi, dış dünyanın yapısı hakkındaki bilgidir; her normal insan, fiziksel, uzamsal, zamansal ve toplumsal çevrenin gündelik taleplerini makul bir başarı oranıyla karşılamasına olanak tanıyan bu bilgiyi, yoğunlaşmış bir çaba sarf etmeksizin edinip uygulayabilir."[46]

Çocuklar yetişkinliğe girdikçe ve yetişkinler olgunlaştıkça, bu genel bilgi yavaş yavaş elde edilir. Örneğin, çocuklar olasılıkla küçük plastik şişelerdeki küçük tabletlerin yutulduğunda zararlı olabile-

45 Bkz. John McCarthy, "Some Expert Systems Need Common Sense", Heinz Pagels (ed.), *Computer Culture: The Scientific, Intellectual and Social Impact of the Computer, Annals of the New York Academy of Sciences*, c. 426, Kasım 1995. Şu internet adresinde: http://www-formal.stanford.edu/jmc/someneed/someneed.html.

46 Benjamin Kuipers, "On Representing Commonsense Knowledge", *Associative Networks: The Representation and Use of Knowledge by Computers* içinde, Nicholas V. Findler (ed.), s. 393-408, New York: Academic Press, 1979. Şu internet adresinde: ftp://ftp.cs.utexas.edu/pub/qsim/papers/Kuipers-csk-79.ps.gz.

ceğini bilmez (şişelerin çocuk güvenlikli kapakları olmasının sebebi budur), ergenler, sekiz yaşındakilerin bilmediği pek çok şeyi bilir ve örneğin *The New Yorker* okurlarına bu derginin kitap ve film incelemelerini anlama olanağı sunan bilgi, genelde ergenlerin bildiklerinin çok ötesine uzanır. Üstelik, farklı ülkelerde ve kültürlerde insanların sağduyu bilgisi elbette farklı olacaktır.

Bana kalırsa her insanın bilgisi, gitgide büyüyen bir ağaç olarak düşünülmeli: Ağacın gövdesi ve alt dalları, "sağduyu"yu meydana getirirken, üst dallar, kişinin öğrenmiş olabileceği özel disiplin "uzmanlıklarını" meydana getirir. Bu ağaç mecazı aynı zamanda, üst dallardaki bilginin, gövdede ve alt dallarda bulunan kavramlardan faydalandığını vurgularken de yararlı.

Önceki bir bölümde, doğal dildeki cümleleri tamamen anlamanın, insanlarda olan ama bilgisayarlarda hâlâ olmayan sağduyu bilgisine gereksinim duyduğunu gördük. Bilgisayarları sağduyuyla donatmaya yönelik göz korkutucu beklenti, birbirine oldukça zıt iki tepki doğurmuştur. Bazılarına göre bu zorluk, yakın gelecek için YZ (en azından güçlü YZ) olasılığını devre dışı bırakıyor. Gerçi başkaları, "bu işi sürdürelim" diyebilir. (Bu hevesli insanların çalışmalarından ileriki bir bölümde bahsedeceğim.) Elbette, nasıl her insanın tüm konular hakkındaki doğal dil cümlelerini anlaması beklenmiyorsa, herhangi bir YZ sisteminin tüm doğal dil cümlelerini anlaması da beklenmez. İnsanların sınırları var, YZ programlarının da kendilerine özgü sınırları olacaktır. Bu beklenti, zeki insanlar yaratma girişimlerimizi nasıl kısıtlamıyorsa, zeki programlar yaratma girişimlerimizi de sınırlamamalı.

24.4 "YZ KIŞI"

1980'lerin ilk yarısı boyunca, devlette ve sanayide pek çok YZ hamisi, YZ'nin yapabileceklerine dair beklentileri epey yükseltti. Bu yersiz iyimserliklerinin suçu kısmen, abartılı vaatlerde bulunan YZ araştırmacılarına düşüyor hiç kuşkusuz. Gerçekçi olmayan umutlara dayanan sistemlerin yaratılamayışı, bahsetmiş olduğum artan eleştirel yorumlarla birlikte, 1980'lerin ortasından sonuna dek uzanan dönemde bir araya gelince, "YZ kışı" denen dönemi getirdi.

Aslında, 1984'te düzenlenen AAAI Ulusal Kongresi'nde önde gelen kimi YZ araştırmacıları, "YZ'nin 'Karanlık Çağı' – Bundan Kaçabilecek miyiz Yoksa Sağ Salim Atlatabilecek miyiz?" başlıklı oturumda bu olasılığa karşı insanları uyarmıştı. Oturum başkanı olan

Yale Üniversitesi'nden Drew McDermott, oturumu açarken şöyle konuşmuştu:[47]

> Bugünlerde YZ etrafında dönen bütün o ticari harala güreleye rağmen, en az son dört yıldır bu alanda çalışan YZ araştırmacıları arasında, pek çoğunuzun aşina olduğuna emin olduğum derin bir huzursuzluk hali var. Bu huzursuzluğun sebebi, belki de YZ'den beklentilerin çok yüksek olup bunun felaketle sonuçlanması kaygısıdır.
>
> ... Bence, "YZ kışının" yaşanmamasını sağlamak amacıyla birtakım adımlar atmamız önemli: Kendimizi terbiye etmeli, kamuyu eğitmeliyiz.

Fakat bu "terbiye" ve "eğitim" hayata geçirilmişse bile, endişe duyulan gerilemeyi engellemeye yetmemiştir. 1980'lerin ikinci yarısında AAAI üyelerinin sayısı yavaşça düştü. 1996'ya gelindiğinde, 4.000 ile 5.000 arası bir rakamda sabitlenmişti. *AI Magazine*'e verilen reklam sayısı da azaldı, tıpkı YZ konferans sergilerine devletin ve sanayinin katılımının seyrekleşmesi gibi. Bazı YZ şirketleri kapılarına kilit vurdu, ayrıca daha büyük kimi bilgisayar donanımı ve yazılımı şirketlerinde YZ araştırmalarına son verildi. Alex Roland'a göre, 1987 ile 1989 arasında DARPA'nın temel YZ ve Stratejik Bilişim araştırmalarına ayırdığı bütçe, 47 milyon dolardan 31 milyon dolara indi. (Ancak, Alex Roland'a göre, o dönemde CMU'nun konuşma anlama programı ve özerk taşıt programı için ayırdığı bütçe artmıştır.)

Ancak bu kış sadece bir mevsim sürdü; kış uykusuyla değil, yola devam etmek üzere tazelenmiş çabalarla geçmiştir bu mevsim. Önümüzdeki bölümlerde anlatacağım gibi, bu esnada bazı yeni fikirler keşfedildi ve eski fikirler ek güçlerle pekiştirildi.

47 Roger Schank, "The Dark Ages of AI: A Panel Discussion at AAAI-84", *AI Magazine*, c. 6, no: 3, Güz 1985, s. 122-134.

25
ANLAŞMAZLIKLAR VE ALTERNATİF PARADİGMALAR

Henüz bahsetmiş olduğum türdeki zorluklar, bizzat YZ araştırmacıları arasında pek çok uyuşmazlığı alevlendirmişti. YZ'nin yavaşlamasıyla hayal kırıklığına uğrayan ve YZ için akıllarında farklı yaklaşımlar olan insanlar, YZ'nin, egemen paradigma yerine şu ya da bu pradigmaya ihtiyacı olduğunu ileri sürer oldular; egemen paradigmaya John Haugeland "güzel eski usül YZ" yani ETYZ diyordu. Elbette ETYZ'nin öncelikli gerekçesi, Simon ile Newell'ın şu inancıydı: "fiziksel simge sistemi, zeki edimler için gerekli ve yeterli araçları barındırır". Fakat 1980'lerde ETYZ eski gücünü yitirmeye başladı ve bizzat YZ araştırmalarının meydan okumaları karşısında duramaz oldu; bu meydan okumaların, Searle, Dreyfus, Penrose ya da alan dışından başka insanların meydan okumalarına kıyasla daha ciddiye alınması şarttı. Bu bölümde, o iç uyuşmazlıkların bazılarını anlatıp ortaya çıkan birkaç yeni paradigmadan bahsedeceğim.

25.1 MANTIK HAKKINDA

ETYZ yaklaşımını benimseyenler arasında, John McCarthy'nin öncüsü olduğu, mantıksal gösterim ve mantıksal uslamlama yöntemi fikirlerinden faydalananlar vardı. Bu insanlara zaman zaman "mantıkçılar" denmiştir. (Ben de onlar arasındaydım; 1987'de basılmış *The Logical Foundations of Artificial Intelligence* [Yapay Zekâ'nın Mantıksal Temelleri] başlıklı kitabın eşyazarıydım.)[1] Yale Üniversitesi profesörlerinden Drew McDermott (doktorasını MIT'den almıştı), YZ'de mantığın rolünden kuşku duymaya başlayanlardan biriydi. Bu, önemli bir olguydu, çünkü bizzat McDermott önde gelen bir mantıkçıydı, fakat 1987'de yayımlanmış etkili makalesinde, "pek çok uslamlamanın, tümdengelimli ya da neredeyse tümdengelimli olarak çözümlenebileceği önermesi yanlıştır", sonucuna varmıştı.[2]

1 Michael Genesereth ve Nils Nilsson, *Logical Foundations of Artificial Intelligence*, San Francisco: Morgan Kaufmann Publishers, 1987.

2 Drew McDermott, "A Critique of Pure Reason", *Computational Intelligence*, c. 3, no: 3, Ağustos 1987, s. 151-160.

Şunu da ekler:

> Ne yazık ki, mantıkçı projeyi ne kadar zorlarsanız, o kadar az tümdengelim bulursunuz. Bunun yerine bulduğunuz çıkarımlar öyle dolaysızdır ki bunların, tümdengelimli olmayan bileşenler barındıran tümdengelimler olması *şart*.
>
> ...
>
> Son kez ne zaman plan yaptığınız anı düşünün ve bu planın işe yarayacağını *kanıtlayabilir* miydiniz diye kendinize sorun. Olasılıkla, planın işe yaramayacağı ön koşulu kolaylıkla sayardınız ama yine de devam edip planı benimserdiniz.

Mantıkçılardan (bendeniz dahil) ve mantıkçılara yakın duran kişilerden "yorum" yazması rica edilmiş ve bu yorumlar McDermott'un makalesinin yanında basılmıştır. YZ'deki önemli uyuşmazlıklardan birini görmek isterseniz bu makaleye göz atın. Yapay zekâda mantığın rolüne dair tartışmalar, YZ'nin mantıktan faydalanışını yeniden şekillendirmeye yardımcı oldu ve uzatmalı biçimiyle mantık, bildirimsel bilginin gösteriminde birincil araç olmayı hâlâ sürdürüyor.

25.2 BELİRSİZLİK

Mantıksal gösterimlerin kullanımına yöneltilmiş başka bir itiraz ise, mantıksal cümlelerin ya doğru ya yanlış olma zorunluluğunun bulunduğu ama insan bilgisinin büyük oranda belirsiz olduğu olgusuna dayanıyordu. Hem MYCIN hem PROSPECTOR (başka uzman sistemleriyle birlikte), belirsizliği uyarlamayı başarmıştı; MYCIN, "kesinlik çarpanları"ndan faydalanırken, PROSPECTOR olasılık değerleri kullanmıştı.

Belirsizlikle başa çıkmak için başka fikirler de öne sürülmüştü. Olasılık kullanımının iki yolundan bahsedeceğim. Bir tanesi, Dempster-Shafer (D-S) kuramıdır; önermelere belli bir inanç derecesi atamaya ve bağımsız bulgulara dayalı inanç derecelerini birleştirmeye yarar.[3] D-S kuramı, bir karara varmak için farklı kaynaklardan gelen verilerin birleştirilmesi (ya da D-S çalışanlarının sözüyle "kaynaştırılması") gereken problemlerde yaygınlıkla kullanılır.[4]

3 İnternette (makalelere atıflarla birlikte) kısa bir özet için bkz. Glenn Shafer, "Dempster-Shafer Theory", 2002; şu internet adresinde: http://www.glennshafer.com/assets/downloads/articles/article48.pdf.

4 Örneğin bkz. David L. Hall ve Sonya A. H. McMullen, *Mathematical Techniques in Multisensor Data Fusion*, Norwood, MA: Artech House, Inc., 2004.



Resim 25.1. Lotfi Zadeh (Fotoğraf, Lotfi Zadeh'in izniyle.)

Olasılık kullanımının başka bir seçeneği ise, bilgisayar bilimcisi Lotfi Zadeh'in (1921-2017 ; Resim 25.1) icat ettiği "bulanık mantık"tır.[5] Bulanık mantık, yargıların doğruluk değerinin 1 (mutlak doğru) ile 0 (mutlak yanlış) arasında bir değer almasına olanak tanıyor. Bu yaklaşım, bulanık küme kuramını temel alır; bu kuramda, bir kümeye üye olmak, "kümede" ya da "kümede değil" arasındaki ara değerleri alabilir; yani, "kısmen kümede olmak" mümkündür. Zadeh, örnek olarak uzun boylu insanlar kümesinden bahseder. Uzun boy tanımına göre, diyelim ki 5 ayak 10 inç (177,8 cm) olan John, 0,7 dereceyle "uzun boy" kümesinden sayılıyor. O halde "John uzun boyludur" önermesinin doğruluk değeri 0,7'dir. Örneğin, 0,95'lik doğruluk değeri, "John oldukça uzun boyludur" önermesine karşılık gelir. "Hafiften", "bir yere kadar", "çok" gibi niteleyiciler, bunları kullanan önermeler için bulanık doğruluk değerlerine kolaylıkla çevrilebilir.

Yargı birleşimlerinin doğruluk değerleri, bulanık mantıkta şöyle hesaplanır: Eğer A ve B iki yargıysa, "bitiştirici" birleşimlerinin (A Ve B) doğruluk değeri, A'nın ve B'nin doğruluk değerlerinden küçük olanına, oysa "ayırıcı" birleşimin (A Ya Da B) doğruluk değeri, A'nın ve B'nin doğruluk değerlerinden büyük olanına eşittir.

5 Zadeh'in özgün makalesi: Lotfi Zadeh, "Fuzzy Sets", *Information and Control*, c. 8, 1965, s. 338-353; şu internet adresinde: http://www-bisc.cs.berkeley.edu/Zadeh-1965.pdf. Özet için bkz. Cox, "Fuzzy Fundamentals", *IEEE Spectrum*, c. 29, no: 10, 1992, s. 58-61. AAAI'nın, bulanık mantığa dair bir internet sayfası var (pek çok makale, uygulama ve öğretici malzemeyle birlikte): http://www.aaai.org/AITopics/html/fuzzy.html.

Zadeh, bahsettiği bu doğruluk değerlerinin ve küme üyeliğinin, olasılık olarak yorumlanamayacağına işaret eder. Onun sebepleri bizi burada ilgilendirmiyor; yine de, bu sorun ihtilaflıdır. (Çoğu istatistikçi, olasılık kuramının, belirsizlikle başa çıkmak için tek somut matematiksel yöntem olduğunu iddia eder.) Bulanık mantık ve birtakım uygulamaları hakkında, özellikle de kontrol sistemlerindeki uygulamaları hakkında kapsamlı bir literatür olduğunu belirtmek yeterli olacaktır.[6]

Bulanık kontrol kullanımına dair sık sık atıfta bulunulan örneklerden biri, Maytag Şirketinin "Intellisense" bulaşık makinesidir. Bir basın açıklamasına göre:[7]

> Iowa'nın Newton kentinde Maytag Şirketi, dünyanın ilk "akıllı" bulaşık makinesi dediği makineyi icat etmiştir. Bilgisayarlı bu makine, düğmesine basılmasıyla, herhangi bir bulaşık yükü için ideal yıkama devir sayısını belirler. Bulaşık makinesinin bariz uslamlama yeteneği, bulanık mantık kontrolünden ve ileri sensör modülünden kaynaklanıyor; pompaya yerleştirilmiş olan bu modül, gıda parçacıklarını, su sıcaklığını, deterjanı ve yıkama devrini ölçer.

Zadeh'e göre, bulanık mantık, "yumuşak bilişim" dediği genel bir çabanın bileşenidir; bu disiplin, "geleneksel (katı) bilişimden farklıdır, zira katı bilişimin aksine, sapmalara, belirsizliğe ve kısmi doğrulara hoşgörülü yaklaşır. Aslında, yumuşak bilişim insan zihnini örnek alır."[8]

Olasılık kuramının bu yollarının aksine, ileriki bir bölümde bahsedeceğimiz Bayes ağlarının icadı, belirsiz bilgiyi göstermenin ve bununla akıl yürütmenin olasılık değerlerini temel alan yöntemleri yeniden canlandırmıştır.

25.3 "YAMAMA SİSTEM"

Başka bir uyuşmazlık da, zeki davranışların temelini oluşturan mekanizmanın (ya da mekanizmaların) doğası hakkındaydı. Arama, öğrenme, mantık ya da çok fazla miktarda sağduyu bilgisine dayanan birleştirici genel bir ilke peşinden koşanların aksine Marvin Minsky,

6 Örneğin bkz. Kevin M. Passino ve Stephen Yurkovich, *Fuzzy Control*, Menlo Park, CA: Addison Wesley Longman, 1998. (Kitap artık basılmıyor ama şu internet adresinden indirilebilir: http://www.ece.osu.edu/~passino/FCbook.pdf.)

7 Bkz. *Machine Design*, Mart 1995.

8 Berkeley İnisiyatifinin Yumuşak Bilişim internet sitesine bkz. http://www-bisc.cs.berkeley.edu/.

zekânın (en azından insan zihninin sergilediği zekânın) bir tür "yamama sistem" olduğunu iddia etmişti. ("Yamama sistem"in farklı sözlük tanımları arasında şunlar da var: 1. Birbirine pek uymayan öğelerden ya da aslında başka uygulamalar için amaçlanmış öğelerden oluşan sistem, özellikle de bilgisayar sistemi. 2. Bir probleme getirilen hantal ya da kaba çözüm.) Minsky'ye göre zekâ, gevşek bir etkileşim içinde bulunan, bazen işbirliği yapan bazen rekabet eden, belki yüzlerce ya da binlerce özel-amaçlı doğaçlama mekanizmanın sonucuydu ve evrimleşen insanın karşılaştığı çok sayıda sorunu çözmek üzere doğmuştu. Minsky'nin sözcükleriyle söylemek gerekirse:[9]

> Beynin işlevleri, küçük bir ilkeler kümesini falan temel almıyor. Bunun yerine, yüzlerce, belki binlerce ilke var. Başka bir deyişle, bana kalırsa beynin her parçası mühendislerin yamama sistem dediği şeydir; yani genel ve toplu bir plan olmaksızın, gerekli olan yerlere birtakım makineler eklenerek bir probleme getirilmiş eğreti bir çözümdür: sonuçta insan zihnine, yamama sistem olarak bakılmalı, işte beynin yaptığı iş budur. Buna dair bulgular son derece açık: Kocaman herhangi bir sinirbilim ders kitabının dizin kısmına göz atarsanız, insan beyninin, farklı işler yapan yüzlerce parçadan (altbilgisayarlardan) meydana geldiğini görürsünüz. Neden beynimiz bunca parçaya gereksinim duyuyor? Eğer zihnimiz birkaç temel ilkeye dayansaydı, böylesi bir karmaşıklığa herhalde ihtiyaç duymazdık.

Elbette, beynin bir tür yamama sistem olması, bilgisayar zekâsının da öyle olması gerektiği anlamına gelmez. Bununla birlikte, bazı YZ araştırmacıları, belli problemleri çözmek üzere tasarlanmış, deneylerle elde edilmiş doğaçlama altprogram derlemelerinden oluşan sistemleri öne çıkarmıştır. Bu araştırmacılar, kuram temelli ilkelere dayalı programları öne çıkaran "muntazamlar"dan kendilerini ayırmak amacıyla, kendilerine "dağınıklar" demiştir. (Bu terimler ilk olarak 1970'lerde Roger Schank tarafından kullanılmıştır; doğal dil işleme sistemi inşa etme yaklaşımını, McCarthy'nin ve benzerlerinin kuram temelli çalışmalarından ayırt etmek amacıyla bu sözcüklerden faydalanmıştı.) 1981'de Bilişsel Bilim Derneği'nin yıllık toplantısında yaptığı açılış konuşmasında Robert Abelson, şunu söyleyerek bu iki tarafı karşılaştırmıştı:

9 Marvin Minsky, "Smart Machines", Sekizinci Bölüm, John Brockman, *The Third Culture: Beyond The Scientific Revolution*, New York: Simon & Schuster. 1995 Şu internet adresinde: http://edge.org/documents/ThirdCulture/p-Ch.8.html. Kitabın bütünü için bkz. http://www.edge.org/documents/ThirdCulture/d-Contents.html.

"Muntazamların birincil kaygısı her şeyin düzgün ve öngörülebilir olmasıyken, dağınıklar hayatın itiş kakışının peşinden koşar..."[10]

Bana kalırsa, YZ gibi çiğ bir alanda hem muntazamlar hem de dağınıklar gereklidir. Dağınıklar, yerli yerine oturmuş kuramın sınırlarının dışındaki cepheleri keşfetmede daha başarılıdır. Muntazamlar, yeni elde edilen bilgileri kodlamaya yardımcı olur, böylece bu bilgi öğretilebilir, bunun hakkında yazılabilir ve böylece hatırlanır.

25.4 DAVRANIŞLAR HAKKINDA

25.4.1 Davranış Tabanlı Robotlar

Grey Walter'ın "tosbağalarını" anımsatan bir yaklaşım kullanan MIT bilgisayar bilimcisi Rodney Brooks, karmaşık gösterimlerden ve uslamlama süreçlerinden kaçınmış, bunun yerine "gerçek dünyada iş gören robot inşası için davranış tabanlı yaklaşım" dediği yaklaşıma odaklanmıştı.

Brooks, yaklaşımının şuradan esin aldığını yazmıştı:

> Tasarım yöntemi olarak evrimi ya da bunun bir kestirimini özetlemek; bu yöntemde, performans iyileştirmeleri, daha duruma özgü devreler [ya da devre gibi örgütlenmiş yazılım] adım adım eklenirken, eski devrelerin yerinde bırakılmasıyla başarılır, böylece yeni devreler iş göremezse (olasılıkla algı koşulları, çalışmanın önkoşullarına uymadığında), eski devreler iş görür. Ek her devre kümesine yeni *katman* denir. Her yeni katman, çevresiyle etkileşime giren sistemde gözlemlenebilir yeni bir davranış üretir.[11]

Resim 25.2'de gösterilen Genghis, Brooks'un katmanlı devre kullanan robotlarının ilk örneklerindendir. Altı bacaklı bu robot, 35 santimetre boyundaydı, bacak uzunluğu 25 santimetreydi ve yaklaşık bir kilogram ağırlığındaydı. Arazide emekleyerek ilerliyor, kızıl ötesi sensörleri sayesinde insanları belirleyebiliyordu. (Genghis'i yürürken gösteren

10 Bu alıntı şuradan: Wendy G. Lehnert. "Cognition, Computers, and Car Bombs: How Yale Prepared Me for the 1990s", *Beliefs, Reasoning, and Decision Making: Psycho-Logic in Honor of Bob Abelson* içinde, Roger Schank ve Ellen Langer (ed.), Hillsdale, NJ: Lawrence Erlbaum Associates, 1994, s. 143-173.

11 Rodney A. Brooks, "From Earwigs to Humans", *Proceedings IIAS The Third Brain and Mind International Symposium on Concept Formation, Thinking and Their Development*, Kyoto, Japonya, Mayıs 1996, s. 59-66. Şu internet adresinde: http://people.csail.mit.edu/brooks/papers/ascona.pdf.



Resim 25.2. Rodney Brooks (üstte) ve Emekleyen Robotu Genghis (altta). (Fotoğraflar, Rodney Brooks'un izniyle.)

kısa film için bkz. http://groups.csail.mit.edu/lbr/genghis/genghis-short2.mov.) Robotun sensörleri arasında iki ön "bıyık", iki eğimölçer (eğimi ve devinimi ölçmek amacıyla) ve öne bakan hareketsiz altı kızılötesi sensör vardı.

Genghis'i denetleyen yerleşik devre, modüllerin üst üste eklenmesiyle inşa edilmişti. Her katman, yürümenin gitgide karmaşıklaşan kiplerini ele alıp, alt katman var olan durumla başa çıkamadığında alt katmanı "kapsıyordu" (yeri geldiğinde onu devre dışı bırakıyordu). Brooks, bu katmanlı örgütlenme türüne "kapsama mimarisi" demiştir. Bu devre, (8-bit mikroişlemcilerle hayata geçirilmiş) "genişletilmiş sonlu makineler" denilen basit bilişim cihazlarından meydana geliyordu.[12]

12 Genghis'in işleyişinin tarifi için bkz. Rodney A. Brooks, "A Robot That Walks: Emergent Behavior from a Carefully Evolved Network", *Neural Computation*, c. 1, no: 2, Yaz 1989, s. 253-262. Ayrıca; *Proceedings of the IEEE International Conference on Robotics and Automation*, Scottsdale. AZ, Mayıs 1989, s. 292-296. MIT YZ Laboratuvarı Raporu (no: 1091) için bkz. http://people.csail.mit.edu/brooks/papers/AIM-1091.pdf.

Önceki simgesel yaklaşımların aksine Brooks'un robotik yaklaşımı, merkezî çevre modelleri ve eylem rotası "planlayan" programlar kullanmıyordu. Brooks, "klasik YZ'yi temel alan simge sistemi hipotezi temelde kusurludur"[13] der ve şöyle yazar:

> ... robotun özgül hedefleri, [davranış tabanlı yaklaşımda] asla açıkça temsil edilmez, ayrıca hiç plan da yoktur; hedefler, algısal koşullarla eşleşen eylemlerde saklıdır ve planların görünürde rotası gerçek zamanda hayat bulurken, bir davranış, robotun dünyadaki yapılanmasını öyle değiştirir ki yeni algısal koşullar, eylemler dizisindeki bir sonraki adımı tetikler.[14]

Brooks, "Filler Satranç Oynamaz" başlıklı makalesinde,[15] MIT'deki laboratuvarında geliştirilmiş oldukça ilginç robot sistemlerinden örnekler verir. Brooks'un makalesinin başlığı, oldukça karmaşık davranışların (örneğin fil davranışlarının), satranç oynamak gibi zekâ isteyen etkinlikler için gerekli temsil ve uslamlama gücünden (olasılıkla) yoksun sistemlerle elde edilebileceği anlamına geliyor. Yine de YZ bilimcileri, fil zekâsına sahip makineler yapabilse gayet mutlu olurdu; ama YZ'nin en son hedeflerine ulaşmak, davranış tabanlı yaklaşımların sunabileceğinin ötesinde, karmaşık temsil ve uslamlama yöntemleri gerektiriyormuş gibi duruyor.

Her ne kadar gitgide yetkinleşen ve zekileşen hayvanların evrim çizgisini (ya da çizgilerini) izlemenin sunacak çok şeyi olduğunu düşünsem de, bırakın fillere ulaşmayı, (Brooks'un bir makalesinin başlığını kullanacak olursam) "sineklerden insanlara" giden yoldan çok uzaktayız bana kalırsa.[16]

25.4.2 Teleo-Reaktif Programlar

Her ne kadar hikâyenin bütünü hiç kuşkusuz bu kadar olmasa da, davranışları, süregelen algısal girdilerle eşgüdümlemek, zeki sistemin önemli bir parçasıdır. Robot Shakey'nin ara düzey eylemleri üzerine çalıştığım günlerden beri bu konuyla ilgileniyordum. 1990'da ve 1991'de akademik izinde olduğum yıl boyunca davranış deneti-

13 Rodney A. Brooks, "Elephants Don't Play Chess", *Robotics and Autonomous Systems*, c. 6, 1990, s. 3-15. Ayrıca, Pattie Maes (ed.), *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, Cambridge. MA: MIT 1990, s. 3-15. Şu internet adresinde: http://people.csail.mit.edu/brooks/papers/elephants.pdf.

14 Rodney A. Brooks, *a.g.e.*

15 Rodney A. Brooks, *a.g.e.*

16 Başka yayınlarına bağlantılar için Brooks'un internet sayfalarına bakınız: http://people.csail.mit.edu/brooks/.

mi üzerine düşünme imkânı bulmuştum. O yılın bir kısmını MIT'de Brooks'un laboratuvarında geçirmiştim. Orada, Stanfordlu öğrenci (kısa süre içinde MIT'li olacaktı) Mark Torrance'ın önemli önerilerinin de yardımıyla, "teleo-reaktif" (T-R) dediğim programlama dilini geliştirdim. ("Teleo", "son" ya da "amaç" anlamına gelen Yunanca *telos* sözcüğünden geliyor.) T-R programı, bir ara düzey etmen (*agent*) denetim programıdır ve robotun dinamik biçimde değişen çevresine dair algısını hesaba katan bir üslupla, robotu hedefine güçlü şekilde yönlendirir. T-R programlarının işleyişinden bahseden bir arasözü hoş görürsünüz umarım. Amaç-güdümlü robotun denetiminde doğan meselelerin bazılarını aydınlatmak için bu arasözden faydalanacağım.

T-R programına örnek, futbol topuna vuran robotu denetleyen programdır. (Program epey basit; bu tür programların nasıl işlediğini açıkladıktan sonra, kendi elinizle "bunu çalıştırmayı" deneyebilirsiniz.) Bu program, acemi bir futbolcunun (altı yaşında diyelim), topa vurmayı öğrenmesini taklit ediyor. Başka neler olup biteceğine aldırmaksızın, çocuk topa yaklaşır, topu karşısına alıp tekmeler:

```
kick(x):
1. Close (x) AND Facing (x) -> foot-swing
2. Close (x) -> face (x)
3. Facing (x) -> move-forward
4. True -> moveto (x)

face (x):
1. Facing (x) -> do-nothing
2. Left (x) -> trotate - ccw
3. True -> rotate - cw

moveto(x):
1. Close (x) -> do-nothing
2. Facing (x) -> move-forward
3. True -> face (x)
```

Bu programın üç parçası var: ana parçası `kick(x)` [x'e vur], iki "altprogramı" ise `face(x)` [yüzünü x'e dön] ve `moveto(x)`'dir [x'e doğru hareket et]. Bu programın nasıl çalıştığını anlamak için, iki önemli bileşen kümesini betimleyeceğim: "algısal altbirimler" ve "ilkel eylem altprogramları". Algısal altprogramlar, robotun durumunun kimi özelliklerinin doğru mu yanlış mı olduğunu belirler. İlkel eylem alt-

programları, robotun temel devinim edimlerini denetler ve bunların, "robotun içine kurulduğu" addedilir ("reflekslerin" hayvanların içine yerleşmiş olması gibi).

Algısal Altprogramlar

- `Close(x)`, robotun, x'e vurma mesafesi içinde olup olmadığını belirler, ki x herhangi bir cisim olabilir. Bunun gibi programlarda, x'e programın "parametresi" ya da "değişkeni" denir. Program çalıştırıldığında, x kesin bir değer alır, örneğin `Ball`, yani futbol topu gibi. Fakat programda kesin değer yerine parametre kullanmak, farklı x "örnekleri" için aynı programın kullanılmasına olanak tanır.
- `Facing(x)`, robotun x'le yüz yüze durup durmadığını belirler.
- `Left(x)`, x'in, robotun baktığı doğrultunun solunda bir yerde (herhangi bir yerde olabilir) kalıp kalmadığını belirler. Eğer öyleyse, robot saat yönünün tersine dönüp x'le karşı karşıya gelir.

İlkel Eylem Altprogramları

- `foot-swing`, robotun "ayağını" hızla ileri hareket ettiren temel eylemdir. Ayağın yolu üzerinde top duruyorsa, topu ötelere fırlatmış olur.
- `move-forward`, robotu, yüzünün dönük olduğu doğrultuda hareket ettirir.
- `rotate-ccw`, robotun (durduğu yerde) saatin tersi yönde dönmesini sağlar.
- `rotate-cw`, robotun (durduğu yerde) saat yönünde dönmesini sağlar.

Öteki eylem altprogramları, yani `kick(x)`, `face(x)` ve `moveto(x)`, ilkel değildir, başka programlardan oluşurlar. Burada gösterilen T-R programlarının numaralı satırlarının, bir okun (`->`) solu ve sağı olmak üzere iki kısımdan meydana geldiğine dikkat edin. Soldaki kısma "koşul kısmı" denir, çünkü belli bir koşulun doğru olup olmadığının yoklanmasından meydana gelir. Sağdaki kısma "eylem kısmı" denir.

T-R programlarının genelde nasıl işlediğini açıklarken ilk adım, belirli bir durumda `kick(Ball)`'un [topa vur] nasıl çalıştığını göstermektir. Diyelim ki robotun yüzü topa dönük ama vuracak kadar

yakın değil. Robot topa vurmak istiyor, dolayısıyla `x` parametresini `Ball`'a ayarlayarak `kick(x)` programını etkinleştiriyor. Robotun etkinleştirdiği program işte şöyledir:

```
kick(Ball):
1. Close(Ball) AND Facing(Ball) -> foot-swing
2. Close(Ball) -> face(Ball)
3. Facing(Ball) -> move-forward
4. True -> moveto(Ball)
```

Programda normalde `x`'in boy gösterdiği her yerde artık `Ball`'un göründüğüne dikkat edin, çünkü vurulacak nesne toptur. T-R programları, program satırlarına, numara sayısına bakılarak ve koşulu yerine getirilmiş ilk program satırı tespit edilerek yorumlanır. Böylece o satırın eylem kısmı etkinleştirilir. Bu özel örnekte, birinci satırın koşul kısmı karşılanmış değil, çünkü robot topa yeterince yakın durmuyor. Aynı sebepten ötürü, ikinci satırın koşul kısmı da geçerli değil. Bununla birlikte, üçüncü satırın koşul kısmı geçerli, böylece robot bununla ilintili ilkel eylemi, yani `move-forward`'ı etkinleştirir. Bu esnada ve `move-forward` eylemi icra edilirken (burası önemli!), programda geçerli ilk satırın hangisi olduğunu yoklayan robot sistemi parçası hâlâ yokluyordur (bir bakıma arka planda). Er geç, birinci satırın koşul kısmı geçerli olacaktır (robotun hareket ederken "konu başlığını" değiştirmediğini varsayarsak). Tam o anda, üçüncü satır artık, programda koşul kısmı geçerli olan *ilk* satır değildir; ama birinci satır öyledir. Dolayısıyla, üçüncü satırın eylem kısmı askıya alınır, birinci satırın eylem kısmı etkinleştirilir ve robot topa vurur.

Şimdi, T-R programlarının gürbüzlüğünü gösterip altprogramların nasıl etkinleştirildiğini açıklayalım: Diyelim ki her şey eskisiyle aynı (yani robotun yüzü topa dönük ama toptan uzakta duruyor), fakat robot ileri hareket ettiği (`move-forward` etkinleştirildiğinde) sırada istemeden yoldan çıkıyor, böylece artık yüzü topa dönük olmuyor. Robot bu değişikliği algıladığı anda, programın üçüncü satırı, koşul kısmı geçerli olan ilk satır değildir artık; ama dördüncü satır öyledir (çünkü onun koşulunun, yani `True`'nun her zaman doğru olduğu varsayılır). Dolayısıyla, `move-forward` faaliyeti kesilir ve bunun yerine `moveto(Ball)` etkinleştirilir.

`Moveto(Ball)`'u etkinleştirmek amacıyla, "program kütüphanesinden" `moveto(x)` alınır ve bunun `x` parametresi yerine `Ball` konulup şu

aşağıdaki program etkinleştirilir:

```
moveto(Ball)
1. Close(Ball) -> do-nothing
2. Facing(Ball) -> move-forward
3. True -> face(Ball)
```

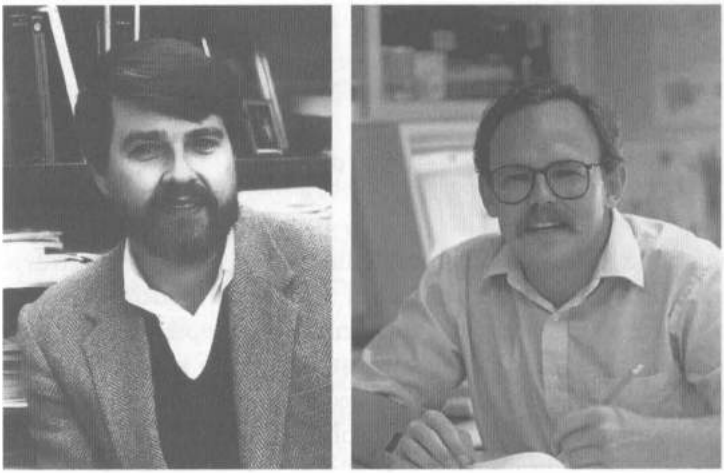
Bu programda, koşul kısmı geçerli olan ilk satır, üç numaralı satırdır; böylece, başka bir altprogram olan `face(Ball)` etkinleştirilmiş olur. Eğer robotun yolundan sapması sonucunda topa göre sola kaydığını varsayarsak, `face(Ball)`'un etkinleştirilmesiyle robot, saat yönünün tersi yönde dönecek demektir. Er geç robotun yüzü yine topa dönük olacak. Şimdi ilginç bir şey daha oluyor. `Moveto(Ball)` altprogramı, bütün o koşul-yoklama düzeneğiyle birlikte arkaplanda çalışmayı sürdürüyor. Bu altprogramın ikinci satırı, şimdi koşul kısmı geçerli olan ilk satırdır (3 numaralı satırın yerine). Dolayısıyla, `face(Ball)` programının işlemesi durduruluyor, `move-forward` ilkel programı etkinleştiriliyor. Eğer aksi bir şey olmazsa, `kick(x)`'in birinci satırı, programda koşul kısmı geçerli ilk satır olur [`moveto(Ball)` askıya alınır] ve `foot-swing` etkinleştirilir. (Vay be! Tüm bunları bir devrenin otomatik olarak denetlemesi, bizim bunun hakkında kafa yormamızdan iyidir.) Eğer yorulmadıysanız, bu programların etkinleştirilmesinin başka yöntemlerini kendiniz gözden geçirmek isteyebilirsiniz.

25.5 BEYİN TARZI HESAPLAMA

25.5.1 Sinir Ağları

Beyin, anlaşıldığı kadarıyla yaptığı işi, birbirine bağlı sinir hücresi ağlarının gerçekleştirdiği devasa paralel hesaplamalarla yaptığı için kimi insanlar, sinir ağlarının sunduğu olasılıkları yeniden keşfetmeye koyulmuştu. 1970'lerin ikinci yarısında, California Üniversitesi San Diego kampüsünde bilişsel psikologlar David E. Rumelhart (1942-2011) ile James L. McClelland'ın (1948-) (Resim 25.3) önderlik ettiği ekip, "paralel dağıtık işleme" (PDP; parallel distributed processing) sistemleri dedikleri ağ araştırmalarına başladı. Bu ekip, PDP ekibi olarak tanınır.

PDP ekibi, beyindeki zihinsel süreçlerin, ağlarda birbirine bağlanan temel sinirsel birimler arasındaki etkileşimlerin sonucu doğduğunu savunuyordu. Bu birimler birbirlerini paralel olarak tetikleyip ketler. Hesaplamaya dair bu görüş, çoğu simge işleme yaklaşımının



Resim 25.3. David Rumelhart (solda) ve James McClelland (sağda). (Rumelhart'ın fotoğrafı, Rumelhart'ın izniyle. McClelland'ın fotoğrafı, James McClelland'ın izniyle.)

yerine getirdiği seri hesaplamalara ve Newell ile Simon'un fiziksel simge sistemi hipotezine ters düşüyor. Dolayısıyla PDP sistemleri, konumu belli veri yapılarında bilgi saklamak yerine, bilgiyi, birimler arasındaki bağlantılara dağıtır. Dahası, PDP sinir ağları ileri beslemeli, katmanlı dizilişlerle sınırlı değildir. Rosenblatt'ın genel perceptronlara dair görüşünü anımsayacak olursak, bazı PDP sistemleri, "özbağlı" bağlantılara olanak tanıyordu; bunlar, çeşitli birimler aracılığıyla döngülerin parçasıydı. Rumelhart'ın daha sonra belirttiği gibi: "Tüm bu çabaların ortak konusu, beyne, geleneksel seri bilgisayarlardan çok farklı olan bir paralel hesaplama cihazının modeli olarak bakmaktır."[17]

PDP çalışmaları, McClelland, Rumelhart ve PDP Araştırma Ekibinin yayımladığı iki kitapla iyice göz önüne çıkmıştı.[18] Birinci ciltte, Rumelhart, Geoffrey E. Hinton ve Ronald J. Williams'ın birlikte yazdıkları "Hata Geri Yayma ile İçsel Gösterimleri Öğrenmek" başlıklı önemli bölüm, ağ ağırlıklarını ayarlamaya yarayan, "geri yayma" adlı yeni bir

17 David E. Rumelhart, "Brain Style Computation: Learning and Generalization", *An Introduction to Neural and Electronic Networks* içinde, Steven Zornetzer, Joel L. Davis ve Clifford Lau (ed.), San Diego: Academic Press, 1990.

18 James L. McClelland, David E. Rumelhart ve PDP Araştırma Ekibi, *Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations*, Cambridge, MA: MIT Press, 1986; James L. McClelland, David E. Rumelhart ve PDP Araştırma Ekibi, *Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Vol. 2: Psychological and Biological Models*, Cambridge. MA: MIT Press, 1986.

teknik tanıtmıştı. Bu teknik, ileriki bölümlerde anlatacağım pek çok yeni uygulamaya can vermiştir.

Fizikçi John J. Hopfield (1933-), başka bir sinir ağı türü icat etmişti.[19] Hopfield ağında her sinirsel öğe, tüm öteki öğelerle bağlantılıdır. Bu bağlantılar üzerindeki ağırlıklar iki yönde aynıdır; yani *i* birimini *j* birimine bağlayan ağırlığın değeri, *j* birimini *i* birimine bağlayan ağırlığınkiyle aynıdır. Ağın işleyişi dinamik bir süreçtir; yani her zaman adımındaki birimlerin değerleri, bir önceki zaman adımındaki değerlere bağlıdır. Birim değerleri hep birlikte, fizikçilerin "enerji fonksiyonu" dediği şeyle ilintilidir ve bu değerler (ağın başlangıç halinden bağımsız olarak), yerel en küçük enerji durumuna karşılık gelen değerlere yakınsamaya eğilimlidir. Bunlara, ağın "kararlı halleri" denir ve ağın sakladığı bir anı kümesi olarak düşünülebilir. Hopfield ağları, çağrışımlı bellek olarak ve kimi basit hesaplamalar için kullanılmıştır. (On şehirlik "gezgin satıcı" problemini çözen Hopfield ağının gösterimi için şu internet adresine bkz. http://to-campos.planetaclix.pt/neural/hope.html.) "Boltzman makinesi", Hopfield ağının gelişmişidir; bu makinede, her zaman basamağının birim değerleri, bir önceki zaman basamağının birim değerlerine göre rasgele değişir.

Bu dönemde gerçekleştirilmiş sinir ağı araştırmalarının büyük kısmına, ETYZ'den farklı olarak "bağlantıcı" ya da "beyin tarzı" hesaplama denmiştir. Bu harekette etkin olan bir başka kişiyse, 1974'te Stanford'dan ayrılıp Rochester Üniversitesi'ne geçerek orada Bilgisayar Bilimi Bölümü'nü kuran ve bağlantıcı-yönelimli araştırmalar yürüten Jerome A. Feldman'dı.[20]

25.5.2 Dinamik Süreçler

Bazı araştırmacılara göre, Hopfield ve Boltzman ağlarında sergilenenlere benzer dinamik süreçler (diferansiyel denklemlerle tanımlanan süreçler de dahil), beynin gerçekleştirdiği pek çok hesaplamanın temelini oluşturuyor. Örneğin, *MIT Bilişsel Bilimler Ansiklopedisi*'ndeki bir makalede Tim van Gelder şöyle yazmıştır:[21]

19 John J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", *Proceedings of the National Academy of Science*, c. 79, no: 8, 1982, s. 2554-2558. Şu internet adresinde: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=346238.

20 Örneğin bkz. Jerome A. Feldman ve diğ., "Computing with Structured Connectionist Networks", *Communications of the ACM*, c. 31, no: 2, Şubat 1988, s. 170-187.

21 T. J. van Gelder, "Dynamic Approaches to Cognition", *The MIT Encyclopedia of Cognitive Sciences* içinde, R. Wilson ve F. Keil (ed.), Cambridge MA: MIT Press, 1999,

> Güncel amaçlar bağlamında dinamik süreç, bir dizi denklemin betimlediği dinamiklik yasaları uyarınca, nicel zaman içinde sürekli, eşzamanlı ve bağımsız olarak değişen bir nicel değişkenler kümesidir. Bu kabul, bilişsel süreçleri anlama bakımından doğru araçları devinbilimin temin ettiği inancıyla kol kola ilerler.
>
> ...
>
> Dinamik sistemler kuramının merkezî içgörülerinden biri, davranışın geometri bağlamında anlaşılmasıdır, yani sistemin olası tüm durumlarında, davranış bir konum ve konum değişikliği meselesi olarak görülebilir. Böylece davranışı, cezbediciler, geçiciler, durağanlık, eşleşme, dallanma, kaos, vesaire bağlamında betimlemek mümkün olur; klasik bakış açısıyla bu özellikleri görmek imkânsızdır.

Ancak, aynı makalede Gelder, "şu an için bilişin pek çok yönü (örneğin bir öyküyü kavrama) dinamik yöntemin üstesinden gelebileceği bir iş değildir" cümlesini de yazmıştır.

Indiana Üniversitesi bilgisayar bilimcisi Randall Beer (1961-) daha iyimserdir. "Bilişsel Bilimin Dinamik Yaklaşımları" başlıklı makalede Beer, "dinamik yaklaşımlar, bilişsel bilimdeki önemli deneysel soruları ele almaya başlıyor", demiştir.[22] Bununla ilgili üç örnek verir; bir tanesi, yatay hareketini on dört sinir hücreli, kesintisiz-zaman devirli sinir ağının hayata geçirdiği bir dinamik sistemin denetlediği temsili bir etmendir. Bu etmenin (*agent*) görevi, farklı şekle sahip, düşen iki nesne arasında ayrım yapmaktır; belli bir şekilden kaçarken (yolundan çekilerek), ötekine çarpmalıdır (yoluna çıkarak). Beer, bu davranışı "en az bilişsel" olarak niteler ve "bilişsel ilgi soruları doğuran en basit davranış" diye tanımlar. Bildiğim kadarıyla, dinamik sistemler, bu en az bilişsel davranışlardan daha fazlasını gerektiren görevlerde henüz kullanılmadı.

Hem Beer'ın hem de başkalarının altını çizdiği bir özellik, ağla çevresi arasındaki etkileşimdir. Doğrusu, çevrenin kendisi çoğu di-

s. 244-246. Şu internet adresinde: http://sites.google.com/site/timvangelder/publications-1/dynamic-approaches-to-cognition/MITDyn.pdf?attredirects=0.

22 Randall D. Beer, "Dynamical Approaches to Cognition Science", *Trends in Cognitive Sciences*, c. 4, no: 3, Mart 2000; şu internet adresinde: http://mypage.iu.edu/~rdbeer/Papers/TICS.pdf. Daha uzun bir makale için bkz. Randall D. Beer, "A Dynamical Systems Perspective on Agent-Environment Interaction", *Artificial Intelligence*, Etkileşim ve Öznelik Konulu Bilişim Araştırmaları Özel Sayısı, Birinci Kısım, c. 72, no: 1-2, 1995, s. 173-215; şu internet adresinde: http://mypage.iu.edu/~rdbeer/Papers/AIJ95.pdf. Beer'ın internet sayfaları (http://mypage.iu.edu/~rdbeer/) pek çok ilave atıf içeriyor.

namik sistemin önemli bir bileşenidir. Çevrenin özelliklerini keşfedip sistemin bütününü basitleştirme çabasını, çalışmalarının bir kısmını Los Alamos Ulusal Laboratuvarı'nda gerçekleştirmiş olan Mark W. Tilden aşırı uca taşımıştır. Örneğin, Tilden'in robotları bilgisayardan faydalanmaz, bunun yerine, engebeli arazide ilerlerken motorlarından gelen direnç girdisinden yararlanarak yürürler.[23]

25.6 EVRİMİ TAKLİT ETMEK

İkinci bölümde, rasgele yaratım ve seçilimli sağkalım gibi evrim süreçlerinden faydalanarak zeki yapılar yaratma girişimlerinden bahsetmiştim. Bunlar arasında John Holland'ın genetik algoritmaları (GA) en umut vaat eden girişim olarak görünüyordu. GA'lar, belirli bir sorunun çözümünü kodlayan simge zincirleri evriltmeye kalkışır. GA konusundaki ilk çalışmaların büyük kısmında iki değerli simgeler (0'lar ve 1'ler) kullanılmıştır; gerçi başka simgeler de kullanılabilirdi.

Gezgin satıcı problemi, GA'ların kullanımını açıklamak için sıklıkla başvurulan bir örnektir. Bu problemde, ziyaret edilmesi gereken şehirlerin bir listesi vardır ve bizler, bir şehirden başlayıp tüm öteki şehirleri sadece bir kez ziyaret ederek başlangıç şehre dönen bir tur bulmak zorundayız. Problem, kat edilecek toplam uzaklığı en aza indiren bir şehir sıralaması bulmakla ilgilidir. Çözümü kodlamak amacıyla, şehir adları simge olarak kullanılabilir. Örneğin, `A, B, ..., N` harfleriyle isimlendirilen on dört şehir varsa ve tura C şehriyle başlayıp yine aynı yerde bitirmek zorundaysak, o zaman (`C, F, N, K, B, L, M, H, D, A, E, G, I, J, C`) simge zinciri, C'yle başlayıp sonra F'yi ve öbürlerini ziyaret eden bir turu temsil eder. Evrim terminolojisi kullanacak olursak, bu turun kat ettiği toplam uzaklık, bu zincirin *evrimsel uygunluğuyla* ilişkilidir. Daha kısa turların evrimsel uygunluğunun daha büyük olmasını istiyoruz, dolayısıyla turun evrimsel uygunluğunu, kat edilen uzaklığın eksi değeri olarak belirleyelim. GA işlemi, en büyük evrimsel uygunluğu olan zinciri evriltmeye çabalar.

Evrimsel süreç, büyük bir rasgele zincir kümesini bir araya toplamakla başlar. Gezgin satıcı problemi örneğimizde, hepsi C'yle başlayıp C'yle biter ve öteki harfler zincirde sadece bir kez görünür. Bu zincir kümesi iki farklı süreçten geçer; bu süreçler, biyolojik evrimde olup bitenlere kısmen benziyor. Birincisinde, bu zincirlerin bazıları rasgele

23 Bu tür basit robotlar hakkında daha fazla bilgi için Wikipedia sitesine bakınız: http://en.wikipedia.org/wiki/BEAM_robotics.

mutasyona uğrar ve kimi bileşenlerinin değeri böylece değişir. Gezgin satıcı zinciri mutasyonuna örnek, zincir içinde rasgele seçilmiş iki simgenin yer değiştirmesi olabilir.

İkincisinde, küme içinde görece yüksek evrimsel uygunluğa sahip zincir çiftleri, "çaprazlama" (*cross-over*) denilen bir işleme katılmak üzere seçilir, böylece "yavru" bir zincir oluşturulur. GA'da, çaprazlama işleminin farklı türleri kullanılır. Gezgin satıcı örneğinde bu işlem, yavru zincirin "geçerliliğini" korumak zorundadır; yani, öbür şehirleri yalnızca bir kez ziyaret eden bir tura denk gelmelidir. Bunu yapmanın yollarından biri, anne zincirlerden birinin ilk *K* simgelerini yavru zincirde tekrarlayıp sonra öteki anne zincirdeki simgeleri tarayarak yavru zinciri, orada olmayan simgelerle doldurmaktır. *K* değeriyse rasgele seçilir.

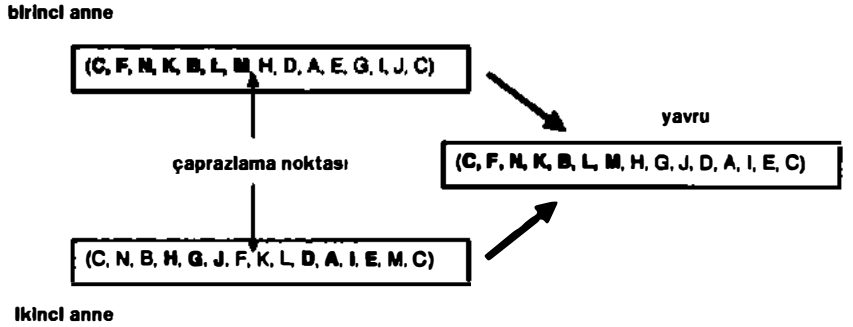
Resim 25.4'teki çizim, bu tarz bir çaprazlamanın nasıl işlediğini gösteriyor.

Evrimsel sürecin her aşamasında, zincirlerin o anki nesli, yeni bir nesli ortaya çıkarır. Yeni nesil, eski nesilden gelen bazı zincirleri (yüksek evrimsel uygunluğu olanları yeğleyerek), mutasyona uğramış zincirleri ve çaprazlama işleminden kaynaklanan yeni zincirleri içerir. İlginç ama, art arda gelen nesiller sonucunda, eldeki problemi çözme konusunda daha başarılı zincirler içerir (bazı problem türleri için).

GA'lar, bilgisayar biliminde, mühendislikte, iktisatta, kimyada ve başka alanlardaki çeşitli bileşimli en iyileme problemlerine uygulanmıştır. İnternet ansiklopedisi Wikipedia, eğitici bilgilerin yanı sıra GA'yla ilgili mükemmel bilgiler sunar: http://en.wikipedia.org/wiki/Genetic_algorithm.[24]

Genetik algoritmaları, "evrimsel uyumluk arazisinde" yüksek tepeleri belirlemeye çalışan bir arama işlemi olarak düşünebiliriz. Olası her GA zinciri, yükselti haritasında bir "nokta" sayılabilir; bu zincirin evrimsel uygunluğuysa, o noktanın yüksekliği olur. En başta araziye rasgele "paraşütçüler" indirilir ve bunlar yüksek noktaları bildirir. Bazı paraşütçüler, o anki konumlarından azıcık ilerlerken (bunlar mutasyonlara karşılık gelir), bazı çiftler var olan konumlarının arasında bir yere uçarak ulaşır (çaprazlama işlemine karşılık geliyor). Ardından bu süreç tekrarlanır. Uyum başarısı arazisinde, bazıları öbürlerinden daha yüksek olan zirveler ve birtakım ovalar olabilir. Birkaç nesil sonra GA işlemi, sadece ufak tepelerin yerini bulmayı başarabilir ya da geniş yayla bulmakta zorluk çekebilir. Fakat ara sıra arazideki

24 Ayrıca bkz., Melanie Mitchell, *An Introduction to Genetic Algorithms*, Cambridge, MA: MIT Press, 1996.



25.4. Çaprazlama işlemi.

en yüksek zirveyi de bulabilir. YZ, matematikçilerin tepe-tırmanma (ya da eğim çıkış) dediği yöntemlerden faydalanmıştır, fakat GA'nın gelişinden önce bu tekniklerde sadece tek "tırmanıcı" vardı. GA, başka evrimsel algoritmalarla birlikte, birkaç tırmanıcının eşzamanlı olarak arama yapmasına olanak tanır, böylece ortaya "paralel arama" çıkar.

John Holland'ın öğrencilerinden John Koza (1943- ; Resim 25.5), Genetik Programlama (GP) adıyla biraz farklı bir evrimsel yöntem geliştirmiştir. GP, zincirler yerine LISP programları yaratır. Bu işlem, eldeki görevi çözmede önemli olduğu düşünülen bazı temel LISP fonksiyonlarını ve sabitlerini içeren programların rasgele bir derlemesiyle işe başlar. Yeni program nesilleri üretmek amacıyla, yine gelişigüzel mutasyonlardan ve çaprazlama işleminden faydalanılır. (GP'nin daha ileri sürümleri, biyolojiden esinlenerek, envanterlerine tersyüz etme, gen kopyalama ve gen silme benzeri işlemler eklemiştir.) Mutasyon kısmı için çeşitli teknikler kullanılabilir; bunlara, programın birtakım parçalarını rasgele seçilmiş yeni program bileşenleriyle değiştirmek dahildir. Çaprazlama işleminde, evrimsel uyumluluğu görece yüksek iki "anne program" seçilir. Her programın rasgele seçilen bir kısmı birbirleriyle değiştirilir ki programın bir sonraki nesli için iki yeni program üretilsin.

Koza, başka şeylerin yanı sıra yeni tür elektrikli süzgeçler, optik mercekler, antenler ve denetim devreleri üreten programlar yaratma amacıyla GP'yi kullanmıştır. Onun da dediği gibi, bu programların pek çoğu, "insanın yaptıklarıyla boy ölçüşebilir". Koza, YZ'nin ana amacının, zeki davranışlarda bulunabilen programlar üretmek olduğu için, bu tür programları doğrudan üretebilecek bir programlama sentezleme tekniği kullanılması gerektiğini ve GP'nin (şimdilik) bu açıdan en iyi sentez tekniği olduğunu iddia etmiştir. Onun da belirttiği gibi, "yapay zekâ, yapay öğrenme, uyarlanımsal sistemler ve otomatik öğrenmede



Resim 25.5. John Koza. (Fotoğraf, John Koza'nın izniyle.)

neredeyse tüm problemler, bilgisayar programı için bir arama işlemi olarak şekillendirilebilir. Genetik programlama, bilgisayar programları uzamında bir bilgisayar programı için, başarıyla arama işlemi yürütmenin yolunu sağlar."[25]

1999'dan beri Koza, GP çalışmasında 1.000-Pentium "Beowulf Tarzı Yığın Bilgisayar" kullanıyor.[26] Eşyazarlarla birlikte Koza, GP hakkında birtakım kitaplar ve makaleler kaleme almıştır.[27]

Bilişim Makineleri Birliğinin (ACM) Genetik ve Evrimsel Bilişim Özel İlgi Grubu (SIGEVO) vardır. Bu grup, GA ve GP'ler dahil, evrimsel bilişimin farklı yönlerini ele alan konferanslar düzenlemiştir.[28] 1991'deki SIGGRAPH konferansında sunulmuş, evrim benzetimi yapan sistemlerin gücüne dair ilgi çekici ve anlatımlı video gösterimini şu internet adresinde izleyebilirsiniz: http://www.archive.org/details/

25 Bu alıntı Koza'nın kendi internet sitesinden: http://www.genetic-programming.com/johnkoza.html.

26 Bkz. Forest H. Bennett ve diğ., "Building a Parallel Computer System for $18,000 That Performs a Half Peta-flop per Day", *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference* içinde, Wolfgang Banzhaf ve diğ. (ed.), San Francisco, CA: Morgan Kaufmann Publishers, 1999, s. 1484-1490.

27 Örneğin bkz. John R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Cambridge, MA: MIT Press, 1992; John R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*, Cambridge, MA: MIT Press, 1994; John R. Koza, Forrest H Bennett III, David Andre ve Martin A. Keane, *Genetic Programming III: Darcoinian Invention and Problem Solving*, San Francisco: Morgan Kaufmann Publishers, 1999; John R. Koza, Martin A. Keane, Matthew J. Streeter, William Mydlowec, Jessen Yu ve Guido Lanza, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*, Norwell, MA: Kluwer Academic Publishers, 2003.

28 SIGEVO internet sitesini ziyaret ediniz: http://www.sigevo.org/index.html

sims/_evolved_virtual_creatures_1994. Bu internet sitesinde belirtildiği gibi: "Süperbilgisayar içinde yüzlerce varlıktan oluşan bir nüfus meydana getirilir ve her varlık, belirli bir görevdeki performansına göre sınanır, örneğin sulu ortam benzetiminde yüzme becerisi gibi."[29]

27.7 YZ'NİN HEDEFLERİNİ DARALTMAK

YZ kışı sırasında pek çok YZ araştırmacısı, önceki yılların hedeflerine göre daha alçakgönüllü ve ulaşılabilir hedeflere odaklandı. YZ'nin *en sonunda* ulaşabilecekleri hakkında cesur öngörüler daha az işitilir oldu. Bunun yerine, YZ'nin (o an için) *gerçekten* ulaşabileceklerine yönelik çabalar gitgide arttı. Sonuçta, "güçlü YZ" yerine kısıtlı, yani "zayıf" YZ üzerine daha fazla çalışma gerçekleştirildi. Artık vurgu, insanların yerini alacak değil, insanlara yardımcı olacak YZ kullanımı üzerineydi.

Araştırmaları destekleyecek ödenek sunan şirketler ve devlet kurumları, kendi problemlerini çözmeye yardımcı olsun diye genel olarak bilgisayar teknolojilerine (özel olarak YZ yerine) bakıyordu. Araştırma ödenekleri, birkaç alan saymak gerekirse, veritabanı sistemleri, kullanıcı arayüzleri, bilgisayar grafiği, bilgisayar ağları, veri madenciliği, bilgisayar oyunları, bilgiye erişim, bilgisayarla görme, sözcük işleme ve çizelge programlarını geliştirmeye yönelmişti. Yeri geldiğinde, arama ve çıkarsama, uzman sistemler, konuşma tanıma ve doğal dil işleme gibi YZ teknolojileri, büyük ve bütünleşik sistemlerin bileşenleri olarak kullanılmıştır. YZ araştırmacıları, bu sistemleri daha faydalı ve cazip hale getirmek amacıyla, söz konusu sistemlerin şurasına burasına zekâ kırıntıları eklemekle yetinir olmuştu.

29 Beni bu videodan haberdar ettiği için Mykel Kochenderfer'e teşekkür ediyorum.

YEDİNCİ KISIM

BÜYÜYEN CEPHANELİK: 1980'DEN İTİBAREN

1980'ler boyunca, YZ, önce gördüğü rağbetin ve ticari başarılarının tadını çıkartıp sonra ödenek kesintisi ve kış mevsimi yaşarken, temel YZ araştırmalarını yürütenler, önemli sayıda güçlü yeni teknik araçlar üretip eskileri de keskinleştirdi. YZ'nin uslamlama ve gösterim, yapay öğrenme, doğal dil işleme ve bilgisayarla görme gibi tüm alt alanlarında yeni sonuçlar boy gösterdi. Önceki çalışmalara kıyasla teknik ve matematik bakımından daha derin olan, istatistik ve kontrol mühendisliği bağlantılarıyla güçlenmiş bu çalışmalar, eski YZ sistemlerine yöneltilmiş kimi eleştirilerin etkisini hafifletirken, YZ'nin becerilerini de büyük oranda artırmıştır.

26
USLAMLAMA VE GÖSTERİM

26.1 TEKDÜZE OLMAYAN YA DA FESHEDİLEBİLİR USLAMLAMA

Bilgi gösteriminde mantık dillerinin ve uslamlamada mantık yöntemlerinin kullanımını öne çıkaran, mantıkçı denilen YZ araştırmacıları, sıradan mantıktaki bir sorunun varlığını kabul eder; adını koymak gerekirse, bu mantık *tekdüzedir*. Bununla kastedilen, mantıksal önerme kümesinden çıkarılan mantıksal sonuç kümesinin, kümeye daha fazla önerme eklendiğinde azalmamasıdır. Belirli bir bilgi tabanından gelen bir önerme kanıtlanabiliyorsa, aynı önerme, daha fazla bilgi eklendiğinde de kanıtlanabilir (yine aynı kanıtla!).

Yine de insan uslamlamasının büyük kısmı o şekilde işliyormuş gibi durmuyor; YZ eleştirmenleri bu olgunun gayet farkındadır (ve bu durumu yüceltirler). Çoğunlukla, makul varsayımlarla birlikte elimizdeki olgulardan faydalanarak birtakım yargılara atlarız, sonra bu varsayımlara ters düşen yeni bir olgu öğrendiğimizde bu yargıyı geri çekmek zorunda kalırız. Bu uslamlama tarzına *tekdüze olmayan* ya da *feshedilebilir* ("etkisiz ya da geçersiz kılınabilir ya da ilan edilebilir" anlamında) uslamlama denir, çünkü yeni olgular, daha önce varılmış kimi yargılardan dönülmesini gerektirebilir.

Tekdüze olmayan uslamlamaya dair örnekleri çocuk öykülerinde bile bulmak mümkün. Margery Cuyler'ın *That's Good! That's Bad!* adlı kitabında,[1] küçük bir oğlan, ailesinin ona hayvanat bahçesinde satın aldığı balona tutunarak gökyüzünde süzülür. "Vay! Ne güzel" denir öyküde. Derken balon, yüksek ve dikenli bir ağaca takılıp patlar. "Of! Çok kötü", denir. Oğlan çamurlu bir nehre düşer, oradaki hipopotamın üstüne tırmanıp hayvanı kıyıya sürer. "Ah, ne güzel". Öykü böyle sürüp gidiyor; balon yolculuğunun "güzel" ya da "kötü" seyretmesine göre bir öyle bir böyle ilerler.

YZ'de (ve bilgisayar biliminin başka dallarında), feshedilebilir uslamlama için kullanılan bazı yöntemler zaten vardı. Öneğin, Carl Hewitt'in önerdiği problem çözme dili PLANNER'da, eğer program bir hedefe, örneğin H'ye ulaşamazsa, o zaman `H Değil` ifadesi belirtilir (H'nin,

1 Margery Cuyler, *That's Good! That's Bad!*, New York: Henry Hoit and Co, 1991.

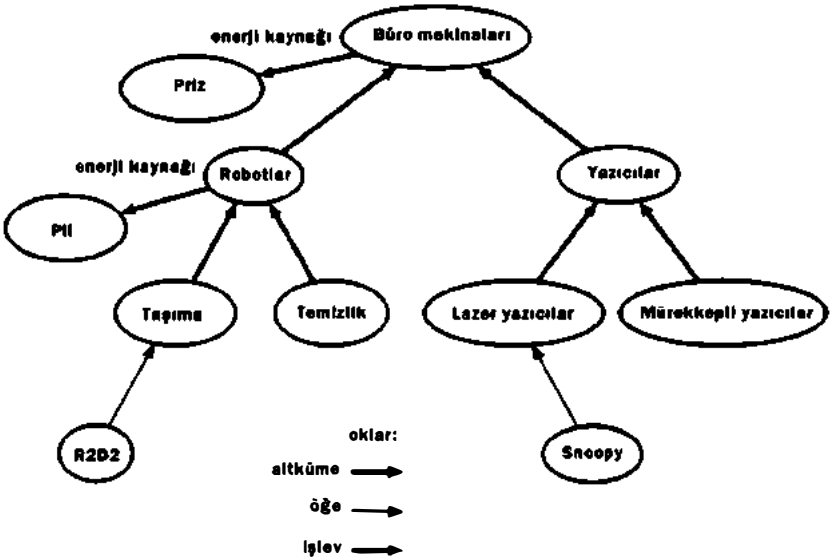
programın doğruluğunu göstermeye çalıştığı bir yargı olduğu varsayımıyla). Bu tür bir uslamlama feshedilebilir niteliklidir, çünkü programa ya da bilgi tabanına ek yargılar eklendiğinde, H'nin doğruluğunu göstermek mümkün hale gelebilir. Benzer biçimde, programlama dili PROLOG'da, eğer bir yargı, program tarafından kanıtlanamıyorsa, o zaman bu yargının yanlış olduğu çıkarsanır. Doğruluğu kanıtlanamayan bir şeyin yanlış olduğu çıkarımını yapmaya, "yetersizlikten ötürü olumsuzlama" denir.

SRI'nın planlama sistemi STRIPS de tekdüze olmayan bir uslamlama sistemiydi. Eylemler yerine getirildikten sonra "aynı kalan" şeyler hakkındaki varsayımlar, kesinlikle varsayımdan ibaretti. Bu gibi varsayımlara dayanarak çıkarılan sonuçlar, içerimleri, o varsayımları olumsuzlayan yeni bilgilerin eklenmesiyle rafa kaldırılabilir.

Feshedilebilir uslamlamanın başka bir yöntemi, veritabanı dünyasında kullanılmıştır. Veritabanları, çok çeşitli bilgileri kodlamak amacıyla kullanılır. Örneğin bir şirket, çalışanları hakkında veritabanı hazırlayabilir. Bu tür bir veritabanı, belli bir çalışanın maaşını, çalıştığı kısmı, vesaire bulmak amacıyla sorgulanabilir. Diyelim ki böyle bir çalışan veritabanından Jack Smith adlı kişi hakkında bilgi edinmek istiyoruz ama adamın ismi veritabanında yok. Bay Smith'in, o şirketin çalışanı olmadığı sonucuna varmamız gayet makul, ki bazı veritabanları da aynı sonuca ulaşacaktır. Bu sonuç, veritabanında *tüm* şirket çalışanlarının isminin yer aldığı varsayımına dayanıyor; kapalı evren varsayımı (KEV) denen durumun bir örneğidir bu. Elbette, Jack Smith ileride şirkete katılabilir ve ismi veritabanına eklenebilir. O zaman, Jack Smith'in şirket çalışanlarından biri olmadığı hükmünden dönmemiz gerekir; bu, feshedilebilir uslamlamanın başka bir örneğidir.

Bertram Raphael'in ta 1964'te hazırladığı soru yanıtlama sistemi SIR'ın, "istisna ilkesi" dediği bir uslamlama tarzı içerdiğini belki hatırlarsınız. SIR'da, bir kümenin tüm öğeleri hakkındaki genel bilgi, belirli öğelere uygulanıyordu; ama sadece, bu belirli öğeler hakkında özgül bilgi olmadığı zaman. Kimi YZ bilgi gösterim şemaları, içerdikleri bilginin bir kısmını, Raphael'in kullandığına benzer "taksonomi hiyerarşileri" içinde temsil edip feshedilebilir uslamlama için istisna ilkesinden faydalanır, ki artık buna çoğunlukla "mirasın reddi" denir.

Resim 26.1'de, bazı büro makinelerinin taksonomi hiyerarşisini gösteriyorum. Bu hiyerarşiden yararlanan bir program, örneğin lazer yazıcının enerji kaynağının, duvardaki priz olduğu sonucuna varır, çünkü bu özellik, genel "büro makineleri" sınıfından miras kalmıştır. Bununla birlikte, robotların enerji kaynağı hakkındaki özgül bilgi, ör-



Resim 26.1. Büro makineleri hiyerarşisi. (Nils J. Nilsson, *Artificial Intelligence: A New Synthesis* içinde, San Francisco: Morgan Kaufmann Publishers, 1998, s. 311.)

neğin, genel büro makineleri sınıfının özelliklerinin mirasını çiğneyerek, R2D2'nin enerji kaynağının pil olduğu hükmüne varır.

1980'ler boyunca, en yaratıcı YZ araştırmacılarının bazıları, feshedilebilir uslamlama problemiyle büyülenmiş, bunun nasıl başarılacağına dair yeni öneriler getirmiştir. Bu önerilere, farklı yaklaşımları kıyaslayıp ayrıştıran, ayrıca bu yaklaşımlardan bazılarının öbür yaklaşımların genellemesi ya da özel hali olduğunu belirleyen epey kuramsal çözümleme eşlik etmiştir.

Kanadalı YZ araştırmacısı Raymond Reiter (1939-2002), yeni yöntemlerden birini önermiştir.[2] En basit haliyle bu yöntem, belli bir bilgi tabanından sonuç çıkarmaya olanak tanıyan özel çıkarsama kurallarından faydalanır, tabii eğer belirli bir koşul yerine getirilmişse ve bu sonuç, o bilgi tabanından normalde çıkarsanan sonuçla ters düşmüyorsa. Reiter'ın özel çıkarım kurallarına *öntanımlı kurallar* denir, bu kuralları kullanan sistemineyse *öntanımlı mantık* denir. Bunun kullanımına örnek olarak, diyelim ki robotun büroda hangi odalara girebileceğini belirlemek amacıyla yine o robotun kullandığı bir bilgi tabanımız var.

2 Raymond Reiter, "A Logic for Default Reasoning", *Artificial Intelligence*, c. 13, 1980, s. 81-132.

İkinci kattaki odalar için, eğer robotun oradaki bir odaya giremeyeceğini kanıtlamak imkânsızsa, robotun o odaya girebileceği sonucuna varılır, diyen bir kuralımız var olabilir. Bu uslamlama da feshedilebilir niteliklidir, çünkü ileride bilgi tabanına eklenecek bir olgu, ikinci kattaki bazı odalara girelemeyeceği çıkarımına olanak tanıyabilir.

Bilginin mantıksal önerme olarak kodlanması önerisini ilk dile getiren kişi John McCarthy de tekdüze olmama sorununu dert etmişti. Bu sorunla başa çıkmak amacıyla, *sınırlama* denilen bir yöntem önermişti.[3] Mantık jargonu kullanmaksızın sınırlamayı açıklamak görece zor. İlkede, bunun "belirtimli sınırlama" (*predicate circumscription*) denen biçimi (birkaç sınırlama biçiminden sadece biri[4]), doğru olduğu kanıtlanabilen varlıklar için belirtimleri geçerli kılan bir varlıklar kümesini sınırlamayı (dolayısıyla "sınırlamayı") içerir. Örneğin, elimizde `Uzun(John)` ve `Uzun(Frank)` gibi yargılar ve başka bir sürü olgu içeren bir bilgi tabanı varsa, `Uzun` belirtimini (istersek) sınırlayabiliriz. Böylece, eğer `Uzun(Susy)`, bilgi tabanı tarafından mantık diliyle ifade edilmemişse, `¬Uzun(Susy)` sonucuna ulaşabiliriz.

McCarthy'nin tekdüze olmayan uslamlamaya ilgi duymasının itici güçlerinden biri, çerçeve problemini çözmede bunun kilit unsur olma ihtimaliydi. Çerçeve probleminin, bir eylemin (örneğin bir robotun eyleminin) ardından nelerin değişip nelerin aynı kaldığını temsil etmenin zorluğuyla ilgilendiğini anımsayın. Yaklaşımlardan biri, eğer dünyanın belli bir halini betimleyen bir belirtim, bir eylemin tarifinde geçmiyorsa (o eylemin önkoşulları ve etkileri dahil), o zaman bu belirtim, eylem tarafından değiştirilmez, varsayımında bulunmaktır. Bu varsayım tekdüze değildir, çünkü daha sonraki (ya da daha ayrıntılı) bir bilgi, bahsi geçmeyen belirtimin aslında değişmiş olduğuna işaret edebilir. Çerçeve problemini, tekdüze olmayan uslamlama kullanmaksızın çözmeye yönelik ilk girişimler, çeşitli teknik zorluklarla karşılaştı (bu zorluklar, burada ele alamayacağımız kadar teknik); fakat çalışmalar sürmüştür. Yakın tarihli bir makale şunu ileri sürüyordu: "İlk ortaya konduğu haliyle Çerçeve Problemi, Shanahan ve Thielscher'ın yaklaşımlarıyla çözülmüştür ve Çerçeve Probleminin en azından mantık bölümü artık kapanmıştır". (Burada adı geçenler,

3 John McCarthy, "Circumscription – A Form of Non-monotonic Reasoning", *Artificial Intelligence*, c. 13, s. 27-39, 1980. Vladimir Lifschitz'in sınırlamayla ilgili birkaç makalesi var, örneğin: "On the Satisfiability of Circumscription", *Artificial Intelligence*, c. 28, no: 1, 1986, s. 17-27.

4 Örneğin bkz. John McCarthy, "Applications of Circumscription to Formalizing Common Sense Knowledge", *Artificial Intelligence*, c. 28, no: 1, 1986, s. 89-116.

Londra Imperial College'dan Murray Shanahan ile Dresden Teknoloji Üniversitesi'nden Michael Tielscher'dir.)[5]

Tekdüze olmayan uslamlama için ileri sürülmüş pek çok önerinin görece benzer olup aralarında pek çok ufak teknik farklılıklar bulunduğu dikkatinizden kaçmamış olsa gerek. Hatta, bahsetmediğim başka öneriler de var, örneğin oto-epistemik mantık, tekdüze olmayan mantık, tahminsel uslamlama, doğruluk koruma sistemleri ve olasılık kuramını temel alan yöntemler gibi.[6] Feshedilebilir uslamlama yöntemlerinin bu bolluğu, bununla ilgilenen pek çok YZ araştırmacısının yaratıcılık ve matematik uzmanlığına, ayrıca birbirlerinin önerilerinin sınırlamalarını belirleyip kaçınma becerilerine yorulabilir.

26.2 NİTEL USLAMLAMA

Muhtemelen pek çoğunuz lise ya da üniversitede ya da ikisinde birden fizik dersi almıştır. Fiziğin işi, fiziksel dünyaya dair kuramlar inşa etmektir ve bu kuramlar çoğunlukla matematik diliyle ifade edilir. Örneğin $F = Ma$ formülü, bir cisme etki eden kuvveti (F), o cismin kütlesiyle (M) ve ivmesiyle (a) ilişkilendirir. Fiziksel süreçleri betimleyen birçok matematik formülü bundan daha karmaşıktır. Örneğin, aşağıdaki "dalga denklemi", bir su dalgasının hızını hesaplamakta kullanılabilir:

$$v = \sqrt{\frac{g\lambda}{2\pi} \tanh\left[2\pi \frac{h}{\lambda}\right]}.$$

Örneğin mühendisler bu denklemi, bir dalganın tepe noktasının belirli bir noktayı aşıp aşmayacağını öngörmek amacıyla kullanabilir.

Biz insanlar, yaygınlıkla deneyimlediğimiz pek çok fiziksel süre-

5 Bu problemin tarihçesiyle, başlıca aktörleriyle ve öne sürülen çözümün teknik ayrıntılarıyla ilgilenenler, M. Kamermans ve Tijn Schmits'in "The History of the Frame Problem" isimli makalelerine bakabilirler. Şu internet adresinde: http://student.science.uva.nl/~tschmits/Bachelorproject/index.html, 2004. (Makalede, bu tarihçeyi özetleyen hoş bir çizelge bulunuyor, ayrıca bkz. http://student.science.uva.nl/~schmits/Bachelorproject/poster_HotFP.PNG.) Ayrıca bkz. Murray Shanahan, *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*, Cambridge, MA: MIT Press, 1997.

6 Örneğin bkz. Matt Ginsberg (ed.), *Readings in Nonmonotonic Reasoning*, Los Altos, CA: Morgan Kauffman Publishers, 1987 ve D. Gabbay, C. Hogger ve J. Robinson (ed.), *Handbook of Logic in Artificial Intelligence and Logic Programming*, c. 3, Oxford and New York: Oxford University Press, 1994.

cin gelecekteki akışını, işe yarar bir doğruluk payıyla öngörebiliriz. Örneğin, insanlar plajda okyanus dalgalarıyla oynarken, dalganın tepe noktasının ne zaman geleceğini çoğunlukla tahmin edip dalganın üzerine tam zamanında atlarlar. Peki beynimiz, o tahmini yaparken üsttekine benzer bir denklemden mi faydalanıyor? Muhtemelen hayır. Bunun yerine, beceri gerektiren eylemlere rehberlik eden öngörü altprogramları, tekrarlanan deneyimler aracılığıyla öğrenilir ve bunlar, psikologların "yordamsal bilgi" dediği şeyin parçasıdır.

Otomatik olarak harekete geçmeye ve çeşitli devinim becerilerimize ilişik yordamsal bilgiyi kullanmaya ek olarak, belirli durumlarda ne olup biteceğini öngören bildirimsel önermelerde bulunabiliriz. Örneğin, dalga kollayan bir sörfçü, arkadaşına şöyle diyebilir: "Bir sonraki dalgayı kaçırma; çok büyük olacak."[7] Belli ki, fiziksel süreçlerle ilgili "nitel bilgiyi" temsil etme ve kullanma imkânımız var; nitel bilgi, ne yordamsal "kas belleğimizin" parçasıdır ne de beyinlerimizde karmaşık matematik formülleriyle temsil edilir. Birkaç örnek verebilirim. Masada duran bir bardak suya çarptığımızda, suyun sonunda masanın kenarına gelip döküleceğini nereden biliyoruz? Ağır kutuları, hafif ve kırılgan kutuların üzerine dizdiğimizde, kırılgan kutuların çökebileceğini nereden biliyoruz? Yolda arabayı daha hızlı kullandığımızda, gideceğimiz yere biraz daha çabuk ulaşacağımızı nasıl biliyoruz?

Kimi YZ araştırmacıları, nitel bilgi gösterimi ve uslamlama becerisini taşıyan sistemler üzerinde çalışmıştır. Scott Fahlman (1948-), MIT'de öğrenciyken yüksek lisans tezi için bu tür bir program yazmıştı. BUILD adlı bu program, yerçekimi ve sürtünme gibi, oyuncak bloklar üzerine etki eden çeşitli kuvvetleri hesaba katarak, oyuncak blok dizmeyi planlıyordu.[8] Bu işi, kesin matematiksel modeller kullanmak yerine, aşağı yukarı nitel bir yöntemle başarıyordu. Dolayısıyla BUILD'in, fizik hakkında nitel uslamlama yapmaya yönelik ilk YZ girişimlerinden biri olduğu söylenebilir.

Kısa süre sonra, başka bir MIT öğrencisi Johan de Kleer (1951-), yüksek lisans tezi için NEWTON adlı bir program yazdı; bu program, nitel uslamlama yapabilen bir bileşen içeriyordu. De Kleer'e göre NEWTON, "yüzeylerde hareket eden cisimlerin mini dünyasındaki mekanik

7 Bu dalga denklemini, bir sörf mamülleri üreticisinin internet sitesinde buldum: http.//www.waveequation.com/wave_equation_data.html

8 Scott E. Fahlman, "A Planning System for Robot Construction Tasks", *Artificial Intelligence*, c. 5, no: 1, 1974, s. 1-49. Bu tez, MIT YZ Laboratuvarı Teknik Raporu no: 283 (Mayıs 1973 tarihiyle ve aynı başlıkla) olarak şu internet adresinde: dspace.mit.edu/bitstream/handle/1721.1/6918/AITR-283.pdf?sequence=2.

problemleri anlayıp çözer."[9] NEWTON, problemlere yaklaşık çözümler üretmek amacıyla nitel fizik bilgisi kullanıyordu; ardından bu program, izleyen nicel hesaplamaları planlamak ve yapmak için bu bilgiden faydalanıyordu. Çoğu fizik öğretmeni, matematiğe dalmadan önce bir problemin "fiziğine" dair nitel uslamlamanın çok önemli olduğunu söyler.

NEWTON, nitel uslamlama yaparken, "neler olup bitebileceğini betimleyen, simgesel bir tanımla kodlanmış sahnelerin devamlılığını yaratmak" amacıyla "canlandırma" denilen bir işlemden faydalanıyordu. Bu işlem, NEWTON'un çözebileceği problem türlerine uygun altı temel eylem tanımı kullanıyordu. Örneğin bu tanımlardan biri UÇMAK içindi. Bu tanım, "eğer cisim, hareketten ıraksayan bir yüzey üzerinde hareket ediyorsa, bu cisim uçuyor olabilir", bilgisini kodlar. De Kleer'in başlıca katkısı, bir bilgisayar programında nitel hesaplamaları ve nicel uslamlamanın nasıl birleştirileceğini göstermek olmuştur.

1979'da Pat Hayes, "Naif Fizik Manifestosu"nu yayımladı.[10] Bunun gözden geçirilmiş sürümü 1985'te yayımlanmıştır.[11] Hayes, yapay zekâ araştırmaları camiasının, "cisimler, şekil, uzam, devinim, katı ve sıvı maddeler, zaman vesaire gibi gündelik fiziksel dünya hakkındaki sağduyu bilgisinin büyükçe bir kısmının formül haline getirilmesine" başladığını öne sürmüştü. Bu konu başlıkları, uzun süredir YZ'nin karşısına özellikle zorlu gösterim ve uslamlama sorunları çıkarıyordu. Bu konularla ilgili gündelik bilgimizi, bilgisayarların akıl yürütebileceği şekilde kodlamak, Hayes'in "naif fizik" dediği nitel fiziğin tam merkezinde yer alır."

Hayes'in manifestoları, fiziksel dünyaya dair sağduyu bilgisi "öbeklerinin" nasıl temsil edileceğiyle ilgili kimi genel fikirler sunmuştur. Örnek olarak, olayları temsil etmek için, John McCarthy'le birlikte daha önce savundukları durumlar ve durum fonksiyonları yerine,

9 Johan de Kleer, "Qualitative and Quantitative Knowledge in Classical Mechanics", Yapay Zekâ Laboratuvarı, Teknik Rapor 352, Aralık 1975, bkz. http://dspace.mit.edu/bitstream/handle/172L1/6912/AITR-352.pdi? sequence=2. (De Kleer'in fikirlerinden bazıları yüksek lisans tezi önerisinde betimleniyor. Bkz. http://www2.parc.com/spl/members/dekleer/Publications/QualitativeandQuantitativeKnowledgeinClassicalMechanics.pdf.)

10 Patrick J. Hayes, "The Naive Physics Manifesto", *Expert Systems in the Micro-Electronic Age* içinde, D. Michie (ed.), Edinburgh: Edinburgh University Press, 1979, s. 242-270.

11 Patrick J. Hayes, "The Second Naive Physics Manifesto", *Formal Theories of the Commonsense World* içinde, Jerry R. Hobbs ve Roben C. Moore (ed.), s. 1-36, Norwood, NJ Ablex Publishing Corporation, 1985.

"tarihçe" kavramını önerdi. Onun tanımına göre tarihçe, "hem zamansal hem uzamsal doğal sınırları olan bir uzay-zaman parçasıdır". Örneğin, "bir alanda dört bloğu yan yana dizme olayı, bir düzlemin tarihçesinin başlangıcıdır ve bu tarihçenin sonunu, blokların birbirlerinden ne zaman ve nerede ayrıldığı getirir."

Hayes, aslında, naif fizik programları yazma konusunda "aceleci olmamamız" gerektiğini söylemiştir; bizzat gösterimlerin üzerine daha fazla temel çalışma yapılana dek, uygulamaları geciktirmek tercih edilmelidir, demiştir. Sıvılarla ilgili böyle bir çalışmayı zaten kısmen tasarlamıştı.[12]

Nitel uslamlamadaki bu ilk keşifler, YZ'nin bu alt alanının pek çok uygulamayla birlikte hızla büyümesini getirdi, özellikle de nitel donanım modellerini temel alan hataların tanısında. (Bunun bir örneğini, uzay gemisi donanımındaki hataların tanısını sekizinci kısımda anlatıyorum.) Northwestern Üniversitesi'nde[13] Profesör Kenneth D. Forbus (1955-) ve Austin Texas Üniversitesi'nde Benjamin Kuipers (1949-), önde gelen ekipler kurdular. (Kuipers artık Michigan Üniversitesi'ne geçmiştir.)[14] Bu konu hakkında dergiler özel sayılar basmış, ayrıca birtakım kitaplar derlenip yazılmıştır.[15]

Uslamlama ve Gösterim konulu bu bölümü, bilgi gösteriminde anlamsal ağların yeni kullanım alanlarını anlatarak bitiriyorum.

26.3 ANLAMSAL AĞLAR

Daha önce feshedilebilir uslamlamadan bahsederken, büro makinelerinin taksonomi hiyerarşisini temsil eden anlamsal bir ağ göstermiştim.

12 Patrick J. Hayes, "Naive Physics 1: Ontology for Liquids", *Formal Theories of the Commonsense World* içinde, Jerry R. Hobbs ve Robert C. Moore (ed.), Norwood, NJ Ablex Publishing Corporation, 1985, s. 71-107. Daha eski bir versiyonu: *Memo 35*, Institu pour les Etudes Semantiques et Cognitives, Université de Geneve, 1978.

13 Bu ekibin internet adresi: http://www.qrg.northwestern.edu/

14 Texas ekibinin internet sayfası: http://www.cs.utexas.edu/~qr/.

15 Örneğin bkz. *Artificial Intelligence*, c. 51, no: 1-3, Ekim 1991; *IEEE Expert: Intelligent Systems and Their Applications*, c. 12, no: 3, Mayıs/Haziran 1997; Yumi Iwasaki'nin tanıtıcı makalesi, "Real World Applications of Qualitative Reasoning: Introduction to the Special Issue", *AI Magazine*, c. 24, no: 4, Kış 2003 (bunun önbaskısı şu internet adresinde: http://ksl-web.stanford.edu/people/iwasaki/my-intro.ps), s. 16-21; Benjamin J. Kuipers, *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*, Cambridge, MA: MIT Press. 1994; ve Daniel S. Weld ve Johan de Kleer, *Readings in Qualitative Reasoning about Physical Systems*, San Francisco: Morgan Kaufmann Publishers, 1990.

```
Printers

   subset_of: Office_machines

   superset_of: {Laser_printers,
                 Ink_jet_printers}

   energy_source: Wall_outlet

   creator: John_Jones

   date: 16_Aug_91
```

Resim 26.2. Bir çerçeve. (Uyarlandığı yer: Nils J. Nilsson, *Artificial Intelligence: A New Synthesis*, San Francisco: Morgan Kaufmann Publishers, 1998, s. 311.)

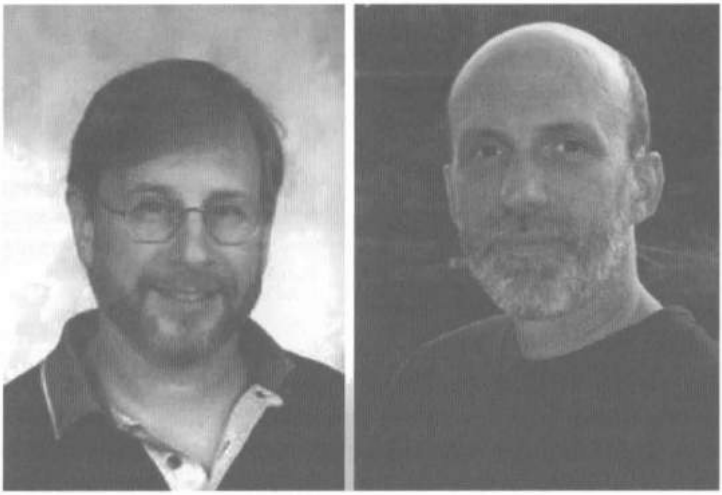
Taksonomi ağları, YZ'de ve bilgisayar biliminde, "varlıkbilim" denilen şeylerin gösteriminde yaygınlıkla kullanılır. YZ'de varlıkbilim, bir kavramlar kümesinden ve o kavramlar arasındaki ilişkilerden oluşur. (Felsefede varlıkbilim, varoluş ya da mevcudiyet araştırmaları anlamına gelir.) Bu ağlarla uslamlama yapan YZ sistemleri genelde, istisna ilkelerinden faydalanan nitelik kalıtım mekanizmalarına sahiptir.

Her ne kadar taksonomi ağlarını anlamanın en iyi yolu, bunları ağaç biçiminde düşünmek olsa da, bunları bilgisayar için kodlarken özel bir veri yapısı kümeleri kullanılır. Bu yapılara, sözcüğü Minsky'nin özgün kullanım biçimi göz önüne alınarak, çoğunlukla "çerçeve" denir. Örneğin, büro makinesi ağının çerçevelerinden biri, Resim 26.2'deki gibi temsil edilebilir.

Bir taksonomide, her birey ya da varlık sınıfı için bir çerçeve olabilir, tıpkı bizzat her varlık için bir çerçeve olması gibi. Sınıf çerçeveleri, ait olduğu üstsınıfı ve ona ait olan altsınıfları adlandırır. Aynı zamanda, bu sınıfa ait varlıkların niteliklerini de belirtir. Ayrıca bir çerçevenin, "meta-enformasyon" taşıması da yaygındır, örneğin çerçevenin yaratıldığı tarih gibi.

26.3.1 Betimleme Mantığı

YZ tarihinin daha erken safhalarında, acaba bilgi, anlamsal ağlar gibi veri yapılarıyla mı gösterilmeli (örneğin çerçeve olarak kodlanarak) yoksa mantıksal yargı kümeleriyle mi gösterilmeli tartışması yaşanmıştır. Bu tartışma yavaş yavaş dindi, çünkü araştırmacılar, anlamsal ağların, belirli mantıksal yargıları temsil etmenin özel bir yolu olarak düşünülebileceği, böylece ağdan bazı çıkarımların doğrudan yapılabileceği fikrini kabul etmişti. Bu görüşün yerleşmesini kolaylaştırmış araştırmacılardan ikisi, Ronald J. Brachman (1949-) ile Hector



Resim 26.3. Ronald Brachman (solda) ve Hector Levesque (sağda). (Ronald Brachman'ın ve Hector Levesque'nun izniyle.)

J. Levesque'ydu (1951-) (Resim 26.3). (İkisi de genel olarak bilgi gösterimi ve uslamlama alanlarında temel çalışmalar yapmıştır.[16])

Brachman, doktorasını[17] Harvard'ta Bill Woods'un yanında yapmıştı. (Woods'un, doğal dil işleme çalışmalarının yanı sıra, anlamsal ağlar ve mantık arasındaki ilişki hakkında da yazıları vardır.[18]) Brachman, tezindeki fikirlerin kapsamını genişleterek, BBN ve USC-ISI'daki çalışma arkadaşlarıyla (Woods dahil) birlikte, KL-ONE[19] adlı bir çerçeve tabanlı bilgi gösterimi sistemi geliştirdi ve bu sistem, *betimleme mantığı* denen mantığın tabanını oluşturdu.

Levesque, üniversite derecelerinin hepsini (lisans, yüksek lisans, doktora) Toronto Üniversitesi'nde tamamlamıştı. 1981'de doktora derecesini aldıktan sonra, Palo Alto Yapay Zekâ Araştırmaları Fairchild

16 Bkz. Ronald J. Brachman ve Hector J. Levesque. *Knowledge Representation and Reasoning*, San Francisco: Morgan Kaufmann Publishers. 2004.

17 Ronald J. Brachman, "A Structural Paradigm for Representing Knowledge", doktora tezi, Mühendislik ve Uygulamalı Fizik Bölümü, Harvard Üniversitesi, Cambridge, MA, 1977.

18 William A. Woods, "What's in a Link: Foundations for Semantic Networks", *Representation and Understanding: Studies in Cognitive Science* içinde, Daniel Bobrow ve Allan Collins (ed.), New York: Academic Press, 1975, s. 35-82.

19 Ronald J. Brachman ve James G. Schmolze, "An Overview of the KL-ONE Knowledge Representation System", *Cognitive Science: A Multidisciplanary Journal*, c. 9, no: 2, 1985, s. 171-216.

Laboratuvarı'nda Brachman'a katıldı; bu ekibi, SRI'den ayrıldığında Hart kurmuştu. Orada Brachman ve Levesque, Richard Fikes'la (o zamanlar Xerox PARC'daydı) birlikte, gösterim ve uslamlama sistemi KRYPTON'u geliştirdiler.[20] KRYPTON, melez bir sistemdi; yani bilgiyi hem mantıksal formüllerle hem de anlamsal ağlarla temsil ediyordu.

Her ne kadar anlamsal ağlar, bireyler ve bunların hiyerarşideki nitelikleri hakkında uslamlama yapmayı kolaylaştırsa da, olumsuzlama ve ayrım içeren önermeleri temsil etmeleri zordur. KRYPTON makalesinin belirttiği gibi: "örneğin 'Çiftçi Jones'un tarlasında duran inek ya Elsie'dir ya da Bessie' gibi bir yargı, tipik bir bildirisel çerçeve sisteminde ileri sürülemez". KRYPTON'un çözümü, iki gösterim türünün bir birleşimini kullanmaktı:

> ... [uslamlama] işlemlerini iki ayrı türe böldük, böylece gösterim sistemimiz için iki ana bileşen ortaya çıktı: terminolojik bileşen [yani ağ tabanlı] ya da *T kutusu* ve bildirisel bileşen [yani mantık tabanlı] yani *A kutusu*. T kutusu, yapılandırılmış koşulların taksonomisini kurmamızı ve bu koşullar arasındaki çözümlemeli ilişkilerle ilgili soruları yanıtlamamızı mümkün kılıyor; A kutusu, ilgilenilen alanların betimlemeli kuramlarını oluşturmamıza ve o alanlar hakkındaki soruları yanıtlamamıza olanak tanıyor.[21]

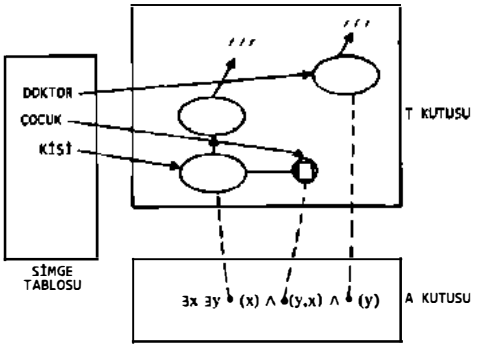
KRYPTON makalesinden alınmış ve Resim 26.4'te gösterilen diyagram, bu sistemin yapısını açıklıyor. Makalenin yazarlarının dediği gibi, sistem, "taksonomik olarak örgütlenmiş, yapılandırılmış koşulların T kutusundan, belirtimleri T kutusu kaynaklı (kabaca) birinci derece cümlelerin A kutusundan ve kullanıcıların atıfta bulunabileceği T kutusu koşullarının adlarını barındıran bir simge tablosundan meydana geliyor". Diyagramdaki T kutusu, (başka şeylerin yanı sıra) çocuğun da kişi olduğunu temsil ediyor. A kutusu, çocuğu olan bir doktorun var olduğunu ifade ediyor.

KRYPTON, AT&T Bell Laboratuvarları'na geçtikten sonra, Brachman ve çalışma arkadaşları tarafından geliştirilmiş CLASSIC dahil birtakım betimleme-mantığı sistemlerinin öncüsüydü.[22] YZ uslamlama sistem-

20 Ronald J. Brachman, Richard E. Fikes ve Hector J. Levesque, "KRYPTON: A Functional Approach to Knowledge Representation", *IEEE Computer*, c. 16, no: 10, Ekim 1983, s. 67-73. Tekrar baskısı: Ronald J. Brachman ve Hector J. Levesque (ed.), *Readings in Knowledge Representation*, San Francisco: Morgan Kaufmann Publishers, 1985, s. 411-429.

21 *A.g.e.*, s. 68-69.

22 Betimleme mantıkları hakkında daha fazla bilgi ve kaynak içeren bir internet sitesi



Resim 26.4. KRYPTON T kutusunun ve A kutusunun parçaları. (Uyarlandığı yer: Ronald J. Brachman, Richard E. Fikes ve Hector J. Levesque, "KRYPTON: A Functional Approach to Knowledge Representation", *IEEE Computer*, sayı 16, no. 10, Ekim 1983, s. 67-73.)

lerindeki kullanımlarına ek olarak, betimleme mantıkları, anlamsal Ağ için varlıkbilim dillerinde de kullanılmıştır, örneğin DAML-ONT[23] ve OWL[24] gibi.

26.3.2 WordNet

WordNet, İngilizce sözcüklere dair büyük bir "kavramsal" sözlüktür, bir dereceye kadar anlamsal ağlar gibi düzenlenmiştir ve insanın sözcük belleğine dair psiko-dilbilim ve hesaplama kuramlarından esinlenilmiştir.[25] Bunun geliştirilmesine 1980'lerde Princeton Üniversitesi'nde Profesör A. George Miller yönetiminde başlandı (daha önce bahsettiğim, "Sihirli Sayı Yedi, Artı Eksi İki" başlıklı makaleyi yazmış olan George Miller). Miller ve çalışma arkadaşları, 1990 tarihli makalelerinde, projenin başlangıcını anlatır:[26]

> 1985'te Princeton Üniversitesi'nde psikologlardan ve dilbilimcilerden oluşan bir ekip, [önceki psiko-dilbilimsel] araştırmaların önerdiği çizgide

için bkz. http://dl.kr.org/.

23 Deborah L. McGuinness, Richard Fikes, Lynn Andrea Stein ve James Hendler, "DAML-ONT: An Ontology Language for the Semantic Web", *The Semantic Web: Why, What, and How* içinde, Dieter Fensel, Jim Hendler, Henry Lieberman ve Wolfgang Wahlster (ed.), Cambridge, MA: MIT Press, 2002; şu internet adresinde: http://www.ksl.stanford.edu/people/dlm/papers/daml-ont-semantic-web.htm.

24 Deborah L. McGuinness ve Frank van Harmelen. "OWL Web Ontology Language Overview", W3C Recommendation, 10 Şubat 2004; www.w3.org/TR/owl-features/.

25 Christine Fellbaum (ed.), *WordNet: An Electronic Lexical Database*, Cambridge, MA: MIT Press, 1998. Ayrıca WordNet internet sitesine bakınız: http://wordnet.princeton.edu/ ve Wikipedia makalesi şu internet adresinde: http://en.wikipedia.org/wiki/WordNet.

26 George A. Miller ve diğ., "Introduction to WordNet: An On-line Lexical Database", *International Journal of Lexicography*, c. 3, no: 4, 1990, s. 235-244.

bir sözcük veritabanı geliştirme işini üstlendi. Başlangıç fikri, sözlükleri alfabetik sıra yerine kavramsal olarak aramaya yardımcı olmaktı; geleneksel türde bir sözlüğün çevrimiçi sürümüyle birlikte kullanılacaktı bu. Ancak çalışmalar ilerledikçe, bu veritabanı, kendi ilke ve hedeflerinin daha iddialı formülasyonlarını gerektirir oldu. Sonuçta ortaya WordNet çıktı. WordNet'in, psiko-dilbilimsel araştırmaların sonuçlarını temel alan hipotezlere kanıt olduğu ölçüde, psiko-dilbilimsel ilkelere dayalı bir sözlük olduğu söylenebilir.

WordNet, bünyesindeki sözcükleri, "eş küme" adlı derlemeler halinde kümeleştirir. Her eş küme, eşanlamlı sözcüklerden, yani aşağı yukarı aynı anlama gelen sözcüklerden meydana gelir. Eş kümelere ve aralarındaki bağıntılara dair bazı örnekler vermek amacıyla, WordNet'in çevrimiçi arama olanağından faydalanacağım (http://wordnetweb.princeton.edu/perl/webwn). Örneğin "bilgisayar" (*computer*) sözcüğü iki ayrı eş kümede bulunuyor. Bu eş kümelerden biri, "bilgisayar" sözcüğünün yanı sıra, eşanlamlısı "hesaplama makinesi", "hesaplama cihazı", "veri işlemcisi", "elektronik bilgisayar" ve ""bilgi işleme sistemi" sözcüklerini kapsıyor. Öbür eş küme, "bilgisayar" sözcüğünün (hesaplama işini insanların yaptığı zamanlardaki) daha eski kullanımının eş anlamlılarını içeriyor, örneğin "hesap makinesi" (kalkülatör), "hesaplayıcı", "işlemci", "kestirici" gibi. Ayrıca eş kümeye kısa bir tanım da eşlik edebilir; "açıklama" adı verilen bu tanım, eş kümedeki sözcüklerin anlamını belirtir. Yukarıda bahsettiğimiz ilk eş kümenin açıklaması, "otomatik hesaplama yapan makine"dir. İkincisinin açıklamasıysa, "hesaplama işinde (ya da hesap makineleri kullanmada) uzman"dır. Bazen açıklama, sözcüğün tipik kullanımını gösteren örnek bir cümle de içerir.

Eş kümeler, başka eş kümelerle bağlantı kurarken, anlamsal ağlarda kullanılan bağıntılara benzer bağıntılardan faydalanır. Bu tür bir bağıntıya "hypernym" denir; bu, (kabaca) "bir tür" anlamına denk geliyor. Örneğin, "bilgisayar" ve "hesaplama makinesi" vesaire gibi sözcükleri içeren eş kümemizin hypernym'i, "makine" sözcüğünü (muhtemelen başka sözcükleri de) içeren ve "insan işlerinin icrasını gerçekleştirmek ya da yardımcı olmak amacıyla enerji ileten ya da dönüştüren herhangi bir mekanik ya da elektrikli cihaz" açıklamasını taşıyan bir eş kümedir. Aşağı yukarı "genel vaka" anlamına gelen "hiponim", hypernym'in tersidir. Yukarıda bahsettiğimiz "bilgisayar" eş kümesinin birkaç hyponym'i birden vardır; bunlar arasında, "sayısal bilgisayar" ve eşanlamlısı sözcükleri (enformasyonu sayısal basamaklarla temsil eden bilgisayar), "sayı canavarı" ve eşanlamlısı sözcükleri (büyük rakamlı matematik işlemlerini saniyede yapabilen bilgisayar), "Turing

makinesi" ve eşanlamlılarından herhangi birini (sonsuz bellek şeridi olan varsayımsal bilgisayar), vesaireyi içerenler vardır.

Başka bağıntılar da var. Cins isim içeren eş kümelerin bağıntısına "meronim" denir ve bu, "parçası olarak barındırmak" anlamına gelir. "Bilgisayar" ve "hesaplama cihazı" sözcüklerini içeren eş kümenin birkaç meronim'i bulunur, örneğin "çip" ve eşanlamlı sözcükleri ("bütünleşik devrede pek çok elektronik işlevi yerine getirmek üzere imal edilmiş, silikon yarı-ileticilerin küçük kristalinden oluşan elektronik donanım" açıklamasıyla), "ekran" ve eşanlamlı sözcükleri ("sinyal alıp bunu televizyon ekranına ya da bilgisayar ekranına yansıtan cihazın oluşturduğu görüntü" açıklamasıyla), vesaire gibi. "Holonim" ise, meronim'in zıddıdır.

Aynı zamanda her eş küme, kapsadığı sözcüklerin cümle öğesi niteliğini de belirtir: isim, yüklem, sıfat, zarf. Eş kümelerin arasındaki ilişkiler, cümle öğesi niteliğine göre değişir. Örneğin, yüklem eş kümelerinin "gerektirim" denilen bir bağıntısı vardır. Mesela, "yürümek" fiilinin eş kümelerinden biri (ilerlemek amacıyla ayağı kullanmak; adım atarak ilerlemek), "adım atmak" fiilini içeren eş kümeyi (adım atarak yer değiştirme ya da hareket etme) gerektirir.[27]

Kendi internet sitesine göre WordNet (bu satırların yazıldığı zaman), 155.287 sözcük ve 117.659 eş küme içeriyor. Bu sözlüğün sürdürülmesini ve genişletilmesini Princeton üstlenmiştir ve internetten indirilmesi halka açıktır. Çevrimiçi sözlük ve eşanlamlılar sözlüğü olarak kullanımının yanı sıra, otomatik metin çözümlemelerini desteklemek amacıyla, doğal dil işleme uygulamalarında, soru yanıtlamada bilgi tabanı olarak ve anlamsal ağ uygulamalarında da bundan yararlanılır. Üstelik, düzinelerce dilde benzer "sözcük ağları" oluşturulmuştur.

WordNet'in, taksonomi bilgi tabanında varlıkbilim olarak kullanımı, isim eş kümeleri arasındaki hypernym/hyponym ilişkisine ve eş kümeyle temsil edilen nesnelerin özelliklerini, bu nesnelerin atalarının özelliklerinden çıkarsamaya yarayan miras mekanizmasını WordNet'in kullanımına göre değişir. Örneğin, bu tür bir hiyerarşide yer alan zincir (özelden genele) şöyle olabilir:

Çalışma istasyonu → sayısal bilgisayar → bilgisayar
→ makine → cihaz → araç → insan eseri → ...

27 WordNet'in ayrıntılarıyla ilgilenen okurlar, WordNet'in internet sitesine ve şu dergideki beş makaleye göz atabilir: *International Journal of Lexicography*, c. 3, no: 4, 1990.

Elbette, bu zincir üzerinde yan dallar da bulunur (WordNet'in çevrimiçi arama olanağını kullanarak bu yan dalları keşfedebilirsiniz).

Bununla birlikte, WordNet'i varlıkbilim olarak kullanırken üzerinde bazı değişiklikler yapmak gerekebilir, zira, Wikipedia'ya göre, "... yüzlerce temel anlamsal tutarsızlık içeriyor, örneğin (i) istisnai kategoriler için ortak bir uzmanlaşmanın varlığı ve (ii) uzmanlaşma hiyerarşisinde gereksiz fazlalık, vesaire gibi."[28]

Karl Kipper Schuler bununla ilintili bir uğraşın sonucunda "VerbNet"i yarattı.[29] Bunun hakkındaki bir internet sayfasına göre,[30] VerbNet, "şu an için İngilizcenin en geniş çevrimiçi yüklem sözlüğüdür. Bu, uzmanlık alanından bağımsız, geniş kapsamlı, hiyerarşik bir yüklem sözlüğüdür ve WordNet, Xtag, Framenet gibi başka sözlük kaynaklarıyla eşlenir."

26.3.3 Cyc

1984'te, pek çok YZ uygulaması, özellikle de doğal dil anlama için büyük miktarda sağduyu bilgisinin gerekli olduğunu fark eden Stanford profesörü Douglas Lenat (1950-), ki daha önce matematiksel kavramlarının keşfini otomatikleştirme ve sezgisel yöntemler üzerine çalışmaları vardı,[31] bilgisayarlara sağduyu bilgisi kazandırmak gibi zorlu bir görev üstlendi (bkz. Resim 26.5). Ona göre ilk adım, sağduyuyu meydana getiren "milyonlarca gündelik terim, kavram, olgu ve göz kararı kuralı hazır hale getirmek" olacaktı. Daha sonra, projenin şu şekilde başladığını anlatacaktı:[32]

> 1984 güzünde Amiral Bobby Ray Inman beni, eğer bu ilk adımı atma konusunda ciddiysem, akademiyi bırakıp Texas Austin'de yeni kurmuş olduğu MCC'sine (Mikroelektronics and Computer Consortium) gelmeye ve bu işi yapacak bir ekip toplamaya ikna etti. Fikir şuydu: önümüzdeki

28 http://en.wikipedia.org/wiki/WordNet.

29 Karin Kipper Schuler, "VerbNet: A Broad-Coverage. Comprehensive Verb Lexicon", Pennsylvania Üniversitesi, doktora tezi, 2005. Bkz. http://repository.upenn.edu/dissertations/AAI3179808/.

30 http://verbs.colorado.edu/~mpalmer/projects/verbnet.html.

31 Davis ve Douglas B. Lenat (ed.), *Knowledge-Based Systems in Artificial Intelligence*, New York: McGraw-Hill, 1982, s. 1-225; Douglas B Lenat "Eurisko: A Program Which Learns New Heuristics and Domain Concepts", *Artificial Intelligence*, c. 21, no: 1-2, 1983, s. 61-98.

32 Lenat'ın şu kitaptaki makalesinden: David G. Stork (ed.), *Hal's Legacy: 2001's Computer as Dream and Reality*, Cambridge, MA: MIT Press, 1998. Bu makale şu internet adresinde: http://www.cyc.com/cyc/technology/halslegacy.html.





Resim 26.5. Douglas Lenat (üstte) ve Cyc bilgi tabanının yapısı (altta). (Fotoğraf, Douglas Lenat'ın izniyle).

> on senede düzinelerce insan, Cyc adlı sağduyu sahibi programı yaratacaktı. Programı bilgiyle donatmak için, milyonlarca önemli olguyla ve göz kararı kuralla Cyc'i elimizle kaşık kaşık besleyecektik.

"Cyc" adı ("sayk" diye okunuyor), "encyclopedia" (ansiklopedi) sözcüğünün ortasındaki üç harften geliyor. Eğer Cyc, ansiklopedi maddelerini anlayacak kadar bilgi edinirse, her türlü metni okuyabileceği ve bilgisine bilgi katabileceği düşünülmüştü; akıllı bilgisayarlara doğru "ikinci adım" budur. ("Üçüncü adım", insanların bildiklerinin ötesinde daha fazla bilgi edinmek amacıyla deneyler tasarlayıp yürüten bilgisayarlar olur.)

Ansiklopedi maddelerini anlamak için, dünya hakkında belirli bir bilgimiz olması gerekir. Lenat'ın da belirttiği gibi:[33]

> Bir ansiklopedi maddesinden herhangi bir cümleyi alıp okurların bildikleri hakkında yazarın ne varsaydığını düşünecek olursak, Cyc'a söyleme-

33 *A.g.e.*

> ye değer bir şeyimiz var demektir. Ayrıca, bir paragrafı ele alıp cümleler arası "geçişlerde" yazarın okurlardan beklediği cümleler "arası" çıkarımlar hakkında kafa yorabiliriz. [Örneğin şu cümlelere bakalım:] "Napoleon, St. Helena'da öldü. Bunun üzerine Wellington epey hüzünlendi". Yazar okurdan, Napoleon'un ölümünü Wellington'ın haber aldığını ve Wellington'ın Napoleon'dan uzun yaşadığını çıkarsamasını vesaire bekler.

Peki, ansiklopedideki maddeleri anlamak için Cyc'ın ne kadar bilgiye ihtiyacı var? Kısa süre önce Lenat bana, aslında Cyc'ın "örneğin 'memeliler tüylüdür' benzeri birkaç milyon genel değerlendirmeye (ayrıca California'nın başkenti gibi sayısız özel olguya)" gereksinim duyacağını düşündüğünü söyledi. Artık "bu rakamın aşağı yukarı 200 milyon" olduğuna inanıyor.[34]

Lenat ve programcılar ile "bilgi giricileri"nden oluşan ekibi, MCC'de on yıl boyunca Cyc üzerinde çalışıp elleriyle bilgi girdiler. Bu projeyi ve hedeflerini betimleyen birkaç rapor, pek çok makale ve bir kitap kaleme alınmıştır.[35] 1994'te Lenat, kısmen MCC'deki zorluklardan ötürü, Texas'ın Austin kentinde, "Yapay Zekâyı araştırmak, geliştirmek ve ticarileştirmek amacıyla" Cycorp'u kurdu. Cyc çalışmaları burada Lenat ve ekibi tarafından sürdürülüyor.

Cyc'ın, bilgi gösterimi için kullandığı dilin adı CycL'dir ve bu dil, birinci dereceden yüklemcil biçimsel dizgenin bir uzantısıdır. Cyc'ın bildiği nesne sınıfları, taksonomik bir hiyerarşide düzenlenmiştir (Resim 26.5'te gösterildiği gibi); bu düzenleniş sayesinde nesne sınıfları, hiyerarşide daha yüksekte duran nesne sınıflarının özelliklerini miras alabilir. Örneğin Cyc hiyerarşisinde "ışık düğmesine basmak" gibi bir "olay sınıfı", "zamansal-şey"in altsınıfıdır, o da "bireyin" altsınıfıdır, o da Cyc'daki en genel sınıf olan "şey"in altsınıfıdır. Cyc, nesneler arasındaki ilişkileri betimlemek için birtakım "kurallar"dan faydalanır (bu kurallar onun mantık diliyle ifade edilir). Örneğin kurallarından biri konuşma diliyle şu anlama gelir: "Tüm *A* ve *B* olaylarında, *A*'nın *B*'ye neden olması, *A*'nın *B*'den önce gelmesi demektir."

Cyc'ın bilgi tabanı (BT), binlerce mikrokurama bölünmüştür; mikrokuram, sınırlı bir alanla ilgili kavramların ve olguların derlemesidir.

34 E-postayla iletişim, 24 Ocak 2008.

35 Ana makale: Douglas B. Lenat ve diğ., "Cyc: Toward Programs with Common Sense", *Communications of the ACM*, c. 33, no: 8, Ağustos 1990, s. 30-49. Cyc projesinin ilk beş yılı şu kitapta anlatılıyor: Douglas B. Lenat ve R. V. Guha, *Building Large Knowledge-Based Systems*, Reading, MA: Addison-Wesley, 1990.

Örneğin, mikrokuramlardan biri, Avrupa coğrafyası hakkında bilgi içerir. "Kimya, biyoloji, askerî teşkilatlar, hastalıklar ve silah sistemleri" konularında uzmanlaşmış mikrokuramlar da vardır. Her mikrokuram kendi içinde tutarlıdır, gerçi bir bütün olarak alındığında, Cyc bilgi tabanı çelişkiler içerebilir. Cyc'ın BT'si, beş milyonu aşkın genel değerlendirme kapsıyor.[36] Bunların çoğunluğu, "alışveriş, akraba ilişkileri, ev cihazları, yemek yeme, ofis binaları, taşıtlar, zaman ve mekân gibi gündelik insan yaşamının nesneleri ve olayları" hakkında sağduyu bilgisi barındırır. Buna ek olarak BT, doğal dil işleme için gerekli olan dilbilim ve sözcük bilgisini de içerir.

Cyc, kendi BT'sinde var olan olgulardan ve kurallardan yola çıkarak yeni olgulara dair yargıda bulunmak için bir "çıkarsama motoru"ndan faydalanır. Başlıca iki çıkarsama yöntemi kullanılır. Bir tanesi, Bölüm 11.1'de de bahsetmiş olduğum, kesin çözüm adlı çıkarsama kuralıdır. Kesin çözüm yöntemi aracılığıyla etkin şekilde akıl yürütmek amacıyla Cyc, tescilli sezgisel yöntemler geliştirmiş ve kendi arama süreçlerinin ölçeğini, mikrokuram kullanımıyla kısıtlamıştır. Öteki çıkarsama yöntemi, nitelik kalıtımıdır; bu yöntem, anlamsal ağ gösterimlerinde yaygınlıkla kullanılmaktadır. "Aynı zamanda Cyc, özgül çıkarsama sınıflarıyla başa çıkmak için 1.000'i aşkın özel amaçlı çıkarsama modülünden faydalanır. Bu tür bir modül, küme üyeliğini/üye olmayışı ilgilendiren uslamlamayı ele alır. Eşitlik uslamlaması, zamansal uslamlama, matematiksel uslamlamayla vs. uğraşan modüller de var... CycL, bir sınırlama biçiminden faydalanır... ayrıca yeri geldiğinde kapalı dünya varsayımından yararlanır."[37]

Cycorp, kimi uygulamalar üzerinde çalışıyor. Bu uygulamalara, internette zeki arama ve bilgi toplama ile doğal dil anlama dahildir. Cycorp'un internet sitesi, bunun, "gerçek dünyada iş dünyasının pek çok problemine hitap eden uygulamalarıyla artık faal bir teknoloji" olduğunu söylüyor.

Ancak, Cyc'a yönelik eleştiriler de yok değil. Örneğin Cyc, insanların kolaylıkla altından kalktığı kimi uslamlama problemlerinde tıkanıp kalır. Engin bilgi tabanı yüzünden, uslamlama işlemleri öyle yavaş yürür ki sonuçta hiçbir işe yaramaz (daha fazla bilgi eklendikçe iyice yavaşlayacağı şüphesizdir). YZ araştırmacılarının çözmekte hâlâ

36 13 Ocak 2009, Doug Lenat'ın e-postası.

37 Cyc'ın, bilgi temsil ediş şekli ve Cyc'ın uslamlama biçimi hakkında daha fazla ayrıntıyla ilgilenen okurlar, şu internet adresindeki eğitici slaytları izleyebilir: http://www.opencyc.org/releases/doc/tut/index_html?tree-e=eJyLLWTUCOVxhAJnIwPbQiYE38Ux2baQOVUPALHZCUs#AAAAAAAADAc=.

zorlandığı belirli gösterim problemlerine tatmin edici sonuçlar bulamaz; örneğin maddelerin gösterimi gibi. Ayrıca, Cyc'la ilgili çalışmaların büyük kısmı gizlilik içinde yapıldığından, genelde meslektaşların değerlendirmesine açık değildir.

Her ne kadar Cyc, doğal dili anlıyormuş gibi görünecek kadar bilgi sahibi olsa da, tipik bir İngilizce soruyu CycL'e otomatik olarak yeterli oranda çeviremez henüz (yani kullanıcıyla etkileşime girmeksizin). Cyc'ı sorgularken, ya külfetli CycL dili kullanılmalı ya da etkileşimli tarzda çalışılmalı. Lenat bana, tıp araştırmacılarının hasta veritabanlarından bilgi almak için Cyc'ı kullandığı Cleveland Kliniğinde hayata geçirilmiş, bu tür bir etkileşimin işe yarayan örneğini anlatmıştı.[38] Araştırmacı, sorgusunu İngilizce dilinde tuşlar. "Cyc elinden geldiğince bu sorguyu çözümler, sorgunun bazı kısımlarını tanır, kullanıcıya da boşlukları doldur tarzı kalıplara benzeyen kısmi sorgu 'parçaları' sunar". Ardından Cyc, tıp konusunda uzman bilgisiyle birlikte sağduyu bilgisini kullanarak, kullanıcının merak ettiği sorguyu nasıl ifade edeceğini belirler. Sorduğu soru hakkında kullanıcının onayını alan Cyc, istenilen bilgiyi getirmek için veritabanı sorgularının yaratılmasında veritabanının nasıl düzenlendiğiyle ilgili bilgisinden yararlanır.

Cyc'ın, internetten indirilebilen iki sürümü var. Bir tanesinin adı Research-Cyc'dır ve araştırmacılar (salt araştırma amacıyla), Research-Cyc lisansı altında buna ulaşabilir.[39] Cyc çıkarsama motorunun yanı sıra, "neredeyse 3.000.000 değerlendirme (olgular ve kurallar) içerir ve kavramları ilintilendiren, kısıtlayan ve aslında (kısmen) tanımlayan 26.000'i aşkın bağıntıdan faydalanır". OpenCyc denen öbür sürüm, Cyc teknolojisinin halka açık kaynak sürümüdür.[40] "Yüzbinlerce terimle birlikte bu terimler arasında bağıntı kuran milyonlarca değerlendirme" içerir. Ayrıca, çevrimiçi bir pencere sayesinde OpenCyc'ın kavram hiyerarşisini de incelemek mümkündür.[41]

İnsanların, sağduyu (ve uzman) bilgisini nasıl örgütleyip kullandığını kimse bilmiyor. Cyc projesinin şimdiye dek toparladığı (ve daha da toparlayacağı) olgu ve bağıntıların miktarı ve örgütlenişi, acaba insan düzeyinde uslamlamaya olanak tanıyacak mı, henüz ispatlanmış değil. Yine de bu çabayı ayakta alkışlıyor, başarılı olmasını diliyorum. Bence, Cyc kadar iddialı bir şeyin gerektiği kesin. (Sağduyu bilgisi

38 E-postayla iletişim, 25 Ocak 2008.

39 Bkz. http://researchcyc.cyc.com/

40 Bkz. http://opencyc.org/.

41 http://www.cycfoundation.org/concepts.

toplamaya yönelik başka bir girişim de MIT Medya Laboratuvarında "Sağduyu Bilişim Girişiminin" çabasıdır. Buradaki çalışmalar http://xnet.media.mit.edu/ internet adresinde anlatılıyor).

Cyc'ın, gerekli bilgiyi (insanların yardımıyla) doğrudan internetten elde edebileceği noktaya ulaşılması mümkündür. Lenat, "FACTory" (olgu-fabrikası) adlı bir oyundan bahseder;[42] bu oyun, oyunu oynayan insanlardan bilgi toplamak üzere tasarlanmıştır. Şu internet adresinde oynayabilirsiniz: http://game.cyc.com/game.html. Bu oyunda Cyc, internette bulduğu İngilizce cümlelerden topladığı birtakım doğal dil ifadeleri yaratır. Bu ifadeleri, rasgele seçilen on oyuncuya sunar. Eğer yeterli sayıda oyuncu bu ifadenin "doğru" olduğu yanıtını verirse, Cyc bu olguyu BT'sine ekler (oyuncu da oyunda puan toplamış olur). Oyunu oynamayı denedim ve Cyc bana, "Tüm spagetti marinara soslarında sarımsak var mı yok mu?" diye sordu. "Doğru" yanıtını verdim ve Cyc öteki oyuncuların %66'sıyla hemfikir olduğumu, dolayısıyla bu cümlenin "doğru" olduğuna inandığını belirtti.

42 E-postayla iletişim, 24 Ocak 2008.

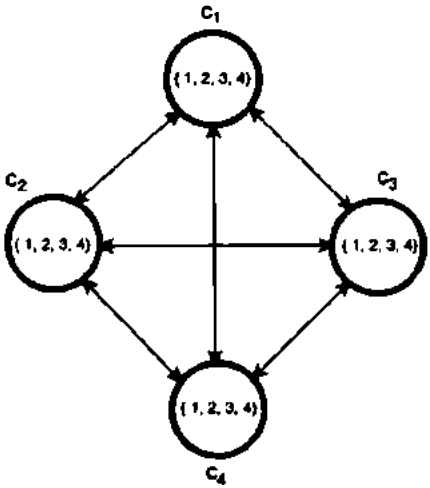
27
USLAMLAMAYA VE GÖSTERİME YÖNELİK BAŞKA YAKLAŞIMLAR

27.1 KISIT SAĞLAMA PROBLEMLERİNİ ÇÖZMEK

Mantık ya da anlamsal ağları temel alan uslamlama yöntemlerine ek olarak, başka teknikler de keşfedilmiştir. Bu bölümde, kısıt sağlama problemi (ya da atama problemi) denen problem sınıfını ve bunları çözme yöntemlerini betimliyorum. Bu problemlerde, belli bir kısıt kümesini sağlayan değerleri alması zorunlu olan bir nesneler kümesi söz konusudur. Atama problemine dair bir örneği daha önce gördük: imgedeki çizgilere etiket atamak. O problemdeki kısıt, imgedeki her çizgiye ancak ve ancak tek etiket atanabilmesiydi.

Kısıtlamalar; veritabanı bağıntıları, mantık formülleri, denklemler ya da eşitsizlikler biçiminde ifade edilebilir. Dolayısıyla, kısıt sağlama problemleri, çizelgeleme, benzetim, bilgisayarla görme ve robotik dahil pek çok ortamda doğal olarak ortaya çıkar. (Örneğin, hesap çizelgesi, basit bir kısıtlamaya uyma sistemidir.) Neyse ki, bu problemler için, uygulamadan bağımsız bazı çok amaçlı çözüm yöntemleri vardır. Bu tür bir yöntemi, küçük bir örnekle birlikte açıklayacağım.

4 × 4'lük satranç tahtasına, hiçbiri birbirini alamayacak şekilde dört vezir yerleştirme problemine bakalım. Dört Vezir probleminde, c_1, c_2, c_3, c_4 adlı dört nesne var ve vezirlerin yerleştireleceği sütunlar 1'den 4'e kadar rakamlarla gösteriliyor. Bu nesnelerden her biri, yatay sıra numaralarına denk gelen 1, 2, 3, 4 gibi dört farklı değer alabiliyor. Dolayısıyla, örneğin c_3, 2 değerini almışsa, vezirlerden biri üçüncü sütunun ikinci sırasına yerleştirilir. Dört Vezir problemi, bu değişkenlerin değerlerini kısıtlar. Örneğin, eğer c_1 1 değerini almışsa, c_2 1 ya da 2 değerini alamaz; c_3 1 ya da 3 değerini alamaz; ve c_4 1 ya da 4 değerini alamaz. Bu kısıtlar, *kısıt grafiği* adlı bir grafikle gösterilir. Bu grafikteki her düğüm noktası, o nesnenin tüm değerlerinin kümesiyle birlikte, o nesnenin adıyla etiketlenir. Eğer ayrıtın kuyruğundaki nesnenin olası değerleri, ayrıtın başındaki nesnenin herhangi bir değeriyle kısıtlanmışsa, düğüm noktası çifti bir ayrıt (doğrultusu olan bir kenar) üzerinden birbirine bağlanır. Resim 27.1'de, Dört Vezir probleminin bu tür bir grafiğini gösteriyorum. Bu problemde, her nes-



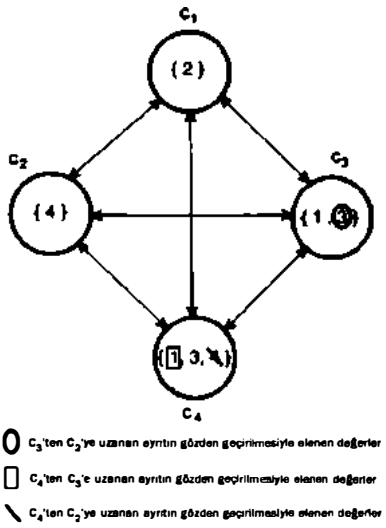
Resim 27.1. Dört Vezir probleminin kısıtlama grafiği. (Nils J. Nilsson, *Artificial Intelligence: A New Synthesis* içinde, San Francisco: Morgan Kaufmann Publishers, 1998, s. 185.)

ne öbürlerini kısıtlar; böylece tüm düğüm noktalarının öteki düğüm noktalarıyla aralarında ayrıt oluşur. (Resimde fazla kalabalık olmasın diye, iki farklı ayrıtı tek çizgiyle ve uçlarında okla gösteriyorum.)

İşe bu nesnelerden birine değer biçerek başlarız. Bu, bir "deneme" değeridir ve arama sürecinin başlangıcıdır. Eğer işe yaramazsa, geri dönüp başka bir değer denememiz gerekir. Diyelim ki c_1 nesnesine 2 değerini atadık (veziri birinci sütunun ikinci satırına koymaya karşılık geliyor). Şimdi, Resim 27.1'deki tüm ayrıtları tekrar tekrar inceleyerek, bir ayrıtın kuyruğundaki nesnenin değeri olup o ayrıtın başındaki *tüm* değerlerle (kısıtlamalar uyarınca) tutarsızlık sergileyen her değeri eleriz. Kısıt yayma adlı bu süreç, elenecek değer kalmadığında durur. Sürecin ilk birkaç adımı şöyle olabilir:

1. Öncelikle, c_2'den c_1'e uzanan ayrıta bak; $c_1 = 1$, $c_2 = 2$ ve $c_1 = 3$ değerlerini eleyebiliriz, çünkü bunların her biri, (tek başına olduğunda) c_1'in değerleriyle tutarsızdır.
2. Sonra, c_3'ten c_1'e uzanan ayrıta bak; $c_3 = 2$ ve $c_3 = 4$ değerlerini eleyebiliriz.
3. Sonra, c_4'ten c_1'e uzanan ayrıta bak; $c_4 = 2$ değerini eleyebiliriz.

Az önce yaptığımız gibi bu değerlerin bazılarını elemek, şimdi daha fazla değerin elenmeye aday olmasını getirir. Ayrıta geri dönüp tutarlılığı yoklamak, hangilerinin tutarlı olduğunu açığa çıkaracaktır. Değer elemenin, kısıt grafiği üzerinde "yayıldığını" söyleyebiliriz böylece.



Resim 27.2. Kısıt yayılmasını açıklayan kısıtlama grafiği. (Nils J. Nilsson, *Artificial Intelligence: A New Synthesis* içinde, s. 185, San Francisco: Morgan Kaufmann Publishers, 1998.)

Yayılma işlemini devam ettirdiğimizde, her düğüm noktası için birer değer dışında, tüm değerler elenmiş olur. Bu noktada, tüm ayrıtlar kısıtlarla tutarlıdır ve hiçbir değer elenemez. Resim 27.2'de gösterilen grafik, sıralanmış üç adımdan sonra kalan değerlerle bu sürecin nasıl ilerlediğini anlatıyor. Bu örnekte, kısıt yayılması, söz konusu problemi çözmüştür (c_1 = 2 değeriyle başlayarak şanslı tahmin yapmışız). Dört vezirin tahtaya yerleşimi Resim 27.3'teki gibi olur.

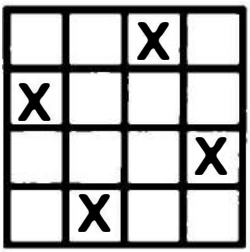
Kısıtlara uyma problemlerini ele alan bu süreç, British Columbia Üniversitesi profesörlerinden Alan K. Mackworth'un (1945- ; Resim 27.4) ileri sürdüğü AC-3 (Arc Consistency Algorithm, no. 3) adlı algoritmayı temel alır.[1] Mackworth, robotik ve etmen denetimi alanlarında kısıt problemleri ve uygulamaları üzerinde çalışmayı sürdürmüştür. (Ayrıca, futbol oynayan robotları ilk önerip yapan kişidir.)

AC-3 için çeşitli uzantılar ve iyileştirmeler önerilmiştir. Bunlar, Rina Dechter'in kitabında[2] (o da bu alana büyük katkılarda bulunmuştur) ve Russell ile Norvig'in kaleme aldıkları kitabın Beşinci Bölümünde[3] gayet güzel anlatılıyor. Vipin Kumar'ın makalesiyse, bütün

1 Alan K. Mackworth, "Consistency in Networks of Relations", *Artificial Intelligence*, c. 8, no: 1, 1977, s. 99-118.

2 Rina Dechter, *Constraint Processing, San* Francisco: Morgan Kaufmann Publishers, 2003.

3 Stuart Russell ve Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2. baskı, Upper Saddle River, NJ: Prentice Hall, 2003.



Resim 27.3. Dört Vezir probleminin çözümü.



Resim 27.4. Alam Mackworth
(Fotoğraf, Alan Mackworth'un izniyle.)

alanı ele alır.[4] ILOG (IBM devralmıştır artık) gibi ticari oluşumlar, çizelgeleme ve benzetim içeren uygulamalar için yaygınlıkla kısıtlamalı programlama dillerini kullanır.

Kısıt yayılmasını açıklamak amacıyla faydalandığım Dört Vezir örneği, arama yapmaksızın çözüme ulaştı (çünkü, $c_1 = 2$ değerini seçerek başladım). Fakat bunun yerine en başta $c_1 = 1$ değerini seçmiş olsaydım, kısıt yayılması, tüm düğüm noktalarındaki tüm değerleri elerdi; yani, birinci sütunun birinci satırında duran bir vezir varsa, Dört Vezir probleminin çözümü yok demektir. (İsteyen bunu deneyebilir.) Bu seçimi yapıp bir çözüm olmadığını görmek, daha yüksek düzeyli bir arama işlemini gerektirir, böylece geri dönüp başka bir değer denemek mümkün olur. Aynı zamanda, deneme seçiminden sonra kısıtlama yayılmasıyla ilerleyiş yüzünden bazı nesne değerlerinin ne olacağının kararlaştırılmaması da olasıdır. Bu durumda, bu nesneler-

4 Vipin Kumar, "Algorithms for Constraint-Satisfaction Problems: A Survey", *AI Magazine*, Bahar 1992, s. 32-44. Şu internet adresinde: http://www.cs.cinvestav.mx/~constraint/papers/kumar-survey.pdf.

den birine değer seçmenin ardından daha fazla kısıt yayılması, olası geri dönüş vesaire gelir. Dolayısıyla, kısıtlara uyma problemlerini çözmek, tipik olarak arama işlemlerini ve birtakım gerileme yöntemlerinin önerilip kullanılmasını gerektirir.

27.2 ÖNERMELİ MANTIK KULLANARAK PROBLEMLERİN ÇÖZÜLMESİ

Mantıksal bilgi gösteriminin ve uslamlamasının önemli ve özel bir örneği, hiçbir mantıksal formülün değişken içermediği durumdur. Her ne kadar bu durumun, `(∀x)[İnsan(x) ⊃ Ölümlü(x)]` gibi formülleri olmasa da, `[İnsan(Sokrat) ⊃ Ölümlü(Sokrat)]` ve `[İnsan(Platon) ⊃ Ölümlü(Platon)]` gibi formülleri olabilir. Burada hiç değişken olmadığı için, bu özel örnek gerçekte önermeli mantıkla aynıdır. Bunun nedeni, `İnsan(Sokrat)` ve `Ölümlü(Sokrat)` gibi ifadelerin, ne zaman bilgi tabanında boy gösterseler, `P014` ve `Q234` gibi hiçbir içsel yapısı olmayan, dolayısıyla bütünüyle ilgisiz önermelerle yer değiştirebilmeleridir. Kendimizi önermeli mantıkla sınırlamanın zararı, tek formül ve her şeyin yerine geçecek değişkenler kullanmak yerine, bahsetmek istediğimiz *tüm* varlıkları kapsayacak muhtemelen çok sayıda formülümüzün olması gerekliliğidir. Fakat bu durumu telafi eden yararı, çok sayıda önermeli formülle uslamlama yapan son derece yetkin yöntemlerin geliştirilmiş olmasıdır.

Basit bir mantık bulmacası üzerinden, bu yöntemlerin nasıl işlediğini göstereceğim. Diyelim ki bir akşam yemeği davetine çağrılanlar arasında görece sorunlu üç kişi var: Ann, Bill ve Charlie. Bu insanların arasındaki toplumsal dinamiklerden haberdar olan bir dost, evsahibine, konuklardan en az birinin kesinlikle davete katılacağını ama Ann gelirse Bill'in gelmeyeceğini, Bill gelirse Charlie'nin gelmeyeceğini ve Charlie gelirse Ann'in gelmeyeceğini anlatıyor. Ev sahibi, bu bilgiyi temel alarak, davete kimin katılacağını bulabilir mi?

Ev sahibi mantıkçı olsaydı, dostunun verdiği bilgiyi şu aşağıdaki önermeli mantık formüllerine dökerdi (A, "Ann geliyor" anlamında, vb):

$$A \vee B \vee C,$$
$$\neg A \vee \neg B,$$
$$\neg B \vee \neg C,$$
$$\neg C \vee \neg A,$$

Daha önce kullandığım mantık formüllerinden, "¬"nin "değil", "∨"nin "ya da" anlamına geldiğini anımsarsınız. Bunlar gibi önermeler-

den (veya olumsuzlamalardan) meydana gelip "ya da" işaretleriyle birbirine bağlanan formüllere "koşul" denir. Tek tek önermelereyse, "değişken" denir, çünkü bunların doğruluk değerleri henüz atanmış değildir.

Ev sahibimiz, bu problemi çözmek istiyorsa, A, B, C önermelerine doğruluk değerlerini (D ya da Y), tüm koşullar D değerini alacak şekilde atamanın bir yolunu bulmalı (çünkü bu önermeler, doğru olduğu varsayılan ifadelerden geliyor). Eğer bir koşul, D değerini almışsa, mantıkçılar o koşulun "yerine getirildiğini" söyler. Örneğin, eğer A, Ann geliyor anlamında D değeri almışsa, ilk koşul yerine getirilmiş demektir (B'nin ya da C'nin hangi değerleri aldığına bakılmaksızın).

Mantıkçılar ve bilgisayar bilimcileri, bir koşullar kümesindeki değişkenlere, tüm koşulların yerine getirileceği şekilde değer atamanın ve bu değerlerin ne olabileceği problemiyle başa çıkmanın yöntemlerini bulmuştur. Bunun zorluğu, koşulun sağlanabilirliğini belirleme probleminin ("SAT problemi" denir) NP-tam olmasıdır, yani en kötü durumda, SAT problemlerini çözmeye yönelik bilinen tüm algoritmaların harcadığı zaman, problemin boyutuyla birlikte üstel olarak artar.

Ev sahibimizin karşılaştığı problem elbette büyük değil; A, B, C koşullarına doğruluk değeri atamanın sekiz farklı yolunu deneyip bu sekizinden hangisinin (eğer içlerinde uyan varsa) tüm koşulları yerine getirdiğini bularak problemi çözmede zorluk yaşamaz. Fakat koşul kümesi olarak kodlanmış pek çok bilişim problemi, binlerce değişken içeren yüz binlerce koşul içerebilir. Bu gibi problemlerde deneme yanılma yöntemini uygulamak epey zor. İyi ki, çok büyük problemleri çözebilen daha etkin yöntemler geliştirilmiştir. Bu etkin yöntemlerden birinin mucitlerinden olan Bart Selman, "... şu anki problem çözücüler, en az bir milyon değişkenli ve birkaç milyon koşullu örnekleri çözebilir", der. Dahası, şunu da iddia ediyor: "Bu, sadece daha hızlı bir donanımın sonucu değil... gerçekten de %95 daha iyi algoritmaların sonucu. Ancak, NP-tam problemiyle ve üstel arama uzamıyla hâlâ boğuşuyoruz. Dolayısıyla, donanımda, algoritma fikirlerinden yoksun iyileştirmelerin fazla etkisi olmaz."[5]

SAT problemlerini çözmenin başlıca iki tür yöntemi var. Sınıflardan biri sistemli yöntemlerden ve diğer sınıf ile yerel arama yöntemlerinden meydana geliyor. Aslında, en iyi problem çözücülerden bazıları, bu iki yöntemin de tekniklerini kullanır. Buradaki temel fikirleri önümüzdeki kısımda anlatıyorum.

5 E-postayla iletişim, 8 Nisan 2008.

27.2.1 Sistemli Yöntemler

Sistemli yöntemlerin çoğu, DPLL algoritması denilen yordamı ve bunun geliştirilmiş çeşitli sürümlerini temel alır.[6] (DPLL algoritması, Martin Davis ile Hilary Putnam'ın önerdikleri daha eski bir algoritma olan DP algoritmasından türetilmiştir.[7]) DPLL algoritması, değişkenlere doğruluk değeri atamanın olası yollarına dair ağacı arayarak iş görür. Arama ağacının her düğüm noktasında, bir daldaki değişkene D (doğru) değeri atanırken, başka daldaki değişkene Y (yanlış) değeri atanır. Bu atamalar, düğüm noktasındaki koşullar kümesini, sonraki iki düğüm noktasında, aşağıdaki sadeleştirme süreci sayesinde yeni kümelere dönüştürür:

1. Her koşulda, henüz atanmış değişkeni, dalına göre ya D ya da Y değeriyle değiştir.
2. D ya da ¬Y içeren koşulları ele. (Bu koşullar, o atamayla birlikte karşılanmış oluyor.)
3. Her ¬D ya da Y'yi, boy gösterdikleri her koşuldan ele. (Koşullar kümesinin yerine getirilebilmesi için, bu koşullarda arta kalmış en az bir değişkenin D değeri almış olması şarttır.)
4. Sadece tek değişken içeren her koşul için, o değişkene, koşulu yerine getiren değeri ver, ve mümkünse sadeleştirme işlemini sürdür.

Şu aşağıdaki şartlardan biri ya da daha fazlası gerçekleştiğinde DPLL durur:

i. Eğer ulaşılan koşul kümesi boşsa, DPLL sona erer ve başlangıçtaki koşul kümesinin yerine getirilebilir olduğuna, şimdiye dek atanmış değerlerin bu koşulları yerine getirdiğine karar verir.
ii. Eğer ağacın bir dalında ulaşılan herhangi bir koşul boşsa (yani, bu koşulu yerine getiriyor mu diye denenecek hiç değişken kalmamışsa), o zaman DPLL, başlangıçtaki koşul kümesinin, şimdiye dek dal üzerinde atanmış doğruluk değerlerin-

6 Bu algoritmaya Martin Davis, Hilary Putnam, George Logemann ve Donald W. Loveland'ın adları verilmiştir. Bkz. Martin Davis, George Logemann ve Donald Loveland, "A Machine Program for Theorem-Proving", *Communications of ACM*, c. 5, no: 7, 1962, s. 394-397.

7 Martin Davis ve Hilary Putnam. "A Computing Procedure for Quantification Theory", *Journal of the ACM*, c. 7, no: 3, 1 960, s. 201-215 .

ce karşılanamayacağını belirlemiş olur. Bu durumda arama işlemi, eğer hâlâ atanmamış doğruluk değerleri barındıran değişkenler varsa, ağacın başka bir dalında sürer. Eğer böyle değişkenler yoksa, DPLL, başlangıçtaki koşul kümesinin yerine getirilemeyeceğini belirleyerek sonra erer.

Örnek olarak, "akşam yemeğine kim geliyor" problemime ilişkin ağaca bakalım. Resim 27.5'te, doğruluk değerlerinin atanmasıyla (A, B, C sırasıyla) ve sadeleştirmeyle oluşan arama ağacının bir kısmını görüyoruz.

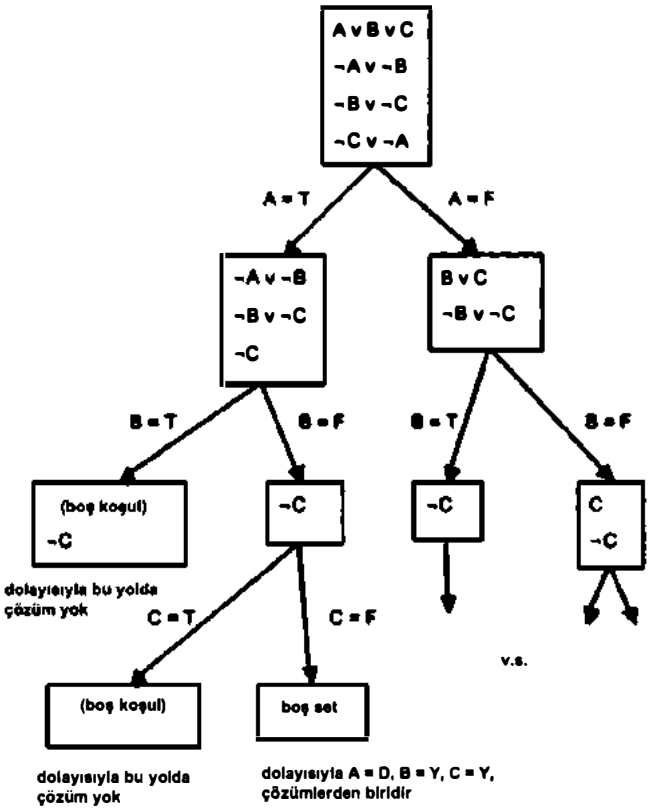
Bu örnekte altı çizilmesi gereken ilginç noktalardan biri, arama sırasına bağlı olarak, DPLL'nin, arama ağacındaki tüm dallar irdelenmeden sona erme olasılığıdır (çoğunlukla da sona erer). Derinliğe öncelik veren (genişliğe öncelik veren yerine) aramayla, hızlı sona erme olasılığı artar. DPLL, yüksek verimine ve hızına, bilgisayar bilimcilerinin "özçağrılı ve geri izlemeli arama" dedikleri yöntemi kullanarak erişir. DPLL'nin daha da geliştirilmesi sonucunda, SAT problemlerini çözen daha hızlı ve güçlü küresel yöntemler ortaya çıktı. Bu iyileştirmeler, "koşul öğrenme" mekanizması denen olanağın kullanımı sayesinde geri izlemeyi daha "zeki" hale getirmeyi ve yerel arama yöntemlerinin faydalandığı bazı stratejilerden yararlanmayı içerir.[8] Bu programlardan biri olan zChaff'ın internet sitesinde şu iddia ediliyor: "Bir milyonu aşkın değişkeni ve on milyon koşulu olan problemleri zChaff kullanarak çözdüğümüz başarı hikâyelerimiz var. (Böyle her problemi çözecek değil elbette!)"[9]

27.2.2 Yerel Arama Yöntemleri

Yerel arama yöntemleri, tepe tırmanma aramasını uygular: Bu arama, tüm koşullar için, başlangıçta rasgele seçilmiş doğruluk değerleri kümesinde, her seferinde tek tek olmak üzere değişiklikler yaparak ilerler. SAT problemleri için, olası her doğruluk değer kümesi, arazi

8 Örneğin bkz. Matthew Moskewicz ve diğ., "Chaff: Engineering an Efficient SAT Solver", *Proceedings of the 38th Design Automation Conference* (DAC'01), 2001 (şu internet adresinde: http://www.princeton.edu/~chaff/publication/DAC2001v56.pdf) ve Niklas Eén ve Niklas Dörensson, "An Extensible SAT-Solver", *Theory and Applications of Satisfiability Testing*, Bilgisayar Bilimi Seminer Notları, Berlin and Heidelberg: Springer-Verlag, 2004 (şu internet adresinde: http://een.se/niklas/Satzoo/An_Extensible_SAT-solver.ps.gz).

9 Bkz. http://www.princeton.edu/~chaff/zchaff.html. Etkili başka bir SAT çözücüsünün internet adresi: http://minisat.se/Main.html.

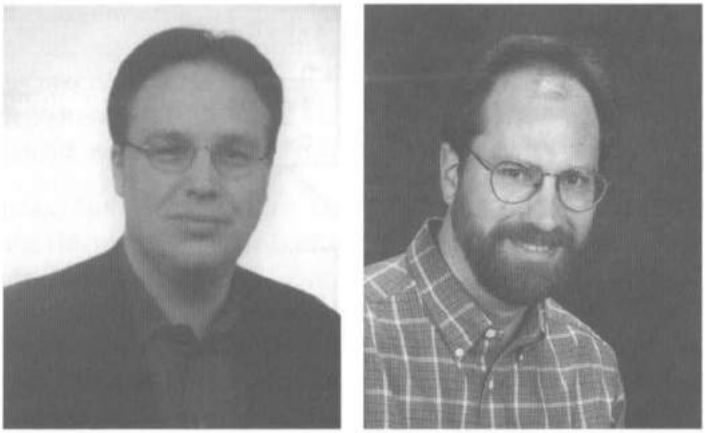


Resim 27.5. DPLL arama ağacı.

üzerinde belli bir yere karşılık gelir ve karşılanan koşulların sayısı (o doğruluk değeri kümesi için), buna denk gelen yerin yüksekliğine ya da rakımına tekabül eder. Arazideki en yüksek mevki (sayısı birden fazla olabilir), karşılanabilecek koşulların azami sayısına karşılık gelir (koşullar kümesi karşılanabilir nitelikliyse, hepsi bu durumdadır).

1992'de Bart Selman (1959- ; Resim 27.6), Hector Levesque ve David Mitchell (1957- ; Resim 27.6), SAT problemleriyle boğuşan, GSAT adlı bir yöntem tanıttılar.[10] (G, "greedy" [açgözlü] anlamına geliyor ve bunun sebebini az sonra göreceğiz.) GSAT ve bunun WALKSAT gibi

10 Bart Selman, Hector Levesque ve David Mitchell, "A New Method for Solving Hard Satisfiability Problems", *Proceedings of the Tenth National Conference on Artificial Intelligence*, Menlo Park, CA: AAAI Press, 1992, s. 440-446. Şu internet adresinde: at http://www.cs.sfu.ca/~mitchell/papers/ai92-gsat.ps.



Resim 27.6. Bart Selman (solda) ve David G. Mitchell (sağda) (Fotoğraflar Bart Selman'ın ve David Mitchell'in izniyle.)

çeşitli uzantıları, sayısı 200.000 değişkene kadar çıkan problemlere başarıyla uygulanmıştır. GSAT, doğruluk değerlerinden oluşan arazide yerel tepe tırmanma araması yürütür.

Bu arama, ana hatlarıyla şu şekilde çalışıyor. En başta rasgele doğruluk değerleri atar ve bu atama sonucunda kaç koşulun yerine getirildiğini değerlendirmeye alır. Eğer hepsini yerine getirmişse, süreç bir çözümle sona erer. Aksi takdirde, her seferde tek bir önermenin doğruluk değerini tersine çevirir. Sonra, karşılanan koşul sayısında en yüksek ("en açgözlü") artışa yol açan doğruluk değerini seçer ve yerel arama, yeni doğruluk kümesiyle sürer (tersyüz edilmiş değerle birlikte).

Çoğunlukla, tek tersyüz ediş, karşılanan koşulların sayısını artıramaz. Yine de, en azından bu rakamı sürdüren tersyüz edişler genelde vardır. Bu durumda, GSAT bunlardan birini (rasgele) seçer ve ulaşılan "yaylada" buna denk gelen adımı alıp, tepe tırmanışına sonradan dönebilmeyi umar. Yoksa, arazideki tüm olası adımların yokuş aşağı olma ihtimali de bulunur. (Bu yerel teknikleri betimleyen bir makale,[11]

11 Bart Selman, Henry Kautz ve Bram Cohen, "Local Search Strategies for Satisfiability Testing", *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, October 11-13, 1993 (DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, c. 26 içinde, David S. Johnson ve Michael A. Trick (ed.), Providence, RI: AMS, 1996.) Bu makalenin bir versiyonu şu internet adresinde: http://www.cs.rochester.edu/u/kautz/papers/dimacs93.ps. Ayrıca, yerel arama yöntemleri hakkındaki bir internet sayfasına bakınız: http://www.cs.rochester.edu/u/kautz/walksat/.

bu tür sonuçların "neredeyse asla görülmediğini" söylüyor.) GSAT nadiren, çıkabileceği kadar yükseğe çıkıp "yerel en yüksek noktaya" ulaşır. Bazı uygulamalarda, tüm koşulları karşılamayan doğruluk değerlerinin atanması faydalı ve makul olabilir; fakat öyle değilse, GSAT yeni bir rasgele doğruluk atamaları kümesiyle "yeniden başlatılır" ve bu yeni denemede daha büyük bir yerel en yüksek noktaya ulaşılması umulur. Her durumda, GSAT, denediği tersyüz edişlerin sayısına sınır koyar ki yaylada sonsuza dek dolanmasın. SAT problemi genelde NP-tam bir problem olduğu için, yerel yöntemlerin (ya da herhangi bir yöntemin) üstel oranda zaman harcayacağı problemlere rast gelmek olasıdır; fakat GSAT'ın yaratıcıları, bu tür problemlerin "son derece nadiren görüldüğünü ve incelediğimiz uygulamalarda doğal olarak görülmediklerini" ileri sürer.[12]

"Akşam yemeğine kim geliyor" problemimiz üzerinde GSAT aşağıda tarif edildiği gibi çalışır, ama önce koşulları tekrarlayalım:

$$A \vee B \vee C,$$
$$\neg A \vee \neg B,$$
$$\neg B \vee \neg C,$$
$$\neg C \vee \neg A,$$

GSAT önce rasgele doğruluk değerleri seçer, örneğin A için D, B için D ve C için D gibi. Bu doğruluk değerleri kümesi, sadece tek koşulu yerine getiriyor, yani $A \vee B \vee C$ koşulunu. Eğer GSAT, doğruluk değerlerinden herhangi birini tersyüz ederse (D'yi Y yaparsa), üç koşul karşılanmış olur; bunların hepsi, "yokuş yukarı" büyük birer adımdır. Diyelim ki GSAT, A'nın değerini tersyüz etmek istiyor, böylece birinci, ikinci ve sonuncu koşul karşılanmış olur. Bu durumda, B'nin ya da C'nin değerini tersine çevirmek, dört koşulun da karşılanmasını getirir; ikisi de çözüme doğru atılmış yokuş yukarı adımdır. Diyelim ki B'nin değerini tersyüz etmeye karar verdi. Bu durumda GSAT tek çözüm bulmuş demektir, yani A için Y, B için Y ve C için D. (Mantık eğilimi olan okurlar, aslında üç çözüm olduğunu fark edecektir: üç davetliden yalnızca birinin yemeğe katılması durumu. Elbette ev sahibi bu üçünden hangisini çağıracağına karar veremez, ama en azından masaya kaç tabak koyacağını bilir.)

GSAT'ın, bu küçük probleme çözüm bulması hiç şaşırtıcı değil. Aslında, rasgele yaratılmış büyük problemlerde, değişkenlerin (A'lar, B'ler, C'ler) sayısı koşulların sayısından çok daha azken, tatmin edici doğruluk atamalarının sayısı olasılıkla çoktur ve GSAT (ve başka yön-

12 Bart Selman, Hector Levesque ve David Mitchell, *a.g.e.*

temler de) muhtemelen çözüm bulur. Bununla birlikte, değişkenlerin sayısı koşulların sayısından çok daha fazla olduğunda, hiç çözümün var olmaması olasıdır.[13]

WALKSAT (bazen WSAT da denir), GSAT'ın önemli bir uzantısıdır; bu yöntemde, karşılanan koşulların sayısında en büyük artışa yol açana kadar önermelerin doğruluk değerlerini tersyüz etmek yerine, bazen rasgele tercihler yapılır. Bu küçük gelişigüzellik eki, arazinin yerel azami noktalarına sıkışıp kalmaktan uzak durmaya yardımcı olur.[14]

Küresel yöntemlerle yerel yöntemleri kıyaslayan Bart Selman şunu ileri sürüyor: "Yerel arama yöntemleri pek çok alanda hâlâ yetkin, fakat... DPLL yöntemleri, problem kodlanışlarına karşı genelde duyarsız olduğu için, bugünlerde çoğunlukla yapılandırılmış problemlerin çözümünde kullanılıyorlar [örneğin, donanım ve yazılım doğrulama gibi.] Bununla birlikte, Selman şunu da der: "DPLL'de rasgeleleştirmenin ve baştan başlamanın kullanımı... DPLL'ye, yerel aramanın sistemli olmayan kimi yönlerini [ekliyor]."[15]

27.2.3 SAT Çözücülerin Uygulamaları

Birtakım önemli problemler, SAT problemi olarak kodlanabilir. Örneğin, Henry Kautz ve Bart Selman, eylem planı yaratmanın, SAT problemi olarak ifade edilebileceğini göstermiştir.[16] SATPLAN[17] ve Blackbox,[18]

13 Bart Selman, Hector Levesque ve David Mitchell, *a.g.e.*

14 Bkz. Bart Selman, Henry Kautz ve Bram Cohen, *a.g.e.*

15 Bart Selman, 8 Nisan 2008 tarihli e-posta.

16 Henry Kautz ve Bart Selman, "Planning as Satisfiability", *Proceedings of the 10th European Conference on Artificial Intelligence*, New York: John Wiley and Sons, Inc., 1992, s. 359-363 (şu internet adresinde: http://www.cs.rochester.edu/u/papers/satplan.ps) ve Henry Kautz ve Bart Selman, "Pushing the Envelope: Planning, Propositional Logic, and Stochastic Search", *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, Menlo Park, CA: AAAI Press, 1996, s. 1194-1201 (şu internet adresinde: https://eprints.kfupm.edu.sa/58089/1/58089.pdf).

17 Henry Kautz, Bart Selman ve Joerg Hoffmann, "SATPLAN: Planning as Satisfiability", *Abstracts of the 5th International Planning Competition*, 2006 (şu internet adresinde: http://www.cs.rochester.edu/u/kautz/papers/kautz-satplan06.pdf). SATPLAN programlarıyla ilgili daha fazla bilgi ve internetten indirme olanakları için bkz. http://www.cs.rochester.edu/u/kautz/walksat/.

18 Henry A. Kautz ve Bart Selman, "Unifying SAT-Based and Graph-Based Planning", *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, San Francisco: Morgan Kaufmann Publishers, 1999, s. 318-325. Şu internet adresinde: http://www.cs.rochester.edu/u/kautz/satplan/blackbox/ijcai99blackbox.ps. Blackbox programları hakkında bilgi ve internetten indirme imkânı için bkz. http://www.cs.rochester.edu/u/kautz/satplan/blackbox/.

planlama görevlerini SAT problemi olarak kodlayan iki sistemdir ve plan yaratırken SAT çözücülerinden faydalanırlar. SATPLAN, eylemlerin etkilerini betimleyen özel olarak tasarlanmış mantık formülleriyle başlar işe, Blackbox ise STRIPS planlama kurallarıyla yola çıkar. (12.1.3'te anlatmış olduğum STRIPS otomatik planlama sistemini anımsarsınız.) Bart Selman'a göre, örneğin lojistik planlama problemleri üzerinde çalışan SAT çözücüleri, birkaç dakika içinde 50 adımlık ideal planlar yaratabilir.[19] SAT-tabanlı planlama sistemlerinin yeni sürümleri, iki yılda bir düzenlenen Uluslararası Planlama Yarışmalarında birincilik ödülleri kazanmıştır.[20]

Etkin SAT çözücüleri ayrıca programların ve sayısal devrelerin[21] doğrulanması ve genomikle ilgili problemlere de uygulanmıştır. Bununla yakından ilintili bir konu, mantık devreleri tasarımlarının doğrulanmasında kullanılan "İkili Karar Diyagramları"nı (BDD) içerir.[22]

27.3 METİNLERİN VEKTÖR OLARAK GÖSTERİLMESİ

Önceki bölümlerde, soruların bilgisayarlarca işlenebileceği bir biçime (belki bir mantık formülüne) dönüştürülüp sonra bilgisayar veritabanını (belki mantık formüllerinden oluşan bilgi tabanını) sorgularken kullanıldığı soru-yanıtlama sistemlerini betimlemiştim. Muhtemelen günümüzün en iyi bilinen soru yanıtlama sistemleri, Dünya Çapında Ağ'ın (www) arama motorlarında yer alıyor. Mantıkçı zihniyeti taşıyan YZ araştırmacısı, Ağ'daki metinlerin en sonunda mantık formülleri olarak temsil edileceğini ve o internet sayfalarının birindeki (ya da daha fazlasındaki) formüllerden yola çıkılarak yanıtlanacak (kanıtlanacak) sorgunun da mantık formülü olarak temsil edileceğini umabilir. İngilizce sorguları bu biçimde yanıtlamaya yönelik kimi yeni başlayan girişimler yok değil,[23] fakat çoğu Ağ arama motoru, daha basit ve verimli teknikler kullanıyor. Bunlardan bazılarının nasıl çalıştığını

19 Bart Selman, 8 Nisan 2008 tarihli e-posta.

20 Uluslararası Planlama Müsabakası için bkz. http://zeus.ing.unibs.it/ipc-5/.

21 Örneğin bkz. Armin Biere ve diğ., "Bounded Model Checking", *Advances in Computers*, c. 58, San Diego: Academic Press, 2003. Bu makalenin eşyazarlarından ve Carnegie Mellon Üniversitesi Bilgisayar Bilimi profesörlerinden Edmund M. Clarke, bu alandaki çalışmalarından ötürü 2007 ACM Turing Ödülünün eşkazananlarından oldu.

22 Bkz. Randy E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation", *IEEE Transactions on Computers*, c. C-35, no: 8, Ağustos 1986, s. 677-691. Şu internet adresinde: http://www.cs.cmu.edu/~bryant/pubdir/ieeetc86.pdf.

23 Örneğin bkz. http://www.powerset.com/.

kabaca anlatacağım. Bunlar, hem belgedeki hem de sorgudaki metni *vektörlere* dönüştürüp, sorgu vektörünü, rakip belgelerin vektörleriyle karşılaştırıyor. Önce, vektörler hakkında bir şeyler söyleyip, sonra da metinlerin vektör olarak nasıl temsil edilebileceğini betimleyeceğim. (Örüntü tanımada vektör kullanımından bahsettiğimi hatırlarsınız.)

Matematikte vektör, büyüklüğü ve doğrultusu olan bir niceliktir. Örneğin, üç boyutlu uzayda vektör, o uzayda doğup belli bir noktaya ulaşan ok olarak çizilir. Ok, vektörün doğrultusunu gösterir, okun uzunluğu, vektörün büyüklüğüne işaret eder. O nokta, vektörü belirlediği için (çıkış noktasından itibaren oku çizmenin sadece tek yolunun olmasından dolayı), "nokta" ve "vektör" sözcükleri çoğunlukla eş anlamlı olarak kullanılır. Sıralı bir rakamlar listesi, bir noktanın koordinatları, dolayısıyla bir vektörün bileşeni sayılabilir. Örneğin, (7, 4, 3, 20) listesi, dört boyutlu uzayda bir vektördür. Vektörlerin pek çok boyutu olabilir; belgeleri temsil etmek amacıyla kullanılan vektörlerin boyut sayısı binlere çıkabilir. Vektörün uzunluğu, tüm vektör bileşenlerinin karelerinin toplamının kare köküne eşittir. (İki boyutlu vektörler söz konusu olduğunda, bu hesaplama, Pisagor teoreminin bir uygulamasıdır sadece: Dik üçgenin hipotenüs uzunluğunun karesi, öbür kenarların karesinin toplamına eşittir.) Örneğin, (7, 4, 3, 20) vektörünün uzunluğu 21,77'dir.

İki vektör arasındaki benzerlik, ya uç noktalarının arasındaki uzaklığın (belki de uzunluklarını hesaba katmak üzere ayarlanarak) hesaplanmasıyla ya da doğrultuları arasındaki açının "küçüklüğüyle" ölçülür; o açı ne ölçüde küçükse, vektörler o ölçüde benzeşir. Açı yönteminde, şu benzerlik hesabı yapılır: Vektörlerden birinin tüm bileşenlerini, öteki vektörün karşılık gelen bileşenleriyle çarp, sonra da tüm bu çarpımları topla. Ardından bu toplamı, vektörlerin uzunluklarının çarpımına böl. Benzerliği gösterdiği için *B* dediğimiz bu son rakam, en yüksek değeri olan 1'e, iki vektör tamamen aynı hizada olduğunda ulaşır (yani aynı yönü gösterdiklerinde). İki vektör birbirine dikse 0 değerini alır ve vektörler zıt yönleri gösteriyorsa eksi değer alır. Dolayısıyla, *B* hesaplaması 1'e ne ölçüde yakınsa, o vektörler o ölçüde benzeşir. (Trigonometriye aşina okurlar, bu hesaplamanın, iki vektör arasındaki açının kosinüsü olduğunu fark edecektir.)

Örnek vermek gerekirse, (7, 4, 3, 20) ve (7, 0, 2, 15) vektörlerinin *B* değeri, (49 + 6 + 300) / (21,77 × 16,67) = 0,978 olarak hesaplanır; bu değer, bu iki vektörün oldukça benzeştiğini gösteriyor.

Peki ama, metni vektöre nasıl çevirebiliriz? Belgelerin bilgisayarda erişimiyle (bilgi erişimiyle) ilgilenen insanlar, bunun için bir yöntem

bulmuştur.[24] Öncelikle, terimlerin (sözcüklerin ya da cümleciklerin) sıralı bir listesi, vektörlerin temsil edeceği belge kümesi olarak seçilir. Eğer bu belgeler yapay zekâ hakkındaysa, "arama", "sezgisel yöntem", "bilgisayarla görme", vesaire dahil uygun yüzlerce terim seçilebilir. Eğer belgeler İngilizceyse ve herhangi bir konu hakkındaysa, yüzbinlerce terim söz konusu olabilir (esasen, İngilizcenin tüm sözcükleri). Çoğunlukla, seçilen terimler sözcük kökleridir, böylece "hesaplama", "hesaplayıcı", ve "hesaplanmış" sözcükleri "hesap" terimiyle kapsanabilir. ("Köktengelim" denilen bu birleştirme yaklaşımı sırasında dikkatli olunmalı, çünkü "flow" {akıntı} ve "flower" {çiçek} gibi kimi sözcükler birbirinin yerine geçebilir.) Ayrıca, "ve", "eğer", "dolayısıyla" gibi sözcükler, belgenin içeriğiyle nadiren ilişkili olduğu için, bu sözcükler terim olarak kullanılmaz.

Belgeyi vektör olarak temsil etme sürecinin bir sonraki adımında, belgedeki terimlerin tekrarı sayılır. Bu tekrar sayılarının listesi oluşturulur (terimler listesiyle aynı sırada); bu liste, belgenin vektör gösterimidir. Dolayısıyla, örneğin "arama" adlı terim, temsil edilecek belgede hiç boy göstermiyorsa, "sezgisel" terimi yedi kez ve "bilgisayarla görü" üç kez boy gösteriyorsa, o zaman liste, örneğin şöyle olur:

(0, 0, 0, 0, 0, 7, 0, 0, 3, 0, 0,...)

Bu listede altı çizili rakamlar, olduğunu söylediğim terimlerin belgede boy gösterme sayısıdır. Elbette, belgede çok ama çok 0 olabilir, çünkü seçilmiş terimler listesindeki pek çok terim belgede hiç görülmez ve sıfır dışında pek çok rakam, o belgede boy gösteren başka terimlerin sayısına karşılık gelir.

Şimdi, diyelim ki "Bilgisayarla görmede hangi sezgisel yöntemler kullanılır" sorusuyla ilgileniyoruz ve bu sorguyu bir internet arama motoruna soruyoruz. Sorguda (ve belgelerde), sözcükleri "köklerine" dönüştüren bir tür önişlem kullanıldığını varsayarsak sorgumuzun gösterim vektörü şöyle olur:

(0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0,...)

Sorgumuz ile belge arasında olduğunu söylediğimiz benzerlik B, iki vektörün uzunluğunun çarpımına bölünür. Bu değer, başka belge-

24 Bu yöntemin ilk kez sunulduğu makale: Gerard Salton, A. Wong ve C S. Yang, "A Vector Space Model for Automatic Indexing", *Communications of the ACM*, c. 18, no: 11, Kasım 1975, s. 613-620. Bu konudaki bazı tartışmalar için bkz. http://www.ideals.uiuc.edu/bitstream/2142/1697/2/Dubin748764.pdf.

lerin benzerlik değerleriyle kıyaslanır ki hangi belgelerin en benzer olduğu, dolayısıyla sorgumuza yanıt olarak öne sürüleceği belirlensin.

Tüm bunlar kulağa basitmiş gibi gelebilir, fakat, her ne kadar temel fikir basit olsa da, bu fikri temel alan belge erişimini ve internet siteleri bulma işlemini elverişli ve yararlı kılmak için bazı iyileştirmelerin yapılması gereklidir (bunlar yapılmıştır da). Öncelikle, belgede geçen terimin sayımı çoğunlukla, o belgedeki metnin uzunluğunu hesaba katacak biçimde ayarlanır. Daha uzun belgelerin, aranan terimi içerme olasılığı görece yüksek olduğu için, bir terim sayımı, o belgedeki tüm terimlerin toplam sayısının yüzdesi olarak hesaplanır. İkincisi, aranan terim, araştırılan tüm belgelerde epey yaygın bulunabileceği için (dolayısıyla kullanışlı bir ayrım söz konusu olmaz bu durumda), bu sayım, o belgelerde terimin genel görülme sıklığına göre değişen bir çarpana göre azaltılır. Daha gelişmiş terim bulma programları, belgenin sorguyla ilintili olma olasılığını hesaplayan çeşitli istatistiksel yöntemleri kullanır. Google'ın bir icadı, internet sitelerini, gördükleri rağbetle ya da "merkezîlikleriyle" ilgili tahminlere göre sıralar. "Yapay öğrenme" yöntemleri (bunlardan bazılarını ileriki bölümde anlatıyorum), erişim sistemlerinin performansını iyileştirme amacıyla gitgide daha fazla kullanılır oldu; elbette verimlilik, uygun dizinleme yöntemlerini ve binlerce bilgisayarın kullanımını gerektiriyor.

27.4 GİZLİ ANLAMSAL ÇÖZÜMLEME

Kimi araştırmacılar, metni vektör olarak göstermenin, metnin "anlamını" yakalayacağını söylemişti. Peki, vektör gösterimleri, belgelerde çeşitli terimlerin ne sıklıkta göründüğüyle hesaplanıyorsa ve görülme sıraları hesaba katılmıyorsa, bu iş nasıl mümkün olabilir? (Ne de olsa "Köpek adamı ısırdı" cümlesinin anlamı, "Adam köpeği ısırdı" cümlesinin anlamından oldukça farklıdır.) Thomas K. Landauer (1932- ; Resim 27.7) ve çalışma arkadaşları, önce 1980'lerin ortasında Bell Communications Research'teki (Bell Laboratuvarlarının mirasçısı) Bilişsel Bilim Araştırma Ekibinde, sonra Colorado Üniversitesi'nde, Gizli Anlamsal Çözümleme (GAÇ) dedikleri, anlam yakalayan vektör tabanlı bir yöntem önerdiler. Tam bir betimlemenin gerektireceği tüm o matematik formüllerini kullanmaksızın temel fikri açıklayabilirim sanırım.

GAÇ yönteminin temelde nasıl işlediğini gösteren dar kapsamlı bir örnek aşağıdaki gibidir. Diyelim ki elimizde belirli bir konu hakkında görece uzun bir belge ya da bir tür metin var. Metni, "pasaj"

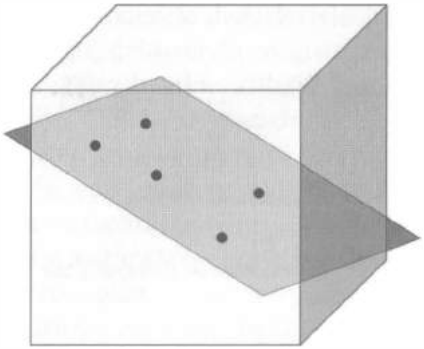


Resim 27.7. Thomas K. Landauer (Fotoğraf, Libby Landauer'in izniyle.)

dediğimiz, yaklaşık 100'er terimlik kısımlara ayırıyoruz. Diyelim ki metnin sözcük dağarcığı 1.000 terimle (bunlar, tekil sözcükler ya da sözcük birleşimleri olabilir) kapsanıyor, ardından bu pasajların her biri, 1.000 boyutlu bir vektörle temsil ediliyor. (Bu vektörlerin inşasında kullanılan terim sayımları, zaten açıklamış olduklarıma benzer yöntemlerle düzenlenir.) Diyelim ki bu tür 100 vektörümüz var.

Vektörlerimizin yer aldığı 1.000 boyutlu bir uzayı gözümüzde canlandırmak zor (aslında imkânsız), fakat belki daha az boyutlu bir "alt uzayın", vektörlerin tümünü ya da çoğunu içerebileceğini en azından kafamızda canlandırabiliriz. Resim 27.8'de gösterilene benzer bir üç boyutlu örneği gözden geçirmek faydalı olacaktır. Bu şekilde, üç boyutlu uzayda yer alan bir düzlemdeki (iki boyutlu) beş noktayı gösteriyorum. İki boyutlu uzay, üç boyutlu uzayın alt uzayıdır. O iki boyutlu uzayda bu beş nokta, üç boyutlu vektörler yerine iki boyutlu vektörlerle temsil edilebilir.

Çeşitli karmaşık matematik teknikleri sayesinde, 100 vektörü yeterince "içeren" daha az boyutlu bir uzay oluşturmak mümkün (diyelim ki, belki 50 boyutlu bir uzay). GAÇ, "Tekil Değer Ayrışması" (TDA) adlı tekniği temel alan yöntemlerden faydalanır, ama bunun ayrıntıları şu an bizi ilgilendirmiyor. Elbette, bu vektörlerin 50 boyutta temsili, 1.000 boyuttaki temsilinden farklı olacaktır. Daha büyük olan



Resim 27.8. Üç boyutlu uzayda iki boyutlu altuzay.

uzaydaki boyutlarla ilintili pek çok terim, daha küçük uzayda yeni bileşenler halinde birleşecektir. Dahası, Landauer ve çalışma arkadaşlarına göre, belgenin ayrı pasajlarından örtük toplu anlamı çıkarmayı mümkün kılan unsur, tam da bu birleşimdir. Bu sürecin bir örneğini açıklarken belirttikleri gibi: "... özgün hücrelerden birinin içeriğini değiştirecek olsaydık, azaltılmış boyut sayısıyla yeniden oluşturulurken, değerler her yerde değişebilir; GAÇ ile çıkarsama ya da tümevarımın matematiksel bağlamı budur."[25]

Vektörleri, daha az bileşenli vektörlere dönüştürmek, temelde, vektörlerin türetildiği asıl pasajlarda boy gösteren (ve göstermeyen) terimleri birbirine bağlar. Bu bağlantı, birbirleriyle ilintili terimleri temel alan üst düzey bir "kavram" yaratmak olarak düşünülebilir. Bir metin belgesini bu kavramların terimleriyle (yani boyutu azaltılmış vektörlerin terimleriyle) ifade etmek, GAÇ araştırmacılarına göre, belgenin esas "anlamını" çekip çıkarır. Boyutu azaltılmış vektörler, aynı pasajda bulunmasalar da benzer anlamlar taşıyan pasajlarda yer alıyorsa bir metnin farklı kısımlarından gelen terimleri birbirine bağlayabilir.

GAÇ işlemi, birbirine karşılık gelen, boyutu azaltılmış iki vektörün arasındaki açının büyüklüğünü hesaplayarak belgedeki herhangi iki pasajın benzerliğini hesaplamayı mümkün kılar. GAÇ yöntemi, boyutu azaltılmış vektörlerle pasajları temsil etme sürecinde ayrıca terim kümesinin bütününde her terimi, aynı şekilde boyutu azaltılmış vektörlerle temsil edebilir. Bu gösterim sayesinde, iki terimin benzerliği de, terim ile pasajın benzerliği de hesaplanabilir. Son olarak, bizzat belge, pasaj vektörlerinin ortalamasından meydana gelen bir vektör

25 Thomas K Landauer, Peter W. Foitz ve Darrell Laham, "Introduction to Latent Semantic Analysis", *Discourse Processes*, c. 25, 1998, s. 25-284. Şu internet adresinde: lsa.colorado.edu/papers/dpl.LSAintro.pdf.

olarak temsil edilebilir. Bu şekilde temsil edilen belgeler arasındaki benzerlik de hesaplanabilir. Bu adım, GAÇ'nın "Gizli Anlamsal Dizinleme" (GAD) denilen bir uygulamasında kullanılıyor. GAD yönteminin, standart terim bulma yöntemlerini bir parça iyileştirdiği bildirilmiştir (gerçi bu konu hâlâ tartışmalıdır).[26]

GAÇ, üniversiteye giriş sınavlarında yazılan kompozisyonları notlandırma, öğrencilerin yazı yazma becerisi edinmelerine yardımcı olma, hastaların sözlü ifadelerinden şizofreni tanısı koymaya yardımcı olma ve belgelerin anahtar sözcüklerden oluşan özetlerini çıkarma dahil çeşit çeşit ortamda kullanılmıştır.[27] Buna ek olarak, insanların kimi becerilerini taklit etmek amacıyla da kullanılmıştır, örneğin TOEFL'ın (ETS'nin, Yabancı Dil Olarak İngilizce Sınavı) eşanlamlı sözcükler kısmında sınava giren ortalama kişiler kadar puan almak ve psikolojie giriş ders kitabının GAÇ çözümlemesiyle elde edilmiş vektörlerinden faydalanarak çoktan seçmeli bir sınavda geçer not almak gibi.[28]

Okurlar, kompozisyona not veren GAÇ sisteminin, bol miktarda uygun sözcüğü tutarlı hiçbir düşünce ifade etmeksizin rasgele yazan birinin aldatabileceği itirazında bulunabilir. Landauer, şunu söylerek bu itirazı karşılar: "İyi bir kompozisyon yazmadan doğru sözcüklere erişmek zor olacaktır... Kötü kompozisyonlar yazıp iyi notlar almaya çalıştığımız oldu ve ancak konuyu gayet iyi bildiğimizde bu işi zaman zaman başarabildik. Bu sistemi kandırmanın en kolay yolu, sıkı çalışmak, konuya hâkim olmak ve sağlam kompozisyon yazmaktır."[29]

1998'de Landauer ve çalışma arkadaşları, GAÇ'ın eğitim uygulamalarını geliştirmek amacıyla Knowledge Analysis Technologies (KAT) şirketini kurdu. 2004'te KAT'ı Pearson Education devraldı; bu şirket, GAÇ-tabanlı eğitim ürünlerini Pearson Knowledge Technologies (PKT) olarak pazarlıyor.

Bazı araştırmacılar, GAÇ yöntemlerinin başlıca gücünün, vektör boyutunu azaltmada yattığına ve boyut sayısını azaltan başka yöntemler de olduğuna işaret ediyor (bunlardan bazıları, GAÇ'da kullanı-

26 Susan T. Dumais, "Latent Semantic Indexing (LSI) and TREC-2", *The Second Text Retrieval Conference (TREC2)* içinde, D. Harman (ed.), National Institute of Standards an Technology Special Publication 500-215, 1994, s. 105-116. Bir nüshası şu internet adresinde: trec.nist.gov/pubs/trec2/papers/txt/10.txt.

27 Thomas Landauer, e-postayla iletişim, 30 Ocak 2008.

28 Bu uygulamalar hakkında daha fazla bilgi için GAÇ internet sitesine ve burada atıfta bulunulan çeşitli makalelere bakınız: http://lsa. colorado.edu.

29 İnternetteki basın açıklamasından; şu adreste: http://lsa.colorado.edu/essay_press.html.

lan yöntemden basittir). Aslında, Landauer ve Susan Dumais, GAÇ'a dair eski makalelerinden birinde, sinir ağını temel alan bir GAÇ benzerini betimler.[30]

GAÇ yönteminin, olasılıkları temel alan bir uzantısını Thomas Hoffman önerip sınamıştır.[31] David Blei, Andrew Ng ve Michael Jordan daha genel bir olasılıksal model geliştirmiştir.[32] Her türlü olasılık modeli, 1980'lerden itibaren yapay zekâda çok belirgin bir rol oynamaya başladı. Şimdi de değineceğimiz konu başlığı bu.

30 Thomas K. Landauer ve Susan T. Dumais, "Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge", *Psychological Review*, c. 104, no: 2, 1997, s. 211-240. 1977 tarihli bir versiyonu şu internet adresinde: http://lsi.research.telcordia.com/lsi/papers/PSYCHREV96.html.

31 Thomas Hofmann, "Probabilistic Latent Semantic Indexing", *Proceedings of the Twenty-Second Annual International SIGIR Conference on Research and Development in Informatic Retrieval*, 1999. Şu internet adresinde: http://www.cs.brown.edu/~th/papers/Hofmann-SIGIR99.pdf.

32 David M. Blei, Andrew Y. Ng ve Michael I. Jordan, "Latent Dirichlet Allocation", *Journal of Machine Learning Research*, c. 3, 2003, s. 993-1022. Şu internet adresinde: http://www.cs.princeton.edu/~blei/papers/BleiNgJordan2003.pdf.

28
BAYES AĞLARI

28.1 OLASILIKLARIN AĞLARDA GÖSTERİMİ

İnsan uslamlamasının çoğunluğu, kesin olmayan önermeler ve nicelikler hakkındadır. Pek çok şey hakkındaki inancımız koşulludur (yani değişebilir) ve niteliklidir (yani farklı güvenirlik düzeyleri vardır). YZ sistemlerinin de belirsiz bilgiyle başa çıkabilmesi gerekir. Bir YZ etmenine ait olgu, önerme ve kuralların, koşullu ve nitelikli olduğunun düşünülmesi yerinde olur. En sonunda, bilgilerinin bir kısmını insanlar sağlar, bir kısmı da sınırlı hassasiyeti ve güvenirliği olan sensörlerden gelir. Yine de YZ'de gerçekleştirilmiş ilk çalışmaların büyük kısmı, bilginin muğlak doğasını göz ardı etmiştir. Aslında Marvin Minsky, eski YZ makalelerini derlediği kitabın, "olasılık kavramlarının açıkça kullanımını" içermediği saptamasında bulunmuştur.[1]

Bununla birlikte günümüzde çoğu YZ araştırmacısı, makinelerin gereksinim duyduğu bilginin büyük kısmının olasılıksal olarak tanımlanması gerektiğini, dolayısıyla bu bilgiyle yapılacak uslamlamada olasılık kuramının olanaklarının kullanılmasının yerinde olduğunu kabul eder. Fakat tıpkı mantıksal uslamlamadaki gibi, olasılıkçı uslamlama da YZ'nin ezeli düşmanı olan birleşimsel patlamaya maruz kalır. Örneğin, bir YZ etmeninin bilgisi bir önermeler kümesinden oluşuyor diyelim. Bu önermeler arasındaki olası etkileşimlerden ötürü, doğru bir olasılıkçı uslamlama, bu önermelerin tek tek olasılığını bilmeye bağlıdır. Bunun yerine, önermelerin çeşitli kombinasyonlarının, "birleşik olasılık" denilen olasılık değerleri istenir çoğunlukla; bu durum, olayın genelinde, kullanıma elverişsiz olacak kadar büyük gösterimler ve denetlemesi zor olan hesaplamalar doğurur.

MYCIN ve PROSPECTOR gibi, belirsizlikle başa çıkabilen eski YZ sistemleri, gösterim ve bilişim zorluklarını gidermek amacıyla sadeleştirici varsayımlardan faydalanıyordu. Bununla birlikte, bu sistemler, inançlar arasındaki önemli karşılıklı bağımlılıkları hesaba almadığı

1 Marvin Minsky (ed.), "Introduction", *Semantic Information Processing*, Cambridge. MA: MIT Press, 1968, s. 14.

için, bulguları aşırı dikkate alma gibi şeyler yüzünden uygunsuz sonuçlar doğuruyordu. 1980'lerde, söz konusu bağımlılıklarla daha iyi başa çıkabilen güçlü kimi yeni yöntemler icat edildi (ve başka alanlardan ithal edildi). Bu yöntemler, hem gösterim hem de bilişim sorunlarını büyük oranda sadeleştirmiştir. Bunlar, muğlak inançları ve bunların bağımlılıklarını, "olasılıksal grafik modeli" diye nitelenen grafik biçiminde temsil etmeyi içeriyordu. Bu tür modellerin en önemli çeşidini bu bölümde betimleyeceğim.

İlkin, belirli inançlara dair uslamlama yapmanın bazı zorluklarını ve bunlarla başa çıkma yollarını açıklamak için, araba motoruyla ilgili çeşitli önermeler içeren bir örneğe göz atalım. Araba motoru ve aksamı hakkında söyleyebileceğimiz bazı şeyler şöyle:

Ö1: Marş dinamosu tamam.
Ö2: Marşa basıldığında, marş dinamosu motoru çalıştırıyor.
Ö3: Yakıt sistemi tamam.
Ö4: Marşa basıldığında araba çalışıyor.

Bu önermeler açıkça ilintilidir. Öncelikle, Ö4, öbür üçüne bağlıdır; Ö4'ün yanlış olduğu gibi üzücü bir saptama, öbür üç önermeye duyduğumuz güveni kesinlikle değiştirecektir. Dahası, Ö1 ile Ö2'nin ilintili olduğunu bilmek için oto tamircisi olmaya gerek yoktur.

Burada yer alan karşılıklı bağımlılıkların tam bir açıklaması, tamam olan ya da tamam olmayan şeylerin *tüm* olasılıklarını sıralamayı gerektirir; böyle on altı olasılık var. Önermenin zıddını, bunun önüne olumsuzlama işaretini (¬) koyarak belirteceksek, ¬Ö1, "marş dinamosu tamam değil" ifadesini niteler. Bu işaretleme sistemiyle belirteceğimiz on altı olasılık şöyle:

Ö1, Ö2, Ö3, Ö4
Ö1, Ö2, Ö3, ¬Ö4
Ö1, Ö2, ¬Ö3, Ö4
Ö1, Ö2, ¬Ö3, ¬Ö4
Ö1, ¬Ö2, Ö3, Ö4
Ö1, ¬Ö2, Ö3, ¬Ö4
Ö1, ¬Ö2, ¬Ö3, Ö4
Ö1, ¬Ö2, ¬Ö3, ¬Ö4
¬Ö1, Ö2, Ö3, Ö4
¬Ö1, Ö2, Ö3, ¬Ö4
¬Ö1, Ö2, ¬Ö3, Ö4
¬Ö1, Ö2, ¬Ö3, ¬Ö4
¬Ö1, ¬Ö2, Ö3, Ö4

$$\neg Ö1, \neg Ö2, \quad Ö3, \quad \neg Ö4$$
$$\neg Ö1, \neg Ö2, \neg Ö3, \quad Ö4$$
$$\neg Ö1, \neg Ö2, \neg Ö3, \neg Ö4$$

Herhalde, motorları ve bunların beklenen güvenirliklerini bilen bir uzman, motor sisteminin içinde bulunabileceği bu on altı "durumdan" her birine belli olasılık değerleri atayabilir. Örneğin uzman, $p(Ö1, Ö2, Ö3, Ö4)$'le gösterilen, her şeyin sorunsuz olduğu durumun toplam birleşik olasılığını 0,999 olarak belirleyebilir. Bu uzmanın, böyle on altı rakam belirlemesi gerekecek. (Aslında sadece on beş, çünkü on altıncı rakam, genel toplamı bir yapmalı. Bunlar sadece olası durumlardır ve bir tanesi geçerli olmalıdır.) Bu birleşik olasılıkları bilen kişi (sabırlı ve olasılık kuramı konusunda donanımlı olduğunu varsayıyoruz), belirli başka olasılıkları da hesaplayabilir, örneğin, ancak yakıt sistemi kesinlikle sorunsuz olduğunda arabanın çalışma olasılığı gibi.

Bu küçük örnek kümesi için on beş rakam belirlemek külfetli bir işmiş gibi durmuyor, fakat, diyelim ki otuz farklı önermenin söz konusu olduğu çok daha gerçekçi bir problemde, $2^{30} - 1 = 1.073.741.823$ adet değerin belirlenmesi gerekir. Dahası, birkaç değer alabilen nicelikler varsa (ikili değeri olan önermelere ek olarak), olasılık sayısı çok daha fazlalaşır.

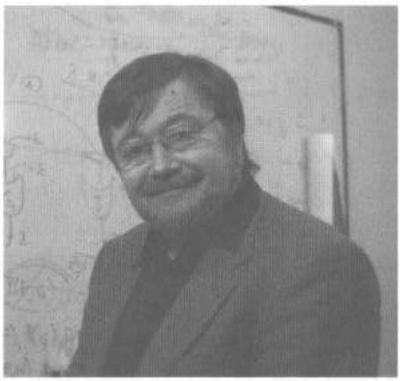
Elbette, en kötü olasılığın olacağını varsaydım, yani dört önermenin de birbirlerine karmaşık biçimde bağımlı olduğunu farz ettim. Öteki uçtaysa, önermelerin birbirlerinden bütünüyle bağımsız olduğu durum var. O zaman, on altı olasılıktan her biri şunun gibi formüllerle hesaplanabilir:

$$p(Ö1, Ö2, Ö3, Ö4) = p(Ö1)p(Ö2)p(Ö3)p(Ö4)$$

(gereken yerlere $\neg$ işareti konulur) ve sadece dört önerme için tek tek olasılık belirlemek yeterlidir.

Araba motorlarıyla ilgili örneğim, bu iki uç arasında bir yerde duruyor. Daha büyük ve gerçekçi problemler de öyle. Bu "aradalık hali", olasılıksal uslamlamayı daha kabul edilebilir kılmanın anahtarıdır. Her ne kadar bu gerçeği daha önce fark edip irdeleyen istatistikçiler olmuşsa da, başlıca gösterim ve hesap yöntemlerinden bazılarını geliştiren kişi Judea Pearl'dü (1936- ; Resim 28.1).

Los Angeles'daki California Üniversitesi bilgisayar bilimi profesörlerinden Pearl'ün kafasını, insanların belirsiz bilgilere dayanarak kolaylıkla mantık yürütmesi ve çıkarım yapmasının kolaylığıyla olasılık hesapları kullanarak bu becerileri kopyalamanın doğurduğu



Resim 28.1. Judea Pearl (Fotoğraf, Judea Pearl'ün izniyle.)

hesaplanma zorluklar arasındaki zıtlık meşgul ediyordu. Daha sonra belirteceği üzere, şu varsayımlarla yola koyulmuştu:[2]

1. [İnsanlara ait] İnanılabilir uslamlama ile olasılık kalkulüsü arasındaki tutarlı uyumun rastlantısal olmayışı, insan sezgilerinin, olasılık hesabına kaba bir biçimine başvurduğunu cidden akla getiriyor.
2. İnsan uslamlamasının hızı ve verimliliği ışığında bakıldığında, eski olasılıkçı sistemlerin başına bela olmuş bilişimsel zorluklar, işin temeliyle ilgili olamaz; dolayısıyla insanların kafasında sakladığı sadeleştirici varsayımlara dair doğru seçimler yapılarak bu zorluk aşılabilir.

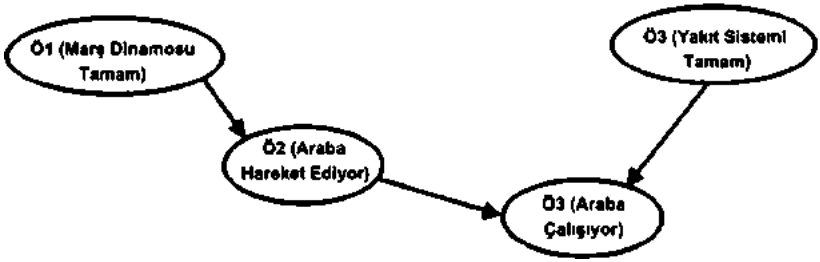
Pearl'ün kilit içgörüsü, önermeler ve başka niceliklerle ilgili inançların, çoğunlukla başka inançların "doğrudan nedeni" sayılabileceği ve bu nedensel bağlantıların, olasılıklar arasındaki ilişkilere dair sadeleştirici varsayımları kodlayan grafik yapılarla temsil edilebileceğiydi.

Olasılık bilgisini kodlamak amacıyla grafik yapılar kullanmayı öneren ilk kişinin Pearl olmadığı kesindir. Bizzat Pearl, eski çalışmalardan bahseder.[3] Russell ile Norvig,[4] Britanyalı istatistikçi I. J. Good'un yap-

2 Judea Pearl, "Two Journeys into Human Reasoning", *Artificial Intelligence: The First Century*, (çıkacak) içinde, Paul Cohen ve Clayton Morrison (ed.). Çevrimiçi versiyonu şu internet adresinde. http://ftp.cs.ucla.edu/pub/stat_ser/r331.pdf.

3 Bu gibi gösterimler için, Pearl'ün temel nitelikli kitabının sayfa 131 ve devamına bakınız: Judea Pearl, *Probabilistic Reasoning Systems: Networks of Plausible Inference*, San Francisco: Morgan Kaufmann Publishers, 1988.

4 Stuart Russell ve Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2. baskı, s. 528, Upper Saddle River, NJ: Prentice Hall, 2003



Resim 28.2. Ağ gösterimi.

tığı çalışmanın, "modern Bayes ağlarının öncüsü sayılabileceğini..."[5] yazmıştır. Ayrıca fizikçiler, Hans A. Bethe'nin bununla yakından ilintili çalışmasına işaret eder.[6]

Bahsettiğimiz araba motoru problemimiz için Pearl'ün kullanabileceği grafik türü Resim 28.2'de gösterilmiştir. İlgi odağı olan her önerme, grafikte "düğüm noktası"yla temsil edilir. Oklar, çeşitli önermeler arasındaki doğrudan etkileri gösteriyor, aynı zamanda bunlar arasındaki belirli bir olasılıksal bağımlılık olmama haline de işaret eder. Örneğin, Ö4'ün (marşa basıldığında araba çalışır) olasılığı, eğer Ö2'yi (marşa basıldığında, marş dinamosu motoru çalıştırır) ve Ö3'ü (yakıt sistemi tamam) zaten biliyorsak, Ö1 (marş dinamosu tamam) olasılığına bağlı değildir. Ö2'yi ve Ö3'ü zaten biliyorsak, Ö1'i bilmek bize Ö4 hakkında yeni bir şey anlatmaz. Olasılık kuramı diliyle söylemek gerekirse, Ö4'ün olasılığı, Ö4'ün anne babası, yani Ö2 ve Ö3 bilinirken Ö1'den *koşullu olarak bağımsızdır*. Gerçek dünyaya dair uslamlama görevlerinde, pek çok bu tür koşullu bağımsızlık vardır ve bunlar, nedensel olarak türetilmiş grafikler tarafından gösterilir. Bunları hesaba katmak, olasılıksal uslamlamanın karmaşıklığını büyük oranda azaltır.

Araba motoru örneğimizde, genel durumda gereksinim duyulan on beş olasılık yerine, şimdi sadece sekiz olasılığa gerek duyuyoruz. Bunlar şunlardır: Ö2 ve Ö3 verilmişken Ö2 ve Ö3'ün dört farklı durumu için Ö4 olasılıkları şöyledir:

$$p(Ö4 \mid Ö2, Ö3)$$
$$p(Ö4 \mid Ö2, \neg Ö3)$$

5 Şuna atıfta bulunuyorlar: I. J. Good, "A Causal Calculus (I)", *The British Journal for the Philosophy of Science*, c. XI, no: 44, 1961, s. 305-318.

6 Hans A. Bethe, "Statistical Theory of Superlattices", *Proceedings of the Royal Society of London*, Seri A, c. 150, no: 871, 1955, s. 552-575.

$$p(Ö4 \mid \neg Ö2, Ö3)$$
$$p(Ö4 \mid \neg Ö2, \neg Ö3);$$

Ö1 bilinirken Ö1'in iki farklı hali için Ö2'nin olasılığı, yani $p(Ö2 \mid Ö1)$ ve $p(Ö2 \mid -Ö1)$; ve Ö1 ile Ö3'ün olasılıkları, yani $p(Ö1)$ ve $p(Ö3)$. Bu olasılık değerleri kümelerinin her biri, ağda bunun karşılığı olan düğüm noktasıyla ilintili, "koşullu olasılık tablosunda" (KOT) depolanır. (Atası olmayan bir düğüm noktasının KOT'u, o düğüm noktası için koşulsuz bir olasılıktır.)

Olasılık kuramından elde edilen sonuç kullanılarak, (doğru bir olasılıkçı uslamlama için gereken) on altı birleşik olasılık da bu sekizinden hesaplanabilir. Aslında burada karşılıksız bir kazanç elde ediyor değiliz. Bunun yerine, ağın belirgin kıldığı koşullu bağımsızlıklarca sağlanan ek bilgiden faydalanıyoruz.

Bayes kuralı, başka düğüm noktaları verildiğinde çeşitli düğüm noktalarının olasılıklarını hesaplamada önemli rol oynadığı için, Pearl, bu tür grafikler için "Bayesçi inanç ağları" terimini uydurmuştur (genelde sadeleştirilerek Bayes ağı ya da inanç ağı denir).[7] Başka önermeleri hangi önermelerin doğrudan etkilediğini ("nedeni olduğunu") özenle not ederek, Bayes ağı kurmanın görece kolay olduğu kanıtlanmıştır. Bu şekilde oluşturulan grafiklere, kuramcılar "dairesel olmayan yönlü grafikler" (DYG) der: "yönlü" denilmesinin nedeni, okların kök düğüm noktasından, bunun eseri düğüm noktalarına uzanmasıdır, "dairesel olmayan" denmesinin nedeniyse, bir düğüm noktasından çıkan okun asla aynı düğüm noktasına dönmeyişidir.

O halde, KOT'daki olasılık değerlerinin nereden geldiği sorulabilir? Bazı ağlar için, belirli önermelerin başka önermeleri nasıl etkilediğine aşina olan bir uzman, olasılıklar hakkında tahminlerde bulunabilir. Bu tür olasılıklara, "öznel olasılık" denir, çünkü neden ve sonuç hakkında uzmanın öznel görüşlerine dayanırlar. Bununla birlikte, KOT'u değerle doldurmanın uzak ara en yararlı yöntemi, bu değerleri, gerçek örneklere dair büyük veri tabanlarından tahmin etmektir. Bu işin nasıl yapıldığını bir sonraki kısımda anlatıyorum.

KOT'lar (ağın yapısıyla birlikte), hangi araçla elde edilmiş olursa olsun, ağdaki kimi düğüm noktalarının, başka düğüm noktalarının olasılıklarından nasıl etkilendiğine dair hesaplamalarda kullanılır. Bu hesaplamalara, "olasılıksal çıkarsama", denir. Bunlar için çeşit çeşit

7 Judea Pearl, "Fusion, Propagation, and Structuring in Belief Networks", *Artificial Intelligence*, c. 29, 1986, s. 241-288.

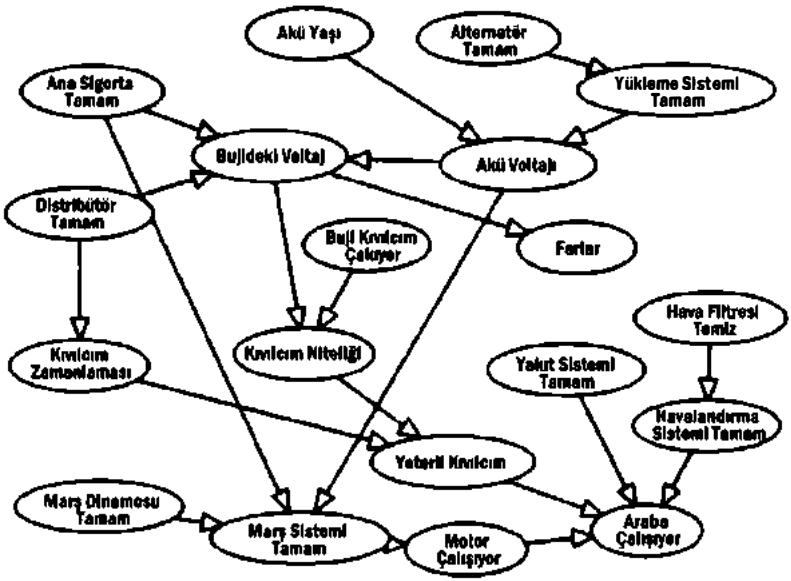
tatbiki hesaplama yöntemi tasarlanmıştır, hatta gerçekçi problemlerde gereken görece büyük ağlar için de.

Herhangi bir hesaplama işine dalmaksızın, Bayes ağlarının başlıca üç olasılıksal çıkarsama tarzını açıklamak amacıyla küçük motor ağından faydalanacağım. Örneğin, marş dinamosunun kesinlikle sorunsuz olduğunu biliyorsak [yani $p(\text{Ö1}) = 1$], arabanın çalışma olasılığını hesaplayabiliriz. Bilinen olasılık değerlerini ağda aşağı doğru (ok yönünde) "göç ettirmeye" çoğunlukla "nedensel uslamlama" denir. Bunun aksine, arabanın çalışmayacağını biliyorsak [yani $p(\text{Ö4}) = 0$], marş dinamosunun sorunsuz olma olasılığını ve yakıt sisteminin sorunsuz olma olasılığını hesaplayabiliriz. Olasılık değerlerini ağda yukarı doğru (okların tersi doğrultuda) göç ettirmeye çoğunlukla "bulgusal" ya da "tanısal" uslamlama denir. Bir belirti üzerinden nedenlerin olasılığını çıkarsamaya çabalayan hekimler (ve bir sistemdeki hataları bulmaya çalışan herkes) aslında bunu yapmaktadır.

Önemli bir uslamlama tarzı daha var ve buna "açıklamayla savuşturma" denir. Bir örneğe göz atalım: Diyelim ki arabanın marşının basmadığını biliyoruz ve yakıt sisteminin sorunlu olma (yani tamam değil) olasılığı ile marş dinamosunun sorunlu olma olasılığını hesapladık. Derken, aslında marş dinamosunun bozuk olduğunu anlıyoruz diyelim. Bu ek bilgiyi hesaba katınca, yakıt sisteminin sorunlu olma olasılığının düştüğünü görürüz. Marş dinamosunun sorunu, arabanın marşının basmayışını "açıklar", dolayısıyla yakıt sisteminden şüphe duyma nedenimiz azalır. Marş dinamosunun çalışmayışı, olası yakıt sistemi sorununu "açıklamayla savuşturmuş" olur. Açıklamayla savuşturma stratejisi, tıp, hukuk, bilim alanlarında çalışanlar tarafından ve gündelik uslamlamada sık sık kullanılır. Örneğin savunma avukatı, bankanın gizli kamera sisteminde saptanan (ve onun müvekkili olmayan) başka bir kişinin varlığını kanıt olarak gösterip kendi müvekkilinin banka soygununa karıştığı iddiasını açıklamayla savuşturabilir.

Açıklamayla savuşturma etkisini, Resim 28.3'te gösterilen, motorlarla ilgili bir parça daha büyük ağda yerine getirilen gerçek çıkarım hesaplamalarıyla açıklayabilirim.[8] Arabanın marşının basmadığı

8 Bu ağ, etkileşimli bir demonun konusudur ve buna, şu ders kitabının çevrimiçi kaynaklarından biri olarak erişilebilir: David Poole, Alan Mackworth ve Randy Goebel, *Computational Intelligence: A Logical Approach*, New York: Oxford University Press, 1998. Şu internet adresindeki uygulamaya bakınız: http://aispace.org/bayes/. Bu uygulamada, düğüm noktaları eklenip çıkarılabilir, koşullu olasılık tabloları değiştirilebilir, ayrıca bileşenlerin performansı gözlemlendikten sonra olasılıkların değeri sorgulanabilir.



Resim 28.3. Etkileşimli bir web sitesinden alınmış Bayes ağı. (Alan Mackworth ile David Poole'un izniyle kullanılmıştır.)

saptamasından sonra, marş dinamosunun sorun olma ihtimali 0,023 olarak hesaplanır (ağın koşullu olasılık tabloları kullanılarak hesaplanır; bu tablolar diyagramda gösterilmiyor) ve yakıt sisteminin sorunlu olma ihtimali 0,283 olarak hesaplanır. Fakat marş dinamosunun çalışmadığı saptamasının eklenişiyle, sorunun yakıt sistemi olma olasılığı yarıdan fazla düşüp 0,1'e iner.

Araba motoruyla ilgili daha gerçekçi ve faydalı bir Bayes ağı kurmak istiyorsak, çok daha fazla bileşen ve altsistemden bahsetmemiz gerekir. Bu tür bir ağ, düğüm noktalarıyla ilintili koşullu olasılık tablolarıyla birlikte yüzlerce düğüm noktası içerebilir. Her ne kadar koşullu bağımsızlıklar, tek tek belirtilmesi gereken olasılıkların sayısını azaltsa da, bunların sayısı yine de öyle büyük olabilir ki kesin olasılıksal çıkarımları hesaplamak fiilen imkânsız hale gelebilir; bu olasılıkların değerlerinin toplandığını varsaysak bile durum bu. Neyse ki, gerek duyulan olasılıkların sayısını azaltmaya olanak tanıyan çeşitli sadeleştirmeler vardır. Bu sadeleştirmelerle birlikte, büyük ağlarda yaklaşık fakat yine de yararlı çıkarımları hesaplamak, uygulanabilir haldedir. Bu sadeleştirmelerden ve yaklaşık hesaplama yöntemlerinden bazılarının görece karmaşık matematik araçları içerdiğini, bunlardan

pek çoğunun istatistik ve kontrol mühendisliği gibi komşu alanlardan kaynaklandığını belirtmeden geçmeyelim.[9] Bu tür Bayes ağı hesaplamaları, önceden hesaplanması fiilen imkânsız olan problemlerin nasıl teknik ilerlemeler doğurduğuna dair başka bir örnek sunuyor.

Resim 28.4'te, görece büyük bir Bayes ağı örneği sunuyorum.[10] Bu ağ, karaciğer-öd hastalığı (karaciğer, safra kesesi ve bağlantılı organların hastalığı) hakkındaki bilgiyi temsil ediyor ve tıp öğrencilerinin kullanacağı bir araç olarak geliştirilmiştir. Bu ağ kısmen, INTERNIST-1'in bilgi tabanından türemiştir. Ağın, 448 düğüm noktası ve 908 oku var. Eğer tam koşullu olasılık tabloları kullanılırsa, 133.931.430 olasılığın belirtilmesi gerekir. Bu ağı geliştirenler, çeşitli sadeleştirmeler kullanarak bu rakamı 8.254 adet değere indirebilmiştir.

Yüzlerce düğüm noktası içeren Bayes ağları, biyoloji, tıp, belge sınıflandırma, görüntü işleme, hukuk, hata-düzeltme kodlaması gibi pek çok alandaki uygulamalarda kullanılmıştır.[11] Bu ağların pek çoğu, büyük veri kümelerinden otomatik olarak türetilir; bu konu başlığını bir sonraki kısımda ele alacağım.

28.2 BAYES AĞLARININ OTOMATİK KURULMASI

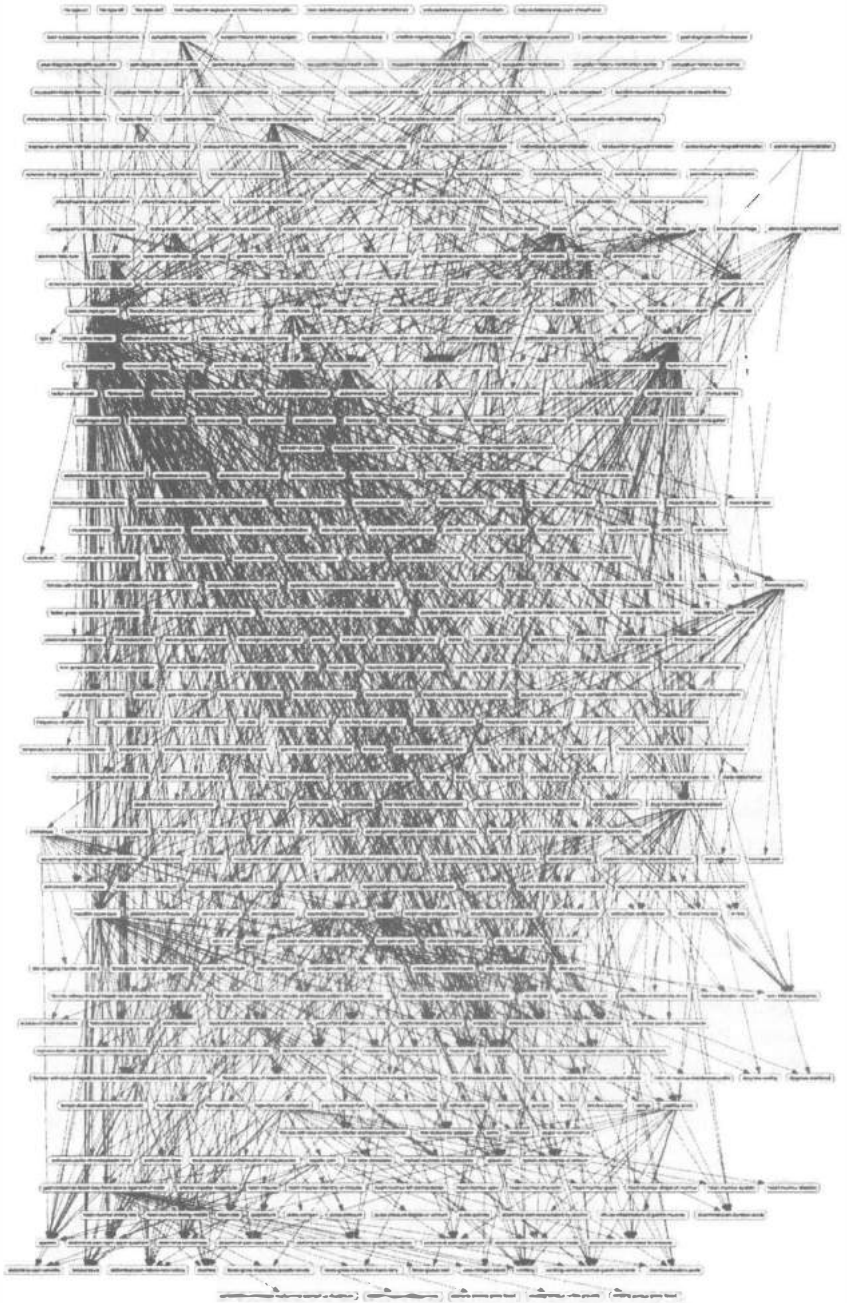
Bayes ağlarının böyle önemli hale gelişinin sebeplerinden biri, büyük veri tabanlarından yola çıkılarak otomatik olarak kurulabilmesidir. Yani bu ağlar "öğrenilebilir" ve öğrenilmiş sürümler, söz konusu alanda uslamlama yapmak amacıyla kullanılabilir. Bu öğrenme yöntemlerini geliştirmenin öncülerinden ikisi Greg Cooper ile Edward Herskovits'di.[12] Bu konu, faal bir araştırma alanı olmayı sürdürüyor

9 Bu tekniklerden bazılarının betimlenişi için bkz. Stuart Russell ve Peter Norvig, *a.g.e.*

10 Malcolm Pradhan ve diğ., "Knowledge Engineering for Large Belief Networks", *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, San Francisco: Morgan Kaufmann Publishers, 1994, s. 484-490. Sıkıştırılmış sürümü şu internet adresinde: ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-94-47.ps.gz.

11 Örneğin bkz. Olivier Pourret (ed.), Patrick Naïm (eş-hazırlayan) ve Bruce Marcot (eş-hazırlayan), *Bayesian Networks: A Practical Guide to Applications*, New York: John Wiley and Sons, Inc., 2008.

12 Greg F. Cooper ve Edward Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data", *Machine Learning*, c. 9, 1992, s. 309-347. Şu internet adresinde: http://www.genetics.ucla.edu/labs/sabatti/Statl80/bayesNet.pdf ve http://bmir.stanford.edu/file_asset/index.php/610/SMI-91-0355.pdf.



Resim 28.4. Tıpla ilgili büyük bir Bayes ağı. (Gregory Provan'ın izniyle kullanılmıştır.)

ve önemli katkılar yapan başkaları da olmuştur.[13]

Genel anlamda süreç şöyle işliyor. Bir ağı öğrenme, o ağın yapısını öğrenmeyi içerir, yani düğümlerinin ve bağlantılarının tanımını, ayrıca ağın KOT'larını öğrenmeyi. İlk olarak, bilinen bir yapının KOT'larının nasıl öğrenileceğini, ardından bizzat yapının nasıl öğrenildiğini açıklayacağım, gerçi bu iki süreç aslında birbirleriyle bağlantılıdır. Araba motoruyla ilgili dört düğüm noktalı Bayes ağına yeniden göz atalım. Peki, ata düğüm noktaları Ö2 (yani araba hareket ediyor) ve Ö3 (yani yakıt sistemi sorunsuz) olan Ö4 düğüm noktasının KOT'unu nasıl öğreniriz? Bu önermeler için kullandığım kısaltmalarla belirtmek gerekirse, KOT, şu aşağıdaki koşullu olasılıklarından meydana gelir:

$$p(\text{Ö4} \mid \text{Ö2}, \text{Ö3})$$
$$p(\text{Ö4} \mid \text{Ö2}, \neg\text{Ö3})$$
$$p(\text{Ö4} \mid \neg\text{Ö2}, \text{Ö3})$$
$$p(\text{Ö4} \mid \neg\text{Ö2}, \neg\text{Ö3})$$

Elimizde arabanın bazen çalıştığı bazen çalışmadığı, motorun bazen çalışıp bazen çalışmadığı, yakıt sisteminin bazen sorunsuz bazen sorunlu olduğu durumlardan oluşan örnek kümelerinin büyük bir derlemesi olsaydı, bunları "örneklem istatistiği" denen tablolar haline getirebilirdik. Örneğin bu örneklemlerde, motorun çalışmadığı ve yakıt sisteminin sorunsuz olduğu durumlarda arabanın kaç kere çalıştığını bir kenara not edip, bu sayıyı motorun çalışmadığı ve yakıt sisteminin sorunsuz olduğu durumların toplam sayısına böleriz. Bu oran, $p(\text{Ö4} \mid \neg\text{Ö2}, \text{Ö3})$'ün tahmini olarak kullanılır. Öbür üç olasılık ve ağdaki başka KOT'ların olasılıkları için de benzer tahminlerde bulunabiliriz. Örneklemin yeterince büyük olması, bu tahminlerin makul bir güvenilirliğinin olması demektir: Örneklem ne ölçüde büyükse, tahminler o ölçüde sağlam olur.

Örneklem istatistiklerinin derlenmesi (bazen ek hesaplamalarla güçlendirilir; gerçi burada o hesaplamalara girmeyeceğim), *bilinen* bir yapıya sahip ağın KOT'larını tahmin etme olanağı sağlar. Peki, bilinmeyen ağın yapısını nasıl öğrenebiliriz? Bu yöntem, aşağıdaki şu adımları içeriyor:

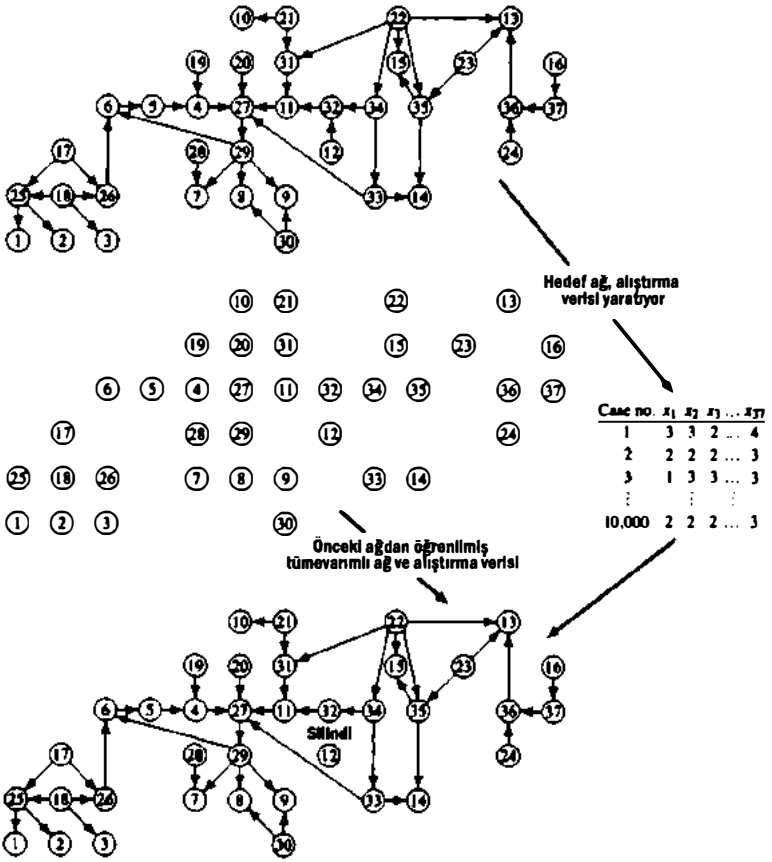
13 Örneğin, David Heckerman'ın öğretici makalesine ve yayınlarına şu internet sitesinden bakınız: http://research.microsoft.com/~heckerman/; ayrıca Michael Jordan'ın (ed.) şu derlemesine bakınız: *Learning in Graphical Models*, Cambridge, MA: MIT Press, 1998.

1. Temel bir aday yapıyla başla, örneğin düğüm noktaları arasında bağlantı olmayan yapı gibi; bu yapının KOT'larını tahmin etmek için veri derlemesinden faydalan. (Atası olmayan bir düğüm noktasının KOT'unun, o düğüm noktası için sadece koşulsuz bir olasılık olduğunu anımsayın. Söz konusu olasılık, bununla ilintili önermenin veri kümesinde kaç kere geçerli olduğunun oranıyla tahmin edilebilir.)
2. Bu ağın "iyilik ölçüsünü" hesapla. Öne sürülen ölçülerden biri, hesaplanmış KOT'larıyla birlikte ağın, özgün veri derlemesini iletmek (yani yeniden oluşturmak) üzere ne ölçüde kullanılabileceğine dayanır.
3. Önceki ağlardan küçük değişikliklerle farklılaşan "komşu" ağları değerlendirmeye alarak, "tepe tırmanma" arama işlemi başlat (bu işlem, bir ayrıt eklemeyi, var olan bir ayrıtı silmeyi, düğüm noktalarını değiştokuş etmeyi içerir). Değişim geçirmiş ağları değerlendirmek amacıyla, bunların KOT ve iyiliği hesaplanır. Değişmiş ağlar içinde, iyiliğinle en fazla artışı gösteren üzerinde karar kıl.
4. İyileştirme yapamadığın noktaya ulaşana dek tepe tırmanma işlemine devam et (ya da önceden tanımlanmış bir durma kısıtası sağlanana dek).

Her ne kadar bu işlem oldukça külfetli görünse de (ki öyledir), bilgisayarlar bu tepe tırmanma işlemini makul bir verimlilikle başarır, bu biçimde görece karmaşık ağlar öğrenilmiştir.

Örnek olarak, Resim 28.5'teki ağlara bakalım. Bu resimde üç ağ gösterilmiş. İlki, hastanenin yoğum bakım birimindeki havalandırma dairesinde kullanılan uyarı sistemini ilgilendiren bir sorun için 37 değişken arasındaki ilişkileri kodluyor. Bilinen bu ağ, 37 düğüm noktası için 10.000 rasgele değer içeren, alıştırma amaçlı bir örnek kümesi oluşturmak üzere kullanılmış. Bu rasgele örnek kümesinden yararlanılarak ve ikinci ağdan yola çıkılarak (düğüm noktaları arasında hiçbir bağımlılığı, dolayısıyla bağlantısı olmayan ağ), yukarıda betimlenmiş yöntemlere benzer yöntemler sayesinde üçüncü ağ öğrenilir (SUN SPARCstation 20 bilgisayarında yaklaşık beş saatte).[14] Yapılar arasındaki çok yakın benzerliğe dikkat edin; sadece bir ayrıt eksik.

14 Ayrıntılar için bkz. Peter Spirtes ve Christopher Meek, "Learning Bayesian Networks with Discrete Variables from Data", *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, San Francisco: Morgan Kaufmann Publishers, 1995, s. 294-299.



Resim 28.5. Bayes ağını öğrenmek. (Nils J. Nilsson, *Artificial Intelligence: A New Synthesis* içinde, s. 350, San Francisco: Morgan Kaufmann Publishers, 1998.)

Bazen, veri kümesinde yer almayan özellikleri temsil eden düğüm noktaları ağa eklenerek, ağ yapısı büyük oranda sadeleştirilebilir. Ağ öğrenme yöntemlerinin kapsamı, icat edilmiş bu özellikleri temsil eden "saklı düğüm noktaları"nı yerleştirmeyi öğrenecek şekilde genişletilmiştir. Verilerden icat edilen özellikler çoğunlukla, verileri doğuran görüngülere dair anlayışımızı derinleştirmek bakımından faydalıdır.

28.3 OLASILIKSAL İLİŞKİ MODELLERİ

Stanford profesörlerinden Daphne Koller (1968- ; Resim 28.6), öğrencileri Avi Pfeffer ve Lise Getoor, ayrıca iş ortağı Nir Friedman'la (eski



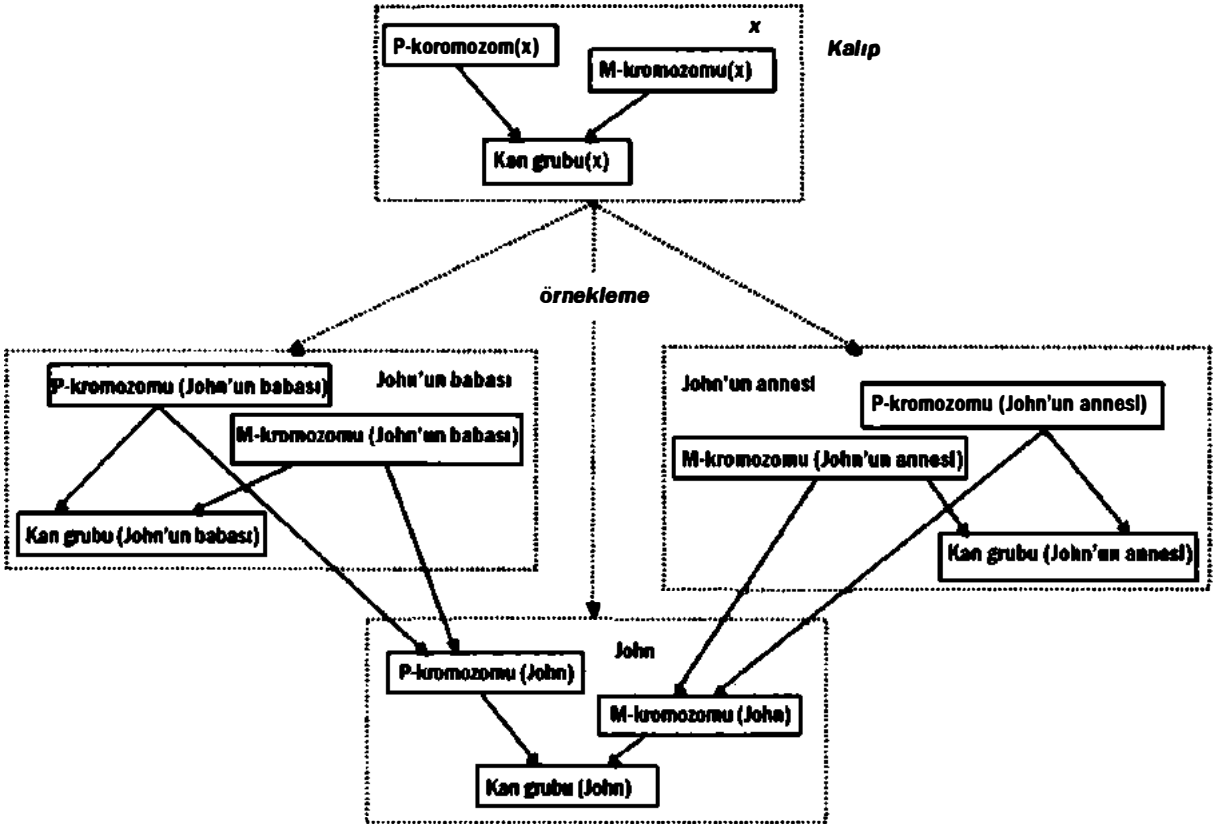
Resim 28.6. Daphne Koller (Fotoğraf, Daphne Koller'in izniyle.)

bir Stanford öğrencisi, şimdi de Hebrew Üniversitesi'nde profesör) birlikte, Bayes ağlarını temel alan "Olasılıksal İlişki Modelini" (OİM) geliştirdi.[15] OİM, olasılığı yüklemcil biçimsel dizgeyle bütünleştirir. [Bu iki gösterim biçimini birleştirmeye yönelik çalışmalara kalkışan başka araştırmacılar daha önce de olmuştu, örneğin bunların en önde geleni British Columbia Üniversitesi'nden David Poole'du (1958-).][16] OİM'ler, ağdaki kimi düğüm noktalarının bazı özelliklerinin ortak olabilmesinden faydalanır, gerçi bu özelliklerin iç değişkenlerinin değerleri ortak olmayabilir (yüklemcil biçimsel dizgede aynı belirtimin, iç değişkenlerine göre farklı değerlerle yazılması gibi).

Örneğin, kişinin kan grubunun ve kromozom bilgisinin, anne babasından miras aldığı kromozomlara göre değiştiğini gösteren ağ, tekrar eden alt ağlar barındırır. Resim 28.7'de bir örnek sunuyorum. Bu örnekte, tek "kalıp" üzerinden farklı alt ağlar oluşturulur ve bu alt ağların özellik değişkenleri, farklı bireylerde cisimleşir. OİM kullanmak, Bayes ağlarının tasarımını, her alt ağı (hafif farklı olacak şekilde) ayrı ayrı tasarlama işlemine kıyasla çok daha verimli kılar. Koller, OİM'ler hakkında kafa yormaya, üç şeritli bir otoyol modelleyen Bayes

15 Daphne Koller ve Avi Pfeffer, "Probabilistic Frame-Based Systems", *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Menlo Park, CA: AAAI Press, 1998, s. 580-587.

16 David Poole, "Representing Diagnostic Knowledge for Probabilistic Horn Abduction", *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Sydney, Australia, Ağustos 1991, s. 1129-1135. Yeniden baskısı: W. Hamscher, L. Console ve J. de Kleer (ed.), *Readings in Model-based Diagnosis*, San Francisco: Morgan Kaufmann Publishers, 1992. Şu internet adresinde: http://www.cs.ubc.ca/spider/poole/papers/ijcai91.pdf.



Resim 28.7. OİM örneği.

ağını dört şeritli otoyol modeline dönüştürmesi gereken bir öğrenciyle sohbet ederken başladığını anlatıyor. Öğrenciye, "fakat sadece tek şerit ekleyeceksin, herhalde yapının bir kısmını tekrar kullanabilirsin", demiş.[17]

OİM'nin yapısını ve KOT'larını ya bir tasarımcı belirleyebilir, ya da bunlar verilerden öğrenilebilir.[18] OİM'nin ek bir faydası da kalıp değişkenlerinin gösterimiyle ortaya çıkan nesnelerin, oluşan Bayes ağında birbirine bağlanmasıdır (diyagramda bazılarının bağlanması gibi); bu nesneler arasındaki ilişkiler dışarıdan belirtilebilir ya da öğrenilebilir. Tıpkı tüm Bayes ağlarındaki gibi, olasılıksal çıkarsama yordamları, belli düğüm noktaları verildiğinde bazı düğüm noktalarının olasılıklarına dair sorguları yanıtlamak amacıyla kullanılabilir.

OİM'ler ve bunlarla ilintili yapılar, örneğin gen anlatımı verilerinden gen faaliyetini düzenleyici ağların elde edilmesi gibi çeşit çeşit uygulamada kullanılmıştır.[19]

Genel olarak Bayes ağlarına gelirsek, daha şimdiden burada sayamayacağımız kadar çok uygulama vardır. Çeşitliliği hakkında fikir vermek amacıyla, genom araştırmalarında,[20] araba trafiği öngörülerinde ve yönlendirilmesinde,[21] salgın hastalık modellerinde[22] ve bilgi-

17 Daphne Koller'in e-postası, 27 Temmuz 2008.

18 Nir Friedman ve diğ., "Learning Probabilistic Relational Models", *Proceedings of the International Joint Conference on Artificial Intelligence*, s. 1300-1309, 1999; şu internet adresinde: http://www.cs.huji.ac.il/~nirf/Papers/FGKPl.pdf.

19 Eran Segal ve diğ., "Module Networks: Identifying Regulatory Modules and Their Condition-Specific Regulators from Gene Expression Data", *Nature Genetics*, c. 34, 2003, s. 166-176. Şu internet adresinde: http://www.wisdom.weizmann.ac.il/~eran/ModuleNetworks.pdf.

20 Nir Friedman, "Inferring Cellular Networks Using Probabilistic Graphical Models", *Science*, c. 303, no: 5659, 6 Şubat 2004, s. 799-805.

21 Microsoft, arabayla giderken yön bulmakla ilgili internet sitesinde Bayes ağlarını temel alan ClearFlow adlı programı kullanıyor: http://maps.live.com/. Bkz. Eric Horvitz ve diğ., "Prediction, Expectation, and Surprise: Methods, Designs, and Study *of* a Deployed Traffic Forecasting Service", *Proceedings Twenty-First Conference on Uncertainty in Artificial Intelligence*, UAI-2005, s. 275-283. Edinburgh, İskoçya, Temmuz 2005; şu internet adresinde: http://research.microsoft.com/en-us/um/people/horvitz/horvitz_traffic_uai2005.pdf. Ayrıca bkz. Cyril Furtlehner, Jean-Marc Lasgouttes ve Arnaud De La Fortelle, "Belief Propagation and Bethe Approximation for Traffic Prediction", Rapport de Recherche 6144, INRIA, Mart 2007; şu internet adresinde: http://www-rocq.inria.fr/Masgoutt/publications/RR-6144.pdf.

22 Gregory F. Cooper ve diğ., "Bayesian Biosurvelliance of Disease Outbreaks", *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence*, ACM International Conference Proceeding Series; c. 70, 2004, s. 94-103; şu internet ad-

sayar kullanıcısının bir sonraki adımını tahmin edip Windows işletim sisteminin daha talep edilmeden uygulama verisini belleğe "getirmesini" mümkün kılmada[23] kullanıldığını belirtelim. Aynı zamanda, Bayes ağlarını temel alan bilgi yakalama ve uslamlama sistemleri satan şirketler de var.[24]

Tüm bu uygulamaların bize öğrettiği şeylerden biri, çok miktarda verinin önemidir; önde gelen YZ ders kitaplarından birinin eşyazarı ve Google'da Araştırma Müdürü olan Peter Norvig'e göre veri miktarı, günümüz YZ'sinde başlıca konulardan biridir.[25] Aslında Peter bana, Google'ın, dünyadaki en büyük YZ sistemi olduğunu söyledi. Ona nedenini sorduğumda, "veri, veri, veri", yanıtını verdi; sonuçta veri, herkesten çok Google'da var.[26]

28.4 ZAMANSAL BAYES AĞLARI

Son birkaç kısımda açıklanan Bayes ağı örnekleri, pek çok uygulamada kullanılan daha büyük ağlarla birlikte, "durağan" diyebileceğimiz türdendir. Yani, düğüm noktalarının ve tüm KOT'ların temsil ettiği önermeler ve nicelikler, bir anlamda zamandan bağımsızdır ve zamanda aynı anı (ya da tüm anları) ele alırlar. Ancak, farklı zamanlardaki nicelikleri içeren bir olasılıksal ağı, yani saklı Markov modellerini daha önce betimlemiştim. 17.3.2'de, konuşma tanımada SMM'lerin nasıl kullanıldığını açıklamıştım. SMM'nin yaygın biçimlerinden biri, Resim 28.8'de gösteriliyor.

Bu diyagram, $x_1, x_2, \ldots x_n$ birimlerinin zaman dizisinin başka bir, $y_1, y_2, \ldots y_n$ zaman dizisini nasıl doğurduğunu gösteriyor. Her x_i'in sonraki x'ler ve y'ler üzerindeki etkisi olasılıksal olarak tanımlıdır. Bu ağın aslında bir Bayes ağı olduğunu anlamak kolay, içerdiği nicelikler bir zaman dizisinde beliriyor olsa da bu SMM'ye *birinci dereceden* Markov süreci denir, çünkü her durumu, hemen bir önceki duruma bağlıdır. Daha yüksek dereceden süreçlerde her durum, kendinden önceki birden fazla durumdan etkilenir.

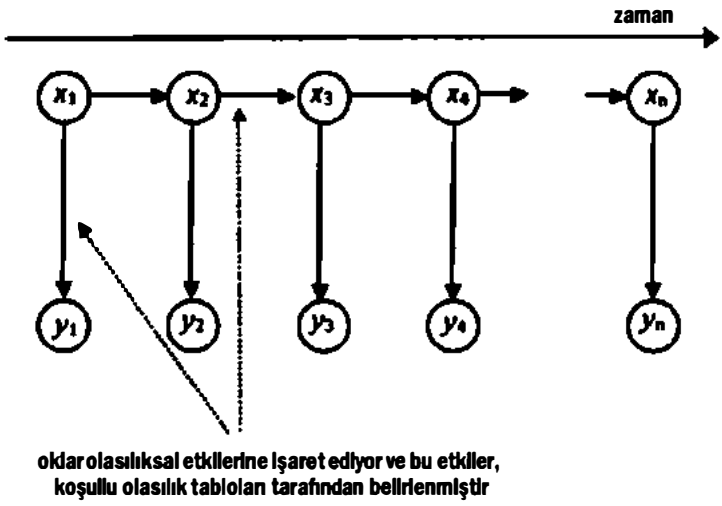
resinde: http://www.dbmi.pitt.edu/panda/papers/UAI2004.pdf.

23 Eric Horvitz, "Continual Computation Policies for Utility-Directed Prefetching", *Proceedings of the Seventh ACM Conference on Information and Knowledge Management (CIKM 98)*, New York: ACM Press, 1998, s. 175-184.

24 Bu şirketlerden biri Hugin Expert'tir. Bkz. http://www.hugin-Com/info/.

25 Peter Norvig, kişisel iletişim, 12 Kasım 2007.

26 Peter Norvig, kişisel iletişim, 21 Kasım 2008.



Resim 28.8. Saklı Markov modeli.

SMM ağında her x_i, "durum değişkenidir" ve y_i'ler "gözlenir değişkenlerdir". Durumların değerlerinin bilinmediği varsayılır (yani "saklı"dır), fakat gözlenebilenler ölçülebilir, dolayısıyla bilinirler. Her durum, bununla ilintili gözlenir değişkenleri ve bir sonraki durumu doğurur. Ağın koşullu olasılık tablolarını bildiğimizi varsayarız, yani gözlenir değişkenlerin olasılıklarını ve belirli durum değerinde bir sonraki durumu biliriz. En az bir gözlenir değişkenin değeri bilindiğinde, Bayes ağlarında olasılık hesaplamada kullanılan yöntemlerin herhangi birinden faydalanarak, durumların güncellenmiş olasılıklarını hesaplayabiliriz.

Bir örneğe göz gezdirelim. Diyelim ki ücra bir havalalanında, uçuş hava koşulları ya sisli ya da sisli değil olarak belirtiliyor. Havaalanında bir sensör, hava koşullarını kaydedip her beş dakikada bir sinyal iletiyor olsun. Fakat havaalanına inmeye niyetlenen uçakların aldığı bu sinyal ara sıra hatalı olabiliyor. Bu sürecin halleri, yani SMM modelindeki *x*'ler ya 1 ya da 0 değerini alır: 1, sis olduğunu gösterir. Uçağın aldığı sinyaller yani SMM'deki *y*'ler de ya 1 ya da 0 değeri alır: 1, sis gözlendiğine işaret eder; fakat gözlemler hatalı da olabilir. Daha somut konuşmak gerekirse, diyelim ki bir sonraki durumun şimdiki durumla aynı değerde olma olasılığı %75 (sis, sürme eğiliminde) ve gözlenir değişkenin hatalı olma olasılığı %5. Bu olasılıklar, Bayes ağı koşullu olasılık tablolarının oluşturulmasına olanak tanıyor. (Her durumun, bir sis

derecesi yansıtması sağlanarak ve önceki tek duruma ek olarak başka durumlar da kullanarak bu örnek daha gerçekçi kılınabilir.)

Uçak pilotu, aldığı y dizisini temel alarak inişe geçip geçmeme kararı vermek zorunda. Örneğin pilot, o anki gözlem de dahil önceki gözlemleri temel alarak iniş kulvarının sisli olma olasılığını bilmek isteyebilir. SMM deyişiyle, bu olasılığı hesaplayan işleme, "süzme" denir. Buna seçenek olarak pilot, gözlemleri temel alarak o andan itibaren 10 dakika içinde iniş kulvarının sisli olma olasılığını hesaplamak isteyebilir. Bu işlemeyse "öngörü" denir. Gerçi pilotun çok işine yaramayabilir ama şimdi de dahil o ana dek yapılmış gözlemlerin dizisini temel alarak on dakika önce iniş kulvarının sisli olma ihtimalini merak edebilir. Bu işleme de "düzleştirme" denir.

17.3.2 numaralı başlıkta konuşma tanımadan bahsederken, SMM durumlarının tek tek sözcüklere ve gözlemlerin, dalgabiçimi kesitlerine denk geldiğini anlatmıştım. Bu uygulamada, o ana dek tüm dalgabiçimi gözlemleri için en olası sözcük zincirini hesaplamak istiyorduk.

Tüm bu hesaplamalar, yani süzme, öngörü, düzleştirme ve en olası durum dizisi, Bayes ağı çıkarsama yöntemleri kullanılarak yapılabilir. Bu yöntemlerin uzmanlaşmış birkaç sürümü var ve bunlardan bazıları, YZ dışındaki alanlarda ortaya çıkmıştır. Bunlar, uygulama ve ağın gözlenir unsurlarına göre değişir. Bu yöntemler arasından ileri-geri algoritmasını, Viterbi algoritmasını ve Kalman süzmesini anıyorum (ama bunları açıklamaya kalkışacak değilim). Matematiğine güvenen cesur okurlar, Russell ile Norvig'in mükemmel ders kitabında berrak açıklamalar bulabilir.[27] Tam açıklamaların görece karmaşık bir matematik içermesi, YZ'nin 1980'lerde başlayan teknik derinliğinin göstergesidir.

Verdiğim sisli/sisli-değil örneği dahil SMM'ler, zamanın her anında sadece tek durum değişkenine sahiptir. Her zaman anında daha fazla durum değişkeni bulunan ağlar oluşturmak mümkündür; bu değişkenlerin her biri, birbirlerini, gözlemleri ve ilerideki durum değişkenlerini etkiler. Bunlara genel olarak "dinamik Bayes ağları" (DBA) denir ve YZ'de ilk olarak Thomas Dean (1950-) ve Keiji Kanazawa tarafından keşfedilmişlerdir.[28] Ek durum ve gözlem değişkenleri, tam hesaplamaları fiilen imkânsız kılar, fakat "parçacık süzgeci" gibi bazı tatbiki kestirim yöntemleri geliştirilmiştir (ileride bunları ayrıntılı anlatacağım). Üstelik, tıpkı sıradan Bayes ağları gibi, DBA'lar, zamansal

27 Stuart Russell ve Peter Norvig, *a.g.e.*

28 Thomas Dean ve Keiji Kanazawa, "*A* Model for Reasoning about Persistence and Causation", *Computational Intelligence*, c. 5, 1989, s. 142-150.

süreçler hakkında bilgi içeren veri tabanlarından öğrenilebilir. Bunlar, bazı uygulamalarda, öncelikle de algı içeren uygulamalarda kullanılmıştır. Bu uygulamalardan birisi, sinemada filmlerin işlenmesidir ve bu süreçte olasılıksal çerçeveden-çerçeveye bağımlılıklar, hareketli nesnelerin tanınıp takip edilmesi amacıyla kullanılır. Başka bir uygulamaysa, insanlı ve insansız uçaklar için çarpışmadan kaçınma sistemlerinin tasdikidir.[29]

Her ne kadar bu bölüm Bayes ağları hakkında olsa da, bu ağlar, "olasılıksal grafik modelleri" denen önemli bir genel sınıfın sadece bir türüdür. Çoğunlukla Markov ağları denilen gelişigüzel Markov alanları, düğüm noktaları arasındaki bağlantıların yönsüz olduğu bu sınıfın başka bir üyesidir. Bunlar aslen, istatistiki fizik problemleri için geliştirilmiştir ve günümüzde görüntü işleme, duyusal algı ve beyin modelleme dahil pek çok alanda uygulanıyor. Bölüm 25.5.1'de bahsetmiş olduğumuz Boltzman makineleri, gelişigüzel Markov alanlarına örnektir. Başka sinir ağlarını da olasılıksal grafik modeli olarak yorumlama yöntemleri vardır.[30]

29 Mykel J. Kochenderfer ve diğ., "A Bayesian Approach to Aircraft Encounter Modeling", *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Honolulu, Hawaii, 18-21 Ağustos 2008.

30 Grafik modeller hakkında daha fazlası için bkz.: Daphne Koller ve Nir Friedman, *Structured Probabilistic Models: Principles and Techniques*, Cambridge, MA: MIT Press, 2009; Kevin Murphy'nin internet sayfası da yararlıdır: "A Brief Introduction to Graphical Models and Bayesian Networks", http://people.cs.ubc.ca/~murphyk/Bayes/bnintro html.

29
YAPAY ÖĞRENME

Otomatik veri toplama teknikleri, ucuz yüksek kapasiteli saklama donanımıyla birlikte, bol miktarda verinin toplanıp tutulmasına olanak tanımıştır. Müşterilerin satış noktalarında yaptıkları alımlar, sıcaklık ve hava basıncı okumaları (hava koşullarıyla ilgili başka verilerle birlikte), haberler, her türlü mali işlem, internet sayfaları ve internet etkileşim kayıtları sayısız örnekten birkaçıdır. Fakat büyük hacimli ham veri, yararlı sınıflandırılma, nicellenme ve bilgi çıkarımı için etkin "veri madenciliği" teknikleri gerektirir. Yapay öğrenme yöntemleri, veri çözümlemesinde gitgide önem kazanan bir rol oynuyor, çünkü muazzam miktarda veriyle başa çıkabilirler. Aslında, veri ne denli fazlaysa, o kadar iyidir.

Çoğu yapay öğrenme yöntemi, verilerden hipotezler oluşturur. Öyleyse (klasik bir örnek kullanmak gerekirse), eğer büyük bir veri kümesi kuğuların beyaz olduğu birkaç durum içeriyorsa ve kuğuların başka renkte olduğu durum içermiyorsa, o halde yapay öğrenme algoritması, "tüm kuğular beyazdır" çıkarımı yapabilir. Bu tür bir çıkarsama, "tümdengelimsel" değil "tümevarımsal"dır. Tümdengelimsel çıkarsamalar, mecburen ve mantıken kendi önermelerinden türer, oysa tümevarımsal çıkarsamalar, ek verilerle çürütülmeye daima açık olan hipotezlerdir. (Siyah kuğuların yaşadığı, keşfedilmemiş bir ada var olabilir.) Yine de büyük miktarda veriyi temel alan tümevarımsal çıkarsamalar son derece faydalıdır. İşin doğrusu, bizzat bilim, tümevarımsal çıkarsamalara dayanır.

Yaklaşık 1980'den önce yapay öğrenme (çoğunlukla sinir ağı yöntemleriyle temsil edilmiştir), YZ'nin kıyılarında dolaşıyor sayılsa da, yapay öğrenme daha sonra modern YZ'de çok daha merkezî konuma geldi. Bir örnekten zaten bahsetmiştim, yani verilerden otomatik olarak inşa edilen Bayes ağlarının kullanımını anlatmıştım. 1980'lerden itibaren başlayan başka gelişmeler, yapay öğrenmeyi YZ'nin en belirgin dallarından biri yapmıştır. Bu çalışmaların bir kısmını bu bölümde anlatıyorum.

29.1 BELLEK TABANLI ÖĞRENME

Büyük miktarda veriyle başa çıkmaya yönelik olağan YZ yaklaşımı, veri miktarını bir şekilde azaltmayı içerir. Örneğin, bir sinir ağı, büyük miktarda öğrenme verisinin önemli noktalarını, ağın yapısı ve ağırlık değerleri aracılığıyla temsil edebilir. Benzer biçimde, verilerden yola çıkarak Bayes ağını öğrenmek, bu verileri ağın düğüm noktası yapısında ve koşullu olasılık tablolarında yoğunlaştırır.

Bununla birlikte, çabuk erişilen bilgisayar belleklerinde büyük miktarda veri saklama ve bu verilerle hesap yapma becerilerimizin artması sayesinde, *tüm* verileri, önceden sıkıştırma yapmaksızın, ihtiyaç duyulduğunda saklayıp kullanan teknikler mümkün olmuştur. Yani bu teknikler, veri bir görevde gerçekten kullanılmadan *önce*, veri miktarını azaltmaya kalkışmaz. Örneğin bir karar için gerekli tüm bu kesintiler, kararın verilmesi gereken anda gerçekleştirilir. Şimdi, *bellek-tabanlı* bu yöntemlerin bazılarını anlatıyorum.

Bölüm 4.3'te, çok boyutlu bir uzayda nokta sınıflandırmaya yarayan "en yakın komşu" yöntemlerinden bahsetmiştim. Örneğin, "en yakın k komşu kuralı", bir veri noktasını ona en yakın k komşu noktanın sınıflarının en çoğuna atar. Benzer bir teknik, bir veri noktasını sayısal bir değerle (ya da veri kümesiyle) ilintilendirmek amacıyla kullanılabilir. Örneğin, k adet en yakın komşuyla ilintili depolanmış değerlerin ortalaması, yeni bir noktaya atanabilir. Kuralın bu sürümü, uygulamaların denetiminde ya da öngörüsünde kullanılabilir. K-en yakın-komşu kuralı, bellek-tabanlı öğrenmenin prototip örneğidir ve olası uzantılarıyla ilgili birtakım sorular akla getirir.

Birincisi, en yakın komşu kuralını (şimdiye dek sunduğum haliyle) uygulamak için her veri bir rakam listesi olmak zorunda, yani çok boyutlu uzayda bir nokta ya da vektör olmalı. O halde sorulması gereken sorulardan biri şu: "Veriler nasıl temsil edilmeli ki en-yakın-komşu yöntemi gibi bir yöntem uygulanabilsin?" İkincisi, veri noktaları arasındaki "uzaklık" nasıl ölçülecek? Veriler, çok boyutlu uzayda yer alan noktalarla temsil edildiğinde, sıradan Öklid mesafesi doğal tercihtir. Bununla birlikte, bu durumda bile, boyutları, verilerin daha "yayılmış" bulunduğu boyutlara gereksiz ağırlık verilmeyecek şekilde "ölçeklendirmek" olağandır. Eğer veriler, bir uzayda yer alan nokta olarak temsil edilmezse, verilerin "yakınlığını" ölçmenin başka bir yolu kullanılmalıdır. Verilerin biçimine bağlı olarak çeşitli yöntemler önerilmiştir.

Üçüncüsü, en yakın k veri noktası arasında, birbirine yakın duranlar, uzaktakilere göre sonucu daha fazla mı etkilemeli? Temel k-en

yakın-komşu yönteminin kapsamı, veri noktalarının öneminin, yakınlıklarına bel bağlayan bir tarzda tartılmasıyla genişletilebilir. Genelde, gitgide uzaklaşan veri noktalarına giderek azalan değerler vermek için "çekirdek" denilen bir yöntemden faydalanılır.

Dördüncüsü, *k* değeri ne olmalı? Yeni veri parçası hakkında karar verirken kaç adet yakın komşu kullanacağız? Doğru çekirdek türüyle, *tüm* veri noktaları gözden geçirilebilir. En uzak komşuların karar üzerinde etkisi sıfır ya da göz ardı edilecek kadar küçük olabilir. Böylece, *k* için hangi değerin kullanılacağı sorusunun yerini, çekirdeğin etkisinin nerelere dek uzanması gerekir sorusu almıştır.

Son olarak, ağırlık biçilmiş tüm komşular hesaba katıldıktan sonra, nasıl karar veriyoruz ya da sayısal bir değeri veya değerleri nasıl belirliyoruz? Bu değer, komşuların çoğunluk oyuyla ya da belki ağırlık biçilmiş komşuların "ortalaması"yla aynı mı olmalı? Bu tercihe bağlı olarak, istatistiksel regresyon yönteminin çeşitli sürümlerinden yararlanmak mümkün.[1]

Andrew W. Moore (1965- ; Resim 29.1) ve Christopher G. Atkeson (1959- ; Resim 29.1), en yakın *k* komşu kurallarının geliştirilmesinde ve bunların veri madenciliğinde ve robot denetiminde önemli problemlere uygulanmasında öncü kişiler arasındadır.

Bu fikirleri kontrol problemlerine uygulamaya dair deneyler, birtakım makalelerde betimlenmiştir. Makalelerden biri,[2] "sopa oyunu" diye bilinen akrobasi oyununu oynayan bir robot cihazının denetiminden bahseder. Sopanın yere düşmemesinin nasıl sağlanacağını öğrenen, bellek-tabanlı bir sistem geliştirilmiştir. Resim 29.2, bu akrobasiyi yapan insan çizimini gösteriyor. Ayrıca, bazı duyu ve kontrol parametreleriyle birlikte robot düzeneği de gösteriliyor.

Robotik ve kontrol alanlarındaki uygulamaların yanı sıra, bellek-tabanlı öğrenme yöntemleri başka alanlarda da kullanılmıştır, örneğin veri madenciliği ve doğal dil işleme gibi.[3]

1 Hoş bir inceleme için, bellek tabanlı öğrenmenin öncülerinden Andrew W. Moore'un toparladığı derlemeye bakınız http://www.autonlab.org/tutorials/mbl08.pdf.

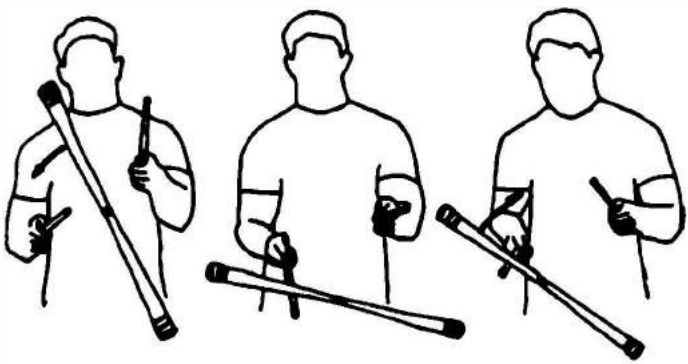
2 Christopher G. Atkeson, Andrew W. Moore ve Stefan Schall, "Locally Weighted Learning for Control", *Artificial Intelligence Review*, c. 11, s. 75-113, 1997; şu internet adresinde: http://www.cs.cmu.edu/~cga/papers/airl.ps.gz. Ayrıca bkz. Stefan Schaal ve Christopher G. Atkeson, "Robot Juggling: An Implementation of Memory-based Learning", *IEEE Control Systems Magazine*, c. 14, no: 1, Şubat 1994, s. 57-71; şu internet adresinde: http://www-clmc.usc.edu/publications/S/schaal-CSM1994.pdf.

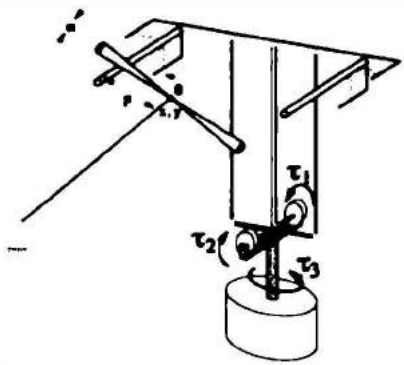
3 Walter Daelemans ve Antal van den Bosch, *Memory-Based Language Processing*,





Resim 29.1. Andrew Moore (solda) ve Chris Atkeson (sağda). (Fotoğraflar, Andrew Moore ile Christopher Atkeson'un izniyle.)





Resim 29.2. İnsan ve robotik bellek-tabanlı öğrenme sistemi tarafından oynanan "sopa denge oyunu". (Çizimler şu makaleden: Christopher G. Atkeson, Andrew W. Moore ve Stefan Schall, "Locally Weighted Learning for Control", *Artificial Intelligence Review*, sayı 11, s. 75-113, 1997; şu internet adresinde: http://www.cs.cmu.edu/~cga/papers/airl.ps.gz.)

29.2 ÖRNEK TABANLI USLAMLAMA

YZ'nin "Örnek Tabanlı Uslamlama" (ÖTU) denilen alt alanı, bellek tabanlı öğrenmenin genelleştirilmiş bir türü olarak görülebilir. ÖTU'da, "örnekler"den oluşturulan kütüphane, yeni örneklerin çözümlenmesinde, yorumlanmasında ve çözümünde kullanılır. Örneğin tıpta, hastaların tanı ve tedavi kayıtları, bir örnek kütüphanesi oluşturur; yeni örnek geldiğinde, tanı ve tedaviye yardımcı olsun diye kütüphanedeki benzer örneklere başvurulur. Hukukta, eski örnek davalardan, yeni davaların yorumlanışında ve hükme bağlanışında faydalanılır (hukuktaki içtihat uygulamasına göre, davalarda verilen kararlar, önceki davalarda çıkmış emsal hükümlere dayandırılır).

Yeni örnekle benzerlik gösteren örnekler, genelleştirilmiş bir örnek "uzayında" onun "komşuları" olarak düşünülebilir. Yakın komşuları belirlemek için, bu uzayda yakınlık fikri, benzerliğin bir ölçüsünü temel almalıdır. Örnek tabanlı uslamlamanın öncülerinden, Georgia Teknoloji Enstitüsü bilgisayar ve bilişsel bilim profesörü Janet Kolodner (1954- ; Resim 29.3), süreci şöyle betimliyor:[4]

> [Erişim bakımından] düzgün örnekler, yeni vaka hakkında geçerli öngörülerde bulunma potansiyeli olanlardır. Erişim ve yeni örneğin özellikleri kütüphanede dizin olarak kullanılır. O özelliklerin bir altkümesiyle ya da o özelliklerden türetilmiş özelliklerle etiketlenmiş örnekler erişime çağrılır.
>
> [Sonra bunlar arasından] uslamlama bakımından en fazla umut vaat eden vakayı ya da vakaları seçeriz... Bazen en iyi tek vakayı seçmek uygundur, bazen de küçük bir küme gereklidir.

Erişilmiş vaka (ya da vakalar) yeni vakaya uygulanacak şekilde uyarlandığında, (eğer başarılı olursa) gelecekteki problemleri çözmede başarılı olabilecek parçaları, gitgide büyüyen vaka kütüphanesine alınır.

Vaka-tabanlı uslamlamanın kökü, Roger Schank'in dinamik bellek modeline dayanır. İlk çalışmaları, Schank'ın iki doktora öğrencisi Janet Kolodner ile Michael Lebowitz yapmıştı.[5] ÖTU'nun başka bir önemli fikir kaynağı da Minsky'nin çerçevelerle ilgili fikirleridir. Amherst Massachusetts Üniversitesi profesörlerinden ve ÖTU öncülerinden Edwi-

Cambridge: Cambridge University Press, 2005.

4 Janet Kolodner, *Case-Based Reasoning*, San Francisco: Morgan Kaufmann Publishers, 1993, s. 18-19.

5 Bkz. Janet Kolodner, "Reconstructive Memory: A Computer Model", *Cognitive Science*, c. 7, no: 4, 1983, s. 281-328 ve Michael Lebowitz, "Memory-Based Parsing", *Artificial Intelligence*, c. 21, 1983, s. 363-404.



Resim 29.3. Janet Kolodner (solda) ve Edwina Rissland (sağda) (Fotoğraflar, Janet Kolodner ve Edwina Rissland'ın izniyle.)

na Rissland (1947; Resim 29.3), kendi ÖTU çalışması hakkında şunu yazmıştı:[6] "Bu çalışma doğrudan, 'kısıtlı emsal yaratma' çalışmasından baş vermişti.... bu sistem, örnek ağından elde edilmiş geçmişteki 'yakın' örneklerden (çerçeve olarak temsil edilir) yeni (karşıt) örneklerin oluşturulmasını modelliyordu."[7] Rissland ve öğrencileri, ÖTU'nun hukukta kullanılmasına büyük katkılar yapmıştır.[8] Rissland bana, ÖTU işleminin zaman zaman şu sözcüklerle özetlendiğini yazmıştır: Erişim, Yeniden Kullanım, Elden Geçirme ve Saklama.[9]

Edinburgh Üniversitesi Yapay Zekâ Uygulamaları Enstitüsü'nün hazırladığı bir internet sayfasına göre, "Vaka-Tabanlı Uslamlama, son yılların en başarılı uygulamalı YZ teknolojilerinden biridir. Ticari ve endüstriyel uygulamalar hızlıca geliştirilebilir ve şirketlerin var olan

6 Edwina L. Rissland, yayımlanmamış notları.

7 Edwina L. Rissland, "Example Generation", *Proceedings Third National Conference of the Canadian Society for Computational Studies of Intelligence*, Victoria, BC, 1980, s. 280-288.

8 Örneğin bkz. Edwina L. Rissland, "Examples in the Legal Domain: Hypotheticals in Contract Law", *Proceedings Fourth Annual Cognitive Science Conference*, University of Michigan, Ann Arbor, 1982, s. 96-99.

9 E-posta, 17 Şubat 2009.

veritabanlardan bilgi kaynağı olarak yararlanılabilir. Danışma ve tanı sistemleri en yaygın uygulamalardır."[10]

29.3 KARAR AĞAÇLARI

Yapay öğrenmenin yeni gelişmeleri listemde sıradaki madde, "karar ağacı" denen yapıların büyük veritabanlarından otomatik olarak oluşturulmasıdır. Karar ağaçları, bir veri kaydına atanacak sınıf ya da sayısal değeri belirlemeye yarayan test dizilerinden meydana gelir. Karar ağaçları, sayısal veriler kadar sayısal olmayan verilerin kullanımına da özellikle uygundur. Örneğin, personel veritabanı, çalışanın diyelim ki pazarlama, imalat ya da muhasebe gibi hangi kısımda çalıştığı bilgisini içerebilir. Veritabanı jargonunda bunun gibi veri birimlerine "kategorik" veri denir (sayısal verilerden ayırt etmek amacıyla). Bu kısımda, o yapıları, oluşturulmasında kullanılan otomatik öğrenme yöntemlerini ve kimi uygulamalarını anlatacağım.

29.3.1 Veri Madenciliği ve Karar Ağaçları

Veri madenciliği, büyük veritabanlarından faydalı bilgi çekip çıkarma işlemidir. Örneğin, insanların kredi kartı davranışlarına dair bir veritabanını gözümüzün önüne getirelim. Bu veritabanında, ödeme kayıtları, ortalama harcama miktarı, gecikme bedeli, ortalama bakiye, vesaire bulunabilir. Uygun veri madenciliği yöntemleri, başkalarının yanı sıra, yüksek gecikme bedeli ödeyen, harcama ortalaması yüksek olan ve başka bazı özellikleri olan insanların ortalamada yüksek bakiyesi sonucuna varabilir.

Önemli bir veri madenciliği yöntemi, karar ağacı oluşturmak amacıyla verilerden faydalanır. Karar ağaçlarının nasıl işlediğini açıklamak için çok basit bir veritabanına göz atalım. Diyelim ki bir şirket örneğin Wal-Mart, bazı ürünleri için indirim kuponu postalamış olduğu haneler hakkında bilgi depolayan bir veritabanına sahip olsun. Diyelim ki bu veritabanı, hanenin yeri (kent, banliyö, kırsal), hanenin tipi (müstakil ya da apartman), daha önce Wal-Mart müşterisi olup olmadığı ve kupon postalarına bu hanenin yanıt verip vermediği hakkında bilgi barındırıyor. (Elbette bu, açıklayıcı olsun diye uydurulmuş bir örnek; Wal-Mart'ın gerçek veritabanları hakkında bir şey biliyor değilim.)

10 http://www.aiai.ed.ac.uk/technology/casebasedreasoning.htm.

Bu tür bir veritabanının tablo halindeki gösterimi şöyle görünürdü:

Hane	Mevki	Tip	Müşteri	Yanıt
3014	Banliyö	Müstakil	Evet	Hayır
3015	Kırsal	Çok katlı	Hayır	Evet
5489	Kent	Müstakil	Evet	Hayır

Tablodaki her sıraya "kayıt" denir. Her sütunun en üstündeki kalemlere "özellik" denirken, sütünlardaki kalemlere, o özelliğin "değerleri" denir.

Daha sonra açıklayacağım yöntemlerle bu veritabanının çözümlenmesi, Resim 29.4'te gösterilen karar ağacının, postalanan kuponlara hangi hanelerin yanıt verdiği hangilerinin yanıt vermediği bilgisini elde ettiğini gösterir. Özellik değerlerine uygulanan testler, ağacın iç düğüm noktalarındadır (kutularda), ve sonuçlar (yanıt var mı yok mu), ağacın uçlarındadır (ya da yapraklarında; oval şekiller). Bu tür bir ağaç, postaya yeni kuponlar vermeden önce beklenen yanıtlarla ilgili öngörü yapma bakımından faydalıdır.

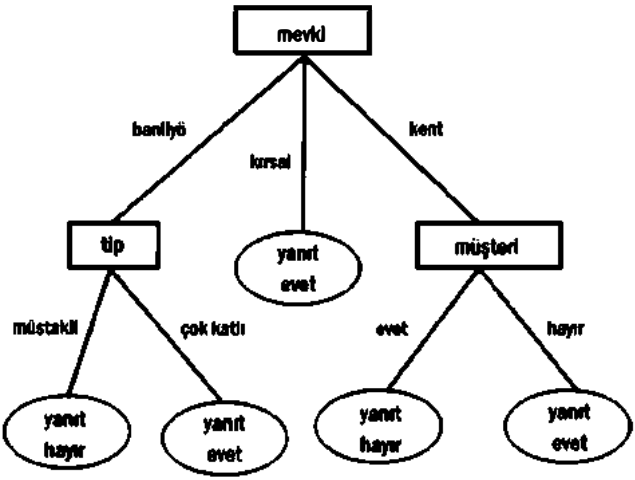
Büyük veritabanlarından bunun gibi karar ağaçlarını (ve çok daha büyüklerini) otomatik olarak oluşturacak (yani öğrenecek) yöntemler geliştirilmiştir. Başlıca yöntemlerin tarihçesinden ve nasıl işlediklerinden bir parça bahsedeceğim.

29.3.2 Karar Ağacının Oluşturulması

A. EPAM

Büyük olasılıkla, karar ağacı oluşturan en eski sistem, 1950'lerin ikinci yarısında Edward Feigenbaum tarafından, Carnegie Mellon Üniversitesi'nde (o zamanlar Carnegie Teknoloji Enstitüsü deniyordu) Herbert Simon'un yanında hazırladığı doktora tezinin parçası olarak geliştirilmiştir.[11] Sistemine EPAM (Elementary Perceiver and Memorizer [Temel Alıcı ve Ezberleyici]) deniyordu. Araştırmanın ama-

11 Feigenbaum bana, 1959 civarında Edward Fredkin'in bağımsız olarak (ve farklı bir amaçla) geliştirdiği başka bir ağaç oluşturma sisteminden bahsetmişti. Bkz. Edward Fredkin, "Trie Memory", *Communications of the ACM*, c. 3, no: 9, Eylül 1960, s. 490-499.



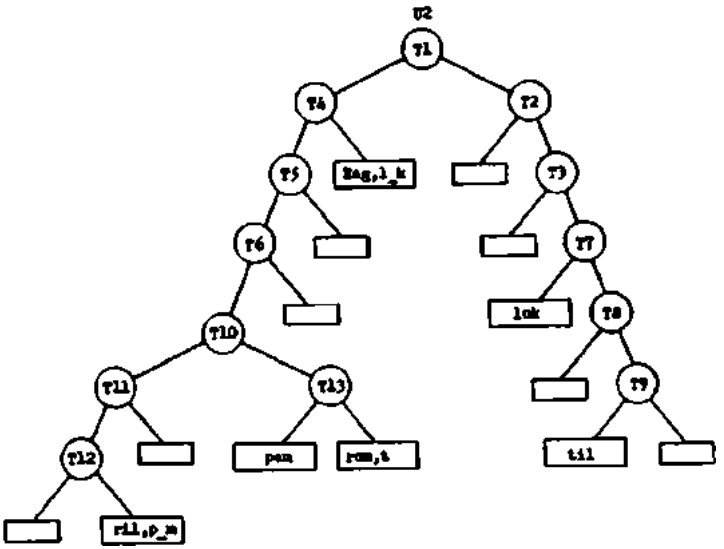
Resim 29.4. Yanıtları öngören karar ağacı.

cı, "[insanda] sözlü öğrenme görüngüsünü açıklayıp öngörmekti". Bu beceriyi sınayan standart bir psikoloji deneyi, insanlara DAX-JIR ve PIB-JUX gibi anlamsız hece çiftleri göstermeyi içerir. Bu çiftin ilk üyesine "uyartı" (*stimulus*), ikinci üyesine "yanıt" (*response*) denir. Bu gibi hece çiftlerini tekrar tekrar gören deneğe daha sonra rasgele bir uyartı gösterilir, böylece deneğin doğru yanıtı verme becerisi sınanır.

EPAM'a, "öğrenme fazı"ndayken bu tür hece çiftleri gösterilmişti. Öğrenme faslı, uyartılar ile yanıtlar arasındaki ilintileri saklamaya yarayan, Feigenbaum'un "ayrım ağı" dediği yapının oluşmasından meydana geliyordu. Bu ağ, iç düğüm noktalarındaki harflerin özelliklerine dair testleriyle ve ağacın ucunda ya da yapraklarında saklanan yanıtlarıyla bir karar ağacıydı. EPAM'ın "sınama fazı"nda, anlamsız uyartı hecesi, doğru yanıtın saklanmış olduğu bir yaprağa ulaşana dek ağaçtaki testlerden geçiriliyordu (umarız ki) . EPAM ayrım ağının bir örneği Resim 29.5'te gösteriliyor. Yuvarlak düğüm noktaları test, kutu düğüm noktaları yanıttır.

EPAM, "çiftli-çağrışım" öğrenme görevinde insanların performansını başarıyla modellemekle kalmamış, unutmayı da modellemiştir. Feigenbaum şunu ileri sürmüştür: "Bildiğimiz kadarıyla, [EPAM], öğrenen bir makinede bu tür bir unutma becerisine dair ilk somut demodur."[12] EPAM, Carnegie'nin liste işleme dili olan IPL-V'le yazılmış-

12 Edward A Feigenbaum, "The Simulation of Verbal Learning Behavior", *Proceedings of the Western Joint Computer Conference*, c. 19, 1961, s. 121-132. Yeniden baskısı:



Resim 29.5. EPAM ayrım ağı. (Alınan kaynak: Edward A. Feigenbaum, "An Information Processing Theory of Verbal Learning", doktora tezi, Carnegie Teknoloji Enstitüsü, s. 99, 1959, RAND Corporation tarafından Rapor P-1817 olarak yayımlanmıştır, Santa Monica, CA, 9 Ekim, 1959. Edward Feigenbaum'ın izniyle kullanılmıştır.)

tı. Aslında, IPL-V gibi dillerin liste işleme özellikleri, karar ağacı *yetiştiren* programları yazmak için gereklidir. Dolayısıyla, EPAM'ın, bu tür programların ilki oluşu şaşırtıcı değildir.

Feigenbaum'un programı hâlâ, hem insan zekâsı kuramlarına hem de YZ araştırmalarına büyük katkı olarak görülür. Simon, Feigenbaum ve başkaları, EPAM programları üzerinde çalışmayı sürdürdü ve bu çalışmalar, PC'de çalışan, IPL-V kodlu EPAM-VI'yla zirve noktasına ulaşmıştır.[13]

Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw-Hill, 1963.

13 Örneğin bkz. Edward A. Feigenbaum ve Herbert Simon, "EPAM-like Models of Recognition and Learning", *Cognitive Science*, c. 8, no: 4, 1984, s. 305-336; Howard B. Richman, J. J. Staszewski ve Herbert A. Simon, "Simulation of Expert Memory Using EPAM-IV", *Psychological Review*, c. 102, no: 2,1995, s. 305-330; ve Howard B. Richman, Herbert A. Simon ve Edward A. Feigenbaum, "Simulations of Paired Associate Learning Using EPAM-VI", Karmaşık Bilgi İşleme Çalışma Makalesi #553, Psikoloji Bölümü, Carnegie Mellon Üniversitesi, 7 Mart 2002. İkinci makale şu internet adresinde: http://www.pahomeschoolers.com/epam/cip553.pdf.

B. CLS

Öğrenen karar ağaçlarıyla ilgili bir sonraki önemli çalışma, 1960 civarında Yale Üniversitesi'nde yapıldı. Psikolog Carl I. Hovland ve onun doktora öğrencisi Earl B. (Buz) Hunt, insanların kavram öğrenme becerisinin bilgisayar modelini geliştirmişti.[14] Hovland 1961'de kansere yenik düştükten sonra, Hunt, kavram öğrenme çalışmalarını sürdürdü ve CLS (Concept Learning System [Kavram Öğrenme Sistemi]) adlı bir dizi karar ağacı öğrenme programı geliştirmek üzere Janet Marin ve Philip Stone'la işbirliği yaptı.[15] Hunt ve çalışma arkadaşları, EPAM'ın da bu konuyla ilgili ve daha eski bir çalışma olduğunu kabul ediyordu.

En azından YZ bakımından, CLS sistemleri kısa süre içinde ID3, CART ve ilişkili programlar gibi başka karar ağacı öğrenme sistemlerinin gölgesinde kalmıştır, örneğin. Bu programların ardındaki ana fikirleri açıklamak amacıyla ID3'ün nasıl işlediğini betimleyeceğim.

C. ID3

J. Ross Quinlan (1943- ; Resim 29.6), 1970'lerin ikinci yarısında (Sydney Üniversitesi'nden aldığı) akademik izin yılını Stanford'da geçirirken ID3'ü (Iterative Dichotomizer [Yineleyen İkiye Ayırıcı]) geliştirmişti.[16] (Bu isim, programın, veri kaydı kümelerini, iki ayrı sınıftan biri olarak sınıflandırana dek tekrar tekrar bölmesinden geliyordu. Programın sonraki sürümleri, bu sınıflandırmada ikiden fazla sınıfa olanak tanımıştır ama isimdeki "D" harfi öyle kalmıştır.) Quinlan eskiden, Washington Üniversitesi Bilgisayar Bilimi Bölümü'nde Earl Hunt'un doktora öğrencisiydi (işin doğrusu, Hunt'ın ilk doktora öğrencisiydi). Quinlan, ID3'ün yaratılışını bana e-postayla açıklamıştı:[17]

14 Carl I. Hovland ve Earl B. Hunt, "Programming a Model of Human Concept Formulation", *Proceedings of the Western Joint Computer Conference*, 9-11 Mayıs, 1961, s. 145-155. Yeniden baskısı: Edward A. Feigenbaum ve Julian Feldman (ed.), *Computers and Thought*, New York: McGraw-Hill, 1963, s. 310-325.

15 Earl B. Hunt, Janet Marin ve Philip J. Stone, *Experiments in Induction*, New York: Academic Press, 1966.

16 Quinlan'ın ID3 betimlemeleri için bkz. J. Ross Quinkan, "Discovering Rules by Induction from Large Collections of Examples", *Expert Systems in the Micro Electronic Age* içinde, Donald Michie (ed.), Edinburgh: Edinburgh University Press, 1979, s. 168-201 ve J. Ross Quintan, "Induction of Decision Trees", *Machine Learning*, c. 1, 1986, s. 81-106. Şu internet adresinde: http://www.cs.toronto.edu/~roweis/csc2515-2006/readings/quinlan.pdf.

17 Quinlan'dan bana gelen 18 Mart 2008 tarihli e-posta.



Resim 29.6. J. Ross Quinlan (Fotoğraf, Ross Quinlan'ın izniyle.)

> Donald Michie'nin [Stanford'da o da misafirdi] dersine giriyordum ve verdiği bir ödev merakımı çekti: satrançta basit bir oyun sonunun sonucuna karar verecek kuralı öğrenmek. ID3, Buz'ın [yani Earl B. Hunt'ın] CLS'sinin baştan kodlanması olarak doğmuştu, fakat iç kısmını bir parça değiştirdim (örneğin örnek kümesini bölme ölçütleri gibi) ve ID3'ün, o zamanlar için muazzam sayılan 29.000 alıştırma örneğiyle başa çıkmasını mümkün kılan yinelemeli yaklaşımı yerleştirdim.

ID3'ün, kurmaca Wal-Mart veritabanını kullanarak, yanıt özelliğinin alacağı değeri öngören bir karar ağacı inşa etme şekli kısaca şöyledir: ID3, öncelikle, yanıt özelliği için evet değeri taşıyan veri kayıtlarını hayır değeri taşıyanlardan ayırt etmek bakımından "en iyi" test olarak kullanılacak tek özelliği arar. ("En iyi"nin nasıl tanımlandığını az sonra anlatacağım.) Verileri kusursuz ayıran tek test yok, ama diyelim ki farklı yazı şekli, ötekilerden daha iyi iş çıkarsın. Ne de olsa bu örnekte, farklı yazı şekli özelliği için kırsal değerini taşıyan *tüm* veri kayıtları, yanıt özelliği için evet değerine sahiptir ve hiçbirinde hayır değeri yoktur. Diyelim ki farklı font değerini taşıyan veri kayıtlarının çoğunluğu (ama hepsi değil), yanıt özelliği için evet değerini taşıyor ve kent değeri taşıyan veri kayıtlarının çoğunluğu (yine hepsi değil), yanıt özelliği için hayır değerini taşıyor. Dolayısıyla, farklı yazı özelliği, yanıt özelliğine kıyasla, verileri ayırma konusunda daha iyidir (ama kusursuz değildir). Bu durumda, farklı yazı özelliğinin değerine dair test, karar ağacı oluşturulmasında faydalanacak ilk test olur.

O halde, şimdilik veritabanını üç altkümeye böldük; bunların ikisi, yanıt özelliği bakımından karma değerler barındırıyor. ID3, aynı ayırma tekniğini bu karma değerli altkümelerin hepsine uygular ve bunların her biri için, test olarak kullanılacak en iyi özellikleri belirler. Bu basit ve görece gerçek dışı örnekte, kullanılacak iki test, yani tip ve

müşteri, "saf" (yani karma değerler taşımayan) ayrımlar oluşturur ve Resim 29.4'te gösterilmiş olan karar ağacını elde ederiz.

Bununla birlikte, eğer ayrımlar saf ya da kabul edilir gibi olmasaydı, ID3, ayrımlar saf ya da kabul edilir sonuçlar verene dek veritabanlarının altkümelerinde test seçmeyi sürdürürdü.

Test edilecek özellik tercihi, faydalı karar ağaçları oluşturmada önemlidir. Quinlan, özgün ID3 programında,[18] test için hangi özelliği kullanacağını belirlerken oluşturulan ayrımın "doğruluğu"yla ilişkili bir ölçü kullanmıştı. Daha sonraki çalışmalarındaysa, "bilgi kazanımı" adlı bir ölçüden faydalanmıştı;[19] bunun tam tanımına burada girecek değilim, ama değerleriyle, istenen sınıflandırma hakkında en fazla "bilgiyi" ileten özellik olduğunu söylemeden geçmeyelim. Quinlan, enformasyon miktarını ölçerken Claude Shannon'ın tanımını kullanmıştı.[20] Daha da sonra, pek çok sonucu olan testler ağır basmasın diye, bilgi edinmenin normalleştirilmiş bir ölçüsünden faydalanacaktı.[21]

Bölüm 18.2'de uzman sistemlerden bahsederken, bunların EĞER-İSE kurallarını temel aldığını anlatmıştım. Quinlan'ın ve başkalarının simge tabanlı yapay öğrenmeye gösterdiği ilgi, verilerden bu tür kuralları öğrenmeye yönlenmişti çoğunlukla. "EĞER" kısmını yaratmak amacıyla testlerin izini takip edip "İSE" kısmı için ağacın uçlarındaki unsurları kullanarak, karar ağacından kurallar oluşturmak oldukça kolaydır. Örneğin, Wal-Mart veritabanı örneğinde, karar ağacından aşağıdaki kuralları türetebiliriz:

18 Yani, Quinlan'ın eserinde betimlenmiş program.

19 Örneğin bkz. J. Ross Quinlan, "Learning Efficient Classification Procedures and Their Application to Chess End Games", Ryszard S. Michalski, Jaime G. Carbonell ve Tom M. Mitchell (ed.), *Machine Learning: An Artificial Intelligence Approach*, San Francisco: Morgan Kaufmann Publishers, 1983, s. 463-482. (Bu arada, bence bu kitabın başlığı, kitabı hazırlayanların, sinir ağı yaklaşımlarıyla yapay öğrenmenin farkını belirtme isteğinden doğmuş.)

20 Klasik bir makale şudur: Claude E. Shannon, "A Mathematical Theory of Communication", *Bell System Technical Journal*, c. 27, Temmuz ve Ekim 1948, s. 379-423 ve 623-656. Şu internet adresinde: http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf.

21 Örneğin bkz. J. Ross Quinlan, "Decision Trees and Multi-Valued Attributes", *Machine Intelligence 11* içinde, Jean E. Hayes, Donald Michie ve J. Richards (ed.), Oxford: Oxford University Press, 1988, s. 305-318. Tüm bu ölçüler şu kitapta anlatılıyor: J. Ross Quintan, *C4.5: Programs for Machine Learning*, San Francisco: Morgan Kaufmann Publishers, 1993.

```
EĞER (mevki = banliyö) ve (tip = müstakil) ise, İSE (yanıt = hayır)

EĞER (mevki = banliyö) ve (tip = çok katlı) ise, İSE (yanıt = evet)

EĞER (mevki = kırsal) ise, İSE (yanıt = evet)

EĞER (mevki = kent) ve (müşteri = evet)ise, İSE (yanıt = hayır)

EĞER (mevki = kent) ve (müşteri = hayır)ise, İSE (yanıt = evet)
```

Quinlan'ın Stanford'daki çalışmasında, ID3, belirli satranç dizilişlerinin siyah için galibiyetle mi mağlubiyetle mi sonuçlanacağını öngören görece büyük karar ağaçları, dolayısıyla kural kümeleri yaratabiliyordu. Donald Michie'nin önerdiği bu problem türü için, ID3, öngörüleri %99,74 doğru olan 393 düğüm noktalı bir ağaç oluştururken 25 özellikten (tahta üzerinde satranç taşlarının konumlarının özelliklerini içeren) ve 29.236 farklı dizilişli bir veri tabanından faydalanmıştı.[22]

Karar ağaçları oluştururken kaçınılması gereken sorunlardan biri, "aşırı öğrenme"dir, yani çok az veriye dayanan testler seçilmesinden ötürü test sonuçlarının bütün olarak veriler içinde anlamlı ilişkiler yakalayamamasıdır. Asıl veritabanı ne kadar büyük olursa olsun, eğer bir dizi test sonucunda saf olmayan ama bir avuç veri kaydına indirgenmiş bir altküme üretiyorsa, bu kümeyi bölmeye yönelik her çaba, verileri aşırı öğrenecek ve faydalı olmayacaktır. Bu yüzden, karar ağacı öğrenme teknikleri, veri altkümelerinin çok az kayıtla kabul edilir sonuçlar verdiği noktaya ulaşmadan ağaç oluşumunu durdurur.

D. C4.5, CART ve Ardılları

Quinlan, karar ağacı oluşturma sistemlerine dair çalışmalarını sürdürüp, bunların gücünü ve uygulanabilirliğini artırmıştır. Bana, "ID3 oldukça basitti, yaklaşık 600 PASCAL satırından meydana geliyordu" demişti.[23] Onun C4.5 sisteminin (bu da yaklaşık 9.000 C satırından meydana geliyordu), üzerinde çalışabildiği veritabanı özellikleri, kategorik değerlere ek olarak kesintisiz sayısal değerler de içerebiliyordu. Hatta, kayıtları kimi özellikleri için eksik değerler barındıran bazı veritabanlarıyla da başa çıkabiliyordu. Son olarak, ağacın kimi kısımlarını budayarak ve ağaçtan türeyen EĞER-İSE kurallarını sadeleştire-

22 J. Ross Quinlan, "Discovering Rules by Induction from Large Collections of Examples", *Expert Systems in the Micro Electronic Age* içinde, Donald Michie (ed.), Edinburgh: Edinburgh University Press, 1979, s. 168-201.

23 E-postayla iletişim, 18 Mart 2008.

rek toplam performansı yükselten yöntemler kapsıyordu.[24] Quinlan'ın 1983'te kurduğu ticari şirket, C4.5'in C5.0 adlı iyileştirilmiş sürümünü pazarlıyor (Windows sürümü See5'le birlikte).[25] Donald Michie de şirket kurmuştur[26] ve bu şirket bağımsız olarak, ID3'ün ACLS adlı ticari sürümünü geliştirmiştir.

Bu dönemde yapay öğrenmedeki önemli gelişmelerden biri, sınıflandırma, tahmin ve öngörü konularında hem temel hem uygulama araştırmaları yapan YZ çalışanları ile istatistikçiler arasındaki verimli işbirliğiydi. Her ekip, başka ekiplerden bir şeyler öğrenmiştir ve bu sebeple yapay öğrenme çok daha zenginleşmiştir. Her ne kadar bu işbirliğine birtakım insanlar katılmış olsa da, özellikle Stanfordlu istatistikçi Jerome Friedman'dan (1939-) bahsetmek istiyorum; Friedman, 1990'larda bazı Stanfordlu YZ doktora öğrencileriyle çalışmaya başlamıştı. Karar ağacı oluşturmaya dair eski çalışmalarını sürdüren Friedman,[27] Leo Breiman, Richard Ohlsen ve Charles Stone'la işbirliği içinde, CART (Classification and Regression Trees) adlı sistemin geliştirilmesine yardımcı olmuştu.[28] CART'ın pek çok özelliği C4.5'le ortaktır (aslında C4.5, sayısal özellikleri ele alırken CART'ın tekniklerinden faydalanır). Bu kitap yazılırken, CART'ın en son sürümü CART 5, ticari bir şirket tarafından piyasaya sürülmüştü.[29]

Ayrıca, karar ağacı öğrenme sistemleri, çeşit çeşit veri madenciliği problemlerine uygulanmıştır.

E. Tümevarımsal Mantık Programlama

Önermeli mantık diliyle ifade edilen, karar ağaçlarından elde edilmiş EĞER-İSE kuralları, $P1 \wedge P2 \wedge \ldots \; PN \rightarrow Q$ biçimine sahiptir. P'ler ve Q'lar, iç yapısı olmayan önermelerdir. Daha önce, belirtim denilen önermelerin içsel savlar barındırdığı yüklemcil biçimsel dizgeden bahsetmiş-

24 Bkz. J. Ross Quinlan, *C4.5: Programs for Machine Learning*, San Francisco: Morgan Kaufmann Publishers, 1993.

25 C4.5'in tam sürümü, Ross Quinlan'ın internet sayfasından ücretsiz indirilebilir: http://www.rulequest.com/Personal/. C5.0'in küçük ölçekli sürümleri şu internet adresinden ücretsiz indirilebilir: http://www.nilequest.com/download.html.

26 Intelligent Terminals, Ltd.

27 Örneğin bkz. Jerome H. Friedman, "A Recursive Partitioning Decision Rule for Nonparametric Classification", *IEEE Transactions on Computers*, c. 26, no: 4, 1977, s. 404-408.

28 Bkz. Leo Breiman, Jerome H. Friedman, Richard A. Okhen ve Charles J. Stone, *Classification and Regression Trees*, Pacific Grove. CA: Wadsworth, 1984.

29 CART 5, Salford Systems'den temin edilebilir. Bkz. http://www.salford-systems.com/1112.php.

tim. Bu dilde, çok daha güçlü kurallar yazılabilir, örneğin $\forall(x, y, z)[Baba(x, y) \wedge Yavru(z, y) \rightarrow Baba(x, z)]$ gibi. Bu tür "ilişkisel" kuralları veritabanlarından ve başka "arkaplan bilgi"sinden öğrenmeye yönelik birtakım teknikler geliştirilmiştir. (Daha önce bununla ilintili "olasılıksal ilişkisel modeller" konusunu anlatmıştım; bunlar, değişkenli belirtimlere olanak tanıyan Bayes ağı sürümleridir.) İlişkisel kuralları öğrenen ilk sistemlerden biri, Quinlan tarafından geliştirilmişti ve adı FOIL'di.[30] Öğrenilen kurallar, bilgisayar dili PROLOG'daki (mantığa dayalı bir dil) ifadelerle aynı biçimi taşıdığından, bu kuralları öğrenmeye adanmış alana "Tümevarımsal Mantık Programlama" (ILP) denir. Her ne kadar ILP yöntemlerinin içerdiği mantık düzeneği burada açıklayamayacağım denli karmaşık olsa da, bunlardan bazıları, karar ağacı oluşturmayla yakın benzerlikler taşır.[31] ILP'nin uygulamalarına, ilaç etkinliği, protein ikincil yapısı ve sonlu sayıda öğeyle ağ tasarımı için ilişkisel kurallar öğrenme dahildir. Bunların hepsi, "ilişkisel veri madenciliği" diyebileceğimiz yaklaşımın örnekleridir.[32]

29.4 SİNİR AĞLARI

1960'larda, sinir ağı araştırmacıları, ağın ayarlanabilir ağırlıklarını değiştirip bütün ağın, "öğrenme" girdilerine uygun çıktılar vermesini sağlayacak çeşitli yöntemler kullandı. Örneğin, Cornell'de Frank Rosenblatt, üç katmanlı alfa-perceptron dediği yapının son katmanında ağırlık değerlerine ayarlıyordu. Bill Ridgway (Bernard Widrow'un Stanford'daki öğrencilerinden), MADALINE dediği sistemin ilk katmanında ağırlıkları ayarlıyordu. SRI'da MINOS II ve MINOS III sinir ağı makinelerinin ilk katmanında ağırlıkları ayarlamak amacıyla benzer bir yöntemden faydalanıyorduk. Başkalarıysa, ağırlık değerlerini be-

30 J. Ross Quinlan, "Learning Logical Definitions from Relations", *Machine Learning*, c. 5, 1990, s. 239-266.

31 Mantık kuramına hakim, ilgili okurlar, şu makaleye başvurabilir: Stephen Muggleton ve Luc De Raedt, "Inductive Lope Programming, Theory and Methods", *Journal of Logic Programming*, sayılar 19-20, s. 629-679, 1994 ve Nada Lavrac ve Saso Dzeroski, *Inductive Logic Programming: Techniques and Applications*, New York: Ellis Horwood, 1994 (şu internet adresinde: http://www-ai.ijs.si/SasoDzeroski/ILPBook/). Ayrıca bkz. Claude Sammut, "The Origins of Inductive Logic Programming: A Prehistoric Tale", *Proccedings of the Third International Workshop on Inductive Logic Programming* içinde, Stephen Muggleton (ed.), Bled, Slovenya, 1993, s. 127-147.

32 Bkz. Saso Dzeroski ve Nada Lavrac (ed.), *Relational Data Mining*, Berlin: Springer-Verlag, 2001.

lirlemek için çeşitli istatistik teknikleri kullanıyordu. Fakat hepimizi engelleyen şey, çok katmanlı ağlarda birden fazla katmanın ağırlıklarını değiştirmekti. (Ekibimizin başı Charles Rosen'ın, bürosunda oturup, ağırlık değişiklikleri için bulduğu yaratıcı yöntemleri sarı kareli tabletlere yazdığını hatırlıyorum; bunlardan hiçbiri işe yaramamıştı.)

29.4.1 Geri Yayılım Algoritması

Bu problem, 1980'lerin ortasında David Rumelhart, Geoffrey E. Hinton ve Ronald J. William'ın önerdiği "geri yayılım" adlı tekniğin icadıyla çözüldü.[33] Geri yayılım tekniğinin özündeki fikir basittir, ama matematiği (bunu atlıyorum) görece karmaşıktır. Ağın çıktısında oluşan hataya bağlı olarak, geri yayılım, ağırlıkların tümünde küçük ayarlamalar yaparak hatayı azaltır. Bu, bir tür tepe-tırmanma (daha doğrusu tepeinme) yöntemi sayılabilir; çünkü ağırlıklar arazisinde hataların düşük değerini arar. Fakat, olası tüm küçük ağırlık değişikliklerini deneyip, en dik yokuş inişine denk gelen hata kümesini belirlemek yerine, geri yayılım, ağırlık değişikliklerinin en iyi kümesini önceden hesaplamak için kalkülüsten faydalanır.

Üniversite (ya da belki lise) kalkulüsünü hatırlayan okurlar, bunun bir eğri ya da yüzey eğimini hesaplamak için kullanılabileceğini hatırlamakta zorlanmasa gerekir. Sinir ağı çıktısında oluşan hata, ağın ağırlıklarının bir fonksiyonu, yani "ağırlık uzamında" bir yüzey sayılabilir. Bu fonksiyon kâğıda yazılabilir ve bizi en dik eğimle yokuş aşağı taşıyacak ağırlık değişiklikleri kümesine yol açan ağırlıklara göre "türevi alınabilir" (kalkulüste bir işlem). Bu fikri, sinir ağları için dosdoğru hayata geçirmenin sorunlu yanı, bu ağların "eşik değerler" barındırması ve bu eşiklerin etkisi yüzünden hata yüzeyinin dik "zirvelerle" dolmasıdır. (Eşikli ağın çıktıları, ağırlıkların bazılarındaki çok küçük değişikliklerle birlikte 1'den 0'a ya da 0'dan 1'e değişebilir.) Kalkülüs işlemleri, düzenli değişen yüzeyler gerektirir ve zirvelerde zorlanır.

Rumelhart ve çalışma arkadaşları, eşiklerin yerine çıktıları yumuşak değişen birtakım bileşenler koyarak bu sorunla başa çıktı; gerçi eşikli ağlarla yaklaşık aynı işi yapmak için yeterince ani değişiklikler

33 David Rumelhart, Geoffrey E. Hinton ve Ronald J. Williams, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol 1: Foundations* içinde, James L. McClelland, David E. Rumelhart ve PDP Araştırma Ekibi (ed.), Cambridge, MA: MIT Press, 1986, s. 318-362. Ayrıca bkz. David Rumelhart, Geoffrey E Hinton ve Ronald J. Williams, "Learning Representations by Back-Propagating Errors", *Nature*, c. 323, Mektuplar, 9 Ekim 1986, s. 533-536.

geçirirler. Bu değişikliklerle birlikte, ağın tüm katmanlarındaki en iyi ağırlık değeri değişikliklerini hesaplamak amacıyla, ağ boyunca hata fonksiyonunu (çıktıdan girdiye) geri yaymak için kalkulüs kullanılabilir. Her ne kadar, kabul edilebilir ağırlık değerlerine odaklanma yavaş olsa da, pek çok sinir ağı öğrenme probleminde kullanılarak etkileyici sonuçlar elde edilmiştir.

Peki, bunu neden *bizler* düşünmedik? Aslında kimi insanlar, Rumelhart ile çalışma arkadaşlarından önce benzer fikirleri gözden geçirmişti. Büyük olasılıkla ilki, Arthur E. Bryson ile Y. C Ho'ydu; bunlar, Euler-Lagrange denklemlerini çözmek amacıyla yinelemeli eğim yöntemlerinden faydalanmıştı.[34] Doktora tezinde Paul Werbos da, çok katmanlı ağları eğitmek için hataların geri yayılımını önermişti.[35]

Yerel arama tekniklerine gelince, geri yayılım tekniği, hata yüzeyinin yerel asgarilerinden birinde sıkışıp kalabilir. Elbette, daha düşük (belki en düşük) bir hata değeri bulmak amacıyla, ağırlıkların farklı başlangıç değerleriyle yola çıkarak öğrenme sürecini tekrarlamak mümkün. Her durumda geri yayılım yöntemi, Laveen Kanal'ın 1993'te yazdığı gibi, "örüntü sınıflandırmada sinir ağlarını eğitmek için büyük olasılıkla hâlâ en yaygınlıkla kullanılan genel yöntemdir."[36]

Sinir ağı öğrenme yöntemleri, hava trafik denetim, kredi kartı sahtekârlığı tespiti, otomatik satış makinelerinin para birimi tanıması ve veri madenciliği dahil çeşitli alanlarda uygulanmıştır. Şimdi de birkaç uygulamasını anlatacağım.

29.4.2 NETtalk

Geri yayılım öğrenme yönteminin çok ilginç uygulamalarından birini, Terrence J. Sejnowski (1947-) ve Charles Rosenberg (1961-) geliştirmiştir. Bir sinir ağına konuşmayı öğretmişlerdi![37] Deneylerinden birinde,

34 Arthur E. Bryson Jr. ve Y. C. Ho, *Applied Optimal Control: Optimization, Estimation, and Control*, Waltham, MA: Blaisdell, 1969.

35 Paul Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences", doktora tezi, Harvard University, Cambridge. MA, 1974. Laveen Kanal, 1992 Kral Sun Fu Uluslararası Örüntü Tanıma Birliği (IAPR) ödülünü alırken, Werbos'la 1975'te yaptığı bir sohbeti hatırlatmıştır. Bkz. Laveen N. Kanal, "On Pattern, Categories, and Alternate Realities", *Pattern Recognition Letters*, c. 14, 1993, s. 241-255. Şu internet adresinde: http://www.lnk.com/prl14.pdf.

36 Laveen N. Kanal, "On Pattern, Categories, and Alternate Realities", *Pattern Recognition Letters*, c. 14, 1993, s. 241-255. Şu internet adresinde: http://www.lnk.com/prl14.pdf.

37 Terrence J. Sejnowski ve Charles R. Rosenberg. "Parallel Networks That Learn



Resim 29.7. Terrance Sejnowski (üstte) ve NETtalk'ta kullanılan sinir ağı (altta). (Fotoğraf ve çizim, Terrance Sejnowski'nin izniyle.)

NETtalk adlı sistemleri, altı yaşındaki bir çocuğun teklifsiz ve kesintisiz konuşmasının kâğıda dökülmüş metnini "okumayı" öğrendi ve akustik çıktı üretti (bu ses, çarpıcı biçimde çocuk sesine benziyordu). (Ses demosunu şu internet adresinde dinleyebilirsiniz: http://www.cnl.salk.edu/ParallelNetsPronounce/.) Ağın yapısı Resim 29.7'de gösteriliyor.

Ağda, yedi harfli zincirleri kodlamak amacıyla tasarlanmış 203 girdi birimi vardı. Metin, bu yedi birimden harf harf akıtılıyordu. Ayrıca bu girdileri, ayarlanabilir ağırlıklara bağlayan 80 "gizli birim" vardı. Gizli birimlerin, "harflerin seslerle eşlenmesi problemini çözmeye uygun içsel gösterimler oluşturması" umulmuştu. Seslerin, yani konuş-

to Pronounce English Text", *Complex Systems*, c. 1, 1987, s. 145-168. Şu internet adresinde: http://www.cnl.salk.edu/ParallelNetsPronounce/ParallelNetsPronounce-TJSejnowski.pdf.

ma seslerinin temel birimlerinin kodlanmış biçimini üreteceği varsayılan 26 çıktı birimi vardı. Çıktı birimleri, ek ayarlanabilir ağırlıklarla gizli birimlere bağlanmıştı. (Toplamda, 18.629 ayarlanabilir ağırlık vardı.) Son olarak, işitilebilir bir çıktı üretebilmesi için ses kodları ticari bir konuşma sentezleyicisini besleyecekti.

Bu ağ eğitilirken, her zaman adımında, çıktı birimlerindeki ses kodu, o zaman adımında kodun metin girdisinde olması gereken öğelerle karşılaştırılmıştı. Geri yayılım, bu hatayı azaltacak şekilde ağırlıkları değiştirmek amacıyla kullanılmıştı. Tekniğin yaratıcıları, "bir ağı yedi harflik pencereyle birkaç gün içinde eğitmenin mümkün olduğu ispatlandı" demiştir. (Bilgisayarların 1987'de çok daha yavaş olduğunu unutmayın.) "Genel olarak, konuşmanın anlaşılırlığı oldukça iyiydi" ve "ağ ne denli sözcük öğrenirse, yeni sözcükleri genelleştirmede ve doğru telaffuz etmede o denli beceri kazanıyor" sonucuna varmışlardı. 1.024 sözcüklük bir kümeyle eğitilen ağ, "aynı konuşmacının kesintisiz 439 sözcükle sınandı [daha fazla eğitilmeksizin]. Sistemin performansı %78'di, bu da gösteriyor ki küçük bir İngilizce sözcük kümesinden sonra bile, öğrendiklerinin büyük kısmını yeni sözcüklere aktarabilmişti". Resim 29.7'de gösterilen özgül ağa ek olarak, daha fazla gizli birimi olan ve iki katmanlı gizli birimi bulunan ağlarla da deneyler yapıldı. Genel olarak, büyük ağlar daha iyi performans sağlamıştır.

29.4.3. ALVINN

Karavan direksiyonuna yön veren bir sinir ağı uygulamasınıysa, Carnegie Mellon Üniversitesi doktora öğrencilerinden Dean Pomerlau geliştirmişti.[38] Karavandan, yola bakan kameradan ve arayüz aygıtından oluşan bu sisteme ALVINN (Autonomous Land Vehicle in a Neural Network [Yapay Sinir Ağlı Özerk Kara Aracı]) deniyordu. ALVINN, CMU Navlab taşıtını kullanıyordu; bu taşıt, ticari bir karavan şasisi üzerine hidrolik motor ve elektrikli dümen sistemiyle birlikte inşa edilmişti. Bir CMU makalesine göre, "Bilgisayarlar, elektrik ve hidrolik servolar aracılığıyla karavana yön verip sürebilir ya da insan şoför, test mevkisine varmak amacıyla ya da bilgisayarı devre dışı bı-

38 İlk makalelerden biri: Dean Pomerleau, "ALVINN: An Autonomous Land Vehicle in a Neural Network", *Advances in Neural Information Processing Systems*, c. 1, San Francisco: Morgan Kaufmann Publishers, 1989, s. 305-313. Pomerleau'nun tezi: "Neural Network Perception for Mobile Robot Guidance", Carnegie Mellon Üniversitesi, Şubat 1992.



Resim 29.8. CMU'nun, ALVINN tarafından kullanılan Navlab taşıtı. (Fotoğraf, Carnegie Mellon Üniversitesi'nin izniyle.)

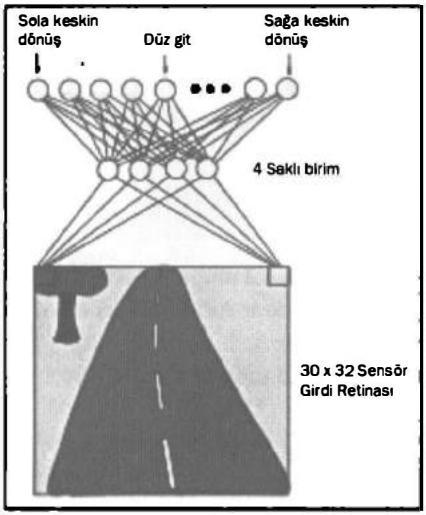
rakmak üzere sürüş denetimini ele alabilir."[39] Resim 29.8'de Navlab'ın bir fotoğrafı gösteriliyor.

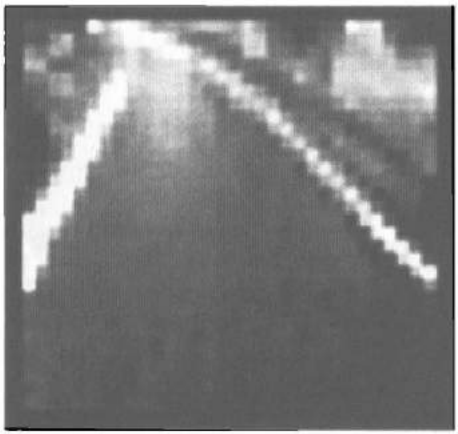
ALVINN'in sinir ağına gelen girdi, karavanın tepesine yerleştirilmiş video kameranın ürettiği, düşük çözünürlüklü 30 × 32 dizilişli gri ölçekli imge yoğunluğu değerleriydi. Bu 960 girdinin her biri, ayarlanabilir ağırlıklar aracılığıyla dört saklı birimin her birine gidiyordu. Ardından saklı birimler, ayarlanabilir ağırlıklar aracılığıyla soldan sağa dizilmiş 30 çıktı birimine bağlanıyordu. Çıktı birimleri, karavanın direksiyon mekanizmasını şu şekilde denetliyordu:[40]

> Merkezdeki çıktı birimi, "düz git" koşulunu temsil ederken, merkezin solunda ve sağında kalan birimler sırasıyla sola ve sağa keskin dönüşleri temsil ediyordu. Çıktı vektörünün en solundaki ve en sağındaki birimler, sırasıyla sola ve sağa 20 metre yarıçaplık dönüşleri ve arada kalan birimlerse, ortadaki "düz git" birimine yaklaşırken sapması doğrusal olarak azalan dönüşleri temsil ediyordu...
>
> Ağın belirlediği direksiyon doğrultusu, en yüksek etkinlik düzeyi sergileyen çıktı biriminin etrafındaki etkinlik "tepesi" yığınının merkezi olarak alınır. Direksiyon doğrultusunu belirlerken en etkin birim yerine kitlesel

39 Charles Thorpe ve diğ., "Vision and Navigation for the Carnegie Mellon Navlab", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, c. 10, no: 3, Mayıs 1988, s. 362-373.

40 Dean A. Pomerleau, "Neural Network Vision for Robot Driving", *The Handbook of Brain Theory and Neural Networks* içinde, Michael Arbib (ed.), Cambridge, MA: MIT Press, 1995. Bu makalenin bir versiyonu şu internet adresinde: http://www.ri.cmu.edu/pub_files/pub2/pomerleau_dean_1995_1/pomerleau_dean_1995_1.pdf.





Resim 29.9. ALVINN ağı (solda) ve tipik yol imgesi (sağda) (Dean A. Pomerleau, "Neural Network Vision for Robot Driving", *The Handbook of Brain Theory and Neural Networks* içinde, Michael Arbib (ed.), Cambridge, MA: MIT Press, 1995. Bu makalenin bir versiyonu şu internet adresinde: http://www.ri.cmu.edu/pub_files/pub2/pomerleau_dean_1995_l/pomerleau_dean_1995_l.pdf.)

etkinliğin merkezinden faydalanmak, direksiyonun daha küçük düzeltmelerine olanak tanır, böylece ALVINN'in sürüş hassasiyetini artırır.

Resim 29.9, ağ düzenini ve ağa sunulan tipik düşük çözünürlüklü yol imgesini gösteriyor.

ALVINN'in çeşitli sürümleri vardı. Bunlardan biri, ağı "seyir sırasında" eğitiyordu, yani ağ, insan şoför karavanı çeşitli yollarda sürerken gerçek zamanlı olarak eğitiliyordu. İstenen direksiyon açısı, şoförün tercih ettiği açı oluyordu ve ağ ağırlıkları, şoförün yaptıklarını taklit etmek üzere geri yayılım tekniği tarafından ayarlanıyordu. Bu yöntemin sorunlarından biri, ağın asla olası "yoldan çıkma" görüntülerine maruz kalmayışıydı. Bu gibi görüntülerin benzetimleri (o durumlarda direksiyonun açısının ne olması gerektiği belirtilerek), eğitim setine ekleniyordu.

Pomerlau, ALVINN'in tipik bir performans testini özetlerken şöyle yazmıştı:[41]

> Ağın, 100 metrelik test yolunda saatte 5 mil hızla karavanı sürdüğü üç sürüşte, taşıtın ortalama konumu, 7,2 santimetrelik standart sapmayla

41 *A.g.e.*

birlikte, merkezin 1,6 santimetre sağında kalmıştı. İnsan denetimi altındaysa, taşıtın ortalama konumu, 5,47 santimetrelik standart sapmayla birlikte merkezin 4 santimetre sağındaydı.

Carnegie Mellon Robotik Enstitüsü, özerk taşıtlar üzerinde çalışmayı sürdürdü (hâlâ sürdürüyor), gerçi imge güdümlü direksiyon idaresine yönelik sinir ağı yaklaşımının yerini daha gürbüz bilgisayarla görme algoritmaları almıştır. Bunların 1995 tarihli görsel algı sistemi RALPH (Rapidly Adapting Lateral Position Handler), yol virajlarını belirlemek için özel görüntü-işleme altprogramlarından faydalanıyordu. Pomerleau'ya göre,[42] "RALPH, yolun yerini belirliyor ve çok farklı koşullar altında değişik yol türlerinde kendi kendine direksiyonu idare edebiliyordu. RALPH, Navlab 5 test taşıtımızı tek şeritlik bisiklet yollarından, kır yollarına, eyaletler arası yollara kadar farklı farklı yollarda en az 3.000 mil sürmüştür."

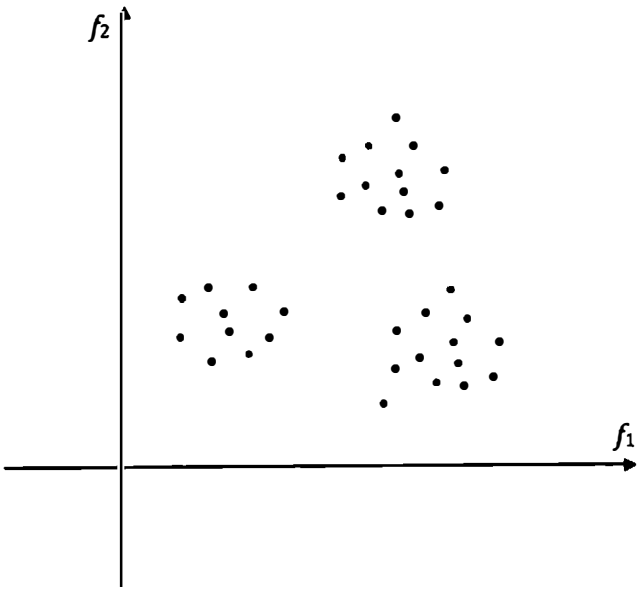
1995 yazında, özel donatılmış taşıtlarından biri, Delco Electronics'in bağışladığı 1990 model Pontiac Trans Sport (Navlab 5), Pittsburgh'dan (PA) San Diego'ya (CA) uzanan 2.849 milin 2.797 milinde kendi başına direksiyon sallamıştı (RALPH'ı kullanarak). (Sadece direksiyon idaresi özerkti, yoksa gazı ve freni Pomerlau ile doktora öğrencisi Todd Jochem idare etmişlerdi.) Taşıtın ortalama hızı, saatte 60 mili aşmıştı.[43]

29.5 GÖZETİMSİZ ÖĞRENME

Bu bölümde şimdiye dek betimlenmiş olan karar ağacı ve sinir ağı öğrenme yöntemleri, "gözetimli öğrenme" örnekleridir; bu öğrenme türünde, sınıflandırmaları bilinen büyük bir öğrenme veri kümesinden yola çıkılarak veri sınıflandırma öğrenilir. Bu sistemlerde öğrenmeyi yönlendiren "gözetim", öğrenme kümesindeki *her* verinin sınıflandırılması hakkında sistemi bilgilendirmeyi içerir. Yine de, salt verileri temel alan faydalı sınıflandırmalar oluşturmak zaman zaman olasıdır. Bu işi yapmanın teknikleri, "gözetimsiz öğrenme" başlığı altına girer.

42 Dean Pomerleau, "RALPH: Rapidly Adapting Lateral Position Handler", *Proceedings of the IEEE Symposium on Intelligent Vehicles*, Eylül 1995, s. 506-511. Bkz. http://www.ri.cmu.edu/pub_files/pub2/pomerleau-dean_1995_2/pomerleau_dean_1995_2.pdf.

43 "no: Hands Across America" internet sitesi şu adreste: http://www.cs.cmu.edu/afs/cs/usr/tjochem/www/nhaa/nhaa_home_page.html. Bu sitede, yolculuğun güncesi ve fotoğrafları da var. Daha yeni çalışmalar için NavLab'ın internet sitesine bakınız: http://www.ri.cmu.edu/labs/lab_28.html.



Resim 29.10. Özellikler uzayında etiketlenmemiş noktalar.

Bölüm 4.3'te, sınıflandırılacak verilerin, iki boyutlu bir "özellikler uzamında"ki noktalarla temsil edildiği bir diyagram göstermiştim (Resim 4.11). Noktaların koordinatları, verilerin sayısal değer verilmiş iki özelliğin, f_1'in ve f_2'nin değerlerine karşılık gelir. Resim 4.11'de, her noktanın sınıfı, bir sınıfa ait olan noktalar için küçük karelerle ve öteki sınıfa ait olan noktalar için küçük dairelerle gösterilmişti. Noktalar bu şekilde etiketlendiği için, gözetimli bir öğrenme yöntemi için öğrenme kümesi olarak kullanılabilirler.

Ancak, diyelim ki Resim 29.10'da gösterilenler gibi etiketsiz örnek noktalarımız var. Bu tür verilerden bir şeyler öğrenmek mümkün mü? Çıplak gözle incelediğimizde, noktaların üç öbek oluşturduğunu görüyoruz. Belki her öbek, aynı sınıfa girdiği düşünülebilecek noktalar içeriyor. Dolayısıyla, eğer öbekleri ve aralarındaki sınırları saptayacak şekilde veri örneklerini otomatik olarak işleyebilseydik, elimize gözetimsiz bir öğrenme yöntemi geçmiş olurdu.

YZ araştırmacıları, öğrenme örnek kümelerini tanımlamak amacıyla birtakım yöntemlerden faydalanmıştır. Rağbet gören ve açıklaması kolay olan bir tanesi, *k*-merkez denen yöntemdir. Bu yöntem, şu adımların defalarca tekrarlanmasıyla çalışır:

1. Örneklem uzamında rasgele yerlere, belli bir sayıda, örneğin *k* adet "öbek arayıcı" yerleştir.
2. Bu öbek arayıcılarının her biri için, öbür öbek arayıcılara kıyasla buna daha yakın duran alıştırma örneklerini aynı kümede topla.
3. Örnek kümelerinin her birinin ağırlık merkezini ("kütleçekim merkezi") hesapla.
4. Her öbek arayıcısını, karşılık geldiği kümenin ağırlık merkezine yolla.
5. Hiçbir öbek arayıcısının hareket ettirilmesine gerek kalmayana dek bu adımları tekrarla.

Bu işlemin sonunda öbek arayıcıları, öbek ya da ayrı veri kategorisi sayılabilecek alıştırma örneklemlerinin ağırlık merkezinde olacaktır. Artık, alıştırma kümesinde olmayan bir noktayı sınıflandırırken, hangi öbek arayıcısına daha yakın olduğunu hesaplamak yeterlidir. Elbette bu süreç, öbek sayısını yani *k*'yı tahmin edebilmeye bağlıdır. Bu işin yöntemleri genelde, öbek içindeki noktalar, öbekler arasındaki uzaklıklara kıyasla birbirlerine daha yakın olacak biçimde seçilir.

İstatistikçiler ve başkaları, verileri öbeklemek için yöntemler geliştirmiştir ve *k*-merkezli yöntemle ilintili çeşitlemeler buna dahildir. Önde gelen tekniklerden biri olan AutoClass, Peter Cheeseman ve NASA'daki çalışma arkadaşları tarafından geliştirilmişti.[44] AutoClass'la ilgili bir internet sitesine göre:[45]

> AutoClass, gerçek ve ayrık değerli özelliklerin birleşiminin betimlediği örneklerden oluşan bir veritabanını ele alıp, bu veriler içindeki doğal sınıfları otomatik olarak bulur. Kaç sınıfın var olduğunu ve neye benzediklerini bu programa söylemeye gerek yoktur; bu bilgiyi verilerden kendisi çıkarır. Sınıflar, olasılıklar cinsinden betimlenir, böylece bir nesne, farklı sınıflarda kısmi üyeliklere sahip olabilir, üstelik sınıf tanımları örtüşebilir.

44 Peter Cheeseman ve diğ., "AutoClass: A Bayesian Classification System", *Proceedings of the Fifth International Conference on Machine Learning*, San Francisco: Morgan Kaufmann Publishers, 1988, s. 54-64. Ayrıca bkz. Peter Cheeseman ve J. Stutz, "Bayesian Classification (AutoClass): Theory and Results", *Advances in Knowledge Discovery and Data Mining* içinde, Usama M. Fayyad ve diğ. (ed.), Menlo Park, CA: AAAI Press and Cambridge, MA: MIT Press, 1996. Şu internet adresinde: http://ti.arc.nasa.gov/m/project/autoclass/kdd-95.ps.

45 http://ti.arc.nasa.gov/project/autoclass/.

AutoClass, yeni bir kızılötesi yıldız sınıfı keşfetmiş olmasıyla ünlüdür. Ayrıca, yeni protein, intron sınıfları ve DNA/protein dizileri verilerinde yeni örüntüler de keşfetmiştir.

Sayısal olmayan verilere uygulanabilecek teknikler bile vardır. İstatistikçiler tüm bu (sayısal ve sayısal olmayan) yöntemleri, "öbek çözümlemesi" adıyla genel bir başlık altında toplar. Bunun iyi bir incelemesi, çevrimiçi elektronik ders kitabı *StatSoft*'ta, http://www.statsoft.com/textbook/stcluan.html adresinde bulunabilir. Ayrıca Duda, Hart ve Stork'un birlikte yazdıkları ders kitabı, gözetimsiz öğrenmeyi (üstelik veri sınıflandırmada başka konu başlıklarını da) bütünüyle irdeliyor.[46]

29.6 PEKİŞTİRMELİ ÖĞRENME

29.6.1 Optimal Politikaların Öğrenilmesi

Gözetimli ve gözetimsiz çeşitlerin arasında bir yerde duran başka bir öğrenme tarzı daha vardır. Bir robotun, sadece eyleminin sonucu belirtildiğinde, süregiden bir dizi deneyimin her aşamasında olası eylemlerden hangilerinin gerçekleşmesi gerektiğini öğrenmesi örnek olarak gösterilebilir. Oyunun sonunda sadece kazanıldığı ya da kaybedildiği bilgisiyle yola çıkarak mükemmel satranç oynamayı öğrenmek aşırı uç bir örnek olurdu. Bu şekilde satranç oynamayı öğrenen makine henüz inşa edilmiş değil, fakat bir programın tavla oynamayı ve helikopter uçuşunu denetlemek gibi ilginç görevleri yerine getirmeyi bu şekilde öğrenmesi mümkün. Psikolojideki öğrenme kuramından terim ödünç alacak olursak, galibiyet ya da mağlubiyet bilgisine (ya da genel olarak, iyi sonuç ve kötü sonuç bilgisine) "ödül" ya da "pekiştirme" diyebiliriz ve bu tür öğrenmeye "pekiştirmeli öğrenme" veya (bazen) "deneme yanılmayla öğrenme" denir.

Pekiştirmeli öğrenmenin uzun ve değişik bir tarihi var. Psikolog Edward L. Thorndike (1874-1949), bu tarz öğrenmeyi hayvanlarda araştırmıştı.[47] Bu alanın öncülerinden Richard S. Sutton (1957- ; Resim 29.11) ile Andrew G. Barto'nun (1948; Resim 29.11) birlikte yazdıkları *Reinforcement Learning: An Introduction* [Pekiştirmeli Öğrenme: Sunuş] başlıklı kitaplarında,[48] ek tarihsel kilometre taşların-

46 Richard O. Duda, Peter E. Hart ve David G. Stork, *Pattern Classification,* New York: John Wiley and Sons, Inc., 2001.

47 Edward L. Thorndike, *Animal Intelligence,* New York: The Macmillan Co., 1911.

48 Richard S. Sutton ve Andrew G. Barto, *Reinforcement Learning: An Introduction,*



Resim 29.11. Andrew Barto (solda) ve Richard Sutton (sağda). (Fotoğraflar, Andrew Barto ile Richard Sutton'ın izniyle.)

dan bahsederler, örneğin Arthur Samuel'in damadaki değerlendirme fonksiyonlarını öğrenmeye yönelik çalışmaları, optimal denetimde Richard Bellman'ın dinamik programlama tekniklerinin kullanımı, John Andreae'nin deneme ve yanılma öğrenme sistemi STeLLA,[49] Donald Michie'nin üçtaş (MENACE[50]) ve denge (BOXES[51]) oyunları öğrenme sistemleri ve Harry Klopf'un "hedonist sinir hücreleri"[52] üzerine çalışması gibi. Pekiştirmeli öğrenme, YZ'nin son derece teknik ve çok dallı hale gelmiş alt disiplinlerinden biridir. Bunun nasıl işlediğine dair matematiksel olmayan hafif bir açıklama yapacağım.

Basit bir şekilde anlatmak gerekirse, pekiştirmeli öğrenme, durumdan duruma geçerek, ödülün elde edileceği duruma ulaşmayı öğrenmektir. Bu problem, labirentte yolunu bulmaya çalışan sıçanların karşılaştığı (ya da bir görevi yerine getirmeyi öğrenen robotun karşılaştığı) soruna benziyor. Aslında, pekiştirmeli öğrenmenin kimi yönlerini

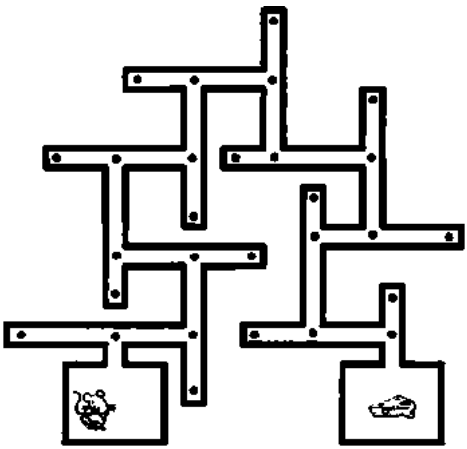
Section 1.6, Cambridge, MA: MIT Press, 1998; şu internet adresinde: http://www.cs.ualberta.ca/~sutton/book/ebook/the-book.html.

49 John H. Andreae, "STeLLA: A Scheme for a Learning Machine", *Proceedings of the 2nd IF AC Congress*, 1963. Şu yayında: *Automation and Remote Control*, Londra: Butterworths, 1964.

50 Donald Michie, "Experiments on the Mechanisation of Game Learning: 1. Characterization of the Model and its Parameters", *Computer Journal*, c. 1, 1963, s. 232-263.

51 Donald Michie ve R. Chambers, "BOXES: An Experiment in Adaptive Control", *Machine Intelligence 2* içinde, E. Dale ve Donald Michie (ed.), Edinburgh: Oliver and Boyd, 1968, s. 137-152.

52 A. Harry Klopf, *The Hedonistic Neuron: A Theory of Memory, Learning, and Intelligence*, Washington, DC: Hemisphere, 1982.



Resim 29.12. Labirent

betimlemek amacıyla labirent örneğine göz atalım. Resim 29.12'de tipik bir labirent gösteriliyor.

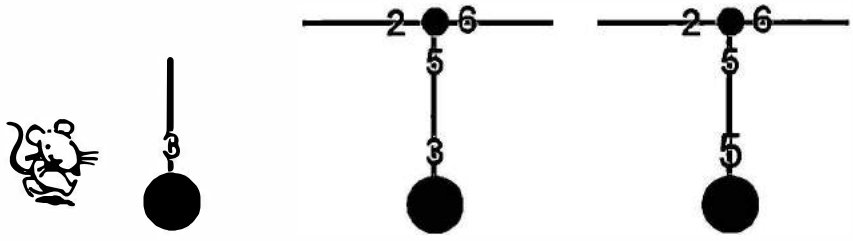
Sıçanın problemi, başlangıç noktasından yola çıkarak, hedef noktasındaki peynire ulaşmaktır. Resimdeki gri noktalar, sıçanın içinde bulunduğu durumu fark edebileceği yerler anlamına gelir. Pekiştirmeli öğrenme terminolojisinde bunlara "durum" denir. Durumların her birinde sıçan, dört eylem arasından seçim yapar, yani ister sağa, ister sola döner, ister ileri ister geri gider. Duruma bağlı olarak, bu eylemlerin ancak bir kısmı mümkündür; örneğin çıkmaza saplandığında artık öne gitmek olmaz. Olası her eylem, sıçanı, labirentteki bir durumdan komşu duruma taşır. Bu durumlar ve onları birbirine bağlayan eylemler birlikte, arama yöntemlerinden bahsederken anlattığıma benzer bir grafik olarak düşünülebilir.

Labirentlerde koşturan gerçek sıçanlardan fazla sapmamak adına, kurmaca bir "robot sıçan"ın, labirenti aşmayı nasıl öğrenebileceğini betimleyelim. Robotun başlıca sorunu, başlarken elinde labirentin haritası ya da eylemlerinin etkileri hakkında bir fikrinin olmayışıdır. Yani, içinde bulunduğu herhangi bir durumda, burada kalkışacağı çeşitli eylemlerin bir sonraki aşamada hangi durumları doğuracağını bilmez. Çünkü elinde, diyelim ki grafikle temsil edilen bu tür bir harita olsaydı, hedef düğüm noktasına ulaşan bir yolu bulmak amacıyla grafiği araştırabilirdi (A* gibi bir yöntem kullanarak). İlerlemenin yollarından biri, deneme yanılma yöntemleri aracılığıyla durumlar grafiğini ve bunların bağlantılarını öğrenip, labirentte yol bulmak için grafik arama yöntemleri kullanmaktır.

Çoğu pekiştirmeli öğrenme yönteminin faydalandığı, buna seçenek olan bir yaklaşım, hedefi ararken robotun rasgele yürüyüşü sırasında karşılaştığı tüm durumları isimlendirmektir. (Robotun sonunda hedefe ulaştığını varsayıyoruz.) Pekiştirmeli öğrenme terminolojisinde, labirentte ilerleme "politikası", tek tek eylemleri, isimlendirilmiş her durumla ilintilendirir. En iyi ya da "optimal politika", labirentteki en kısa (ya da en masrafsız) yolu doğuran eylemi her durumla ilintilendirir. Pekiştirmeli öğrenme, en iyi politikayı ya da en azından iyi politikaları öğrenmekle ilgilidir.

Politika öğrenmenin yöntemlerinden biri, bir "değer" sayısını her durumdaki olası her eylemle ilişkilendirmek, böylece bu sayılara, hedefe giden yolu işaret edene dek ayarlamaktır (deneyimleri temel alarak). Bu yönteme "Q-öğrenme" denir ve ilk olarak Christopher Watkins (1959-) tarafından Cambridge Üniversitesi'nde yaptığı tezinde ileri sürülmüştür.[53] Robot, öğrenme sürecinde önce, başladığı duruma bir isim verir ve bu durumdaki olası her eyleme, rasgele seçilmiş değerler atar. Öğrenme süreci, karşılaştığı her *yeni* durumun alabileceği tüm eylemlere isimler ve değer atayarak bu tablonun kapsamını genişletir. (Robotun, kendi tablosunda, öğrenme sürecinde ziyaret ettiği tüm durumların adlarını hatırladığını ve bunları yeni durumlardan ayırt edebileceğini varsayıyoruz.) Robotun başlangıç durumu, sadece olası eylemlerine atanmış ve rasgele seçilmiş değerlerle birlikte Resim 29.13'te sol tarafaki çizimde gösteriliyor. Öğrenme sürecinin her aşamasında robot, en yüksek değeri taşıyan eylemi gerçekleştirir. Robotun başlangıç durumunda sadece tek eylem olduğu için, bu eylemi gerçekleştirir, kendini yeni bir durumda bulur ve bu yeni durumdaki olası eylemlere rasgele değerler biçer. Bu adım, Resim 29.13'ün ortasındaki çizimde gösteriliyor. Şimdi de öğrenmede kilit adım geliyor. Robot, en yüksek değerin 6 olduğu eylemler barındıran yeni bir duruma ulaştığını "bildiği" için, onu bu duruma ulaştıran eylemin değer rakamını, yani 3'ü güncelleyerek, 6 değerinde olduğunu hayal ettiği bir eylemi gerçekleştirmekle daha tutarlı bir rakama ayarlar bunu. Henüz tamamladığı eylemin "bedelinin" hesabını vermek için, 3 rakamının ayarlanışı 6'ya kadar uzanmaz, örneğin 5'e varır. Sonuç, Resim 29.13'te sağdaki çizimde

53 Christopher J. C. H. Watkins, "Learning from Delayed Rewards", doktora tezi, Cambridge Üniversitesi, Cambridge, İngiltere, 1989.



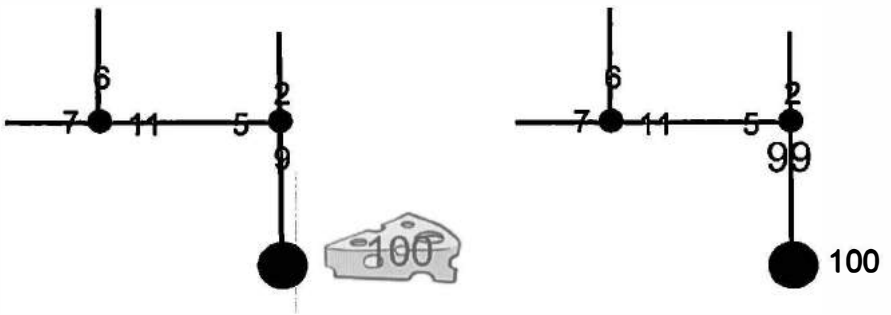
Resim 29.13. Q-öğrenme sürecinin başlangıç aşamaları.

gösteriliyor. Bu çizimde, ayarlanarak biçilmiş değer, öbür rakamlardan daha büyük ve gölgeli gösteriliyor.

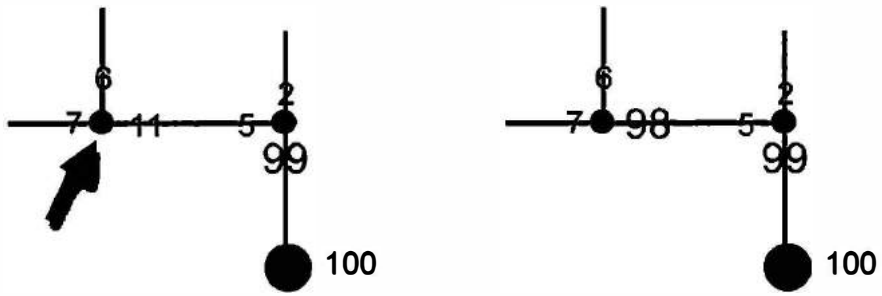
Bu işlem böyle sürer gider. Robot, her durumda, değer rakamı en yüksek olan eylemi gerçekleştirir, sonra da bu değer rakamını ayarlayarak henüz girilmiş durumdaki en yüksek değer rakamına sahip eylemin değerine yakın bir değer verir. Üstelik bu işlem rasgele seçilmiş değer biçme rakamlarıyla başlamış olsa bile, sonuçta deneme yanılma süreci hedef duruma denk gelip yüksek "ödüle" ulaşacaktır. O safhada, ödüle götüren son eylemin değer rakamı, ödülün değeriyle aynı düzeye (ya da biraz daha azına) çekilecektir. Bu adımı Resim 29.14'te gösteriyorum. Resmin sol tarafındaki çizimde, robot, hedefe ulaşan eylemi gerçekleştirdiğinde, durumlardan bazıları ve eylem değerleri gösteriliyor. Resmin sağ tarafındaki çizimde, hedefe götüren eylemin ayarlanmış değerini (gölgeli) gösteriyorum. Şimdi, ilk defa bir eyleme değer biçmek, rasgele belirlenmek yerine ödül almaya dayanıyor. Eğer robot, yine hedef durumun komşusuna denk gelirse, kesinlikle aynı eylemi gerçekleştirir. Daha da önemlisi, izleyen deneyimde, sondan bir önceki duruma ulaştığında, bu ödül tabanlı değeri geri yayacaktır.

Geri yayılımın nasıl işlediğini Resim 29.15'te açıklıyorum. Diyelim ki soldaki çizimde, bir şekilde robot, okla işaretlenmiş duruma varmış olsun. Bu durumda, en yüksek değeri taşıyan eylemi gerçekleştirir ve hedefe komşu olan duruma ulaşır. Bu durumdan dışarı uzanan eylemlerin en yüksek değeri 99'dur, dolayısıyla henüz gerçekleştirdiği eylemin değerini, sağ taraftaki çizimde gösterildiği gibi 11'den 98'e değiştirir. Hedefe yakın durumlardan eylemlerin değerini geri yayılımla yükseltmek, aslında, o durumları, sanki bizzat hedef durummuş gibi, kendinden "ödüllendirici" kılar.

Dikkatli okurlar, "rasgele" değer biçme rakamlarını işime geldiği gibi ayarladığımdan, böylece robot hedefe yakın durumlara vardığında hemen hedefe ulaşabildiğinden yakınacaktır. Peki, eğer bu değer-



Resim 29.14. Hedefe götüren aşamalar.



Resim 29.15. Hedefe ulaşan eylemlerin değerlerinin geri yayılımı.

ler, robot hedefe yaklaştığında yine uzaklaşmasını sağlayacak şekilde olsaydı, ki büyük olasılıkla öyle olurlardı, o zaman ne olurdu? Eğer değerler benim tarif ettiğim gibi ayarlanmışsa, daima hamlelerin maliyetini hesaba katıyorsa, en sonunda bu rakamların robotu hedefe doğru gitmeye zorlayacağını, öteki tüm yolların kapanacağını biraz düşünerek anlayabiliriz.

Tekrar tekrar denendiğinde, hedefe giden yolla ilgili eylemlerin değerleri yavaş yavaş hedeften geri yayılacaktır. En sonunda, bol bol deneme yanılma deneyiminden sonra (ve birtakım "makul" varsayımlarla birlikte), değerler bir araya toplanıp optimal politikayı yürürlüğe sokacaktır, yani robotu daima hedefe en verimli yolla getiren politika oluşturulacaktır.

Pekiştirmeli öğrenmenin çoğu sürümüne şu incelikler katılmıştır:

- Birden fazla durumda ödül verilebilir. Yani, sadece tek hedef durum olması şart değil, pek çok durum ödüle katkıda bulunabilir. Ödüller, sayısal değerlerle temsil edilir ve bu rakamlar artı (gerçek "ödül"), sıfır ya da eksi ("ceza") değerli olabilir.

- Tek hedef duruma giden optimal yola karşılık gelen politikayı bulmaya çalışmak yerine, ödül miktarını zamanla en yükseğe çıkaran politikalar öğrenilmeye çabalanır. Çoğunlukla, bir politika öğrenilirken, uzak gelecekte beklenen ödüllerin değeri "iskonto" edilir, yani daha çabuk beklenen ödüller kadar hesaba katılmazlar.
- Bir durumda gerçekleştirilen eylem, robotu her zaman aynı duruma götürmeyebilir. Kişi, bir durumda belirli eylemlerin robotu belli durumlara götürme olasılıklarını ve "öncelikli tarama"[54] gibi bazı pekiştirmeli öğrenme yöntemlerinin bu işi yaptığını öğrenmeye çalışabilir. Q-öğrenme işlemi, bu olasılıkları öğrenme ihtiyacından açıkça kaçınır, çünkü, bunlar her neyse (ödüllerle birlikte), öğrenme sürecinin, durum-eylem çiftlerine atadığı değerleri uygun şekilde etkiler.
- Ek bir zorluk olarak, duyu aygıtının yeterince doğru ya da bilgilendirici olmayışından ötürü, robotun hangi durumda bulunduğuna dair bilgisi kusurlu olabilir. Bu durumda, robotun içinde bulunduğu asıl durumun, robottan "saklı" olduğu söylenir ve bu durum, optimal politika öğrenme sorununu daha da zorlaştırır.

Bu inceliklerle birlikte problem, "Markov Karar Süreci" (MKS) problemi durumuna gelir. Durumlarla ilgili bilgi eksikliyse, "Kısmen Gözlemlenir Markov Karar Süreci" (KGMKS) denir. Hem kontrol kuramı hem de YZ araştırmacıları MKS'ler ve KGMKS'ler üzerinde iyice çalışmıştır.[55]

Kullanışlı uygulamalarda pekiştirmeli öğrenme kullanımı için önemli olan birkaç noktadan bahsetmek amacıyla robot-labirent örneğinden faydalanabilirim. Öncelikle, robotun rasgele keşif yaparak en sonunda hedef duruma ulaşacağını varsaydım. Karmaşık prob-

54 Andrew Moore ve Christopher G. Atkeson, "Prioritized Sweeping: Reinforcement Learning with Less Data and Less Real Time", *Machine Learning*, c. 13, Ekim 1993. Şu internet adresinde: http://www.ri.cmu.edu/pub_files/publ/moore_andrew_1993_l/moore_andrew_1993_l.pdf.

55 Daha fazla bilgi için örneğin aşağıdakilere bakınız: Richard S. Sutton ve Andrew G. Barto, *Reinforcement Learning: An Introduction*, Cambridge. MA: MIT Press, 1998 (kitabın html versiyonu şu internet adresinde: hrtp://www.cs.ualbero.ca/~sutton/book/ebook/the-book.html.) ve Leslie P Kaelbing, Michael L Liftman ve Andrew W. Moore, "Reinforcement Learning: A Survey", *Journal of Artificial Intelligence Research*, c. 4, 1996, s. 237-285. Pek çok makale ve gösterim sunan internet sayfası şu adreste: http://rlai.cs.ualberta.ca/RLAI/rlai html.

lemlerde bir hedefe (ya da başka ödüllere) rasgele ulaşma şansı ya cılızdır ya da sıfırdır. Problemi, ödüle ulaşmanın daha kolay olduğu alt problemlere ayrıştırmak bazen öğrenme sürecini hızlandırır. Buna ek olarak, robotu en başta, hedefe rasgele ulaşabileceği kadar yakın bir noktaya bırakan "biçimlendirici" stratejiler kullanılabilir. Ardından, hedefe yakın eylemlere hedefle ilintili değerler biçildikten sonra, başlangıç durumları hedeften gitgide uzağa çekilebilir. Bir seçenek de, robota belki ara ödül biçiminde ipuçları verilmesidir, böylece robot şimdilik iyi gittiğini bilir. Bu gibi stratejiler, hayvanlara ve insanlara belli beceriler öğretilirken kullanılıyor.

Başka bir problem de, zaten öğrenilmiş bir politikadan "faydalanma" ile daha iyi politikalar "keşfetme" arasında seçim yapmaktır. Öğrenme sürecinin başında elde edilmiş eylem değerleri, çoğunlukla olası en iyi değerler kümesi değildir. Daha iyi bir değerler kümesi öğrenmesi için, robotun, bilinen politikadan rasgele ayrılmaya ve daha iyi bir stratejiye kilitlenmeye bir şekilde yüreklendirilmesi gerekir. Son olarak, pek çok problemin "durum uzayı" öyle büyük olabilir ki tüm durumlar ile bunların eylemleri ve değerleri, robot labirenti problemine varsaydığıma benzer bir tabloda sıralanamaz. Bu durumda, belli bir durumda gerçekleştirilebilecek eylemlerin değerleri saklamak yerine hesaplanır. Bunun nasıl yapıldığına dair bir örneği önümüzdeki birkaç sayfada gösteriyorum.

29.6.2 TD-GAMMON

Yapay öğrenme yöntemlerinin gücünün en etkileyici gösterimlerinden biri, IBM'de Gerald Tesauro'nun geliştirdiği TD-GAMMON sistemidir.[56] TD-GAMMON sürümleri, kendi kendilerine oynadıkları milyonlarca oyun sayesinde mükemmel tavla oynamayı öğrenmiştir. TD-GAMMON, sinir ağlarında öğrenme ile "zamansal farklılık öğrenmesi" (TD ön eki buradan geliyor) denilen bir tür pekiştirmeli öğrenmenin birleşimini kullanıyordu.

TD-GAMMON'un sinir ağı üç katmandan meydana geliyordu. Bu programın sürümlerinden birinde, 198 girdi birimi, 40 saklı birim ve 4 çıktı birimi vardır. Çıktı birimlerinden her biri, 0 ile 1 arasında bir çıktı değeri alabilir. (Ağda, eşik birimleri yerine, daha önce bahsettiğim türde bileşenler var, yani çıktıları 0 ile 1 arasında düzenli olarak ama

56 Gerald Tesauro, "Temporal Difference Learning and TD-GAMMON", *Communications of the ACM*, c. 38, no: 3, Mart 1995. Makalenin html versiyonu şu internet adresinde: http://www.research.ibm.com/massive/tdl.html.

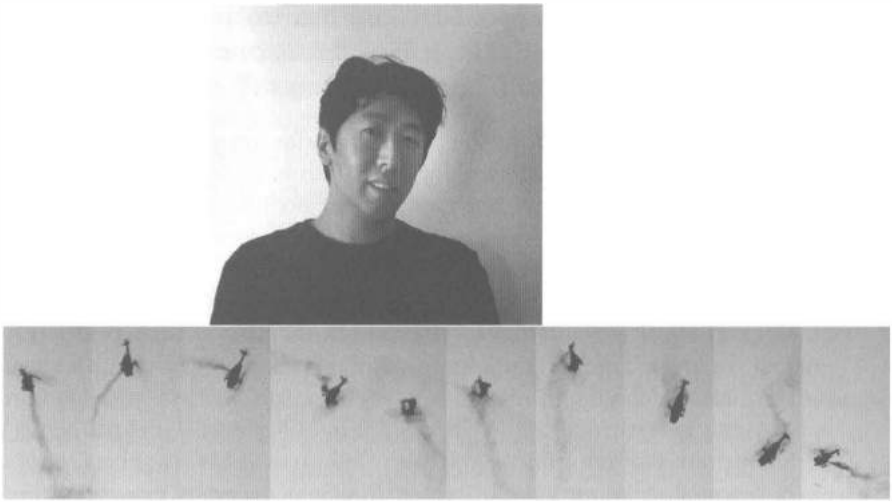
yine de ani şekilde değişen bileşenler.) Çıktıların her biri, oyundan belirli bir sonucun olasılığını tahmin etme görevini üstlenmişti. Dört olası sonuç şunlardı: beyazlar kazanır, beyazlar mars yapar, ya da siyahlar kazanır, siyahlar mars yapar. Girdi birimleri, tahtada taşların dizilişini temsil etmek üzere kodlanmıştı. Dört çıktının değerleri, beyaz taşların bakış açısından bir tahta konumunun tahmini "değerini" veren bir rakam ortaya çıkarır.

Ağın nasıl öğrendiğini az sonra betimleyeceğim. Öncelikle, bir hamleyi seçerken ağdan nasıl yararlanıldığına bakalım. (Okurun tavla oyununa aşina olduğunu varsayıyorum ama aşina değilse bile betimlemem ona anlaşılır gelecektir.) Oyunun her aşamasında zarlar atılır ve program, gelen zara göre olası tüm hamleleri gözden geçirir. Ağ, zarla oluşan olası her pul dizilişinin değerini hesaplar ve program, en iyi değeri barındıran pul dizilişine yol açacak hamleyi seçer (hamle sırası beyazdayken en yüksek değer, hamle sırası siyahtayken en düşük değer).

Şimdi ağın nasıl öğrendiğini görelim: Gerçek oyun sırasında karşı karşıya gelinen her pul dizilişi için, geri yayılım kullanılarak ağın ağırlıkları ayarlanır, böylece o pul dizilişi için hesaplanan değer, bir sonraki geçici pul dizilişi için hesaplanan değere yakınlaşır ("zamansal farklılık" teriminin buradan doğduğunu böylece anlıyoruz). Ağ, rasgele seçilen ağırlık değerleriyle işe koyulur, dolayısıyla öğrenme sürecindeki ilk hamleler ve ağırlık ayarlamaları rasgeledir. Fakat en sonunda, rasgele seçilmiş hamleler bile oyunculardan birinin galibiyetiyle sonuçlanır. Oyunculardan biri kazanınca, dört olasılık değeri kesin biliniyor demektir; bunlarından bir tanesi "1", öbürleri "0"dır. Böylece ağın ağırlıkları öyle ayarlanabilir ki sondan bir önceki pul dizilişinin değeri, kazanan son pul dizilişininkine daha yakın kılınır. Tüm pekiştirmeli öğrenme yöntemlerindeki gibi, değerler, oyun sonundan başlangıç konumuna kadar yavaş yavaş geri yayılır. Milyonlarca oyundan sonra, ağın ağırlıklarının aldığı değerler usta düzeyinde tavla oynamayı sağlar. Sutton ve Barto, öğrenmeye ek olarak arama da kullanan bir TD-GAMMON sürümü hakkında yorum yaparken şunu yazmışlardır:[57]

> TD-GAMMON 3.0, dünyadaki en iyi insan oyuncuların oyun gücünde ya da buna yakın görünüyor. Şimdiden dünya şampiyonu olabilir. Bu programlar, en iyi insan oyuncuların oynama tarzını çoktan değiştirdi. Örneğin TD-GAMMON, belirli açılış hamlelerini, en iyi insan oyuncuların geleneksel hamlelerinden farklı oynamayı öğrenmiştir. TD-GAMMON'un

57 Alıntı, pekiştirmeli öğrenme hakkındaki kitaplarının html versiyonundan: http://www.cs.ualberta.ca/~sutton/book/11/node2.html.



Resim 29.16. Andrew Ng (üstte) ve yuvarlanma manevrası yapan model helikopteri (altta). (Fotoğraflar, Andrew Ng'nin izniyle.)

başarısını ve ek çözümlemeleri temel alan en iyi tavla oyuncuları, artık bu hamleleri TD-GAMMON gibi oynuyor.

29.6.3 Başka Uygulamalar

Büyük olasılıkla, pekiştirmeli öğrenme yönteminin yüzlerce önemli uygulaması vardır. Tipik ve dramatik örneklerden biri, Andrew Ng (1976-) ve Stanford'daki ekibinin, hava akrobatik helikopter manevralarını öğrenme üzerine çalışmalarıdır.[58] "Yuvarlanmayı" öğrenen model helikopterin bazı fotoğrafları Resim 29.16'da gösteriliyor. Bu yöntem, asansör hareketi, iş-dükkân çizelgesi, enerji tüketimi idaresi ve dört ayaklı yürüyen robot alanlarında da uygulanmıştır.

Pekiştirmeli öğrenme hakkında son bir yorum olarak, yapay öğrenme teknolojisinin ismini psikolojiden almış bir parçasının, hayvan beyninin öğrenme tarzına sinir fizyolojisi düzeyinde kuramsal bir çer-

58 Pieter Abbeel ve diğ., "An Application of Reinforcement Learning to Aerobatic Helicopter Flight", *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference* içinde, Bernhard Scholkopf, John Platt ve Thomas Hofmann (ed.) Cambridge, MA: MIT Press, 2007, s. 1-8. Pdf versiyonu için bkz. http://www.cs.stanford.edu/~ang/papers/nips06-aerobatichelicopter.pdf. Videolar için bkz. http://www.cs.stanford.edu/group/helicopter. Yuvarlanma videosu için bkz. http://www.cs.stanford.edu/group/helicopter/video/rolls_080130_web960.mp4.

çeve temin ederek borcunu geri ödediğini belirtmek ilginç olacaktır. Christopher H. Donahue ve Hyojung Seo, *The Journal of Neuroscience* adlı dergide çıkmış makalelerinde şöyle yazmıştır:[59]

> Belirsiz ortamlarda yön bulurken etkin kararlar vermek amacıyla hayvanlar, eylemlerinin sonuçlarını doğru tahmin etme becerisi geliştirmek zorunda. Pekiştirmeli öğrenme, hayvanların bu hüneri nasıl edindiğini anlamada önemli bir kuramsal paradigma olarak ortaya çıkmıştır...
>
> Hayvanın yeğlediği davranışı başarıyla öngörmeye ek olarak, pekiştirmeli öğrenme modeli, hedef güdümlü davranışlarda bazal sinir düğümlerinin işlevini açıklamada başarıyla kullanılmıştır. Ön tavan bölgesinde ve siyah maddede dopamin sinir hücrelerinin, bir ödül-öngörü hatasını kodladığı gösterilmiştir; bu, hayvanın gelecekteki seçimlerinin sonuçlarını iyileştirmek amacıyla kullanılır. Bir görevde serbest seçimlerde bulunan maymunlarla gerçekleştirilmiş başka bir araştırma, çizgili cisim (*corpus striatum*) sinir hücreleriyle eylem değerleri arasında ilinti olduğunu göstermiştir, ki bu değerler, her eylemle ilgili sonuç mazisi bütünleştirilerek tahmin edilir.

29.7 GELİŞTİRME

Bahsettiğim birçok yapay öğrenme yöntemi çeşitli yollarla geliştirilebilir. "Hesaplamalı öğrenme kuramı" üzerine çalışan kimileri, bunlardan bazılarını istatistikçilerin ve başkalarının çalışmaları temeline oturtmuştur. "Torbalama" ([*bagging*] bootstrap aggregating) adlı tekniğin babası, Berkeley California Üniversitesi'nden Profesör Leo Breiman'dı.[60] Sınıflandırma problemlerinde, torbalama tekniği, örneğin m sayıda ayrı sınıflandırıcının çıktılarını birleştirerek iş görür. Her sınıflandırıcı, özgün alıştırma kümesinin farklı bir altkümesiyle eğitilir. Bu altkümeler, özgün altkümenin kimi örnekleri rasgele seçilerek (ve geri yerine konarak) elde edilir. (İstatistikçiler bu örnek kümelerine "rasgele seçilmiş örneklem", der.) M sayıda sınıflandırıcının her biri

59 Chrisopher H. Donahue ve Hyojung Seo, "Attaching Values to Actions: Action and Outcome Encoding in the Primate Caudate Nucleus", *The Journal of Neuroscience*, c. 28, no: 18, 30 Nisan 2008, s. 4579-4580. Yazarlar, hem Sutton ile Barto'nun kitabına hem de şu makaleye atıfta bulunuyor: Wolfram Schultz, Peter Dayan ve P. Read Montague, "A Neural Substrate of Prediction and Reward", *Science*, c. 275, no: 5306, 14 Mart 1997, s. 1593-1599.

60 Leo Breiman, "Bagging Predictors", İstatistik Bölümü Teknik Raporu no: 421, Berkeley, California Üniversitesi, Eylül 1994. Şu internet adresinde: http://salford-systems.com/doc/BAGGING_PREDICTORS.pdf; ve Leo Breiman, "Bagging Predictors", *Machine Learning*, c. 24, no: 2, 1996, s. 123-140.

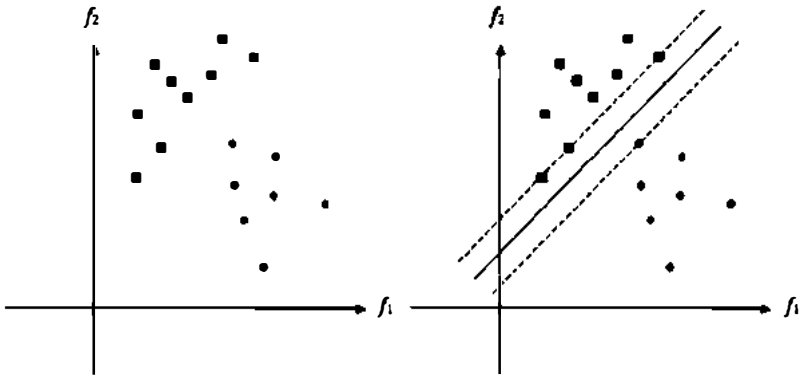
eğitildikten sonra, çoğunluk oyuyla son bir sınıflandırma yapılır. Bu teknik, tek başına kullanılan sınıflandırıcıdan bağımsız olarak uygulanabilir; sinir ağı, karar ağacı, en-yakın-komşu ya da elinizde artık ne varsa. Torbalama tekniği aynı zamanda (sınıf yerine) rakamları örnekle ilintilendirme probleminde de kullanılabilir. Bu durumda çıktıların, oylama yerine ortalamaları alınır. Oylama ve ortalama alma işlemleri, verileri aşırı öğrenmeden kaçınmaya yardımcı olur, dolayısıyla tüm verilerle eğitilen tek sınıflandırıcıya kıyasla daha iyi performans sağlarlar. [1960'ların MADALINE sinir ağının, her eşik birimi rasgele seçilmiş örneklemle eğitilmiş olsa performansının nasıl artacağını insan merak ediyor.]

Bununla ilintili bir fikir, "yükseltme", Robert E. Schapire tarafından önerilmişti.[61] Her ne kadar bunun pek çok sürümü olsa da, çalışma şeklinin ana hatları aşağıdaki gibidir. Gözetimli yapay öğrenme tekniklerinden herhangi birini kullanan sınıflandırıcı, her örneğe eşit "ağırlık verilmiş" özgün bir öğrenme kümesiyle eğitilir. (i numaralı örneğin ağırlığı olan w_i, örneğin öğrenme kümesine o örneği w_i kere eklemekle belirlenebilir.) Ardından, yanlış sınıflandırılmış örneklerin ağırlığının "iteklendiği", doğru sınıflandırılmış örneklerin ağırlığının düşürüldüğü yeni örnek kümeleri oluşturulur. Bu yeni öğrenme kümesi kullanılarak, başka bir sınıflandırıcı eğitilir. (Bu seferki büyük olasılıkla, yanlış sınıflandırılmış eski örnekler üzerinde daha fazla çalışacaktır.) Bu işlem, diyelim ki m sayıda sınıflandırıcı elde edilene dek tekrarlanır. Şimdi, her sınıflandırıcı, yeni örneklerin sınıflandırılmasına oy verir. Bunların oyları, özgün öğrenme kümesindeki performanslarına göre ağırlıklandırılır. Daha güvenilir sınıflandırıcıların oyları, o kadar güvenilir olmayan sınıflandırıcıların oylarına kıyasla daha fazla hesaba katılır. Özgün sınıflandırıcılar "zayıf" (yani güvenilmez) olsa bile, m sayıda sınıflandırıcının birleşik kümesinin toplu doğruluk oranı oldukça iyi olabilir, dolayısıyla sonuçları "iteklemiş" olur.

İteklemeyi gerçekleştirmenin çeşitli yolları önerilmiştir. Yoav Freund ile Robert Schapire'e göre rağbet görenlerden bir tanesi "Adaboost"tur.[62]

61 Robert E. Schapire, "The Strength of Weak Learnability", *Machine Learning*, c. 5, 1990, s. 197-227. Şu internet adresinde: http://www.cs.princeton.edu/~schapire/papers/strengthofweak.pdf.

62 Yoav Freund ve Robert E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting", *Journal of Computer and System Sciences*, c. 55, no: 1, 1997, s. 119-139. Sıkıştırılmış ek versiyonu şu internet adresinde: http://www.cs.princeton.edu/~schapire/papers/FreundSc95.ps.Z.



Resim 29.17. İki boyutlu uzamda noktalar ve doğrusal ayıran sınırlar.

Ayrıca torbalama tekniği ile yükseltmeyi birleştirmek de mümkündür.[63]

Son olarak, "Destek Vektör Makineleri"nden (DVM) bahsedeceğim. Bunların tam bir betimlemesi, burada sunmak istediğimizden daha fazla matematik içerir, fakat geometriden bir örnek vererek nasıl işledikleri hakkında kabaca fikir verebilirim. Özellikler uzamında ayırt edici bir sınırı anlatmak amacıyla, Resim 29.17'nin sol tarafında, Resim 4.11'de kullanmış olduğum aynı noktaları gösteriyorum. Küçük karelerle gösterilen noktalar bir sınıfa ait örnekleme ve küçük dairelerle gösterilen noktalar başka bir kategoriye ait örnekleme karşılık gelir.

Hatırlatmak amacıyla, diyagramdaki noktaların, sınıflandırmak istediğimiz birimlerden (konuşma sesleri, görüntüler ya da başka veriler gibi) hesaplanmış özelliklere, f_1 ve f_2'ye eşit koordinatları olduğunu söyleyelim. Bu örnekte, noktaları iki sınıfa eksiksiz ayıran pek çok düz çizgi (yani doğrusal) sınır vardır. Dolayısıyla, ("alıştırma örneklemi" sayı) noktaları sınıflandırmak üzere bir sinirsel öğeyi eğitme girişimi başarılı olacaktır. Eğer alıştırma için standart hata düzeltme yöntemi kullanırsak, kesinlikle *birtakım* doğrusal sınırlar elde ederiz, fakat iş DYM'lere gelince, sınırdan alıştırma örneklemini ayırmanın ötesinde daha fazlasını isteriz. Bu sınırın öyle olmasını isteriz ki bu sınırdan ("kenar payı" denir), zıt sınıfların en yakın noktalarına ulaşan uzaklıklar olabildiğince en uzun olsun. Bu tür doğrusal bir sınır, Resim 29.17'nin sağ tarafında gösteriliyor. İki tarafta en yakın noktalardan geçen paralel kesik çizgilere "destek vektörleri" denir. Kıyısı

63 Bkz. S. B. Kotsiantis ve P. E. Pintelas. "Combining Bagging and Boosting", *International Journal of Computational Intelligence*, c. 1, no: 4, 2004, s. 324-333. Bkz. http://www.math.upatras.gr/~esdlab/en/members/kotsiantis/ijci paper kotsiantis.pdf.

olabildiğince en büyük olan sınırlar istenir, çünkü bunlar, alıştırma kümesinde olmayan yeni noktaları sınıflandırmada daha iyidir; yani bunların daha iyi "genelleştirme" niteliği bulunur.

SRI'da örüntü tanımaya (gözetimli öğrenme türüne) dair ilk çalışmalarımız, alıştırma örneklerinden yalıtılmış ayırt edici sınırları bulmaya çalıştığımız bazı deneyleri kapsıyordu. Bunu yapmanın yöntemlerinden biri, özgün alıştırma örneklerine küçük miktarda "gürültü" eklenerek bunlardan türetilmiş öğrenme örneklerini işe dahil etmeyi içeriyordu. Buradaki temel fikir, bu güçlendirilmiş kümeye uygulanan hata düzeltme eğitim yöntemini, özgün örneklerden uzaklaşmaya zorlamaktır. H. Glucksman daha şık bir yöntem önermişti; bu yöntemde, hata düzeltme alıştırmaları, örnek kümeleri ile ayrıt edici sınırlar arasında en az uzaklığa ulaşılana dek sürdürülüyordu.[64] Fakat kıyıları olabildiğince büyük kılmak, bazı karmaşık optimizasyon yöntemleri gerektirir. (Matematiğe yatkın okurlar, Tristan Fletcher'ın çevrimiçi tanıtımına (http://www.tristanfletcher.co.uk/SVMExplained.pdf) ya da Nello Christianini ile John Shawe-Taylor'un DYM'ler hakkındaki ders kitaplarına bakabilir.[65])

Şimdi, doğrusal olarak ayırt edilebilen özellik uzaylarının nasıl elde edildiğini sorabilirsiniz. Bunun yollarından biri, Rosenblatt'ın alfa-perceptronu gibi bir şey kullanmaktır. Alfa perceptronun birinci eşik öğesi katmanındaki örneğin *N* adet öğenin her birinin kendi girdisini, rasgele veri ölçümü derlemesinden aldığını anımsayın (örneğin piksel ya da konuşma dalgabiçimi değerleri gibi). Bu "bağdaşım birimlerinin" (ilk katman öğelerine öyle deniyordu) ikili çıktıları, iki boyutlu örnekte kullandıklarıma benzer özellikler olurdu. Bunlar, *N*-boyutlu özellikler uzamında, doğrusal olarak ayırt edilebilen noktaları belirliyordu (Rosenblatt'ın umudu böyleydi en azından). Rosenblatt'ın çalışmasında genelde öyleydiler.

DYM'lerle çalışanlar, özellikleri tanımlarken farklı bir yöntemden faydalanır. Bunların yöntemi, ortaya çıkan özellikler uzayının doğrusal olarak ayırt edilmesini (en azından buna yakın olmasını) sağlar. Bunların özellikleri, "çekirdek" denilen şeyin kullanımını içerir ve bu gibi özelliklerden yararlanan makinelere "çekirdek makineleri" denir.

64 H. Glucksman, "On the Improvement of a Linear Separation by Extending the Adaptive Process with a Stricter Condition", *IEEE Transactions on Electronic Computers*, c. EC-15, no: 6, 1966, s. 941-944.

65 Nello Cristianini ve John Shawe-Taylor, *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*, Cambridge, UK: Cambridge University Press, 2000.

Bunun da matematiği burada anlatamayacağımız kadar karmaşık, fakat konuyla ilgilenen okurlar, Nello Christianini ile John Shawe Taylor'ın kitabına göz atabilir. Bu kitabın işaret ettiği gibi, çekirdek makinelerini ve DYM'leri doğuran matematiğin tarihçesi, 20. yüzyılın başına kadar uzanır ve optimizasyon kuramında, istatistikte ve hesaplamalı öğrenme kuramında çalışan insanları ilgilendirmiştir.

DYM'ler ve çekirdek makineleri, son derece matematiksel olanaklardan faydalanan farklı disiplinlerdeki çalışmaların, yapay zekâda güçlü yeni tekniklerin geliştirilmesine katkı verişinin güzel örnekleridir. Yapay öğrenmede yeni çalışmaların anlatıldığı önemli yerler arasında, her yıl Sinirsel Bilgi İşleme Sistemleri Derneği'nin hamilik ettiği Sinirsel Bilgi İşleme Sistemleri (NIPS) Konferansları vardır.[66]

Bu bölümde yapay öğrenmeye dair onca yöntemi gördükten sonra, hangi yöntemin en iyi olduğunu sorabilirsiniz. En yakın-komşu yöntemi mi, karar ağacı mı yoksa başka bir şey mi kullanılmalı? Bu soruyu araştırmacılar da sormuş ve farklı yöntemlerin, örneğin yazı karakteri tanıma gibi çeşitli standart problemler üzerinden rekabet ettiği "yarışmalar" yapılmıştır. Avrupa Topluluğu ESPRIT projesi StatLog tarafından düzenlenen bu tür bir yarışma, Donald Michie, D. J. Spiegelhalter ve C. C. Taylor'ın derledikleri kitapta anlatılıyor.[67] Çeşitli yöntemlerin başka bir kıyaslaması, Russel ile Norvig'in YZ kitabında bildiriliyor.[68] Kimi problemlerde bazı yöntemler daha iyi iş çıkartır, fakat bu farklılıklar çoğunlukla miniktir ve sahada çalışan çoğu insan, sonuçta daha fazla veri toplamanın, kullanılan yapay öğrenme algoritmasından daha önemli olduğu konusunda anlaşır. Yani, belirli bir yöntemle ayrıntılı bir şekilde uğraşmaktansa veri toplamaya daha fazla vakit ayırın.[69]

66 Bkz. http://nips.cc/.

67 Donald Michie, DJ. Spiegelhalter ve CC. Taylor (ed.), *Machine Learning, Neural and Statistical Classification*, Chichester: Ellis Horwood, 1994. Kitap artık basılmıyor ama şu internet adresinde var: http://www.maths.leeds.ac.uk/~charles/statlog/.

68 Stuart Russell ve Peter Norvig, *Artificial Intelligence: A Modern Approach*, 2. baskı, s. 752-754, Upper Saddle River, NJ: Prentice Hall, 2003.

69 Farklı algoritmaların kıyaslamasında ilave bakış açıları için bkz. David J. Hand, "Classifier Technology and the Illusion of Progress", *Statistical Science*, c. 21, no: 1, 2006, s. 1-15. Şu internet adresinde: http://arxiv.org/pdf/math.ST/0606441.

30
DOĞAL DİLLER VE DOĞAL MANZARALAR

Önceki bölümlerin bazılarında, verilerin ve bu verileri damıtıp kullanan yapay öğrenme tekniklerinin önemli rolünü gördük. Eğer mantıksal gösterimlerden ve Cyc uslamlamasından sinir ağlarının kararlarına ve öngörülerine varana dek ayrı ayrı YZ yaklaşımlarını birleştirmeye başlayan tek bir konu varsa, hepsinin muzzam miktarda veriye bel bağlamasıdır. Oyuncak problemler ve basit bulmacalardan sıyrılıp gerçek dünyaya dair problemlere dalmak, gerçek dünyaya ait veriler gerektirir. Bu bölümde, doğal dil işlemenin ve bilgisayarla görünün en yeni sistemlerinin, başa çıkmaları gereken girdileri temsil eden verilerden nasıl faydalandığını betimliyorum.

30.1 DOĞAL DİL İŞLEME (DDİ)

Yeni teknik ilerlemeler, büyük veritabanları ve artan bilgisayar gücüyle birlikte yazılı ve sözlü dillerle başa çıkabilecek sistemlere duyulan gereksinim, metin parçalarını özetlemek, sorguları yanıtlamak, dil çevirisi yapmak gibi görevleri yerine getiren gelişmiş sistemlerin doğmasına yol açtı. Bu kısımda, son yirmi otuz yılda DDİ'de (doğal dil işleme) yaşanmış kimi teknik gelişmeleri anlatıyorum. Bu gelişmeler etkileyici olsa da, Terry Winograd'ın ta 1971'de dile getirdiği şu umudunu henüz hayata geçirememiştir: "Bilgisayar sistemleriyle, araştırma asistanları, kütüphaneciler ya da sekreterle konuşur gibi konuşacağız; komutlarımızı yerine getirip istediğimiz bilgileri temin edecekler". Pek çok insana göre, bu gibi sistemleri hayata geçirme problemi, "tam YZ" problemidir, yani bu sistemler *genel olarak* insanlar kadar zeki olup insanlar kadar usavurum yapmalı ve problem çözmeli. Her durumda, bu gibi sistemlerin, en sonunda onları elde ettiğimizde, burada betimleyeceğim teknolojilerin tümünü ya da bir kısmını barındıracak olması olasıdır.

30.1.1 Dilbilgileri ve Ayrıştırıcı Algoritmalar

Daha önce, dilbilim kuramının kimi temel fikirlerini anlatmıştım. Örneğin, cümlelerin, bağlamdan yoksun dilbilgileri (BYD) kullanılarak sözdizimi yapılarına göre çözümlenebileceğinden bahsetmiştim. Kesin koşullu dilbilgisi (KKD), sistemli dilbilgisi, geçişli ağ dilbilgisi ve DIAGRAM gibi daha karmaşık dilbilgilerinden de söz etmiştim. Doğal dil cümlelerini çözümlerken dilbilgisinden faydalanan sistemler, girdi cümlesine uyanını bulmak amacıyla aday "ayrıştırma ağaçlarını" tararken, ayrıştırıcı algoritmalar kullanmak zorunda. Geçerli cümle saydığımız sözcük zincirlerini kabul eden ve cümleden saymadığımız sözcük zincirlerini reddeden gerçeğe uygun dilbilgileri için, her biri ayrı bir anlam ileten olası pek çok çözümlemenin var olması olasıdır. Tüm bunların arasında "en iyi" tek ayrıştırma ağacını seçmek, anlamsal ve edimsel çözümlemelere bel bağlar; bu çözümlemeler, cümlenin içinde bulunduğu bağlamı ve sağduyu bilgisini hesaba katar.

Sağduyu bilgisi hesaba katılmadığında sorun çıktığını gösteren mizahi bir örnek olarak, Daniel Jurafsky ve James Martin, 1930 tarihli sinema filmi *Animal Crackers*'dan bir cümle alıntılar: Groucho Marx şöyle der: "Bir sabah pijamamın içinde bir fil vurdum. Pijamama nasıl girmiş bilmiyorum."[1]

Doğal dil işleme konusundaki çalışmalar, yeni ve daha karmaşık dilbilgileri, ayrıştırıcı algoritmalar ve anlamsal işleme teknikleri keşfetmeyi sürdürüyor. Yeni dilbilgileri, artık İngilizcenin daha büyük altkümeleriyle daha etkin bir şekilde başa çıkabiliyor, hatta bunların pek çoğu İngilizce dışındaki dilleri de kullanabiliyor. Birtakım örnekler saymak gerekirse: sözcüksel işlevsel dilbilgileri (LFG),[2] ağaç bitiştiren dilbilgileri (TAG),[3] bağımlılık dilbilgisi, baştan güdümlü öbek yapısı

1 [Cümlenin orijinali: *One morning I shot an elephant in my pajamas. How he got into my pajamas I don't know.*] Alıntı: Daniel Jurafsky ve James H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 2. baskı, Upper Saddle River. NJ: Pearson Prentice Hall, 2008, s. 432.

2 Ronald Kaplan ve Joan Bresnan, "Lexical-Functional Grammar: A Formal System for Grammatical Representation", *The Mental Representation of Grammatical Relations* içinde, Joan Bream (ed.), Cambridge, MA: MIT Press, 1982, s. 173-281.

3 Aravind K. Joshi, L. S. Levy ve M Takahashi, "Tree Adjunct Grammars", *Journal Computer Systems Science*, sayı10, no: 1, 1975.

dilbilgileri (HPSG),[4] idare ve bitiştirme[5] ve kategorik dilbilgileri.[6]

Ayrıştırıcı algoritmalarda da pek çok gelişme kaydedildi. Gerçeğe uygun dilbilgileriyle birlikte kullanıldığında, genişliğe öncelik veren arama teknikleri (ya aşağıdan yukarı ya da yukarıdan aşağı yöntemle birlikte), saklama alanını çabucak tüketir. Derinliğe öncelik veren arama tekniği, bellek bakımından daha hesaplı olsa da, arama işlemi sorunla karşılaşıp cümlenin daha önceki kısımlarına dönmek zorunda kaldığında, aramanın çoğunu bitirmiş olma tehlikesi taşır. Geri dönüp aynı kısımları tekrar ayrıştırma işinden kaçınmak amacıyla, cümlenin önceden hesaplanmış ayrıştırmalarını olası tekrar kullanım için saklayan çizelgeler ve başka yapılar devreye sokan ayrıştırıcılar icat edilmiştir. Martin Kay, ilk çizelgeli ayrıştırıcı geliştirmişti.[7] Çizelge benzeri yapılar kullanan başka ayrıştırıcılar, Earley ayrıştırıcısı (Jay Earley'nin icadıdır)[8] ile Cocke-Younger-Kasami (CYK) algoritmasıdır.[9] Modern ayrıştırıcılar, daha önce bahsetmiş olduğum dinamik programlama tekniğinin sürümlerinden herhangi birini kullanır. Bu teknik, ara sonuçların kaydedilmesine olanak tanır. Sırf DDİ'nin bu yönlerinin genişliğini ve derinliğini açıklamak için, bu dilbilgisi ve ayrıştırıcı örneklerini betimlemeye kalkışmaksızın sıralıyorum (bunlar oldukça tekniktir).[10]

Doğal dil işleme araştırmaları ve uygulamaları, büyük metin dosyaları barındırmanın çok faydasını görmüştür. Bu gibi dosyalar, milyonlarca cümle içerir ve pek çok dilde olabilir. Gazete makaleleri, ede-

4 Carl Pollard ve Ivan A. Sag, *Head-Driven Phrase Structure Grammar,* Chicago: University of Chicago Press, 1994.

5 Noam Chomsky, "Some Concepts and Consequences of the Theory of Government and Binding", *Linguistic Inquiry Momograph* 6, Cambridge, MA: MIT Press, 1982.

6 Mark Steedman, "Categorial Grammar (Tutorial Overview)", *Lingua,* c. 90, 1993, s. 221-258.

7 Bkz. Martin Kay, "The MIND System", *Natural Language Processing* içinde, Randall Rustin (ed.), New York: Algorithmics Press, 1973, s. 155-188.

8 Jay Earley, "An Efficient Context-Free Parsing Algorithm", *Communications of the Association for Computing Machinery,* c. 13, no: 2, 1970, s. 94-102.

9 Bkz. John Cocke ve Jacob T. Schwartz, "Programming Languages and Their Compilers: Preliminary Notes", Teknik Rapor, Courant Matematik Bilimleri Enstitüsü, New York Üniversitesi, 1970; Tadao Kasami, "An Efficient Recognition and Syntax -Analysis Algorithm for Context-Free Languages", Bilimsel Rapor AFCRL-65-758, Air Force Cambridge Research Lab, Bedford, MA, 1965; ve Daniel H. Younger, "Recognition and Parsing of Context-Free Languages in Time n^3", *Information and Control,* c. 10, no: 2, 1967, s. 189-208.

10 Daha derine inmek isteyen okurlar, DDİ konulu bir ders kitabına bakabilir, örneğin: Daniel Jurafsky ve James H. Martin, *a.g.e.*

biyat metinleri ve başka yazıları kapsarlar. Büyük cümle dosyalarına, bütünce (*corpora*) denir. Stanford'un DDİ internet sitesi, http://nlp.stanford.edu/links/statnlp.html, birtakım örnekler sunuyor. "Dilbilim Veri Konsorsiyumu"nun internet sitesi, http://www.ldc.upenn.edu/, başka bir bütünce içeriyor. Bu dosyalardan alınan cümleleri, bazen insanlar ayrıştırıcı algoritmaların yardımıyla ayrıştırırlar ve açıklayıcı ek notlar, bunlarla ilintili cümlelerle birlikte "ağaç bankası" denilen yapılarda saklanır. Pennsylvania Üniversitesi'nde geliştirilmiş "Penn Ağaç Bankaları", önde gelen örneklerdir.[11] Penn Ağaç Bankası Projesi, şu adreste bir internet sitesine sahip: http://www.cis.upenn.edu/~treebank/. Ağaç bankaları, açıklayıcı ek notlarıyla birlikte, içlerinde barındırdıkları cümleleri kapsayan daha güçlü dilbilgilerinin oluşturulmasını tetikleyebilir. Her zamanki gibi, ağaç bankası ne denli büyükse, tetiklenmiş dilbilgisi de o denli iyidir. Bu süreçte, istatistik tabanlı yapay öğrenme teknikleri kullanılır ve bu da beni bir sonraki konu başlığına getiriyor.

30.1.2 İstatistiksel DDİ

A. Olasılıklar Kullanan Bağlamdan Bağımsız Kurallar

Daha önce bahsettiğim üzere, bir dilbilgisi, dilde geçerli olan cümlelerle geçerli olmayanlar arasında ayrım yapabilmelidir. Fakat Christopher Manning ile Hinrich Schütze'nin işaret ettikleri gibi, "doğru oluşturulmuş sözleri, yanlış oluşturulmuş söz sayılan öteki tüm sözcük zincirlerinden kesin hatlarla ayıracak mutlak ve tam bir tanımlama mümkün değildir. Bunun nedeni, insanların kendi iletişim ihtiyaçları için 'kuralları' daima esnetip bükmesidir."[12] Bu olgu, dil araştırmalarında gayet erken bir aşamada fark edilmiştir. Dilbilimci ve antropolog Edward Sapir, 1921 tarihli kitabında şöyle yazmıştı: "Maalesef ya da ne iyi ki, hiçbir dil aman vermez şekilde tutarlı değil. Tüm dilbilgileri sızıntı yapar."[13] Elbette Sapir, ne kadar karmaşık olursa olsun her dilbilgisinin, insanların kabul edilmez saydığı kimi cümleleri kabul ettiğini,

11 Mitchell P. Marcus, Beatrice Santorini ve Mary Ann Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank". *Computational Linguistics*, c. 19, 1993, s. 313-330. Sıkıştırılmış ek baskısı şu internet adresinde: ftp://ftp.cis.upenn.edu/pub/treebank/doc/cl93.ps.gz.

12 Christopher D. Manning ve Hinrich Schütze, *Foundations of Statistical Natural Language Processing*, Cambridge, MA: MIT Press, 1999.

13 Edward Sapir, *Language: An Introduction to the Study of Speech*, Bölüm II, The Elements of Speech, New York: Harcourt Brace, 1921.

insanların reddettiği kimi cümleleriyse kabul ettiğini kastediyordu.

Bu zorluğun farkına varmış ilk YZ araştırmacılarından Eugene Charniak, sözdizimi çözümlemelerinin, olasılıklarla nitelenmesini önermişti. Bazı cümleler "büyük olasılıkla" geçerlidir, bazılarıysa büyük olasılıkla değildir ve ikisi arasında her türlü aşamalanma vardır.[14] Bu tür bir yaklaşımın anında sağladığı yararlardan biri, bir çözümlemenin olasılığının, anlamı muğlak cümlelerin alternatif çözümlemeleri arasından seçim yapmayı mümkün kılmasıdır. Örneğin, Grouchovari *John shot elephants in pajamas* [John, pijama içinde fil vurdu] cümlesini okumanın iki alternatif yoluna bakalım:

- *John (while in pajamas) shot elephants.* [John (üzerinde pijamaları olduğu halde), filler vurur.]
- *John shot elephants (which were in pajamas)* [John, (pijamalı) filler vurur].

Cümlenin bu yorumlarından her birinin farklı bir ayrıştırma ağacı bulunur. Peki, bunlardan birini daha olası saymanın bir yolu var mı?

1969'da, otomatlar kuramcısı Taylor L. Booth, dilbilgisi tanımlamak için kullanılan kurallara olasılıklar atayan, bağlamdan bağımsız dilbilgilerinin bir çeşitlemesini önermişti.[15] Bu tür dilbilgilerine "Olasılıksal Bağlam Bağımsız Dilbilgileri" (OBBD) denir. Bu fikri açıklamak amacıyla, aşağıdaki çok basit (ve eksik) dilbilgisinden yararlanacağım:[16]

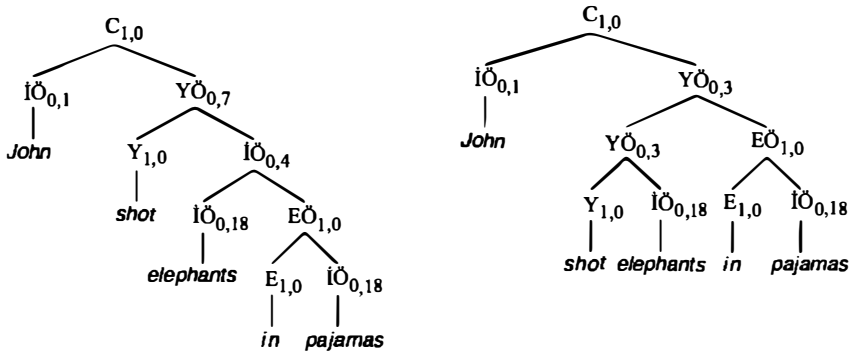
```
C → İÖ YÖ (1,0)     İÖ → İÖ EÖ (0,4)
EÖ → E İÖ (1,0)     İÖ → John (0,1)
YÖ → YÖ İÖ (0,7)    İÖ → pajamas (0,18)
YÖ → YÖ EÖ (0,3)    İÖ → shot (0,04)
E → in (1,0)        İÖ → elephants (0,18)
Y → shot (1,0)      İÖ → uniforms (0,1)
```

Kuralın ardından gelen parantezlerdeki rakam, o kuralın olasılığını temsil ediyor. Dolayısıyla, örneğin bu dilbilgisine göre, cümlede isim

14 Eugene Charniak, *Statistical Language Learning*, Cambridge, MA: MIT Press, 1993.

15 Taylor L. Booth, "Probabilistic Representation of Formal Languages", *Tenth Annual IEEE Symposium on Switching and Automata Theory*, 1969, s. 74-81.

16 Bu dilbilgisi, Manning ve Schütze'nin kitabının on birinci bölümündeki dilbilgisinden uyarlanmıştır; bu kitap, yardımcı slaytlarla birlikte şu internet adresinde: nlp.stanford.edu/fsnlp/pcfg/fsnlp-pcfg-slides.pdf. Manning ile Schütze'nin belirttikleri gibi, İÖ kuralları bir parça sıradışıdır, çünkü içinde barındırdığı dilbilgisi, "Chomsky Normal Biçimi" denilen dilbilgisidir.



Resim 30.1. "John pijamamalarıyla filler vurur" cümlesinin iki ayrıştırma ağacı. [C: Cümle, E: Edat, EÖ: Edat Öbeği, İÖ: İsim Öbeği, Y: Yüklem, YÖ: Yüklem Öbeği]

öbeğinin "elephants" [filler] sözcüğü olma olasılığı 0,18'dir. İsim öbeği *bir şey* olmak zorunda olduğu için, tüm isim öbeklerinin olasılıklarının toplamı 1,0'dır.

B. Ayrıştırma Ağaçlarının Olasılıkları

Bu kuralların olasılıklarının bağımsız olduğunu varsayarak (gerçeğe uygun dilbilgileri için pek de uygun olmayan bir varsayımdır bu), ayrıştırma ağacının olasılığını hesaplayabiliriz, zira ağaçta kullanılmış tüm kuralların olasılıklarının çarpımını alabiliriz. Bu cümle için iki ayrıştırma ağacı Resim 30.1'de gösteriliyor. John'un ateş ederken pijamalarını giymiş halde olduğunu gösteren sağdaki ağaç, belki çizgi filmler dışında, çoğu ortamda daha uygun görünüyor.

Ağaçlardaki her dilbilgisi terimine yazılmış rakamlar, ilgili kuralların olasılıklarıdır. Soldaki ayrıştırma ağacının olasılığı şudur:

$$\text{Olasılık}_{\text{sol}} = 1{,}0 \times 0{,}1 \times 0{,}7 \times 1{,}0 \times 0{,}4 \times 0{,}18 \times 1{,}0 \times 1{,}0 \times 0{,}18 = 0{,}0009072.$$

Sağdaki ayrıştırma ağacının olasılığıysa şöyledir:

$$\text{Olasılık}_{\text{sağ}} = 1{,}0 \times 0{,}1 \times 0{,}3 \times 0{,}7 \times 1{,}0 \times 0{,}18 \times 1{,}0 \times 1{,}0 \times 0{,}18 = 0{,}0006804.$$

Dolayısıyla soldaki ağaç tercih edilir. (Pekâlâ, gerçekten filleri vurmaktan bahsediyor değilim. Aklımda, pijama giymiş çizgi film filleri vardı.)

OBBD'lerin başka bir önemli yönü de, cümlenin toplam olasılığını öngörmede kullanılmalarıdır. Yani, *John shot elephants in pajamas* gibi bir cümlenin ortaya çıkma olasılığı nedir? Bu cümlenin olası tüm ayrıştırma ağaçlarının olasılıklarını toplayarak bunu hesaplayabiliriz.

Bu durumda, iki olasılığı toplayarak 0,0015876 rakamını elde ederiz. Bu örnekteki olasılıklar, açıklayıcı olsun diye uydurulmuştur ve ciddiye alınmamalıdır. Daha gerçekçi olasılık değerleri, daha büyük bir dilbilgisini ve cümleler bütüncesini temel alır; bu da beni bir sonraki konuya getiriyor.

C. OBBD'leri Öğrenmek

OBBD'de kuralların olasılık değerleri nasıl elde edilir? Daha özel olarak, gerçek cümleleri uygun biçimde modelleyen değerler nasıl elde edilir? Açıklamalı bir ağaç bankası, o ağaç bankasındaki cümlelere uygun değerleri sağlamanın bir yolunu sunar, çünkü her cümlesinin, ilgili bir ayrıştırma ağacı bulunur. Ayrıştırma ağaçları, hep $sl \rightarrow s$ biçiminde kurallar kullanır; *sl*, kuralın sol tarafını gösterirken (YÖ gibi), *s*, kuralın sağ tarafını gösterir (YÖ İÖ gibi). $sl \rightarrow s$ biçimli bir kuralın olasılık değerini elde etmek için, ağaç bankasında bu kuralın kaç kere boy gösterdiğini hesaplarız ve bu sayıyı, *sl*'nin boy gösterdiği sayıya böleriz. Bu şekilde elde edilen OBBD, yeni cümleleri çözümlerken kullanılabilir.

Eğer elde, bir cümle bütüncesine (*corpus*) uygulanabilecek sıradan (olasılıksal olmayan), bağlamdan bağımsız bir ayrıştırıcı varsa, ağaç bankası olmaksızın da OBBD yaratılabilir. Fakat ağaç bankasının aksine, bütüncedeki her cümlenin birden fazla ayrıştırmasının olması olasıdır ve bazı cümlelerin pek çok ayrıştırması olur. Peki, kuralların boy gösterişini sayarken, bol bol ayrıştırması bulunan cümlelerdeki kurallara gereğinden fazla ağırlık vermekten nasıl uzak durabiliriz? Aşağıdaki yöntem, gayet işe yarıyormuş gibi görünüyor:

1. Özgün BYD'yi eşit kural olasılıklarıyla birlikte OBBD'ye dönüştür.
2. Cümleleri bu OBBD'yle çözümle, her muğlak çözümleme için tek olasılık hesapla.
3. Her cümle için her çözümlemedeki kuralları say ve bu sayımı, o çözümlemenin olasılığıyla ağırlıklandır.
4. Bu ağırlıklı sayımları kullanarak kurallar için yeni olasılıklar, dolayısıyla yeni bir OBBD hesapla.
5. Bu işlemi kural olasılıkları değişmeyi bırakana dek tekrarla (önünde sonunda olacaktır bu).

Bu yöntem, yapay öğrenmede sık sık kullanılan Beklentileri Büyütme (BAÇ) algoritmasının bir sürümüdür.

Cümle yapısının bağlamdan-bağımsız-olmayan yönlerini ve özgül sözcüklerle ilgili ayrıntılı bilgiyi açıklamak amacıyla, kullanışlı uygulamalar genelde, çeşitli yollarla güçlendirilmiş OBBD'ler kullanır. OBBD sürümleri için birtakım ayrıştırıcılar geliştirilmiştir. Bu noktada, A* arama algoritmasını temel alan ayrıştırıcıdan bahsetmekten kendimi alamıyorum. Dan Klein ve Christopher D. Manning, bunu sunarken şöyle yazmıştı:[17]

> Ortalama uzunluğa sahip Penn ağaç bankası cümlelerine dair [sezgisel fonksiyon olarak kullanımları bakımından] en ayrıntılı tahminlerimiz, [A* arama tekniği kullanılarak] işlenmiş kenar çizgilerinin toplam sayısını, tüm olasılıkları göz önüne alan bir çözümleyicinin gerektirdiği rakamdan en fazla %3 azaltır ve daha basit bir tahmin, ki bir dakikadan kısa bir önhesaplama gerektirir, iş yükünü en fazla %5 azaltır.

Cümle çözümlemesinde, istatistik tabanlı başka yöntemler de geliştirilmiştir. Bunların sadece birkaçını betimleyeceğim.[18] Rens Bod ve Amsterdam Üniversitesi'ndeki çalışma arkadaşları, "Veri-Yönelimli Çözümleme" (VYÇ) [Data-Oriented Parsing] dedikleri bir teknik geliştirmiştir; bu teknik, şu fikre dayanıyor: "insanlarda dili algılama ve üretme, *soyut* dilbilgisi kurallarının gösterimi yerine, *somut* dil deneyimlerinin gösterimleriyle işliyor."[19] Ayrıca istatistiksel yöntemler, hem VYÇ fikirlerinden[20] hem de Josef van Genabith ile Dublin City Üniversitesi'ndeki ekibinin açıklamalı ek notlarıyla ağaç-bankası verilerinden, SİD dilbilgisi öğrenme üzerine çalışmalarından[21] faydalanarak Sözcüksel İşlevsel Dilbilgilerini (SİD) de güçlendirmiştir. Son olarak, Ron Kaplan ve ticari doğal dil sorgulama şirketi Powerset'teki (artık Microsoft bünyesinde-

17 Dan Klein ve Christopher D. Manning, "A* Parsing: Fast Exact Viterbi Parse Selection", *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – vol. 1*, Morristown, NJ: Association for Computational Linguistics, 2003, s. 40-47.

18 Ronald Kaplan bana bundan 9 Haziran 2008 tarihli e-postasında bahsetmişti.

19 Rens Bod ve diğ., "Data Oriented Parsing (DOP)", *Proceedings COLING '92*, Nantes, France, 1992. Ayrıca bkz. Data-Oriented internet sayfasına bakınız: staff.science.uva.nl/~rens/dop.html.

20 Rens Bod ve diğ., "A Data-Oriented Approach to Lexical-Functional Grammar", *Computational Linguistics in the Netherlands 1996* içinde, Jan Landsbergen (ed.), Eindhoven, Hollanda, 1996.

21 Anette Frank ve diğ., "From Treebank Resources to LFG F-Structures: Automatic F-Structure Annotation of Treebank Trees and CFGs extracted from Treebanks", *Treebanks: Building and Using Syntactically Annotated Corpora* içinde, Anne Abeille (ed.), Dordrecht/Boston/Londra: Kluwer Academic Publishers, 2003.

dir) ekibi, (öğrenilmiş olmak yerine) elle işlenmiş bir ayrıştırıcı kullanılarak üretilmiş bir cümlenin çoklu ayrıştırma ağaçlarına, sıralama olasılıklarının nasıl atanacağını öğrenmeye çalışıyor.

Doğal dil işlemede istatistiğin başka kullanımları, sözcüklerin belirli birleşimlerinin çeşitli metin kaynaklarında ne sıklıkta görüldüğü verisini içerir. Bu gibi sözcük birleşimlerine "*n*-gram" denir. Örneğin, "hemen şimdi" gibi iki sözcüklük bir dizi 2-gramdır ve "onu rafa kaldır" gibi üç sözcüklük bir dizi 3-gramdır. "En iyi veri, çok veridir" sloganını kullanan Google, kamuya açık İnternet sayfalarından bir trilyonluk bir sözcük oluşumunu, örneğin "en az kırk kez boy gösteren 1.176.470.663 adet beş-sözcüklük dizilerin sayısını" yayımlamak amacıyla çözümlemişti.[22]

Manning ve Schütze, istatistiksel yöntemlerin kullanımının etkisini özetlerken şöyle yazmıştır: "İşin doğrusu, doğal dil işleme alanında istatistiki yöntemler için duyulan heyecan, geleneksel DDİ yöntemlerinin kullanımında çözümden kaçmış gerçek problemlere, istatistiki yöntemlerin kullanışlı çözümler getirebileceğini gören insanların beklentisinden kaynaklanmıştır". Bu alan için birtakım yeni isimler bile önerirler, örneğin DDİ yerine "Dil Teknolojisi" ya da "Dil Mühendisliği" gibi.[23]

30.2 BİLGİSAYARLA GÖRME

Bu kısımda, bilgisayarla görme alanında gerçekleştirilmiş son çalışmaların birtakım temsili örneklerinden bahsedeceğim; bunların büyük kısmı, daha önceki bölümlerde anlatmış olduğum temel görüntü işleme teknikleri üzerine kurulmuştur. Aslında, çoğu araştırmacı, eski çalışmalara duyduğu minneti ifade eder, tıpkı yeni bir makaleden alınmış şu alıntıdaki gibi:[24]

> Bilgisayarla görme alanında, bölge ve kenar çizgisi betimleyiciler, üstpikseller, yukarıdan aşağı ve aşağıdan yukarı süreçleri birleştirmek, Bayesçi formülasyon, öznitelik ayıklama, vesaire gibi modern sayılan pek çok fikrin otuz yıl önce pekâlâ bilindiğini belirtmek ilginç olacaktır!...

22 Bkz. http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html.

23 Christopher D. Manning ve Hinrich Schütze, *a.g.e.*, s. 7.

24 Derek Hoiem, Alexei Efros ve Martial Hebert, "Recovering Surface Layout from an Image", *International Journal of Computer Vision*, c. 75, no: 1, 2007, s. 151-172. Şu internet adresinde: http://www.ri.cmu.edu/pubs/pub_5818.html.

> Bununla birlikte, ilk öncülerin, zamanlarının ilerisinde oldukları anlaşılıyor. Sezgisel yöntemlere güvenmekten başka çareleri yoktu, çünkü büyük miktarda veriden ve görsel dünyamızın yapısına hükmeden ilişkileri öğrenmeye yarayacak bilişim kaynaklarına sahip değildiler. Son on yılda öğrenme yöntemlerinde kaydedilmiş ilerlemeler, manzara anlama konusunda bütün bir çözüm için yeni bir umut getiriyor.

Artık elimizde gerekli veriler ve bilgisayar kaynakları var. Bunların ötesinde, bilgisayarla görme başka alanlardan gelen katkılardan da faydalanmıştır, örneğin optik, matematik, bilgisayar grafikleri, elektrik mühendisliği, fizik, sinirbilim ve istatistik gibi. Tüm bu disiplinler, fikir ve teknik sağlamayı sürdürüyor, fakat bunlardan bir tanesi, ismini koymak gerekirse yapay öğrenme, ağır basmaya başlamıştır.

Bazı insanlar, "bilgisayarla görme" [*computer vision*] ve "yapay görme" [*machine vision*] arasında ayrım yapar; bilgisayarla görmeyi genelde robotikle sınırlarlar ve yapay görmeye bu uygulamayı ve başka uygulamaları dahil ederler. En sonunda robotların bu uygulamaların çoğunu gerçekleştirmesini istediğimiz için, bu ayrımın çok faydalı olduğunu düşünmüyorum, dolayısıyla alanın bütününe bilgisayarla görme demeyi sürdüreceğim.

Başka bir ayrım da, bir taraftan "manzara çözümleyici görme" [*scene analysis vision*] ile öte yanda "amaçlı (ya da etkin) görme" [*purposive (or active) vision*] arasındaki ayrımdır. Manzara çözümleyici görme yaklaşımı, ta ilk günlerinden itibaren görme araştırmalarının büyük kısmına kılavuzluk etmiştir. Bu görüş, bilgisayarla görmenin hedefinin, iki boyutlu görüntüyü üç boyutlu görüntü betimlemesine dönüştürmek olduğunu savunuyordu. Örneğin, MIT "Kopyalama Demosu"nun görme sistemi, oyuncak blokların dizilişinin üç boyutlu modelini oluşturuyordu (bkz. Resim 10.1). Bunun aksine kimi araştırmacılar, görmenin amacının, sadece ama sadece devimin denetimi için gerekli özgül bilgiyi sağlamak olduğuna işaret ediyordu. Örneğin Shakey'nin yararlandığı çeşitli görme altprogramlarında bu yaklaşımın benimsendiğini görebiliriz. Shakey, görsel dünyasının tam modelini inşa etmek yerine, devinim eylemlerine ve plan yapmaya kılavuzluk etmek için gerekli bilgiyi temin etmek amacıyla görme sisteminden faydalanıyordu. Bu tür "amaçlı görme", tam bir manzara çözümlemesine kıyasla bilgisayar kaynaklarını daha az kullanır.

Hayvanların (insanlar dahil) görsel süreçlerini inceleyen insanlar da bu iki yaklaşımdan birini savunmuştur. İnsanın görme süreçlerini modellemekle ilgilenen David Marr, manzara çözümlemesi yaklaşımını savunuyordu. Bununla birlikte, kurbağalarda görsel algıyı çözüm-

leyenler, bu hayvanın görsel sisteminin, örneğin sinekleri yakalamak gibi bir amaç etrafında örgütlendiğini belirtmiştir. Hesaplamalı sinirbiyoloğu Terrence Sejnowski (NETtalk üzerinde çalışmış aynı Sejnowski) ve çalışma arkadaşları, insan görüsünün amaçlı olduğunu, "saf görme" dedikleri görüntü yapılandırma niteliği taşımadığını gösteren biyolojik ve fizyolojik bulgular betimlemiştir. Şöyle yazmışlardır:[25]

> Görme yetisi ne işe yarar? Üç boyutlu dünyayı eksiksiz şekilde yeniden yaratmak gerçekten gerekli mi? Bu sorulara verilen biyolojik ve bilişimsel yanıtlar, saf görme kavramından oldukça farklı bir görme kavramı doğurmuştur. [Bu makalede] ana hatları betimlendiği haliyle etkileşimli görme, eylemlere kılavuzluk ederken başka duyu sistemlerini de ortak olarak işe dahil eder.

Yaklaşık son yirmi yılda üretilmiş pek çok bilgisayarla görme sistemine baktığımda, bu iki türü de görüyorum. Örneğin, özerk taşıtlara yollarda rehberlik eden yetkin sistemler var; her ne kadar manzaraya dahil olsalar da sürme işiyle ilgisi olmayan yol kenarındaki evleri çözümlemeksizin, hatta onların varlığının farkında olmaksızın salt yola ve öbür taşıtlara dikkat eden sistemler bunlar. Ayrıca, binaların ve bina içlerindeki nesnelerin üç boyutlu modellerini oluşturmak amacıyla fotoğrafları çözümleyen sistemler de var. Buna ek olarak, iki yaklaşımdan da izler taşıyan sistemler de var; bunları, önümüzdeki kısımda anlatıyorum.

30.2.1 Derinlik ve Yüzey Bilgisi Elde Etmek

Carnegie Mellon Üniversitesi Robotik Enstitüsü'nde Derek Hoiem, Alexei Efros ve Martial Herbert, tek görüntünün bölütlerini, çeşitli türlerde ve doğrultularda yüzeylerin parçası olarak sınıflandırabilen bir program geliştirmişlerdi.[26] Her ne kadar bu sınıflandırmalar, imgeyi doğuran görüntünün üç boyutlu modelini oluşturmasa da, görüntünün önemli fiziksel nitelikleri hakkında bilgi verir; bir yerde David Marr'ın 2½ boyutlu eskizinin yaptığı gibi. Örneğin bu tür bilgi, o görüntüde yolunu bulması ve nesneleri tanıması gereken robot için faydalı olabilir.

25 Patricia S. Churchland, V. S. Ramachandran ve Terrence J. Sejnowski, "A Critique of Pure Vision", *Large-Scale Neuronal Theories of the Brain* içinde, Christof Koch ve Joel L. Davis (ed.), Cambridge, MA: MIT Press, 1994, s. 23-60. Şu internet adresinde: http://papers.cnl.salk.edu/PDFs/ACritiqueofPureVision1994-2933.pdf.

26 Derek Hoiem, Alexei Efros ve Martial Hebert, *a.g.e.*



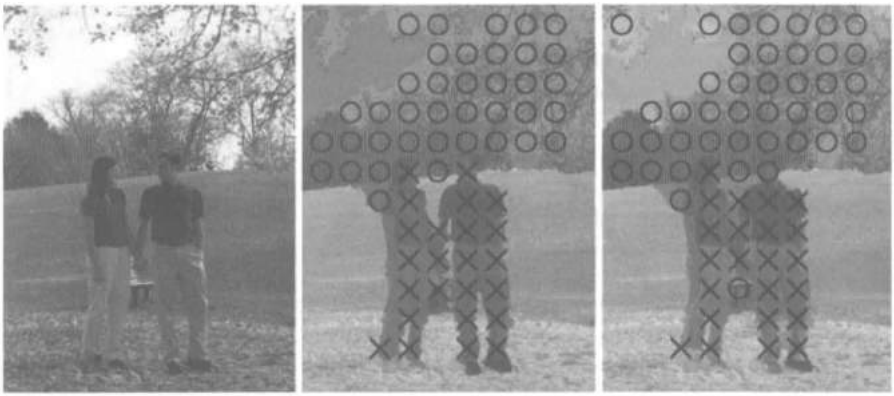
Resim 30.2. Tipik dış mekân imgeleri. (İmgeler, Derek Hoiem'in izniyle.)

Onların çalışması, çeşitli koşullar altında çekilmiş "(karlı, güneşli, bulutlu, alacakaranlık)", "orman, şehir, yol, plaj, göl vesaire" gibi dış hususların imgelerini kullanıyordu. Resim 30.2'de iki örnek gösteriliyor.

Bu program, bir görüntünün bölgelerini üç ana yüzey sınıfından biri olarak sınıflandırıyordu: "destek", "dikey", "gökyüzü". Programın yaratıcıları bu sınıfları şöyle tanımlıyor: "Destek yüzeyleri, kabaca zemine paraleldir ve potansiyel olarak katı nesneleri destekleyebilir. Örnekler arasında yol yüzeyleri, çimenler, toprak yollar, göller ve masa üstleri sayılabilir. Dikey yüzeyler, nesneleri destekleyemeyecek kadar dik olan katı yüzeylerdir, örneğin duvar, falez, bordür kenarı, insanlar, ağaçlar, inekler gibi. Gökyüzü, açık havaya ve bulutlara karşılık gelen bir görüntü bölgesidir sadece". Programın yaratıcıları, bu sınıflandırmayı haklı çıkarmak amacıyla, Google görüntü arama motoruyla topladıkları 300 görüntüde "piksellerin %97'sinden fazlasının yatay (destek), hemen hemen dikey yüzeylere ya da gökyüzüne ait olduğunu" belirtir (imgelerin insanlarca incelenmesiyle belirlenmiş yüzdedir bu).

Daha sonra program, her dikey yüzeyi şu alt sınıflardan biri olarak sınıflandırır: "yüzü 'sola,' 'merkeze' ya da 'sağa' dönük düzlemsel yüzeyler ve düzlemsel olmayan 'gözenekli' ya da 'katı' yüzeyler. Düzlemsel yüzeylere bina duvarları, falez sırtları ve kabaca düzlemsel olan başka yüzeyler dahildir. Gözenekli yüzeylerse, katı ve kesintisiz yüzeyi olmayanlardır. Örneğin, ağaç yaprakları, makiler, telefon kabloları, zincirli çitler, gözenekli yüzeylere örnektir. Katı yüzeylerse, katı ve kesintisiz yüzeyleri olan arabalar, insanlar, plaj topları ve ağaç gövdeleri gibi, düzlemsel olmayan dikey yüzeylerdir."

Bu program, 300 Google görüntüsünden oluşan alıştırma kümesini kullanarak bu sınıflandırmaları (ve alt sınıflandırmaları) yapmayı öğrenmişti. Bu kümedeki her alıştırma görüntüsünün komşu piksel

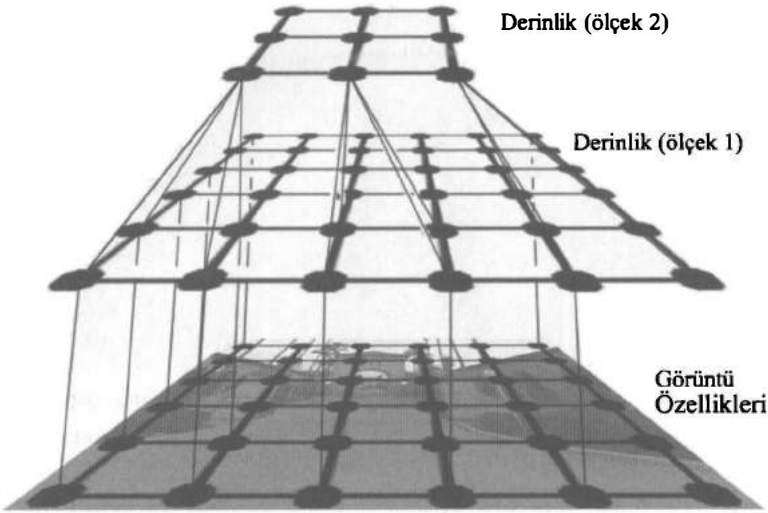


Resim 30.3. Özgün görüntü (solda), elle etiketlenmiş görüntü (merkezde) ve sistemin çıktısı (sağda). (Resimler, Derek Hoiem'in izniyle.)

toplulukları, renk ve yoğunluk temelinde neredeyse birörnek bölgeler oluşturarak bir araya gelmişti ve bunlara "üst-pikseller" [*superpixels*] denir. Sonra her üst-piksel (üşenmeden) elle bir sınıflandırma ve alt sınıflandırma değeri atanır. Bundan sonra üst-pikseller, bölüt [*segment*] denilen daha büyük bölgeler olarak gruplandırılır; bunlar, bileşenleri olan üst-piksellerin sınıflandırmalarını kalıtımla alır. Bu noktadan itibaren işin matematiği, burada girmek istediğim düzeyden (ya da okumak isteyeceğinizden) daha karmaşık hale geliyor, fakat özünde öğrenme süreci, öğrenme görüntüleri kümesinde elle sınıflandırılmış bölgelere yeterince uyan bir karar ağacı kurulabilir. Böylece, eğitilmiş karar ağacı, herhangi bir imgenin bölgelerini sınıflandırmak için kullanılabilir. Karar ağacının düğüm noktaları, pikseli ve yer, renk, doku ve bakış açısı içeren bölüt özelliklerini temel alıyordu; tüm bunlar, daha önce icat edilmiş tekniklerle hesaplanabilir (bunlardan bazılarını önceki bölümlerde betimlemiştim).

Resim 30.3'teki görüntüler, tüm sonuçları tamamen temsil ediyor olmasa da, bu programın ne kadar performanslı olduğunu gösteriyor. Resim 30.3'teki görüntülerde yeşil renk destek yüzeyini, kırmızı renk dikey yüzeyi, mavi renk gökyüzünü gösteriyor. Dikey yüzeylerin altsınıfları, sola bakan düzlemler için sola bakan oklarla, merkeze bakan düzlemler için yukarı bakan oklarla, sağa bakan düzlemler için sağa bakan oklarla gösteriliyor ve "O", gözenekli yüzeylere, "X" de katı yüzeylere işaret ediyor.

Stanford profesörü Andrew Ng ve öğrencileri daha da ileri gitmiş, tek kamera görüntülerden güncel derinlik bilgisi ve görüntü yapısı bil-



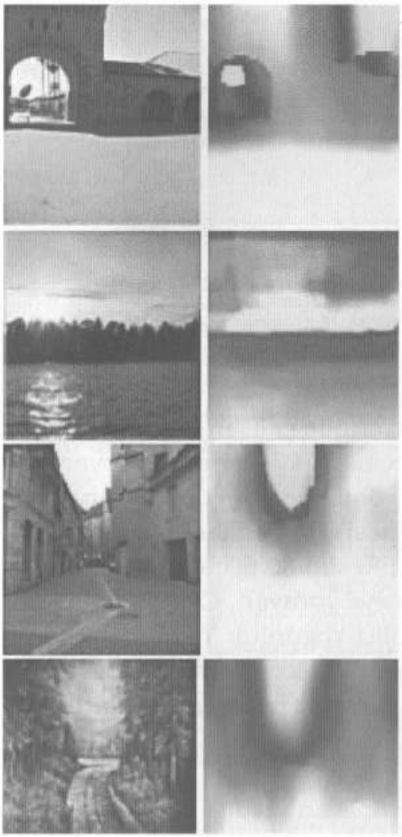
Resim 30.4. Çok ölçekli gelişigüzel Markov alan ağı. (Çizim, Derek Hoiem'in izniyle.)

gisi çıkarmışlardır. Bir alıştırma görüntü kümesinin "arazi verileri"ne dair derinlik bilgisi, öncelikle üç boyutlu bir lazer tarayıcı tarafından toplanır. Öğrenme algoritması, birtakım görüntü özelliklerinden faydalanarak, derinliğe dair öngörülerini arazi verilerinin derinliğiyle kıyaslar.[27] Bu özellikler, doku çeşitlemelerini, aşamalı doku değişimini, rengi ve cephe sistemi bilgisini içerir. Yakın nesnelere dair derinlik bilgisi uzak nesnelere kıyasla daha büyük ölçeklerde elde edildiği için, özellikler, çoklu görüntü ölçeğinde çıkarılır. Öğrenme süreci, bir görüntü parçasının derinliği ile komşu parçaların derinlikleri arasındaki ilişkiyi temsil etmek amacıyla çok ölçekli hiyerarşik bir Markov rastlantısal ağı oluşturur. Resim 30.4, bu tür üç katlı bir ağın, iki katının basitleştirilmiş bir çizimidir. (Sistemin, örneklerden yola çıkarak nasıl öğrendiği, yine, burada açıklayamayacağım kadar karmaşıktır.)[28]

Resim 30.5, bu sistemin öngördüğü, görüntülerle ilintili "derinlik haritaları"yla (farklı derinlikler, farklı renkler tarafından gösteriliyor) birlikte internetten indirilmiş bazı görüntüleri gösteriyor.

27 Betimlememi, bunların pek çok makalesinden birine dayandırıyorum: Ashutosh Saxena, Sung H. Chung ve Andrew Y. Ng, "3-D Depth Reconstruction from a Single Still Image", *International Journal of Computer Vision (IJCV)*, Ağustos 2007. Bkz. http://ai.stanford.edu/~asaxena/learningdepth/saxena_ijcv07_learningdepth.pdf.

28 Bu ayrıntılarla ilgilenen okurlar şu makaleye bakabilir: Saxena ve diğ., *a.g.e.*



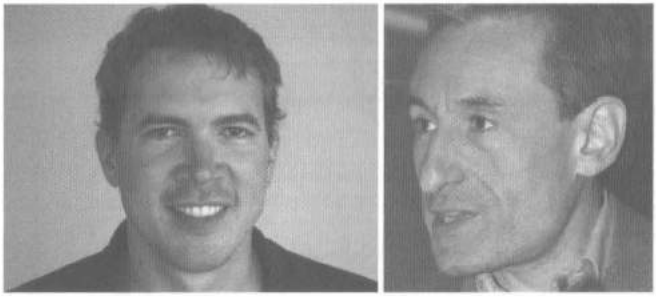
Resim 30.5. Tipik görüntüler (solda) ve öngörülen derinlik haritaları (sağda). (Fotoğraflar, Andrew Ng ile Ashutosh Saxena'nın izniyle.)

Ashutosh Saxena ve Andrew Ng, bu teknikleri geliştirmeye devam ediyor. İnternet sitelerinde (http://make3d.stanford/edu/), Youtube ya da başka yazılımlardan yararlanarak, sistemlerinin, çeşitli tek kamera görüntülerinden yola çıkarak inşa ettiği üç boyutlu modelleri "evirip çevirebilirsiniz". (Bu evirip-çevirme demoları, çok sayıda görüntüyü ve istatistiksel yöntemi temel alan bilgisayarla görmenin ne kadar ilerlediğinin etkileyici bir örneğidir.) Ayrıca bu internet sitesine kendi fotoğraflarınızı yükleyip bunları üç boyutlu hale dönüştürebilirsiniz.[29]

30.2.2 Hareketli Nesneleri İzlemek

Eğer görme sistemleri gerçek dünyadan doğal görüntülerde iş görecekse, başa çıkmaları gereken şeylerden biri hareketli nesnelerdir. Nesne-

29 Bilgisayar kodları ve görüntü verileri için bkz. http://make3d.stanford.edu/code.html.



Resim 30.6. Michael Isard (solda) ve Andrew Blake (sağda). (Fotoğraflar, Michael Isard ve Andrew Blake'in izniyle).

leri görsel olarak takip etme problemi üzerinde çalışan kimi araştırmacılar olmuştur; en eski çalışmaların geçmişi 1970'lerin ikinci yarısına kadar uzanıyor. Devreye sokulan bazı yöntemleri açıklamak amacıyla faydalanacağım örnek, Oxford Üniversitesi'nde Michael Isard (1971- ; Resim 30.6) ile Andrew Blake'in (1956- ; Resim 30.6) çalışmalarıdır. Bu araştırmacılar, hareketli nesneleri izlemeye yarayan, CONDENSATION (Conditional Density Propagation [Koşullu Yoğunluk Yayılımı]) adlı bir algoritma geliştirmiştir.[30] Bu algoritma, "eğriler olarak modellenmiş önplan nesneler önemli ölçüde karışık hareket ederken, bunların ana hatlarını ve özelliklerini izleyebiliyordu, hem de bunu film karesi hızında ya da buna yakın hızda yapıyordu."

Sistemlerinin nasıl çalıştığına yine onların verdiği örneklerden biriyle kısaca göz atalım: benzer yaprakların bulunduğu bir arkaplana karşı, rüzgârla uçan bir çalı yaprağının filmi. Filmin ilk karesinde, ilgilendiğimiz yaprağın ana hatları, Resim 30.7'nin sol kısmında gösterildiği gibi, elle çizilerek belirtilmiştir. Hareket eden yaprağın ana hatlarını izlemek, yaprağın dinamikleri hakkında bilgi gerektirir. Yani, zamanın belli bir anında görüntüdeki konumu ve şekli belliyken, yaprak, bir sonraki zaman anında muhtemelen hangi konumu ve şekli alacaktır? Peki, sonraki anlarda ne olacak? Bunu kesin bilemeyiz, fakat tahmin yapmak amacıyla, dinamik Bayesçi ağlardan (DBA) faydalanabiliriz, ki bu ağlar, ayrık değişkenlerin olasılık dağılımları yerine kesintisiz olasılık

30 Bkz. Michael Isard ve Andrew Blake, "CONDENSATION: Conditional Density Propagation for Visual Tracking", *International Journal of Computer Vision*, c. 29, 1998, s. 5-28. Şu internet adresinde: http://www.cs.cmu.edu/~efros/courses/AP06/Papers/isard-blake-98.pdf. Örneklerle ve makalelerle birlikte algoritmanın şu internet sitesine de bakınız: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/ISARD1/condensation.html.



Resim 30.7. Rüzgârda yaprağı izlemek. (İlk kareden yarım saniye sonraki kareden fotoğraflar, şu internet adresinden: http://homepages.inf.edu.ac.uk/rbf/CVonline/LOCAL_COPIES/ISARD1/images/leafmv.mpg. Michael Isard ile Andrew Blake'in izniyle kullanılmıştır.)

dağımı kullanmak üzere tadil edilmiştir. Gerekli olasılıkları bir öğrenme işlemi tahmin eder ve bu olasılıklar, hareket eden yaprağın gözlemlenmesiyle yavaşça iyileştirilir. Bununla birlikte, yaprağı gözlemlemek için onu izlememiz şart, buysa olasılıkları bilmeyi gerektirir; Isard ve Blake, bu "tavuk-yumurta" probleminin hakkından gelebilmiştir.

Zaman ilerledikçe, yaprağın görüntüdeki konumuna ve şekline dair olasılıklar yayılır, yaprağın ana hatları hakkında gitgide daha fazla belirsizlik yaratır. Fakat gözlem yapmayı sürdürürüz; her an yeni bir görüntü alırız. Hassas olmayan bu gözlemler de yaprağın konumuna ve şekline dair olasılıklar sunar (Bayes kuralı sayesinde). Gözlemlerden doğan bu yeni olasılıklar, görüntüde yayılan, yaprağın dinamikleriyle ilgili olasılıkları öyle bir noktaya kadar keskinleştirir ki görece hassas tahminler yapılabilir. Örneğin, yirmi beş zaman adımı sonrası (0,5 saniye) için sistem, Resim 30.7'nin sağ tarafında gösterilen hattı tahmin eder. (Yaprağın takip edildiği filmi şu internet adresinde izleyebilirsiniz: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/ISARDI/images/leafmv.mpg.)

Isard ve Blake, tüm bunları elde etmek amacıyla bir dizi karmaşık teknolojiden faydalanır. Karşılaştıkları sorunlardan biri, yaprak dinamiğinin olasılık fonksiyonlarını nasıl göstermek gerektiği ve bu gösterimlerin bir zaman adımından öbürüne nasıl geçirileceğidir. "Parçacık süzgeci" adlı bir teknik benimsemişlerdir; bu teknik, ana hat belirleyen bir eğrinin olasılığını, ana hatların parçacık denilen büyük bir ağırlıklı örneklemiyle gösterir. Her zaman adımında bir grup parçacık öne çıkarılıp sonraki zaman adımına aktarılır ve bu küme, bir olasılık fonksiyonu olarak gösterilir. Parçacık süzgeci tekniği, görüntü işleme gibi algıya dayalı problemlerde yaygınlıkla kullanılır.

Nesne takip etmeyi açıklamak amacıyla Isard-Blake çalışmasından bahsettim ama başka pek çok çalışma daha var. Washington Üni-



Resim 30.8. Çok sayıda insanın izlenmesi. (Filmden. Dieter Fox'un izniyle kullanılmıştır.)

versitesi, Robotik ve Durum Tahmini Laboratuvarı'nda Dieter Fox ve çalışma arkadaşları, pek çok uygulamada parçacık süzgeci kullanmıştır. İnternet sitelerinin birinde (http://www.cs.washington.edu/ai/Mobile_Robotics/mcl/), "parçacık süzgeçlerini iş başında" görebilirsiniz. Orada var olan özellikle etkileyici bir gösterim, *hareketli* bir robotun lazerli menzil bulucularını kullanarak değişken sayıda insanı aynı anda izlediği bir filmdir.[31] Tipik bir sahne, Resim 30.8'de gösteriliyor. Soldaki görüntü (robotun kullandığı görüntü değil; bu, bizim için), insanların ve robotun asıl yerlerini gösteriyor. Sağdaki görüntü, insanların ve robotun, grafiksel nesnelerle temsil edilen hesaplanmış yerlerini gösteriyor. Bu uygulama, parçacık süzgecinin bir uzantısını kullanır; sistemin yaratıcıları buna "örneklem-tabanlı ortak olasılık verilerini ilintilendirme süzgeçleri" der.[32]

Münih'teki Bundeswehr Üniversitesi Sistem Dinamiği ve Uçuş Mekaniği Enstitüsü'nde önderliğini Ernst D. Dickmanns'ın (1936) yaptığı ekip, 1970'lerin ikinci yarısından beri sürücüsüz otomobiller için görme ve denetim sistemleri üzerinde çalışıyor. Bunların, "4 Boyutlu" yaklaşım dedikleri dinamik görme sistemleri, nesne devinimin uzamsal-zamansal modellerini kullanarak yakın taşıtları saptayıp takip edebilir.[33] Belki de onlar, görsel nesne takibinde Kalman süzmeyi kullanan ilk

31 Bkz. http://www.cs.washington.edu/ai/Mobile_Robotics/mcl/animations/floor3D.avi

32 Dirk Schulz, Wolfram Burgard, Dieter Fox ve Armin B. Cremers, "People Tracking with a Mobile Robot Using Sample-based Joint Probabilistic Data Association Filters", *The International Journal of Robotics Research (IJRR)*, c. 22, no: 2, 2003, s. 99-116.

33 Bkz. Ernst D. Dickmanns, "Dynamic Vision-Based Intelligence", *AI Magazine*, c. 25, no: 2, 2004, s. 10-30. Şu internet adresimde: http://www.aaai.org/ojs/index.php/aimagazine/article/viewFile/1758/1656.

ekipti. Aslında, onların çalışmasına "bilgisayarla görünün ilk önemli gerçek dünya uygulaması" denmiştir.[34]

Bunların, çeşitli Mercedes-Benz taşıtlarına yerleştirilmiş olan görme ve denetim sistemleri, uzun mesafelerde özerk araba sürebilmiş, şerit değiştirip yavaş arabaları sollayabilmiştir. 1995'te bunların VaMP taşıtı (Mercedes-Benz 500 SEL), Münih'ten Danimarka'nın Odense kentine gidiş gelişi kapsayan 1.785 kilometrelik yolculuğu saatte 175 kilometreyi aşan hızlarla tamamlamıştır. Yolculuğun yaklaşık %95'i, toplamda 400 şerit değiştirme manevrasıyla birlikte tamamen özerk olarak kat edilmiştir. Bunların özerk taşıtına ve görme projelerine dair ek ayrıntılar, Dickmann'ın "Dinamik Görme" hakkındaki kitabında bulunabilir.[35]

Yer sıkıntısından ötürü, başka nesne izleme projelerini betimleyemiyorum, ama bir iki tanesinden daha bahsedeceğim. Jitendra Malik, Berkeley California Üniversitesi'nde, (başka görme çalışmalarıyla birlikte) nesne-takibi araştırmalarının yürütüldüğü bir görme ekibinin başındadır.[36] Büyük Britanya'da Leeds Üniversitesi Görme Ekibinde, örneğin futbol oyuncularını ve otomobilleri izleme çalışmaları yapılmıştır.[37] Leeds'te, "nesnelerin uzamsal-zamansal sürekliliği üzerindeki temel kısıtlamaları" uslamlayarak nesne izleme hassasiyetini iyileştirme amacını taşıyan başka çalışmalar da yapılıyor.[38]

30.2.3 Hiyerarşik Modeller

Bilgisayarla görmede (hatta belki tüm YZ dallarında) potansiyel olarak en çok umut vaat eden gelişmelerden birinin hiyerarşik modelleri içerdiğine inanıyorum. Bu modellerin farklı sürümleri bulunur ve bunları oluşturmanın farklı yöntemleri vardır, fakat ayrıntılara yeterince uzaktan bakarsak, hepsinin benzer yapıları ve nitelikleri olduğunu görürüz. Öncelikle, daha üst düzey öbekler oluşturmak amacıyla ham pikseller uzamsal olarak bir araya getirilir (bazı sistemlerdeyse

34 Alıntı, Sebastian Thrun'un e-postasından, 27 Haziran 2008.

35 Ernst D. Dickmanns, *Dynamic Vision for Perception and Control of Motion*, Berlin: Springer-Verlag, 2007.

36 Bkz. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/vision_group.html.

37 Bkz. http://www.comp.leeds.ac.uk/vision/behaviour.html.

38 Bkz. http://www.comp.leeds.ac.uk/vision/cogvis/continuity.html ve Brandon Bennett ve diğ., "Enhanced Tracking and Recognition of Moving Objects by Reasoning about Spatio-Temporal Continuity", *Image and Vision Computing*, c. 26. no: 1, Ocak 2008, s. 67-81. Şu internet adresinde: http://www.comp.leeds.ac.uk/qsr/pub/Bennett08imavis.pdf.

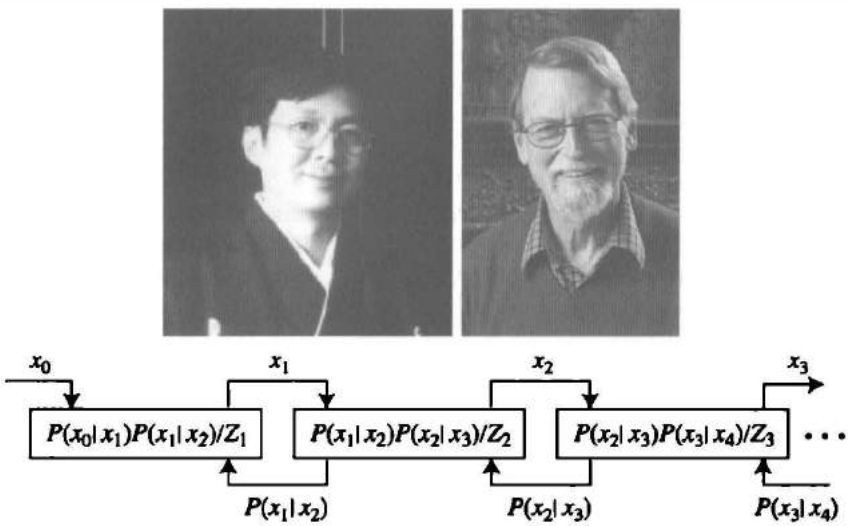
zamansal olarak). Bu öbekler, işlenecek olan görüntü türüne uygun küçük kenar çizgileri, köşeler ya da başka ilkel görüntü bileşenleri içerebilir. Hiyerarşinin bir sonraki düzeyinde, birinci düzey öbeklenmeler tekrar oluşturulup daha üst düzey bileşenler meydana getirilir, ta ki görüntüdeki tanınır nesneler en üst düzeyde temsil edilene dek.

Bu sistemlerde kullanılan pek çok fikir, eski sistemlerin (Pandemonium, Neocognitron,[39] Karatahta mimarileri, konuşma tanıma sistemleri, PDP yinelemeli ağlar gibi) belirli özelliklerini anımsatır, fakat daha yeni sistemlerin pek çoğu, daha önce tek başına hiçbir sistemin yapmadığı şekilde bu özellikleri birleştirip kapsamlarını genişletiyor. Özellikle, izninizle şunlardan bahsedeyim:

1. Çeşitli düzeylerde oluşturulmuş öbekler, muazzam veri kümeleri kullanılarak *öğrenilir*; bunlar, önceden elle tasarlanmış değildir. Üstelik kimi sistemlerde bu öğrenme işlemi "gözetimsizdir"; bu işlem, nesne kimlikleri hakkında bilgi temin eden zamansal görüntü akışında nesnenin sürekli görünmesini temel alır.
2. Her düzeyde ortaya çıkan öbeklenmeler, başlıca gösterimsel ve bilişimsel mekanizmaları temin eden olasılıksal grafik modellerin (rastlantısal Markov alanları gibi) olasılıklarıyla nitelenir.
3. Belli bir düzeydeki öbeklenmelerin olasılıkları, sırf daha üst düzeydeki öbeklenmelerin olasılıklarını değil, ayrıca aynı ya da daha alt düzeydeki öbeklenmerin de olasılıklarını etkiler. Yani, Pandemonium'un ve ileri-beslemeli sinir ağlarının aksine, bu yeni sistemlerde üst düzeylerden alt düzeylere "geriye dönük" bağlantılar da vardır. Geriye dönük bu bağlantılar, görüntüde belirsiz görünse de ve var olmasa da, büyük olasılıkla görüntüde var olan şeyler hakkında sistemin öngörüde bulunmasına olanak tanır.

Kimi araştırmacılar, hiyerarşik modellerin geliştirilmesiyle ilgileniyor. Bunların bazılarının hevesi, insanların ve primatların görsel korteksindeki saklama ve çıkarsama mekanizmalarını modellemeye dayanır. Yine de bunların modelleri, sinirbilimden gelen içgörüleri ve bulguları, hiyerarşik grafik modeller ve istatistiki örnek toplama teknikleri dahil

39 K. Fukushima, "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position", *Biological Cybernetics*, c. 36, no: 4, 1980, s. 93-202.



Resim 30.9. Tai Sing Lee (üstte solda), David Mumford (üstte sağda) ve görsel işleme katmanları (altta). (Fotoğraflar, Tai Sing Lee ve David Mumford'un izniyle. Diyagram şu makaleden uyarlanmıştır: Tai Sing Lee ve David Mumford, "Hierarchical Inference in the Visual Cortex", *Journal of the Optical Society of America*, A, sayı 20, no. 7, Temmuz 2003.)

oldukça incelikli hesaplama tertibatlarıyla birleştiren YZ araştırmacılarına hayli ilginç gelir. Daha güçlü bilgisayarla görme sistemleri kurmak amacıyla, biyolojik açıdan makul olup olmadığına bakmaksızın hiyerarşik modeller ve ileri bilişim yöntemleri kullananlar da var.

Öncelikle, görsel korteksin işlemci katmanlarının hiyerarşisinin Resim 30.9'daki gibi modellenebileceğini söyleyen Tai Sing Lee (1961-) ile David Mumford'un (1937) önerisini anlatacağım.[40]

Lee-Mumford modelinde (kısmen, Ulf Grenander'in örüntü kuramına dair çalışmasına dayanır[41]), resme soldan giren "aşağıdan yukarı" görsel gözlemler, sağ tarafta oluşturulan "yukarıdan aşağı" hipotezlerle bütünleştirilir. Bu diyagramda x_0'ın, piksel dizisi halindeki bir görüntünün gösterimi yerine geçtiğini düşünün. x_1 ise, görüntünün çok daha soyut bir gösterimidir, örneğin kısa çizgi bölütleri gibi özellikler bağlamındaki bir gösterimidir. Bir adım sağa ilerlediğimizde, hesaplamalar daha da soyut bir gösterimi, yani x_2'yi oluşturur; bu da x_1'le ilgili bir hipotez olarak hizmet eder. Kutu içindeki formüller

40 Tai Sing Lee ve David Mumford, "Hierarchical Inference in the Visual Cortex", *Journal of the Optical Society of America A*, c. 20, no: 7, Temmuz 2003.

41 Ulf Grenander, *General Pattern Theory*, Oxford: Oxford University Press, 1993.

(bunları açıklamaya kalkışacak değilim) ve bunları bağlayan oklar, her düzeyde bir gösterimin, x_i'nin olasılığının, hem x_{i-1}'e (girdi sayılır) hem de x_{i+1}'e (x_i'le ilgili bir hipotez sayılır) bağlı olduğunu gösteriyor.

Lee ve Mumford, bu ileri beslemeli-geribildirim sürecini şöyle betimliyor:

> İleribesleme girdisi, hipotezlerin oluşturulmasını sağlar ve daha üst çıkarsama alanlarından gelen geribesleme, daha erken düzeylerdeki çıkarsamaları şekillendiren önkoşulları temin eder. Ne ileribesleme iletileri, ne de geribesleme iletileri durağandır: Bir görüntünün yorumu ilerledikçe, yeni önkoşulları geribesleyen yeni üst düzey yorumlar ortaya çıkar ve düşük düzey yorumlar incelik kazandıkça, ileribesleme iletisi iyileştirilir. Bu tür hiyerarşik Bayesçi çıkarsama, birden fazla alanda eş zamanlı ilerleyebilir... bu esnada görsel hiyerarşideki ardışık korteks alanları, yorum belirmeye başladıkça birbirlerinin çıkarsamalarını küçük döngüler içinde hızlı ve kesintisiz olarak [kısıtlarlar]. Bütün olarak bu tür bir sistemin, tüm alt düzey ve üst düzey bilgi kaynaklarını kapsayan, görsel manzaranın tutarlı bir yorumu olarak çabucak kaynaşmasını umabiliriz; fakat birtakım sorunlar yok değil...

Bu "sorunlar"dan biri şu: Düzeylerden hiçbiri kendi yorumundan kesin emin olamayacağı için, birden fazla yüksek olasılıklı küresel yorumlar var olabilir. Lee ve Mumford, süregiden başka YZ çalışmalarını temel alan bir çare öneriyor, yani hiçbir düzeyde "sonuç çıkarmaya hemen atlamamak" ama bütün zincir için genel bir yorum, en olası yorum olarak belirdiği müddetçe, bazı yüksek olasılıklı yorumların "tedavülde kalmasına" olanak tanımak. (Barrow ve Tenenbaum sistemlerinin ikisinin, yani MSYS'nin ve içkin imgeleri kullanan sistemin, bunu daha 1970'lerde yapmaya çalıştığını anımsayın.) Lee ve Mumford, fikirlerini geçerlemek için parçacık süzgeci tekniğini kullanmayı önermişlerdi; bu teknik, daha önce bahsettiğim gibi, her düzeyde yorumların olasılık dağılımını göstermek amacıyla ağırlıklı örneklem kullanır. Deneyimlerden öğrenilen bu dağılımları ve düzeyleri birbirine bağlayan formülleri kullanan sistemler, her düzeyde en olası yorum üzerinde karar kılabilir.

Her ne kadar Lee ve Mumford, kendi olasılık hesaplamaları için, rastlantısal Markov alanlarının kullanımı gibi, uygulamaya yönelik fikirler ileri sürmüş olsa da, kendi modellerini hayata geçirmemişlerdir. Şöyle açıklarlar:

> Önerimize eşlik eden bir benzetim ileri sürmedik; bunun sebebi kısmen, pek çok ayrıntının henüz çözülmemiş olması ve kısmen, model tercihinin



Resim 30.10. Suratın daha berrak görülmesi. (Şu makaleden uyarlanmıştır. Tai Sing Lee ve David Mumford, "Hierarchical Inference in the Visual Cortex", *Journal of the Optical Society of America*, A, c. 20, no. 7, Temmuz 2003.)

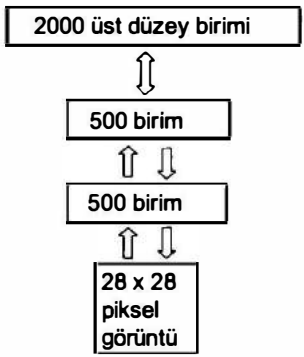
oldukça kısıtlamasız oluşu ile belirli herhangi bir benzetimin, bizimki gibi üst düzey hipoteze ancak zayıf bir destek sağlamasıdır.

Fakat, modellerini destekleyen sinir fizyolojisi ve psikolojik fizik bulgularına atıfta bulunurlar. Kendilerininki gibi modellerin, görüntülerin işlenmesini nasıl iyileştirdiğini açıklamak için Resim 30.10'daki çizimi kullanırlar. Görüntünün parlak renkle aydınlatılmış kısmı, bunun bir yüz görüntüsü olduğunu akla getiriyor. Yeri gelince bu hipotez, görüntünün alt düzey işlenişini, yüzün sönük kenar çizgisine karşı daha hassas hale getiriyor; böylece saptanmasına olanak tanıyor. (İnsanlar, "ah evet, o çizgiyi görüyorum", diyebilir.)

Geoffrey E. Hinton (1947), Simon Osindero (1977-) ve Yee-Whye Teh (1977), "derin inanç ağı" denen başka bir hiyerarşik model için (görece karmaşık) gözetimsiz öğrenme yöntemleri tasarlamışlardı.[42] Bunun, Resim 30.11'de gösterilen sürümüyle deneyler yürüttüler. Yapının bütünü, katmanlı bir sinir ağıdır ve üst düzey, aşağı düzeydeki birimlerle çift yönlü bağlanan 2.000 birimden meydana gelir. Ağın eğitilmesi, en dipten başlayarak adım adım, düzey düzey ilerler. Her

42 Geoffrey E. Hinton, Simon Osindero ve Yee-Whye Teh, "A Fast Learning Algorithm for Deep Belief Nets", *Neural Computation*, c. 18, no: 7, Temmuz 2006, s. 1527-1554. Şu internet adresinde: http://www.cs.utoronto.ca/~hinton/absps/ncfast.pdf.





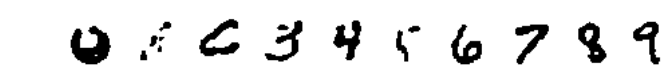
Resim 30.11. Geoffrey Hinton (solda) ve derin inanç ağı (sağda). (Fotoğraf, http://www.schoolarpedia.org/article/Image:Geoffnew3.jpg internet adresinden. Ağ diyagramı ve fotoğraf, Geoffrey Hinton'ın izniyle kullanılmıştır.)

düzey eğitildikten sonra, bunun ağırlıkları "dondurulur" ve buradan çıkan sonuçlar, bir üstteki düzeyi eğitirken girdi olarak kullanılır. Açgözlü denilen bu eğitim yöntemi, izlenen görüntülerin dağıtımına dair sağlam bir hiyerarşik modelin oluşmasıyla sonuçlanır.

Bu sistemin yaratıcıları, daha önce eğitilmiş bir hiyerarşik ağın tepesine on karar biriminin eklendiği deneyler de betimlemiştir. Ardından karar birimleri, her biri 28×28 pikselli imge olarak sunulan, elle işlenmiş basamaklar arasında ayrım yapmak üzere eğitilir. Büyük, standart bir basamak, veritabanı eğitimi için kullanılır ve bir başkasıysa sınama için kullanılır. Bu sonuçlar, daha geleneksel tekniklerin sonuçlarına üstün gelmiştir. Eğitilmiş ağın en üst düzeyinin "aklında ne olduğunu" görmek amacıyla, aşağı doğrultulu oklar, dip düzeyde, en üst düzeyden girilmiş etiket kodlarını temel alan görüntüler yaratılırken kullanılır. Yaratılan bu görüntülerin kimi örnekleri Resim 30.12'de gösteriliyor.

İlk Palm Pilot'un tasarımcısı Jeff Hawkins (1957-), neokorteksin hiyerarşik bir zamansal bellek olduğunu, bunun katmanlarının (aşağıdan yukarı doğru) duyu girdisi dizilerinin gitgide daha soyut gösterimlerini sakladığını ve işlevinin (yukarıdan aşağı), gelecekteki deneyimlere dair gitgide daha ayrıntılı öngörülerin yapılmasına olanak tanımak olduğunu söylemiştir.[43] Görsel korteksin, zamanla görüntü

43 Jeff Hawkins, Sandra Blakeslee'yle birlikte, *On Intelligence*, New York: Times Books, 2004.



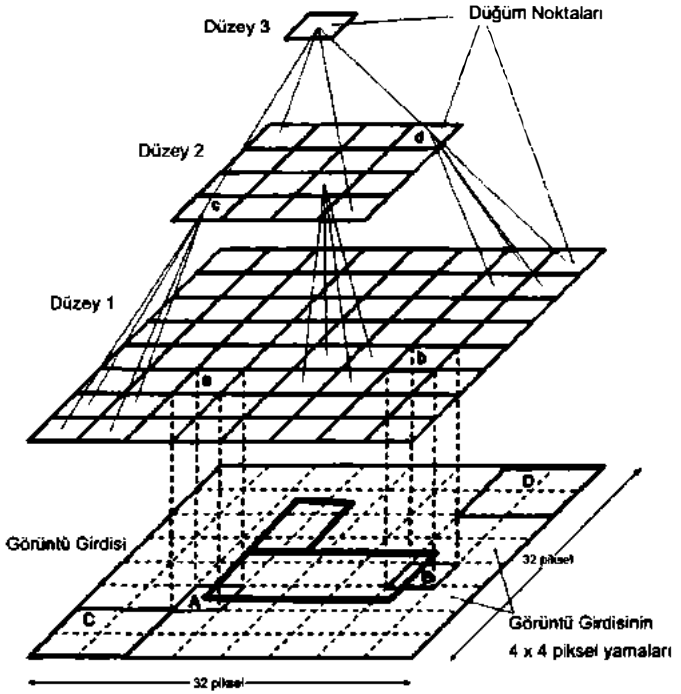
Resim 30.12. Eğitilmiş ağın yarattığı bazı görüntüler. (Geoffrey Hinton'ın izniyle kullanılmıştır.)

dizilerine maruz kalarak gözetimsiz bir tarzda öğrendiğini ileri sürer. Görüntüleri, zamanda kesintisiz olarak boy gösterdiklerinde gördüğümüz için, bunlar, görsel alanımıza giren nesnenin her imgesinin dizileri olmak zorunda; her ne kadar farklı konum, ölçek ve yönlenimlerde boy gösterseler de. Bu aynılık, hiyerarşinin tüm düzeylerinde öğrenme gösterimlerinde yararlanılan örtük bir etiketleme sağlar. Dahası, der Hawkins, hiyerarşik bellek ve bunun öğrenme yöntemleri, sadece görsel girdi için değil başka duyu kipleri için de kullanılır. Hiyerarşinin en yüksek düzeylerinde bu ayrı kipler birleşip, görmeye, dokunmaya ve işitmeye yaslanan duyu dünyamızın bütünleşik modelini sunar; bu modeli, az sonra ne olacağına dair öngörüler yaparken kullanırız.

Hawkins ve Stanfordlu doktora öğrencisi Dileep George (1977-), bu fikirleri temel alarak, "Hiyerarşik Zamansal Bellek" (HZB) dedikleri bir ağ modeli geliştirmişti.[44] George, kendi doktora tezinde, bu modelin, Resim 30.13'te açıklanan bir sürümünü hayata geçirmişti. Dip düzey, 32 × 32'lik bir piksel dizilişidir ve buna bir dizi görüntü sunulur. Birinci düzey, 8 × 8'lik ağ düğüm noktaları dizilişinden meydana gelir; her düğüm noktası 4 × 4'lük görüntü pikseli girdisi yamalarından girdi alır. Örneğin, "a" düğüm noktası, kendi "alıcı alanı"ndan girdi alır, yani "A" işaretli piksel yamasından ve "b" düğüm noktası, "B" işaretli piksel yamasından girdi alır. İkinci düzey, düğüm noktalarının 4 × 4'lük bir dizilişidir ve her düğüm noktası, birinci düzey düğüm noktalarının 2 × 2'lik kümelerinden girdi alır. Bu tür bir düzenek, üçüncü düzeydeki tek düğüm noktasına dek sürer. O düğüm noktasının amacı, görüntü girdilerinin sınıf etiketlerini yani kategorilerini tanımaktır.

Her katmandaki düğüm noktaları, alt katmandaki alıcı alanında yaygınlıkla boy gösteren dizileri tanımak üzere eğitilmiştir. Örneğin, Resim 30.13'te "a" işaretli birinci düzey düğüm noktası, kendi alıcı alanı "A"daki piksel öbeklerinin sık sık boy gösteren dizilerinin olasılıklarını temsil etmek üzere eğitilmiştir. Örneğin, bu tür bir yüksek

44 Dileep George, "How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition", doktora tezi, Elektrik Mühendisliği Bölümü, Stanford Üniversitesi, Haziran 2008. Şu internet adresinde: http://www.numenta.com/for-developers/education/DileepThesis.pdf.



30.13. HZB modeli (Dileep George'un izniyle kullanılmıştır).

olasılıklı dizi, sağa hareket eden küçük köşeler olabilir. Sırası gelince, ikinci düzeydeki düğüm noktaları, birinci düzeydeki alıcı alanlarındaki yüksek olasılıklı dizilerin sık sık boy gösteren dizilerinin olasılıklarını temsil etmek üzere eğitilir, vesaire. Ağın eğitilmesi, girdi imgesi olarak filmler sunulmasını içerir ve bu eğitim, hiyerarşide düzey düzey ilerler. Eğitimden sonra, her düzeyde temsil edilen dizilerin olasılıkları, yukarıdan gelen geribildirimle koşullanır. Örneğin, eğer "A" işaretli piksel yamasında küçük bir köşenin soldan sağa hareket ettiği bir film sunulursa ve eğer böyle küçük bir köşe, eğitim sırasında sık sık bu şekilde hareket etmişse, o zaman birinci düzeydeki "a" düğüm noktası, o köşenin bu devinime devam edeceğini öngörür.

Düzeyler hiyerarşisinde yukarı çıktıkça, her düğüm noktası, dolaysız da olsa, görüntünün gitgide daha büyük bölütlerinden girdi alır. Son olarak, en üstteki düğüm noktası (diyagramda üçüncü düzey), ağın görmüş olduğu görüntü kategorilerinin olasılık dağılımını temsil eder. Ağ, (eğitimden sonra) "tanıma kipi"nde işlerken, en üstteki düğüm noktası, retinaya düşen görüntünün en olası kategorisini

tanımlar. Bu ağ, George'un kendi doktora tezinde kullandığı çeşitli basit görüntüleri tanımayı öğrenebilmişti. George, bu tür ağları geliştirmek amacıyla Hawkins'in kurmuş olduğu Numenta adlı şirkette çalışmalarını sürdürüyor.

Her ne kadar şimdiye dek betimlenmiş modeller, algı görevleri uyarınca geliştirilmiş olsa da, birtakım iyileştirmelerle birlikte bu modeller, zeki etmenlerin genel mimari şemasının temeli olarak hizmet edebilir. (Bir sonraki bölüme bakınız.) Bu amaçla, söz konusu iyileştirmeler, başka şeylerin yanı sıra, var olan algı olanaklarının rehberlik ettiği eylem planlama ve gerçekleştirme imkânlarını içermelidir. Elbette, eğer bu modeller yeni-korteksin işleyişiyle alakalıysa (yandaşları, öyle olduklarını iddia ediyor), o zaman eylem planlaması ve icrası dahil, yeni-korteksin yaptıklarından fazlasını yapabilmeleri gerekir. Her durumda, korteks modeli araştırmaları, YZ araştırmacıları ile sinirbilimciler arasında bir işbirliği alanı açmıştır. Yeni-korteksin olasılık modellerini kurmuş olan Thomas Dean'in işaret ettiği üzere, "Korteks ölçeğindeki modellerin varoluşu, beyni anlamamızı kolaylaştırmakla kalmayacak, araştırmacıların, biyolojiden öğrenilen derslerle son model yapay öğrenme tekniklerini birleştirip biyolojik ve geleneksel yaklaşımların en iyi taraflarını bir araya getiren melez sistemler tasarlamasına olanak tanıyacaktır."[45]

Yer azlığı yüzünden, hiyerarşik model geliştirmiş önde gelen başka görme araştırmacılarının çalışmalarını anlatamıyorum, fakat birkaç tanesinden kısaca bahsedeceğim; ilgilenen okurlar, bunların internet sitesine bakabilir.

Tomaso Poggio ve MIT Beyin Araştırmaları McGovern Enstitüsü'ndeki çalışma arkadaşları, beynin, görsel nesneleri nasıl tanıdığını modellemeye yardımcı olması amacıyla matematiksel ve istatistiki öğrenme mekanizmaları uyguluyor.[46] Uygulama alanlarından biri, yüz tanıma olmuştur.[47]

45 Eskiden Brown Üniversitesi'nde bilgisayar bilimi profesörü olan Dean, şimdi Google'ın bilim ekibindedir. Brown'daki internet sayfası: http://www.cs.brown.edu/research/projects/cortex.html.

46 Örneğin bkz. Thomas Serre, Aude Oliva ve Tomaso Poggio. "A Feedforward Architecture Accounts for Rapid Categorization", *Proceedings of the National Academy of Sciences (PNAS)*, c. 104, no: 15, 2007, s. 6424-6429. Şu internet adresinde: http://cbcl.mit.edu/projects/cbcl/publications/ps/serre-PNAS-4-07.pdf.

47 Örneğin bkz. R. Brunelli ve Tomaso Poggio, "Face Recognition: Features Versus Templates", *IEEE PAMI*, c. 15, 1993, s. 1042-1052. Şu internet adresinde: http://cbcl.mit.edu/people/poggio/journals/brunelli-poggio-IEEE-PAMI-1993.pdf.

New York Üniversitesi Courant Matematik Bilimleri Enstitüsü Bilişimsel ve Biyolojik Öğrenme Laboratuvarı'nda Yann LeCun, "derin mimariler" dediği konuda çalışıyor, yani "doğrusal olmayan ve eğitilebilir modüllerden oluşan çoklu katmanların meydana getirdiği" mimarileri inceliyor. Ekibinin odaklandığı konulardan biri, "enerji-tabanlı modeller"dir (ETM); bunlar, fiziksel enerjiyle ilintili bir kavramın değişkenlerle (sıradan olasılıklar yerine) bağdaştırıldığı grafiksel modellerdir.[48]

30.2.4 Görüntü Dilbilgileri

Doğal dil işlemede dilbilgileri ve sözdizimsel çözümlemelerin başarılı kullanımından ötürü, benzer fikirlerin resim ve görüntü işlemede kullanımına yönelik girişimlerin olması şaşırtıcı değil. Aslında, Russell Kirsch verdiği bir söyleşide şunu söylemiştir: "1957'ye gelindiğinde, dilbilimcilerin bilgisayarda dilbilgisiyle yaptıkları işler merakımı çeker olmuştu... Böylece, bana bariz görünen o soruyu sordum: Aynısını resimlerle de yapabilir miyiz?"[49] Derken, Kirsch ve karısı Joan, resim çözümleyen (ve üreten) bir dilbilgisi geliştirdiler.[50] Yukarıda andığımız söyleşiye göre, dilbilgilerini kullandıkları bilgisayar, "[ressam Richard Diebenkorn] tarzında çizgiler ve örüntüler yaratabiliyordu. Resim bittiğinde, karı koca Kirschler, yaratılmış bu imgeyi bizzat sanatçıya gösterdiler. Diebenkorn, bu resmin, kendi yaptığı tablolalara epey benzediğini kabul etmiştir. Aslında bilgisayar benzetimi, Diebenkorn'un daha önce yaptığı bir tabloyla neredeyse özdeşti."

"Resim dilbilgileri" üzerine başka bir çalışma da Profesör Azriel Rosenfield ve Maryland Üniversitesi'ndeki ekibi tarafından yapılmıştır.[51]

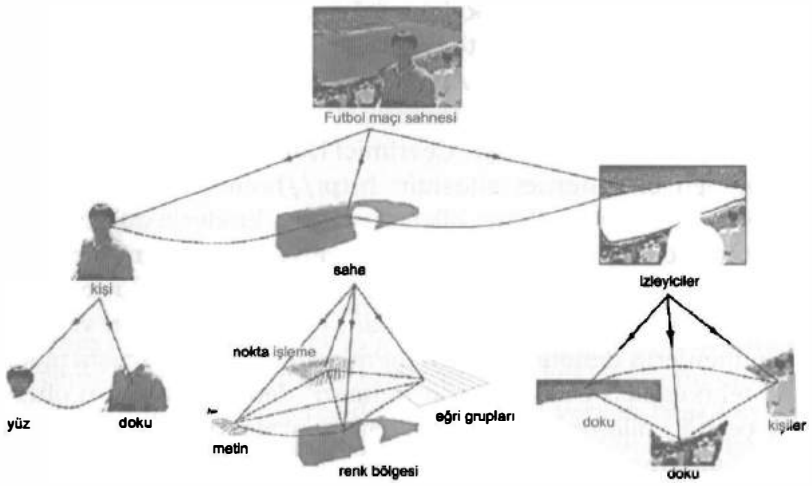
UCLA Görüntü ve Görme Bilimleri Merkezini yöneten Song-Chun Zu (1969-), görmeyle ilgili problemlere, istatistiki ve fizik-temelli tek-

48 Bkz. Yann LeCun ve diğ., "A Tutorial on Energy-Based Learning", *Predicting Structured Data* içinde, G. Bakir ve diğ. (ed.), Cambridge, MA: MIT Press, 2006. Şu internet adresinde: http://yann.lecun.com/exdb/publis/pdf/lecun-06.pdf.

49 Şu söyleşiden: "Russell Kirsch: The Language of Shapes", söyleşiyi yapan Kennedy Smith, *Portland's Daily Journal of Commerce,* 28 Temmuz 2006. Şu internet adresinde: http://www.mel.nist.gov/msid/shape.pdf.

50 Russell Kirsch ve Joan Kirsch, "The Structure of Paintings: Formal Grammar and Design", *Environment and Planning B: Planning and Design,* c. 13, 1986, s. 163-176. Bkz. http://www.nist.gov/msidlibrary/doc/kirsch_1986_structure.pdf.

51 Azriel Rosenfeld, "Isotonic Grammars, Parallel Grammars, and Picture Grammars", *Machine Intelligence* 6 içinde, Bernard Meltzer ve Donald Michie (ed.), Edinburgh: Edinburgh University Press, 1971, s. 281-294.



Resim 30.14. Bir görüntüyü çözümlemek. (Song-Chun Zhu ve David Mumford, "A Stochastic Grammar of Images", *Foundations and Trends in Computer Graphics and Vision*, sayı 2, no. 4, s. 259-362, 2006.)

nikler uygulamıştır. Çalışma arkadaşlarıyla birlikte, "rastlantısal değişkenli görüntü dilbilgisi"ni geliştirdi; bu teknik, görüntüleri bileşenlerine ayrıştırmak için kullanılabiliyordu.[52] (Ayrıştırma yöntemi, Bölüm 30.2.3'de betimlenmiş olan Lee ve Mumford'un çalışmasındaki kimi fikirleri hayata geçirir.) Resim 30.14, ayrıştırma ağacı olarak sunulmuş bir görüntüyü ayrıştırmanın örneğini sunuyor.

Bilgisayarla görme çalışmaları son birkaç yılda büyük ilerleme kaydetmiştir ve pek çok uygulamanın önemli bir parçasıdır, örneğin olay saptama (trafik ihlalleri gibi), tıbbi görüntüleme, nesnelerin izlenmesi, (yüzler, yayalar, taşıtlar gibi), görsel protezler, fotoğraflarda belli nesneleri bulma, depolarda envanter denetimi, robot taşıtların yön bulması ve harita çıkarması, yazı karakteri ve elyazısı tanıma, tehlike uyarı sistemleri, süreç denetimi, devre kartı kontrolü, meyve ve sebze derecelendirme, topografik haritalama, orman etütleri, kalabalıkta yüzleri tanıma ve saptama, internette görüntü arama, görüntü sıkıştırması ve tarımda mahsul saptaması gibi.[53]

52 Song-Chun Zhu ve David Mumford, "A Stochastic Grammar of Images", *Foundations and Trends in Computer Graphics and Vision*, c. 2, no: 4, 2006, s. 259-362. Şu internet adresinde: http://www.stat.ucla.edu/~sczhu/papers/Reprint_Grammar.pdf.

53 British Columbia Üniversitesi Bilgisayar Bilimi Bölümü profesörlerinden David Lowe, bilgisayarla görme ürünlerini satan şirketleri internet sitesinde sıralıyor: http://www.cs.ubc.ca/spider/lowe/vision.html.

Daha fazlasını öğrenmek isteyen okurlar, ders kitapları, bilgisayarla görme yayınları ve internette zengin bir malzeme bulacaktır. *Computer Vision – A Modern Approach* [Bilgisayarla Görme; Modern Yaklaşım], tavsiye edilen bir kitaptır.[54] Edinburgh Üniversitesi'nde Robert B. Fisher'in hazırladığı "Çevrimiçi Bilgisayarla Görme Kaynağı" tavsiye edilen bir internet sitesidir: http://homepages.inf.ed.ac.uk/rbf/CVonline/. Bu site, ilginç bilgilere ulaşan linklerle doludur.

YZ tarihçemin bu kısmında betimlenmiş YZ'nin tüm yeni teknik aygıtlarını gördükten sonra, bunların hepsini nasıl bir araya getirip zeki bir tarzla akıl yürüten, plan yapan, eyleyen, algılayan ve iletişim kuran etmenlerin denetleneceğini merak edebilirsiniz. Araştırmacılar, bileşen teknolojileri bütünleştirip "mimari" dedikleri yapıları oluşturmanın çeşitli yollarını bulmuştur. Bir sonraki bölümde bunlardan bazılarını anlatıyorum.

54 David Forsythe ve Jean Ponce, *Computer Vision – A Modern Approach*, New York: Prentice Hall, 2002. Bölümlerin tam taslaklarının çevrimiçi versiyonu şu internet adresinde: http://decsai.ugr.es/mia/complementario/tl/book3chaps.html.

31
ZEKİ SİSTEM MİMARİLERİ

Bilgisayar bilimcileri, pek çok uzman altprogramdan meydana gelen büyük programları oluşturmanın çeşitli yollarını geliştirmiştir. Çoğu programın işleyişini denetleyen geleneksel çerçeve, komutlarını adım adım veren, bellekteki verilere erişip bunları saklayan, bu gibi veriler üzerinde çeşitli işlemler gerçekleştiren ve izin verilen başka eylemlerde de bulunan bir ana program içerir. Ana programdaki komutlardan bazıları, bir altprogramı "çağırıp" kontrolü ona bırakmak olabilir. Ardından altprogramlar da başka altprogramları çağırır, vesaire. Bir altprogram, çağrılma sebebi olan işi bitirince, denetim, onu çağırmış olan programa döner, bu da öbür altprogramı çağırır ve denetim en sonunda ana programa geçene dek bu böyle sürer. En sonunda, yapılması gereken iş başarıldığı için ana program işlemi keser ya da işlemeye devam eder (prensipte sonsuza dek), çünkü işi asla bitmiyordur, tıpkı isteyenlere uçak bileti rezervasyonu yapan program gibi. Bu yapıya, von Neumann mimarisi denir.

Bu genel fikre pek çok incelik katılmıştır. Örneğin, programlara ve altprogramlara "kesintiler" dahil edilebilir. Bunlar, bizzat bilgisayar sisteminin ya da çevrenin içindeki özel koşulları kollar; bu koşullar oluştuğunda, denetim, bu gibi koşullarla başa çıkabilecek programlara geçer. Örneğin bilgisayar işletim sistemleri, kullanıcı girdilerine ve bilgisayar donanımında olan bitene tepki verebilen kesintileri kullanır.

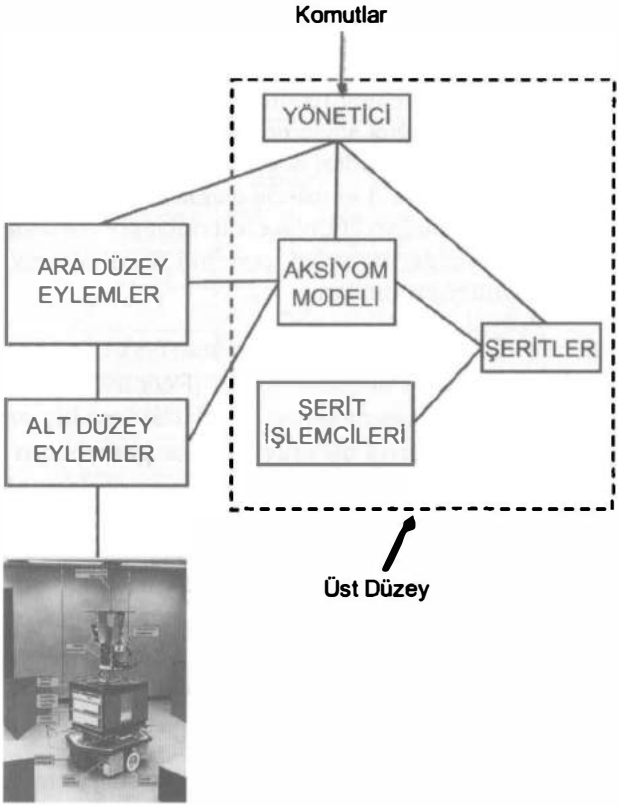
İlk YZ programları, von Neumann mimarisi kullanan bilgisayarlarda çalışıyordu, dolayısıyla programların mimarisinin (yani kendilerini düzenleme biçiminin), bilgisayarın von Neumann tarzı işletimine bağlı kalması doğaldı. Yavaş yavaş zamanla, aslında bilgisayarı denetleyen alt düzey programlar, programcıların fark etme ihtiyacı duymayacağı biçimde gittikçe daha karmaşıklaştığı halde o bilgisayarlarda çalışmaya devam etmişlerdi. Örneğin, LISP'le yazılmış programların çalışması için önemli olan yeniliklerden biri, bilgisayarın değerli saklama kaynaklarını daha verimli kullanmayı içeriyordu. Çöp toplama denilen altprogramlar, bir daha asla kullanılmayacak liste yapılarını bulmak amacıyla zaman zaman bilgisayarı tarıyordu. Böylece, bu

yapıları saklamaya ayrılmış bellek, yeni liste yapılarının saklanması amacıyla kullanılabilirdi. Programcılar, alt düzey bilgisayar yazılımı mimarisinin bu özelliğini görmezden gelip, sanki ellerinde bol bol bellek alanı varmış gibi kendi von Neumann tarzı, sıralı işleyen programlarını yazmayı sürdürebiliyordu.

Komutları birbiri ardına sırayla yerine getiren von Neumann fikrinin aksine, pek çok komutun aynı anda gerçekleştiği bir mimari düşünülebilir. Bu tür bir "paralellik", ya programları gerçekten de birkaç donanım işlemcisine yüklenmesiyle ya da programların aslında sırayla gerçekleştiği ama programcıların, işin aslını bilseler de eşzamanlı çalıştıklarını düşündüğü daha basit bir von Neumann mimarisinin paralel işleminin *benzetimi* yapılarak elde edilebilir. Örneğin, sinir ağlarının simgesel olmayan dünyasında, eşzamanlı işleyen bir grup sinirsel öğe düşünülebilir, gerçi bu ağların benzetimleri, her sinirsel öğeyi tek tek sırayla göz önüne almak zorundadır. Pandemonium'da, cinler (*demon*) (bazıları belki sinirsel öğeler tarafından bazıları da programlar tarafından yürürlüğe sokulmuştur) büyük olasılıkla paralel olarak işleyebilir, fakat Selfridge'in programları bu tür bir paralelliği taklit etmek zorundadır. Paralelliğin benzetimi ayrıca, "zaman paylaşımı" sistemiyle de elde edilebilir; bu sistemde kullanıcı (ya da farklı kullanıcılar), tüm programların eşzamanlı çalıştığını düşünülebilir.

UNIX, Windows ya da Mac OS gibi günümüzün bilgisayar "işletim sistemleri", düzenlenişi (yani mimarisi) çok dikkatli tasarlanması gereken çok karmaşık program öbekleridir. Hem gerçek paralel donanımdan (çok çekirdekli sistemler denir) hem de zaman paylaşımından yararlanırlar, böylece kullanıcılar, örneğin e-posta programlarını çizelge programlarıyla eş zamanlı olarak (öyle zannederler) çalıştırabilir.

Bu bölümde, araştırmacıların, zekâ eseri davranış elde etmek amacıyla programlarını düzenleme yöntemlerinden bazılarını anlatıyorum. Bunlardan bazılarına mühendislik ve hesaplamaya dair fikirler ilham vermişken, kimilerini ise psikoloji verilerini modellemeye çabalayan bilişsel bilim esinlemiştir. Hatta bazılarını, çeşitli beyin bölgelerinin işlevleriyle ilgili fikirler etkilemiştir. Bu mimarilerin pek çoğunda paralel işletim benimsenmiştir, gerçi çoğunlukla paralellik benzetimle elde edilen türdendir.



Resim 31.1. Shakey'nin üç düzeyli mimarisi.

31.1 HESAPLAMA MİMARİLERİ

31.1.1 Üç Katmanlı Mimariler

YZ sistemi Shakey'nin bileşenlerinin, yüksek-, ara- ve alt-düzey öbekler olarak, "üç katmanlı" bir mimari olarak nasıl örgütlendiğini zaten betimlemiştim. Resim 31.1'de, Shakey programlarının ve verilerinin düzeyler olarak nasıl kümelendirildiğini gösteriyorum. Bu düzeylerdeki programlar arasındaki etkileşim, bağlantı çizgileriyle gösterilmiştir. Shakey'nin tüm algısal ve temel devinim programları, alt düzey eylemlere iliştirilmişken, ara-düzey eylemler, alt düzey eylemleri çeşitli yollarla birleştirip belirli sıradan görevleri yerine getirebilir. Üst düzey ise, planlama ve planların gerçekleşmesinin genelinden sorumludur.

Shakey'nin kullandıklarına benzer üç katmanlı mimarilerden, başka robot sistemlerinde de yararlanılmıştır (ve hâlâ yararlanılıyor).

Bu mimarileri Jet Propulsion Laboratuvarı'nda kullanmış olan araştırmacı Erann Gat'ın yazın tarama makalesinde belirttiği gibi:[1]

> Üç katmanlı mimari, hareketli robotları denetleyen etkin algoritmaların üç ayrı kategoriye ayrıldığını söyleyen görgül gözlemden doğar: 1) Çok az ya da sıfır iç durumla, sensörleri doğrudan eylemcilerle eşleştiren tepkisel denetim algoritmaları, 2) kapsamlı olarak iç duruma bağlı olan ama hiç arama yapmayan olağan etkinlik dizilerini idare eden algoritmalar, 3) planlayıcılar gibi, zaman harcayan (çevrenin değişme hızıyla ilintili olarak), arama tabanlı algoritmalar.

Gat'ın betimlemiş olduğu üç katmanlı mimarilerin bazıları, R. James Firby'nin "Tepkisel Eylem Paketleri" (TEP) [Reactive Action Packages] kullanan üç katmanlı şemasını temel alır.[2] TEP'ler, bir araya toplanmalarıyla ve farklı durumlarda bir görevi yerine getirmenin bilinen tüm yollarını betimlemeleriyle, uzaktan tepkisel programlara oldukça benzerler.[3]

Üç katmanlı mimarinin modern bir örneği, Ernst D. Dickmanns ile çalışma arkadaşlarının betimlemiş olduğu, Almanların sürücüsüz "gören yolcu arabası"nın kullandığı VaMoRs-P'dir.[4] Sistemlerinin mimarisine dair diyagramlardan biri Resim 31.2'de gösteriliyor.

31.1.2 Çokkatmanlı Mimariler

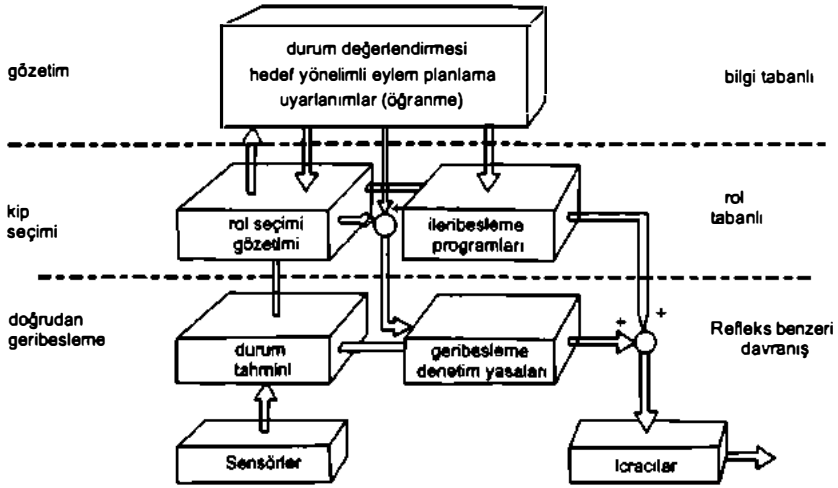
Hepsi de bir planlama düzeyi içeren üç katmanlı şemalara seçenek olarak, Rodney Brooks ve başkaları, planlama gereksinimi olmaksızın çevredeki değişikliklere (duyumsandığı haliyle) doğrudan tepki veren robot eylemlerini denetleyen mimariler önermiştir. Aslına "kapsama mimarileri" denilen bu yapılara daha sonra "davranış-tabanlı" mimari denmiştir, çünkü özellikle programlanmış robot davranışlarından

1 Erann Gat, "On Three-Layer Architectures", şu yayında: David Kortenkamp, R. Peter Bonnasso ve Robin Murphy (ed.), Cambridge, MA: MIT Press, 1998, s. 195-210. Şu internet adresinde: http://www.flownet.com/ron/papers/tla.pdf.

2 R. James Firby, "Adaptive Execution in Dynamic Domains", doktora tezi, Yale Üniversitesi, Bilgisayar Bilimi Bölümü Teknik Raporu no: 672, 1989. Özet için bkz. R. James Firby, "An Investigation into Reactive Planning in Complex Domains", *Proceedings of AAAI-87*, s. 202-206, 1987.

3 Üç katmanlı mimariler dahil, robotları denetlemeye yarayan başka mimariler şu kitapta betimlenmiştir: Bruno Siciliano ve Oussama Khatib (ed.), *Springer Handbook of Robotics*, Berlin and Heidelberg: Springer-Verlag, 2008.

4 Ernst D. Dickmanns ve diğ., "The Seeing Passenger Car 'VaMoRs-P,'" *Proceedings of the IEEE 1994 Symposium on Intelligent Vehicles*, 24-26 Ekim 1994, s. 68-73.



Resim 31.2. Sürücüsüz arabanın üç katmanlı mimarisi. (Ernst D. Dickmanns izniyle kullanılmıştır.)

oluşuyorlardı.[5] Davranış tabanlı mimari tiplerinden biri Resim 31.3'te gösteriliyor.

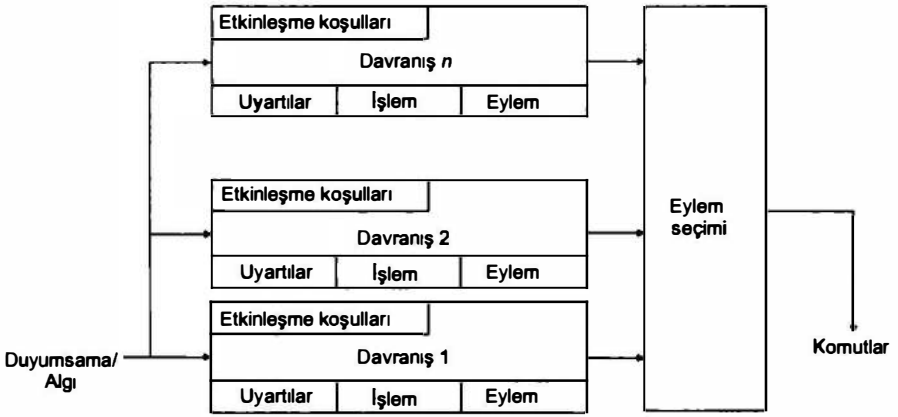
Örneğin, "etrafta gezinme", "engellerden sakınma" ve "keşfetme" gibi farklı davranışlar, düzeyler olarak düzenlenmiştir ve her biri kendi çevresel uyartı kümesine tepki verir, ayrıca her biri, duyumsanan duruma göre robotu denetleyebilir. Çevrede olan bitenle bu yakın eşleşme ve etkileşim, bazılarının "kendi kendine beliren davranış" dediği davranışları doğurur. Maja Matarić ile François Michaud'nun belirttikleri gibi:[6]

> Örneğin, öteki robotlarla sürü oluşturan bir robot, özgül bir *sürüleşme* davranışı içermeyebilir; bunun yerine, çevreyle ve öteki robotlarla etkileşimi sonucunda sürüleşme ortaya çıkabilir, gerçi sürüleşme sadece *çarpışmalardan kaçınma*, *topluluğa yakın durma* ve *hareketi sürdürme* davranışlarından meydana geliyor olabilir.

Ulusal Standartlar ve Teknoloji Enstitüsü'nde (USTE) (eskiden Ulusal Standartlar Bürosu) James Albus (1935- ; Resim 31.4), "referans model mimarisi" dediği yapıyı geliştirmişti. Bu mimari, daha önce USTE'de "zekâ kuramının" bileşeni olarak geliştirilmiş, çoklu "gerçek zamanlı

5 Konuyla ilgili bir inceleme için şu kitabın 38'inci bölümüne bakınız: Bruno Siciliano ve Oussama Khatib (ed.), *a.g.e.*

6 *A.g.e.*, s. 895.



Resim 31.3. Davranış tabanlı mimari.

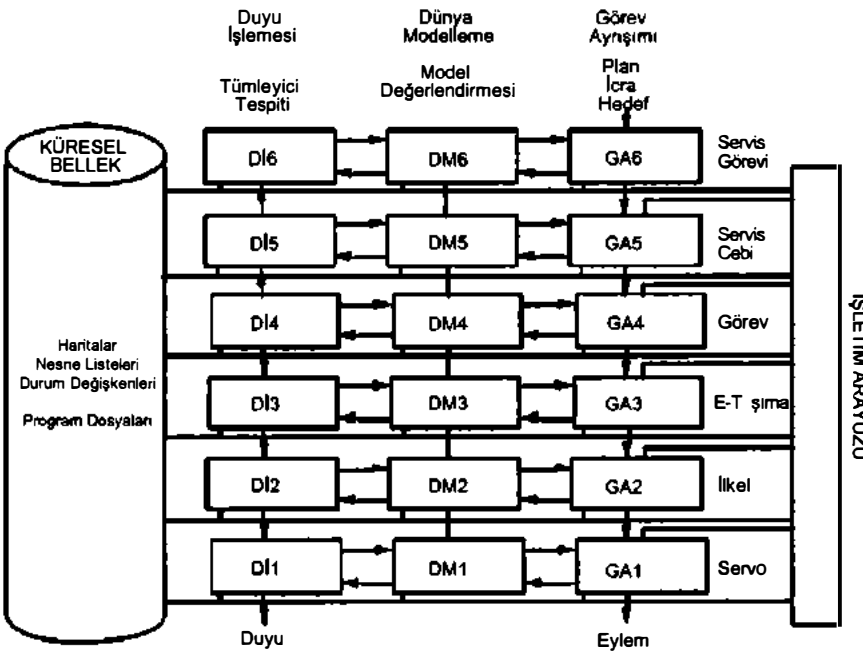


Resim 31.4. James Albus (Fotoğraf, James Albus'un izniyle.)

denetim sistemleri"nin (GDS) katmanlarından meydana geliyordu.[7] (Albus, kendi GDS modelinin aslen, David Marr'la birlikte tasarladıkları bir beyincik modelinden ilham aldığını söyler.[8]) Her GDS, "denetim problemini dört temel öğeye ayırır: davranış yaratımı (ya da görev ayrışımı), dünyayı modelleme, duyu işleme ve (daha yeni sürümlerde) değer

7 Bkz. James S. Albus, "Outline for a Theory of Intelligence", *IEEE Transactions on Systems, Man, and Cybernetics*, c. 21, no: 3, Mayıs/Haziran 1991.

8 Bkz. James S. Albus, "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)", *Journal of Dynamic Systems, Measurement and Control*, c. 97, American Society of Mechanical Engineers, Eylül 1975, s. 220-227; bir versiyonu şu internet adresinde: http://www4.cs.umanitoba.ca/~jacky/Robotics/Papers/Albus-NewApproachForManipulatorControlCMACS.pdf.

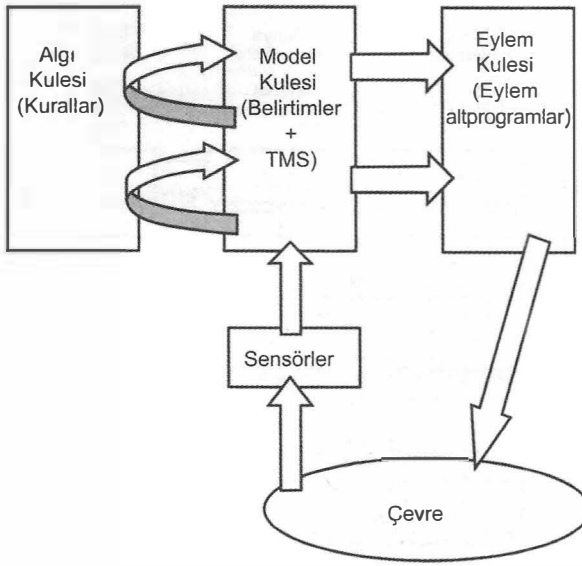


Resim 31.5. NASREM mimarisi. (Şu makalenin ikinci resminden: James S. Albus, "A Reference Model Architecture for Intelligent Systems Design", *An Introduction to Intelligent and Autonomous Control*, İkinci Bölüm, P. J. Antsaklis ve K. M. Passino (ed.), Dordrecht: Kluwer Academic Publishers, 1993, s. 27-56.)

yargısı. Bu öğeleri, belirli altsistemlerden sorumlu hesaplama düğüm noktaları olarak kümelendirir ve bu düğüm noktalarını hiyerarşik katmanlar olarak öyle düzenler ki her katmanın kendine özgü bir işlevselliği ve zamanlaması olur."[9]

GDS'lerin, NASREM (NASA/NBS Standard Reference Model) denilen katmanlı bir yapısı, uzay istasyonunda uzaktan denetimli uçuş hizmet robotu mimarisi olarak önerilmiştir (ama bildiğim kadarıyla hayata geçirilmemiştir). Bu yapı, Resim 31.5'te gösteriliyor. Her katmanda GDS birimleri, duyu işleme (Dİ) bileşenleri, dünya modelleme (DM) bileşenleri ve görev ayrışımı bileşenleri (GA) barındırır. En alt GDS katmanı, esasen bir servo denetimcidir; hiyerarşide yukarı çıkıldıkça, GDS'ler gitgide daha stratejik görevlerle başa çıkar. Albus ve NIST'teki

9 James S. Albus, "A Reference Model Architecture for Intelligent Systems Design", *An Introduction to Intelligent and Autonomous Control* içinde, İkinci Bölüm, Dordrecht: Kluwer Academic Publishers, 1993, s. 27-56.



Resim 31.6. Üçlü Kule mimarisi.

ekibi, katmanlı GDS'ler kullanan çeşitli mimariler geliştirmiştir.

Albus'un mimarisinden esinlenerek, Resim 31.6'da gösterdiğim "üçlü kule mimarisi" dediğim yapıyı geliştirdim.[10] Merkezde duran Model Kulesinin en alt düzeyi, sensörleri aracılığıyla çevreden doğrudan girdi alır ve bunları ilkel algısal belirtimler olarak depolar. Algı Kulesindeki programlar (kural olarak temsil edilir), bu ilkel belirtimleri daha soyut olarak temsil eder; bunlar, bu gösterimleri Model Kulesine ekler. Gitgide daha soyut gösterimler yaratma süreci, Algı ve Model Kulelerinde basamak basamak yükselir. Eylem Kulesinde, en alttaki eylem altprogramları basit reflekslerdir ve bunlara, ilkel belirtimlere denk gelen Model Kulesi belirtimleri başvurur. Daha karmaşık eylemlere, o eylemlere uygun daha soyut belirtimler başvurur. Yüksek düzey eylemler, süreç, çevreyi gerçekten etkileyen ilkel eylemlerde dibe vurana dek başka eylemlere başvurur. Eylem Kulesindeki eylemlerin hepsi, benim teleo-reaktif dilimle programlanmıştır (bkz. s. 425). Ardından, bu eylemlerin algılanan etkileri, Model Kulesindeki belirtimlerin değerini değiştirir, belki de farklı eylemlere başvurur. Süregi-

10 Nils J. Nilsson, "Teleo-Reactive Programs and the Triple-Tower Architecture", *Electronic Transactions on Artificial Intelligence*, c. 5, Kısım B, 2001, s. 99-110. Baskı sonrası versiyonu şu internet adresinde: http://www.ep.liu.se/ej/etai/2001/006/

den çevresel değişiklikleri aslına sadık kalarak modellemek amacıyla, model kulesinin parçası olarak bir Doğruluk Sürdürme Sistemi (DSS) dahil edilmiştir. DSS, Model Kulesinin o an var olan bileşenlerinden artık (algı kuralları aracılığıyla) türetilemeyecek belirtimleri ve değerleri Model Kulesinden siler. Bu mimarinin bildiğim tek uygulaması, blok sıralamasını taklit eden robotun denetimidir.[11]

Carnegie Mellon Üniversitesi'nde HEARSAY-II konuşma anlama sistemi için tasarlanmış olan Karatahta mimarisini zaten betimlemiştim (Bölüm 25.4.2.). Bu mimari aynı zamanda HASP/SIAP okyanus gözetimi sisteminde de kullanılmıştır. Russel ve Norvig'den yaptığım alıntıda söylendiği gibi, "Karatahta sistemleri, günümüzün kullanıcı arayüzü mimarilerinin temelidir". Bunlar aynı zamanda bazı bilgisayar uygulamalarında da kullanılmıştır, örneğin gen konumlarını ve yapılarını öngörmede kullanılan otomatik genom bilgi ekleme sistemleri gibi.[12]

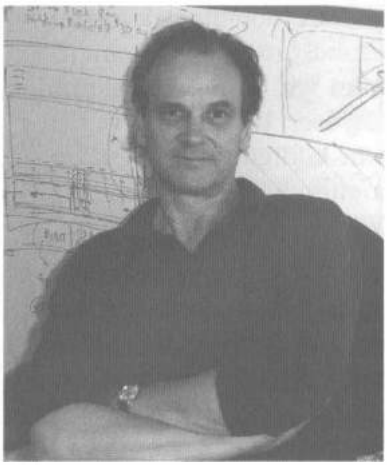
Daha önce söylediklerimi gözden geçirmek gerekirse, Karatahta, katmanlı bir bellek yapısıdır ve bu yapıda, "bilgi kaynakları" (BK) denilen programlar, çeşitli katmanlardan veri okuyup yazabilir (bkz. Resim 17.5). Tipik olarak bir BK, çeşitli katmanlarda veri arayıp okuyabilir, o veriyi kullanarak hesap yapabilir ve o hesapların sonucunu birden fazla katmana işleyebilir. Denetim birimi, üzerine etki edebileceği verileri gören BK'lerden hangisinin etkin olması gerektiğine karar verir. Bazı uygulamalarda, aynı anda birkaç BK birden faal olabilir. Tüm bunların sonucu, çok dinamik bir süreçtir ve bu süreç sırasında, Karatahtadaki veriler sürekli değişiklik geçirir ve en sonunda istenen bilgi üretilir, örneğin bir gen konumunun tahmin edilmesi, bir cümlenin tanınması, okyanus sonar sinyalinin yorumlanması gibi. Karatahtanın bir katmanındaki bilgi, BK'ler aracılığıyla, başka bilgilerin başka katmanlarda boy göstermesini sağladığı için, Karatahta mimarisi, daha önce betimlemiş olduğum korteks modellerindeki aşağı ve yukarı olasılık yayılımının öncüsü olmuştur.

31.1.3 İAN Mimarisi

Michael Georgeff (1946- ; Resim 31.7) ve birtakım başkaları, *inanç*, *arzu* ve *niyet* gibi felsefi kavramları temel alan etmen (*agent*) mimarileri

11 *A.g.e.*

12 Bkz. Stephane Descorps-Declere ve diğ., "Genepi: A Blackboard Framework for Genome Annotation", *BMC Bioinformatics*, c. 7, 12 Ekim, 2006, s. 450vd. Şu internet adresinde: http://www.biomedcentral.com/1471-2105/7/450.



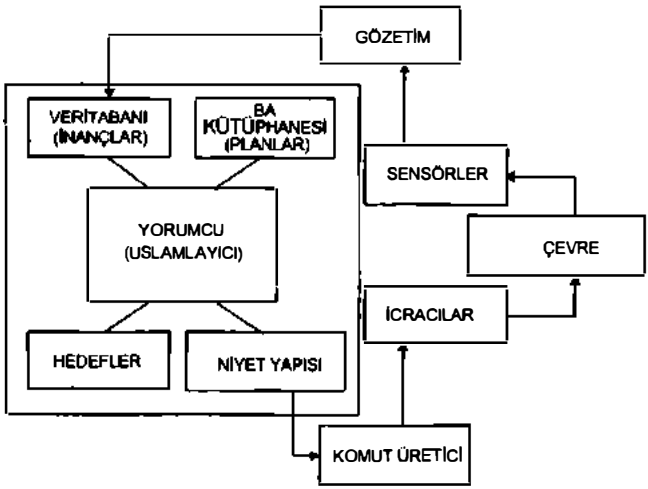
Resim 31.7. Michael Georgeff (Foroğraf, Michael Georgeff'in izniyle.)

önermiştir.[13] Bunlara İAN mimarisi denir. Bir etmenin inancı, çevresi (kendisi ve başka etmenler dahil) hakkındaki bilgisini temsil eder ve çoğunlukla bir tür mantık diliyle ifade edilir, örneğin birinci dereceden yüklemcil biçimsel dizge gibi. ("Bilgi" sözcüğü yerine "inanç" sözcüğü kullanılır, çünkü etmenin inançları değişikliğe tabidir ve çevresini doğru olarak modellemeyebilir.) Etmenin arzuları, onun hedeflerini, yani ulaşmak istediği durumları temsil eder. Etmenin niyetleriyse, ulaşmak üzere gerçekten seçtiği arzularını gösterir. Yani, bu arzularına ulaşmak maksadıyla bir plan gerçekleştirmeye başlamıştır. Örneğin davranış tabanlı, tepkisel mimarilerden ayrı olan İAN mimarileri, gerçek veri yapıları olarak inançları, arzuları ve niyetleri açık açık temsil eder.

Genel hatlarıyla konuşmak gerekirse, daha önce bahsetmiş olduğum kimi mimari şemalar İAN mimarisi olarak düşünülebilir. Örneğin Shakey'nin inançları vardı (kendi dünya modeli), ona her an bir arzu veriliyordu (hedefi) ve icracı sistemi zaman zaman, o hedefe ulaşmak için bir plan yürütüyordu (niyeti). Bununla birlikte Georgeff ve çalışma arkadaşları, İAN mimarisinin özgül bir sürümünü önermişlerdi; buna Yordamsal Uslamlama Sistemi [Procedural Reasoning System (PRS)] demişlerdi.[14] Bunu Resim 31.8'de görebilirsiniz.

13 Bu fikirlerin felsefi bir incelemesi için bkz. Michael E. Bratman, *Intention, Plans, and Practical Reason*, Stanford CA: CSLI Publications, 1999.

14 PRS hakkında çeşitli makaleler ve raporlar var. Bir tanesi şudur: Michael P. Georgeff ve Francois F. Ingrand, "Decision-Making in an Embedded Reasoning System". *Proceedings of the Eleventh International Joint Conference an Artificial Intelligence*, Ağustos 1989, s. 972-978. Şu internet adresinde: http://www.agent.ai/download.



Resim 31.8. PRS, bir İAN mimarisi. (Michael P. Georgeff ve Francois F. Ingrand, "Decision-Making in an Embedded Reasoning System", *Proceedings of the Eleventh International Joint Conference an Artificial Intelligence*, Ağustos 1989, s. 972-978.)

Mimarinin çalışma şekli kısaca aşağıdaki gibidir. (Daha fazla ayrıntı için, Georgeff ve Ingrand makalesine bakınız.)

- Veritabanı, etmenin (*agent*), çevresi hakkındaki güncel inançlarından (kendisi dahil) ve ilgilendiği alandan meydana gelir. Bazı inançları en başta tasarımcı yerleştirir ve bazılarını etmen, kendi algı tertibatı ve çıkarsama mekanizmaları aracılığıyla elde eder. PRS'de inançlar, birinci dereceden yüklemcil biçimsel dizgeyle temsil edilir.
- Hedefler (etmenin arzuları), ulaşılması gereken koşullardır ve hem dış dünyaya hem de etmenin içsel durumlarına atıfta bulunabilir.
- Planlardan oluşan BA kütüphanesi, "Bilgi Alanı" (BA) denilen şeyleri içerir. Her BA, bir nesneyi tutmak gibi görevleri başarmaya yönelik planları belirten özgül bir yordamdır. BA, o yordamın adımlarını betimleyen gövdeden ve BA'nın hangi koşullarda verimli biçimde uygulanabileceğini belirten yürütme koşulundan meydana gelir. "İlkel" BA'larda gövde bulunmaz, fakat bunlar, sistemin doğrudan gerçekleştireceği eylemlere

atıfta bulunur. Ayrıca "meta-düzey" (*meta-level*) BA'lar da vardır; bunlar, uygulanabilir çoklu BA'lar arasından seçim yapabilir, niyetleri değiştirebilir yönlendirebilir ve gerçek zaman kısıtlamaları koşullarında belli bir probleme ayrılacak uslamlama miktarını hesaplayabilir.

- Niyet yapısı, sistemin gerçekleştirmek üzere seçtiği görevleri kapsar. Niyet, ana BA olarak ifade edilir ve bunun yanında, ana BA'nın gerçekleşmesi için kullanılan tüm alt-BA'lar da vardır.
- Yorumcu, sistemi çalıştırır. Sistemin öteki bileşenlerini korur ve niyet yapısından (bir BA), gerçekleştirmek üzere bir niyet seçer. PRS'nin özelliklerinden biri, BA'nın ugulamasının, algılanmış belirli bir durum tarafından (acil durum gibi) kesintiye uğramasıdır, böylece PRS, ortamdaki beklenmedik değişikliklere hızlıca tepki verme yeteneği kazanır.

PRS-tarzı mimarinin kimi uygulamaları olmuştur, örneğin uzay mekiklerinin işlev bozukluklarıyla başa çıkma[15] ve özerk robotların denetimi gibi.[16]

Yukarıda betimlediğim mimari fikirlerin yanı sıra, zeki sistemlerin nasıl düzenleneceği konusunda pek çok genel öneri de yapılmıştır ve bunlardan bazıları işler programlarla (*running program*) sonuçlandırılmıştır (en azından, işler programlar yazmaya olanak tanıyan proglamlama dilleriyle). "meta-düzey uslamlama", yani uslamlama hakkında akıl yürütme becerisini barındıracak sistemler için de birtakım önerilerde bulunulmuştur. Bunlar arasından, Brian Smith'in 3-LISP sistemini[17] ve Richard Weyhrauch'un FOL sistemini[18] anacağım;

15 Michael P. Georgeff ve Amy L. Lansky. "A System for Reasoning in Dynamic Domains: Fault Diagnosis on the Space Shuttle". SRI YZ Merkezi Teknik Notu 375, 1986. Şu internet adresinde: http://www.ai.sri.com/pubs/files/584.pdf.

16 Michael P. Georgeff ve Amy L. Lansky, "Reactive Reasoning and Planning: An Experiment With a Mobile Robot", *Proceedings of the Sixth National Conference on Artificial Intelligence,* 1987. Daha uzun bir versiyonu: Michael P. Georgeff, Amy L. Lansky ve Marcel J. Schoppers, "'Reasoning and Planning In Dynamic Domains: An Experiment with A Mobile Robot", Teknik Notu 380, YZ Merkezi, SRI International, Nisan 1987. Şu internet adresinde: http://www.ai.sri.com/pubs/files/579.pdf.

17 Brian Cantwell Smith, "Reflection and Semantics in a Procedural Language", MIT doktora tezi ve MIT Bilgisayar Bilimi Laboratuvarı Teknik Raporu 272, 1982. Bkz. http://publications.csail.mit.edu/lcs/pubs/pdf/MIT-LCS-TR-272.pdf.

18 Richard Weyhrauch, "Prolegomena to a Theory of Formal Reasoning", *Artificial Intelligence,* c. 13, no: 1 ve 2, Nisan 1980, s. 133-176. Ayrıca, Stanford Bilgisayar

iki sistem de kendi işlemleri hakkında "derin düşünme" yeteneğine sahipti. Meta-düzey uslamlama sistemlerini Pat Hayes,[19] Michael Genesereth[20] ve Stuart Russell ile Eric Wefald[21] da önermiştir.

Taban düzeyindeki bir problemi en iyi çözme yolunu kararlaştırma gibi meta-düzey bir problemde önemli konular, farklı çözüm yöntemlerinin masraf ve kazanç beklentilerine dair tahminleri içerir. Eric Horvitz (1958-), "uslamlama kaynaklarında kısıtlama ve belirsizliklerin olduğu koşullarda zorlu problemlerin çözümünü denetleyecek karar kuramı"nı uygulamanın öncüsüdür.[22] O zamandan beri, meta-düzey uslamlamada olasılık yöntemlerinin ve karar kuramının kullanımı, YZ araştırmalarının önemli bir parçası olmuştur. *Artificial Intelligence* adlı dergi, 2001 yılında bu konu için özel sayı hazırlamıştı.[23]

Marvin Minsky'nin "Zihin Topluluğu"[24] hakkındaki fikirleri de, anında yürürlüğe sokulacak kadar hatları belirli olmasa da zeki sistemler için olası tasarımlar akla getiriyor. Bu tür bir "topluluk", çok sayıda basit "etmen"den meydana gelirdi ve bunların hiçbiri, kendi başına zeki bir varlık olacak kadar güçlü ya da tam olmazdı, fakat "zihin", bunların ortak davranışlarından ve etkileşiminden türerdi. Benzer bir biçimde, William Kornfeld ve Carl Hewitt, zeki sistemlerin, "bilimsel bir topluluk" tarzında örgütlenip, adeta bireysel ve paralel araştırmalar, yayınlar ve eleştirilerden yararlanması gerektiğini söylemiştir.[25]

Bilimi Bölümü Teknik Raporu CS-TR-78-687, 1978: http://www-db.stanford.edu/TR/CS-TR-78-687.html.

19 Patrick J. Hayes, "Computation and Deduction", *Proceedings of the Second Mathematical Foundations of Computer Science Symposium*, Çekoslavakya Bilimler Akademisi, 1973, s. 105-118.

20 Michael R. Genesereth, "An Overview of Meta-Level Architecture", *Proceedings of the Third National Conference on Artificial Intelligence*, Los Altos, CA: Morgan Kaufmann Publishers, 1983, s. 119-124.

21 Stuart Russell ve Eric H. Wefald, *Do the Right Thing: Studies in Limited Rationality*, Cambridge, MA: MIT Press, 1991.

22 Eric J. Horvitz, "Reasoning about Beliefs and Actions under Computational Resource Constraints", *Proceedings of the Third Workshop on Uncertainty in Artificial Intelligence*, Seattle WA, Temmuz 1987, s. 429-444. Şu internet adresinde: ftp://ftp.research.microsoft.com/pub/ejh/u87.ps.

23 *Artificial Intelligence*, c. 126, no: 1-2 (Sınırlı Kaynakların Varlığında Feragat Edilen Bilişim Özellikleri Hakkında Özel Sayı), Şubat 2001.

24 Marvin Minsky, *The Society of Mind*, New York: Simon and Schuster, 1988.

25 William Kornfeld ve Carl Hewitt, "The Scientific Community Metaphor", *IEEE Transactions on Systems, Man, and Cybernetics*, c. SMC-11, no: 1, 1981, s. 24-33. Ocak 1981 tarihli MIT YZ Laboratuvarı Raporu no: 641 olarak şu internet adresinde:

31.1.4 Etmen Toplulukları için Mimariler

Zeki etmenlerin, hem insan hem makine gibi başka zeki etmenler barındıran ortamlarda var olacağı beklenmeli. Bu etmenlerden pek çoğu, görevlerini yerine getirirken işbirliği yapacak ya da rekabet edecek. Etmenler arası iletişim stratejileri ve çoklu etmen mimarileri, önemli YZ başlıkları olmuştur.

YZ sistemleri ile insanlar arası etkileşime daha önce değindik. Bu etkileşimler, doğal dilin kısıtlı sürümlerinden ya da başka bir kullanıcı arayüzü tertibatından faydalanır. İşin doğrusu, dünya, özel tasarım protokoller kullanan ağlar üzerinden birbirleriyle iletişim kuran bilgisayarlarla dolu. Burada bahsetmek istediğim konu ise, YZ yöntemleri sayesinde, YZ etmenleri arasında, sabit iletişime ve örgütlenme protokollerine kıyasla daha esnek ve etkili iletişimin nasıl kurulacağıdır. YZ etmenleri, eylemlerini planlamanın yanı sıra başka etmenlerle kurdukları iletişimi de planlayabilmeli. Dahası, algısal verileri yorumlamanın yanı sıra, başka etmenlerden gelen iletişimi de yorumlayabilmeliler. Bu işi başarmak için, öteki etmenlerin eylemlerini, bilgisini ve hedeflerini hesaba katmak zorundalar.

Günümüzde "çoklu etmen sistemleri" denilen konuyla (eskiden "dağıtık YZ" denirdi) ilgili ilk çalışmaların bir kısmını Massachusetts Üniversitesi'nde Victor Lesser (1944-) ve Carnegie Mellon Üniversitesi'nde Lee Erman (1944-) yapmıştır. DISTRIBUTED HEARSAY-II dedikleri sistemi geliştirmek amacıyla, HEARSAY-II'den Karatahta mimarisiyle ilgili birtakım fikirleri uyarlamışlardı. Bu sistem, her biri kendi BK'sını barındıran ve farklı farklı kaynaklardan gelen gürültülü sinyalleri işlemek için aralarında iletişim kuran birtakım dağıtık Karatahtaların birleşimiydi.[26] Lesser ve Erman, çeşitli alanlarda uygulamalar hayal etmişti, örneğin "sensör ağları (düşük güçlü radardan, akustik ya da optik dedektörlerden, sismometrelerden, hidrofonlardan, vesaireden oluşuyordu), ağ (otomobil) trafik denetimi, envanter denetimi (örneğin araba kiralama), enerji şebekeleri ve hareketli robotlar gerektiren görevler gibi". Belirttikleri üzere: "Sensörlerin olduğu yere veri işleme kabiliyeti yerleştiren ve işlemciler arasında sadece kısıtlı bir iletişim gerektiren bir mimari, özellikle avantajlıdır, belki de gerçek zamanlı tepkinin, kısıtlı iletişim bant genişliğinin ve güvenirliğin taleplerini karşılamanın tek yolu budur."

http://dspace.mit.edu/bitstream/1721.1/5693/2/AIM-641.pdf.

26 Victor R. Lesser ve Lee D. Erman, "Distributed Interpretation: A Model and Experiment", *IEEE Transactions on Computers*, c. C-29, no: 12, Aralık 1980, s. 1144-1163. Bkz. ftp://mas.cs.umass.edu/pub/lesser/LesserIEEE1980.pdf.

Lesser ve çalışma arkadaşları, Amherst'te Massachusetts Üniversitesi MAS Laboratuvarında çok etmenli sistemler üzerinde çalışıyor. DISTRIBUTED HEARSAY-II çalışmasını temel alarak, "Dağıtık Taşıt Gözetimi Test Cihazı"nı (Distributed Vehicle Monitoring Testbed[DVMT]) geliştirdiler.[27] Bu test cihazıyla yürütülen araştırmalar, dağıtık bir sensör ağı kullanarak taşıt hareketini izlemeye odaklanmıştı ve işbirlikçi dağıtık problem çözmeye yönelik test yöntemlerinin kaynağı olmuştur. DVMT çalışmasının ardından başka çoklu etmen sistemi projeleri de gelmiştir.[28]

1970'lerin ikinci yarısı boyunca ve 1980'lerin ortalarına dek, çoklu etmenlerin etkinliklerini eşgüdümleme konusunda pek çok fikir geliştirildi. Bunlardan biri, Reid Smith'in geliştirdiği Contract Net sistemiydi.[29] Bu sistem, "problem çözme iletişimine ve dağıtık problem çözücüde düğüm noktası denetimine" yönelik bir protokolü ve "uzlaşma işlemini" temel alıyordu. İlk uygulamalarından biri, sensörler konuşlandırılana dek sensör yerlerinin ve tiplerinin bilinmediği dağıtık bir sensör ağıydı.[30]

Önemli sistemlerden biri de, Southern California Üniversitesi'nde Les Gasser ve çalışma arkadaşlarının geliştirdiği "Çoklu Etmen Hesaplama Ortamı"ydı (Multi-Agent Computing Environment [MACE]).[31] MACE hakkındaki makale, onu şöyle betimliyor:

> MACE... çeşit çeşit deneysel Dağıtık Yapay Zekâ sistemi kurmaya yarayan aletli test düzeneğidir... MACE hesaplama birimleri (bunlara "etmen" [*agent*] denir), paralel çalışır ve iletiler üzerinden iletişim kurar. Bu birimler, bilgi gösterimi (dış dünya bilgisi, öbür etmenlerin modeli,

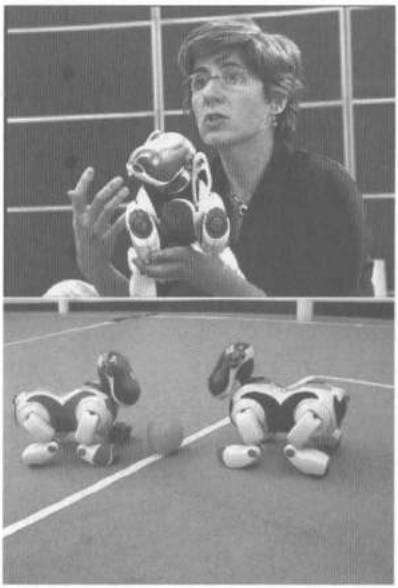
27 Victor R. Lesser ve Daniel Corkill, "The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks", *AI Magazine*, c. 4, no: 3, 1983, s. 15-33.

28 Laboratuvarın internet sitesine bakınız: http://mas.cs.umass.edu/, ayrıca şu internet adresinde Laboratuvarın çalışmalarının tarihçesine bakınız: ftp://mas.cs.umass.edu/pub/LabHistory_Web-Article.pdf

29 Reid G. Smith, "The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver", *IEEE Transactions on Computers*, cilt C-29, no: 12, Aralık 1980, s. 1104-1113.

30 Ayrıca bkz. Randy Davis ve Reid G. Smith, "Negotiation as a Metaphor for Distributed Problem Solving", *Artificial Intelligence*, c. 20, no: 1, 1983, s. 63-109.

31 Les Gasser, Carl Braganza ve Nava Herman, "MACE: A Flexible Testbed for Distributed AI Research", *Distributed Artificial Intelligence* içinde, Michael N. Huhns (ed.), Londra: Pitman Publishers, 1987, s. 119-152. Şu internet adresinde: http://www.isrl.uiuc.edu/%7Egasser/papers/gasser-braganza-herman-mace-a-flexible-testbed-for-dai-research-1987.pdf.



Resim 31.9. Manuela Veloso (üstte) ve futbol oynayan Aibo robotları (altta). (Fotoğraflar, Manuel Veloso'nun izniyle.)

hedefleri ve planları, rolleri, becerileri, vesaire) ve uslamlama yetenekleri için olanak seçenekleri sağlar.

Çoklu etmen sistemlerinin ilginç bir uygulaması, işbirlikçi (ve rekabetçi) robotları içeriyor. Carnegie Mellon Üniversitesi'nde Profesör Manuela Veloso (1957- ; Resim 31.9), bu alanda çalışan başlıca araştırmacılardan biridir. Veloso, "işbirliği yapan, gözlemleyen, uslamlayan, eyleyen ve öğrenen zeki robotlar üzerine araştırmalarına"[32] ek olarak, futbol oynayan robotların RoboCup maçlarında da faaldir. Bu maçlarda her robot, kendi duyumsama ve işleme becerilerini barındırır. Her robot, öteki oyuncuların eylemlerini ve yapabileceklerini hesaba katmak zorundadır.

Robocup, "uluslararası ortak bir projedir... YZ ve zeki robotik araştırmalarını desteklemek amacıyla, geniş bir yelpaze içinden teknolojilerin bütünleştirilip incelendiği standart bir problem sağlar."[33] Nihai hedefi, "[2050 yılına dek], insanlar âleminde dünya futbol şampiyonu olan takımı yenebilecek, bütünüyle özerk insansı robotlar geliştirmektir". Yakında, bu robotları etrafta görmeye başlayabiliriz.

32 CORAL internet sitesine bakınız: http://www.cs.cmu.edu/~coral/main/.
33 Bkz. http://www.robocup.org/.

Ortamda, etmenin işbirliği yapmak ya da rekabet etmek zorunda olduğu başka etmenler varsa, o etmenin, kendi çevre modelinde öbür etmenlerle ilgili modeller içermesi önemlidir. Söz konusu modeller, öbür etmenlerin inançları ve bu inançların nasıl değiştirilebileceği hakkında bilgi kapsamalıdır. Araştırmacılar, bunun gibi sorunlarla başa çıkabilmek için, bir *A* etmeninin kendi kendine, *B* etmeninin P olgusunu bildiğini ve *A* etmeninin hangi koşullarda *B* etmenine P olgusunu söylemesi gerektiğini nasıl sunmalı türü problemler hakkında kafa yormaya başladı. Başlıca zorluklardan biri, *B*'nin bildiğini *A*'nın bilmesi (P ∨ Q) ile *B*'nin ya P ya da Q olgusunu bildiğini *A*'nın bilmesi arasında nasıl ayrım yapılacağıydı. Başka bir zorluk da, *A* etmeninin, örneğin OB adlı bir nesne için *B*'nin hangi ismi kullandığını bilmeksizin *B* etmenine bu nesneyle ilgili bir olgu anlatma hakkında fikir yürütmesiydi. Başka bir zorluk da, *B*'nin kendi uslamlama süreçleriyle varacağı sonuçlar hakkında *A*'nın yapacağı varsayımları içeriyordu. Bu zorluklara dair çeşitli çözümler önerilmiştir. Bu çözümlerin en önde geleni, "epistemolojik mantık" (inanç mantığı) denilen yaklaşımı kapsıyor.[34]

Başka bir etmenin bildikleri ve varabileceği sonuçlar hakkında bilgisi olan etmen, öbür etmenin bildiklerine ekleme yapacak, düzeltecek ya da bundan bir şeyler öğrenecek konumdadır. Başka etmenin bilgisine ayarlamalar yapıp bundan bir şeyler öğrenmek, etmenler arasındaki işbirliği için kilit bir unsurdur ve vericiden alıcıya iletişimi, ayrıca alıcının vericiyi anlayıp ona uyum göstermesini gerektirir. Araştırmacılar, çeşitli iletişim edimi tipleri olduğunu belirtmiştir. Bunlara genelde "konuşma eylemi" denir, gerçi iletişim, konuşma dışında bir araçla kurulur. Bu tiplerin pek çoğunu, John L. Austin'in çalışmalarının izinden giden John Searle sınıflandırmıştır.[35] Bu tiplerin, çoklu etmen kullanımı bakımından başlıcaları, etmenden etmene olgu aktaran "iddiacılar", alıcıdan belli eylemi talep eden ya da buyuran "buyrukçular" ve vericinin belli bir eylemi gerçekleştirmesini vaat eden "üstleniciler"dir.

34 Matematiğine güvenen okurlar, örneğin şu makaleye danışabilir: Robert C. Moore, "Reasoning about Knowledge and Action", *Proceedings of IJCAI-77*, c. 1, 1977, s. 223-227. Genişletilmiş versiyonu: Robert C. Moore, "A Formal Theory of Knowledge and Action", SRI YZ Merkezi Teknik Notu 320, Şubat 1984, şu internet adresinde: http://www.ai.sri.com/pubs/files/632.pdf. Ayrıca bkz. Kurt Konolige, "A Deduction Model of Belief and Its Logics", SRI YZ Merkezi Teknik Notu 326, Ağustos 1984, şu internet adresinde: http://www.ai.sri.com/pubs/files/626.pdf.

35 John L. Austin, *How to Do Things with Words*, New York: Oxford University Press, 1962; John Searle, *Speech Acts*, New York: Cambridge University Press. 1969.

Etmenler arasındaki iletişime eylem bağlamında bakacak olursak bu eylemleri kullanarak plan yaratma hakkında kafa yorabiliriz. BBN'de Philip R. Cohen (1950-) ve Toronto Üniversitesi'nde C. Raymond Perrault (1949-), tam da bunu yapanlar arasındaydı.[36] Özellikle, REQUEST ve INFORM (daha eski "iddiacılara" (*assertives*) ve "buyrukçulara" (*directives*) dayanıyorlardı) adlı konuşma fiillerini ele aldılar ve bu fiillerin gerçekleştirilebileceği koşulları önerip etkilerinin ne olacağını söylediler. Koşullar ve etkiler, vericinin ve alıcının bilgi tabanlarında boy gösteren (ya da buradan türetilebilecek) mantıksal ifadeler bağlamında ifade edilmişti. (Hem vericinin hem de alıcının, birbirlerinin bilgisi hakkında bilgileri olduğu ve bu bilgiyle uslamlama yapabilecekleri varsayılmıştı.) Böylece STRIPS gibi bir planlama sistemi, istenen etkilere ulaşabilecek konuşma fiili örneklerinden oluşan planlar yaratabilirdi.

Cohen ve Perrault'nun konuşma fiilleri, KQML (Knowledge Query Manipulation Language) dilinin zeminini döşemişti. KQML, DARPA'nın desteğiyle "bilgi paylaşma inisiyatifi" bünyesinde geliştirilmişti.[37] Bu dil, etmenler arasında gerçekleşen, `sor-eğer`, `bildir`, `anlat` ve `yanıtla` gibi çeşitli iletişim edimlerini tanımlar.

KQML, iletinin içeriğini ifade etmeye yarayan, birinci dereceden yüklemcil biçimsel dizgeyi temel almış bir dil olan KIF'i (Knowledge Interchange Format) kullanır.[38] Dolayısıyla, *A* etmeni *B* etmenine ileti göndermek istediğinde, iletinin içeriğini KIF'le kodlayıp bunu uygun KQML iletişim edimiyle sarmalar. Örneğin, tipik bir KQML/KIF diyaloğu şöyledir:[39]

```
A'dan B'ye: (sor-eğer (> (büyüklük chip1) (büyüklük chip2)))
B'den A'ya: (yanıt doğru)
B'den A'ya: (bildir (=(büyüklük chip1) 20))
B'den A'ya: (bildir (=(büyüklük chip2) 18))
```

36 Philip R. Cohen ve C. Raymond Perrault, "Elements of a Plan Based Theory of Speech Acts", *Cognitive Science*, c. 3, no: 3, 1979, s. 177-212. Şu internet adresinde: http://www.cs.huji.ac.il/~imas/readings/cohen79.pdf.

37 KQML şu makalede betimleniyor: Tim Finin ve diğ., "KQML as an Agent Communication Language", *Software Agents* içinde, Jeff Bradshaw (ed.), Cambridge, MA: MIT Press, 1997. Bkz. http://www.cs.umbc.edu/kqml/papers/kqmlacl.pdf. DARPA bilgi paylaşımı inisiyatifi şu makalede betimleniyor: Robert Neches ve diğ., "Enabling Technology for Knowledge Sharing", *AI Magazine*, c. 12, no: 3, 1991, s. 16-36.

38 Bkz. http://www-ksl.stanford.edu/knowledge-sharing/kif/.

39 Bu örnek, Michael Wooldridge'in çok etmenli sistemler dersinin slaytlarından birinden alınmıştır; şu internet adresinde: http://www.csc.liv.ac.uk/~mjw/pubs/imas/.

KQML ve KIF gibi projelerle destek alan Zeki Fiziksel Etmenler Kurumu (FIPA), 1996'da İsviçre'de kuruldu. Günümüzde FIPA, IEEE Bilgisayar Cemiyetinin standartlar komitesinden biridir. Bu kurumun standartları, "heterojen etmenlerin karşılıklı işlemlerini ve temsil edebildikleri hizmetleri teşvik etmeyi amaçlar". Bu standartlar, "Etmen İletişim Dili" (ACL) iletilerini ele alır ve "ileti değişimi etkileşim protokolleri, konuşma fiili kuramı tabanlı iletişim fiilleri ve içerik dili gösterimleri" sağlar.[40]

Günümüzde, çoklu etmen sistemlerini yürürlüğe sokmaya yarayan çeşitli sistemler ve diller vardır. Örneğin, mantık tabanlı AgentSpeak dili için açık kaynak Jason çeverici, kullanıcılara, çok etmenli karmaşık sistemler tanımlamaya yarayan bir yazılım platformu sağlar.[41]

Her ne kadar pek çok çoklu etmen sistemi, birkaç etmenin genel problemleri çözmek amacıyla işbirliği yaptığı uygulamalara yoğunlaşmış olsa da, etmenlerin kendilerine dönük olması da olasıdır, bu da aralarında rekabet doğurur. Futbol oynayan rakip robot takımları buna bir örnektir. Satın alma ve satma gibi ticari işlerle uğraşan etmenler başka bir örnektir (insanları taklit ettikleri varsayılır). Çoklu etmen araştırmasının bu yönleri, pazarlığı ve açık artırmayı da içerir. Oyun kuramı, bir etmenin tercih yapmadaki başarısının, öbür etmenlerin seçimlerine bağlı olduğu durumları ele alan ünlü bir çerçevedir. Çoklu etmen sistemi araştırmalarına oyun kuramını sokan kişiler, "Akılcı Etmenler Arasında Alışveriş" başlıklı makaleleriyle Jeff Rosenschein (1957-) ve Michael Genesereth (1948-) olmuştur.[42] Günümüzde çoklu etmen sistemleri YZ'nin büyük bir alt konusunu oluşturur, ayrıca konuşma fiili kuramı ile oyun kuramı, bunun önemli kuramsal payandaları arasındadır.[43]

40 FIPA internet sayfasına bakınız: http://www.fipa.org/.

41 Rafael H. Bordini, Jomi Fred Hübner ve Michael Wooldridge, *Programming Multi-Agent Systems in AgentSpeak Using Jason*, New York: John Wiley and Sons, Inc., 2007. Jason internet sitelerinden biri şu adreste: http://jason.sourceforge.net/JasonWebSite/Jason%20Home.php.

42 Jeffrey S. Rosenschein ve Michael R. Genesereth, "Deals among Rational Agents", *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985, s. 91-99. Şu internet adresinde: http://dli.iiit.ac.in/ijcai/IJCAI-85-VOL1/PDF/017.pdf.

43 Örneğin bkz. Michael Wooldridge, *An Introduction to MultiAgent Systems*, Chichester, England: John Wiley and Sons, Inc., 2002; Yoav Shoham ve Kevin Leyton-Brown, *Multiagent Systems: Algorithmic, Game Theoretic, and Logical Foundations*, New York: Cambridge University Press, 2008; ve Tim Finin'in "Agents 101" adlı internet sitesi, etmenler hakkındadır: http://agents.umbc.edu/.

31.2 BİLİŞSEL MİMARİLER

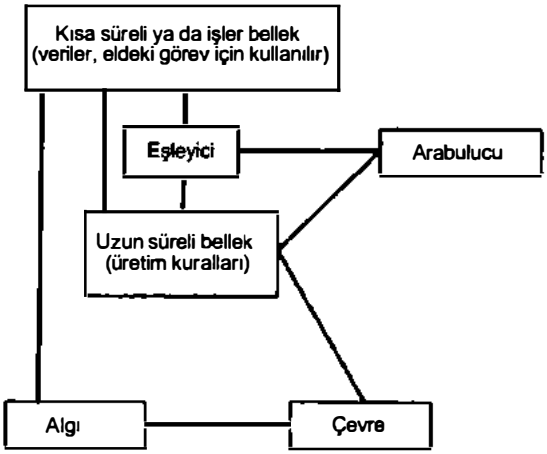
31.2.1 Üretim Sistemleri

Allen Newell ve Herb Simon, insanların problem çözme yetisinin bilişimsel modelleriyle ilgilenen ilk araştırmacılar arasındaydı. Bilişsel süreçlerin ilk mimarilerinden biri sayılabilecek Genel Problem Çözücüyü (GPÇ) daha önce anlatmıştım. Bu kısımda, bilişsel mimari denilen birkaç yapıdan daha bahsedeceğim. Bu mimari ailelerinden birinin geliştiricisinin daha sonra söyleyeceği gibi:[44]

> [Bilişsel mimari], zeki davranışın zeminini oluşturan sıkı sıkıya eşlenmiş mekanizmalardan oluşan sabit bir tabandır. [Bu tür] bir mimari, uzman sistemleri ve psikolojisel modelleme gibi alanlardaki uygulamalar kadar, problem çözme, planlama, öğrenme, bilgi gösterimi, doğal dil, algı ve robotik gibi zekâ gerektiren temel becerilerin de geniş kapsamlı incelenmesinin zeminini oluşturur.

Newell ve Simon, GPÇ üzerine çalışmalarından sonra, "üretim sistemlerine" odaklandı; bu modeller, ürünler denilen EĞER-İSE kurallarını kullanıyordu. Bu tür kurallardan ilk olarak, Bölüm 18.2'de uzman sistemleri betimlerken bahsetmiştim. O sistemlerde bu kurallar temel olarak çıkarsama amacıyla kullanılır, bunlardan ya ileri çıkarsamayla çıkarsanmış yargılar elde edilir ya da geri çıkarsamayla althedefler yaratılır. Newell ve Simon, bu kuralların, eylem üretmekteki kullanımıyla ilgileniyordu. EĞER kısmı, her zamanki gibi bir koşulu belirtir ve İSE kısmı, modelde koşul yerine getirildiğinde gerçekleşen bir eylemi belirtir. (Bölüm 25.4.2'de betimlemiş olduğum kendi teleo-reaktif programlarım da, eylem üreten bu kurallardan yararlanıyordu.) Newell ve Simon, iki tür bellek yapısından oluşan bir mimari tasarlıyordu. Bunlardan biri olan "uzun süreli bellek", üretim kurallarından meydana gelir. Öteki, yani kısa süreli ya da "işler bellek", yerine getirilecek görev hakkındaki dinamik bilgiyi barındırır. Uzun süreli bellek zamana dayanıklıdır ve binlerce kural içerebilir. İşler bellekse, kuralın koşul kısmının sınayacağı verileri kapsar. Bir kuralın koşul kısmı, işler bellekteki verilerle eşlendiğinde, o kural "sinyal gönderir"; yani eylem kısmı hayata geçirilir. Bu eylem, işler belleğe veri yazılmasıyla ya da silinmesiyle (ya da ikisi birden), ya da dış ortamda bir eylemde bulunulmasıyla sonuçlanabilir. İşler bellekteki veriler değiştirildiğin-

44 Paul S. Rosenbloom, John E. Laird ve Allen Newell (ed.), *The SOAR Papers: Research on Integrated Intelligence*, Birinci Bölüm, Cambridge, MA: MIT Press, 1993.



Resim 31.10. Üretim kuralı mimarisi.

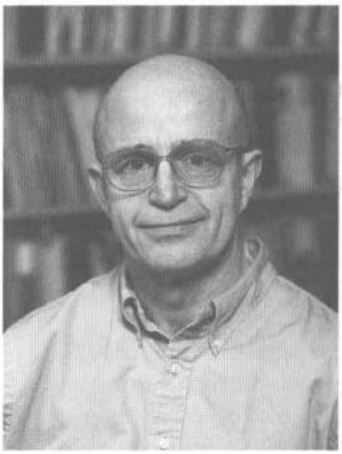
de farklı kurallar harekete geçer, bunlar da veriyi yine değiştirir, ve böyle sürer.

Resim 31.10, bu üretim-kural mimarisinin basit bir sürümünü gösteriyor. Kuralın birden fazla koşul kısmının işler bellekteki verilerle eşlenmesi durumunda (bu, olağan bir durumdur), "Arabulucu" birimi, hangisinin (ya da paralel işletim söz konusuysa, hangilerinin) eyleme geçeceğini seçer. Dış ortamda birtakım eylemler üreten bu mimari sürümünde, "Algı" sistemi, çevrenin güncel halinin göze batan özelliklerini temsil etmek üzere kısa süreli belleğe veri işleyebilir.

Newell ve Simon, problem çözme görevleri yerine getiren insan deneklerle kapsamlı deneysel çalışmalar gerçekleştirmiştir; böylece, insanların performansının, bu mimari şemayla pekâlâ modellenebileceğini göstermiş oldular. *Human Problem Solving* [İnsanlarda Problem Çözme] başlıklı kitapları, bu çalışmaları büyük oranda açıklar.[45] Genel olarak, üretim-sistemi mimarilerini, YZ sistemlerinin yapılanmasına dair bir öneri değil, insan bilişsel yetisine dair bilimsel araştırmalara büyük bir katkı sayarlar. Newell, kendi kitapları hakkında yorum yaparken ileride şunu yazacaktı: "Amaç, *psikolojinin* bilgisayarla ilintili diye gözardı edilemeyecek bir yöntemle yapılabileceğini ortaya koymaktı."[46]

45 Allen Newell ve Herbert A. Simon, *Human Problem Solving*. Englewood Cliffs, NJ; Prentice-Hall, 1972.

46 Allen Newell, "This Week's Citation Classic", *Current Contents*, no: 34, 25 Ağustos, 1980, s. 167. Bkz. http://www.garfield.library.upenn.edu/classics1980/A1980KD-04600001.pdf.



Resim 31.11. John Anderson (Fotoğraf, John Anderson'un izniyle.)

Bununla birlikte, üretim sistemlerini, en başta OPS5 bilgisayar dilinin kullanımı aracılığıyla bilgisayarlarla ilintilendirenler de çıkmıştır.

ACT-R ve SOAR gibi, daha sonraki bilişsel mimari önerileri, üretim sistemi modelinden etkilenmişti; ikisi de YZ sistemlerinde problem çözme ve mimari modeli olarak kullanılmıştır. Şimdi de bu sistemleri anlatıyorum.

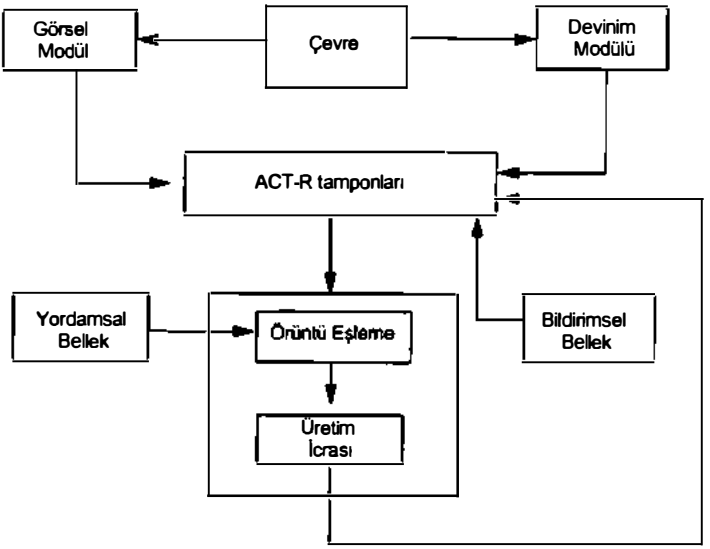
31.2.2 ACT-R

John R. Anderson (Resim 31.11) ve başkaları, Carnegie Mellon Üniversitesi'nde, ACT (Adaptive Control of Thought; uyarlanımsal Düşünce Denetimi) denilen bir dizi bilişsel mimari geliştirmeyi sürdürüyor.[47] (11.4'te bahsettiğim gibi, Anderson, Stanford'da öğrenciyken, Gordon Bower'la birlikte, insanın çağrışımlı belleğiyle (HAM) ilgili bir kuram geliştirmişti; bu, ACT çalışmasının öncülü sayılabilir.) Bu model dizisinin sonuncusu, ACT-R'dir ("R", rasyonel (akılcı) anlamında.) Bunun internet sitesine göre:[48]

> ACT-R, bilişsel bir mimaridir: insanın bilişsel yetilerini taklit etmeye ve anlamaya yönelik bir kuramdır. ACT-R üzerine çalışan araştırmacılar, in-

47 Yeni bir tanımlama şu makalede bulunabilir: John R. Anderson ve diğ., "An Integrated Theory of the Mind", *Psychological Review*, c. 111, no: 4, 2004, s. 1036-1060; şu internet adresinde: http://act-r.psy.cmu.edu/publicarions/pubinfo.php?id=526. Ayrıca bkz. John R. Anderson, *How Can the Human Mind Occur in the Physical Universe*, New York: Oxford University Press, 2007.

48 http://act-r.psy.cmu.edu/.



Resim 31.12. Temel ACT-R mimarisi. (John Anderson'ın izniyle kullanılmıştır.)

sanların bilgiyi nasıl örgütleyip zeki davranış ürettiğini anlamaya çabalar. Araştırma ilerledikçe, ACT-R, insan bilişsel görevlerinin hepsini yapabilecek bir sistem olmaya gitgide yaklaşır: Dünyayı algılama, dünya hakkında kafa yorma ve eyleme biçimimizi ayrıntılarıyla kavrayacak hale gelir.

ACT-R mimarisinin temel fikri, Resim 31.12'de gösteriliyor. Üç ana bileşeni bulunur: modüller, tamponlar ve örüntü eşleyiciler. Devinim Modulü, devinim altprogramları aracılığıyla Çevre üzerine ya da ACT-R Tamponları üzerine etki eder. Resimde gösterilen Görsel Modülün yanı sıra işitme, dokunma, vesaire için başka algı modülleri de olabilir.

ACT-R'de iki tür bellek modülü vardır. Bildirimsel Bellek, olgulardan meydana gelir, örneğin "Washington DC, Amerika Birleşik Devletlerinin başkentidir, Fransa Avrupa'da bir ülkedir ya da 2 + 3 = 5'tir", gibi. Bildirimsel bilgi, ACT-R'de öbek denilen birimlerle temsil edilir. Yordamsal Bellek, "şunu bunu nasıl yaptığımız hakkındaki bilgiyi temsil eden üretim kurallarından meydana gelir: Örneğin, klavyede 'Q' harfine nasıl basılacağı, nasıl araba sürüleceği ya da toplama işleminin nasıl yapılacağı hakkında bilgi."[49] Belirli şeyleri nasıl yaptığımız hakkındaki bilgimizi çoğunlukla söze dökemeyiz: Bu işleri yaparız işte; dolayısıyla bunlar hakkındaki bilgi, bildirimsel değil yordamsal sayılır.

49 Alıntı şu internet adresinden: http://act-r.psy.cmu.edu/about/.

ACT-R'deki tamponlar, modüllerin arasında arayüz olarak hizmet eder. "Belirli bir zaman anında tamponların içerikleri, o anda ACT-R'nin durumunu temsil eder... Örüntü eşleyici, tamponların güncel durumuna uyan bir üretim arar. Sadece bu tür bir üretim, herhangi bir anda yerine getirilebilir. O üretim, yerine getirildiğinde, tamponları ve dolayısıyla sistemin durumunu değiştirmiş olur. Böylece, ACT-R'de biliş, üretimi tetikleyen ardışık sinyaller olarak gün yüzüne çıkar."

ACT-R internet sitelerinden birine göre:[50]

> ACT-R, *melez bir bilişsel mimaridir*. Bunun *simgesel* yapısı, bir üretim sistemidir; *altsimgesel yapı*, pek çok matematiksel denklemin özetleyebileceği son derece paralel süreçlerin bir kümesiyle temsil edilir. Altsimgesel denklemler, simgesel süreçlerin pek çoğunu denetler. Örneğin, birtakım üretimler, tamponların durumuna uyuyorsa, altsimgesel olanaklar denklemi, her üretimle ilintili göreli masrafını ve kazancını tahmin eder ve en yüksek kullanabilirliği olan üretimin gerçekleşme seçimine dair karar verir. Benzer biçimde, bir olgunun bildirimsel bellekten çağrılıp çağrılmaması (ya da ne kadar hızlı çağrıldığı), altsimgesel erişim denklemlerine bel bağlar, ki bunlar o olgunun kullanımının bağlamını ve tarihçesini hesaba katar. Ayrıca altsimgesel mekanizmalar, ACT-R'deki çoğu öğrenme sürecinden de sorumludur.

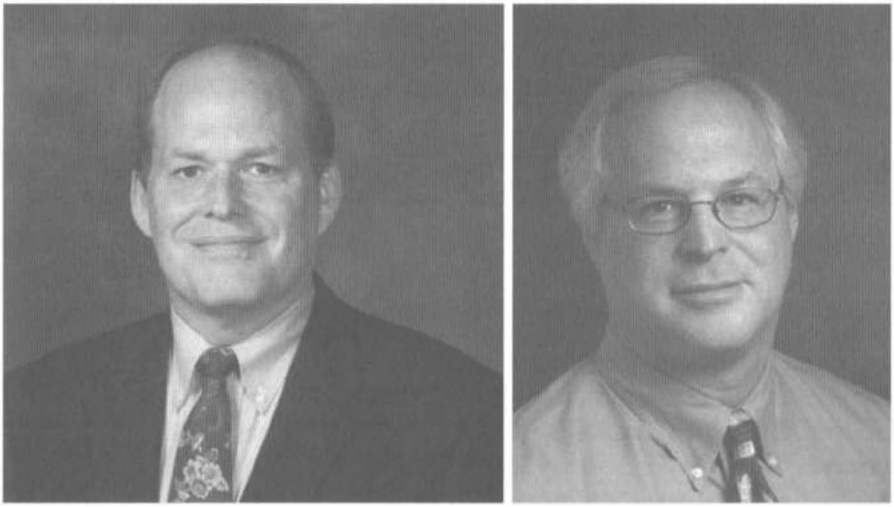
ACT-R modelleri, insanlarda öğrenme, dil işleme, algı, problem çözme ve karar verme dahil çeşit çeşit bilişsel davranışı açıklayıp taklit etmek için kullanılır. Bu çalışmayı betimleyen yüzlerce (belki binlerce) makale var.[51] ACT-R uygulamaları, örneğin "cep telefonu tuşlamanın sürücü performansına etkisinden" zeki öğretici sistemlere kadar geniş bir yelpazedeki konuları kapsar. YZ'yle daha doğrudan ilintili uygulamalara bakacak olursak, Deniz Kuvvetleri Yapay Zekâ Uygulamalı Araştırmalar Merkezinde (NCRAI) Greg Trafton ve Alan C. Schultz, ACT-R/E dedikleri bir ACT-R sürümü kullanarak "bilişsel öğeler iliştirilmiş robot" imal ediyorlar. Bu robot, algı ve eylem için "görsel ve işitsel modüller" ve "devinim ve uzam modülleri" içeriyor.[52]

İşlevsel manyetik rezonans görüntüleme (fMRI) araştırmaları, ruhsal süreçleri açıklamadaki rolüne ek olarak, karmaşık görevlerde

50 http://act-r.psy.cmu.edu/about/.

51 Bir liste için bkz. http://act-r.psy.cmu.edu/publications/index.php.

52 Bkz. Greg Trafton ve diğ., "Integrating Vision and Audition within a Cognitive Architecture to Track Conversations", *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, 2008, s. 201-208. Şu internet adresinde: http://www.nrl.navy.mil/aic/iss/aas/documents/trafton.hri08.pdf.



Resim 31.13. John Laird (solda) ve Paul Rosenbloom (sağda) (Fotoğraflar, John Laird ile Paul Rosenbloom'un izniyle.)

aktif olan beyin bölgelerini, ACT-R mimarisinin bileşenleriyle bağdaştırmak amacıyla kullanılmıştır.[53]

ACT-R yazılımı, kaynakçalarla ve eğitici malzemelerle birlikte şu internet adresinde bulunur: http://act-r.osy.cmu.edu/actr6/.

31.2.3 SOAR

1980'lerin ilk yarısında John Laird (1954- ; Resim 31.13), Allen Newell ve Paul Rosenbloom (1954- ; Resim 31.13), SOAR adlı (aslen, Sate, Operator And Result'ın kısaltması olduğu söylenir) bir dizi bilişsel mimari geliştirmeye başladı. SOAR'ın geliştiricileri, "SOAR mimarisinin en son hedefinin, hem insan hem de yapay biliş için bir temel olarak hizmet etmek olduğunu", söylemiştir.[54] ACT gibi SOAR da Newell ile Simon'un GPÇ ve üretim sistemleri üzerine çalışmalarından evrimleşmiştir, ayrıca problem uzamları, bulucu arama, bilişsel yeti edinme ve öğrenmeyi ilgilendiren fikirler içerir. Laird ve Rosenbloom, Newell'in

53 Bkz. John R. Anderson ve diğ., "A Central Circuit of the Mind", *Trends in Cognitive Science*, c. 12, no: 4, 2008, s. 136-143 (şu internet adresinde: http://act-r.psy.cmu.edu/publications/pubinfo.php?id=800) ve John R. Anderson ve diğ., "An Integrated Theory of the Mind", *Psychological Review*, c. 111, no: 4, 2004, s. 1036-1060 (şu internet adresinde: http://act-r.psy.cmu.edu/publications/pubinfo.php?id=526).

54 Paul S. Rosenbloom, John E. Laird ve Allen Newell (ed.), *a.g.e.*

doktora öğrencileriydi ve SOAR'ın kimi özellikleriyle ilgili tezler yazmışlardır.[55] SOAR mimarisi ve uygulamaları, kitaplarda, yüzlerce makalede[56] ve SOAR internet sitelerinde (şu internet adresinden ulaşılabilir: http://sitemaker.umich.edu/soar/home) betimlenmiştir.

SOAR1'den (1982) SOAR9'a (2008) kadar, her biri öncülüne kıyasla daha gelişkin olan bir dizi SOAR mimarisi yaratıldı.[57] SOAR yazılımının en yeni sürümleri şu internet adresinden indirilebilir: http://sitemaker.umich.edu/soar/soar_software_downloads. Bu çeşitli SOAR sürümlerinin nasıl çalıştığını tam olarak anlamak için, makalelerdeki ve kitaplardaki kimi SOAR örnekleri gözden geçirilmeli. Buradaki kısa açıklamamı, ana fikirlerden bazılarını anlatarak sınırlandıracağım.

SOAR, bir dizi sabit altprogram barındıran bir programlama diline benzer. Bu altprogramlar sayesinde, farklı görevler SOAR'la "programlanabilir". YZ'nin gözdesi "oyuncak" problemlerden (örneğin sekizli bulmaca) "gerçek dünya" uygulamalarına (bilgisayar sistemi konfigürasyonu ve robot denetimi gibi) dek çeşitli örnekler vardır. SOAR, kendisine verilen her görevi, bir altgörev hiyerarşisi yaratıp çözerek halleder. Her görev (ana görev ve altgörevler dahil), yeni bir durum üretmek üzere güncel duruma uygulanan işlemciler kümesinden oluşmuş "problem uzamı"nda, arzulanan durumu bulma hedefi olarak belirlenir.

SOAR, problem uzamı kurarken güncel durumunu ve o durumda hangi işlemcilerin uygulanabileceğini bilmeli. Eğer bu gibi şeyleri, örneğin önceki deneyimler sayesinde doğrudan bilmiyorsa, hedefi bunları keşfetmek olan bir problem uzamı şubesi kurar (bu biçimde devam eder). SOAR, problem uzamını tanımlar tanımlamaz, o uzamda var olan duruma uygulanacak işlemciyi seçmeli. Hangi işlemcinin

55 John E. Laird, "Universal Subgoaling", doktora tezi, Bilgisayar Bilimi Bölümü, Carnegie Mellon Üniversitesi, Pittsburgh, PA, 1983; Paul S. Rosenbloom, "The Chunking of Goal Hierarchies: A Model of Practice and Stimulus-Response Compatibility", doktora tezi (ayrıca Teknik Rapor no: 83-148), Bilgisayar Bilimi Bölümü, Carnegie Mellon Üniversitesi, Pittsburgh, PA, 1983.

56 İki ciltlik kitapta altmış sekiz makale derlenmiştir (kapsamlı bir kaynakçayla birlikte): Paul S. Rosenbloom, John E. Laird ve Allen Newell (ed.), *a.g.e.* (Kitabın giriş bölümü şu internet adresinde: http://www.isi.edu/soar/papers/soar-papers-book/intromosaic.ps.) SOAR'la ilgili başlıca makale şudur: John E. Laird, Allen Newell ve Paul S. Rosenbloom, "SOAR: An Architecture for General Intelligence", *Artificial Intelligence*, c. 33, 1987, s. 1-64.

57 SOAR9, şu makalede betimleniyor: John E. Laird, "Extending The SOAR Cognitive Architecture", *Proceedings of the Artificial General Intelligence Conference*, Memphis, TN, Mart 2008, Amsterdam: IOS Press; şu internet adresinde: http://ai.eecs.umich.edu/people/laird/papers/Laird-GAIC.pdf.

geçerli olduğunu henüz bilmiyorsa, amacı bunu bulmak olan bir yardımcı problem uzamı kurar. Pek çok SOAR makalesinde ifade edildiği gibi: "SOAR mekanizmaları sıkı sıkıya eşlenmiş bir katmanlar hiyerarşisi oluşturur; bellek, karar ve hedeften meydana gelir bu katmanlar; bu hiyerarşide her katman, üzerindeki katman için bir iç devre işlevi görür. Hiyerarşide dipten yukarı çıkıldıkça katmanların karmaşıklığı da zaman ölçeği de gitgide artar."

Yardımcı problemler kurmaya, "evrensel althedef belirleme" denir; bu süreç sonunda, derin bir althedefler ağacı ve poblem uzamı ortaya çıkar. Bunlar, denetim kararları içerdiğinde (örneğin hangi işlemcinin uygulanacağı gibi), SOAR'ın kendi problem çözme davranışı üzerine "kafa yorduğu" söylenebilir. Evrensel althedef belirleme süreci, SOAR'ın üzerinde çalıştığı görev hakkında daha önce öğrenmiş olduğu bilgiye bağlı olarak, zayıf denilen bir dizi yöntem yürürlüğe sokabilir, örneğin tepe tırmanma, araç-amaç çözümlemesi ve bulucu arama gibi.

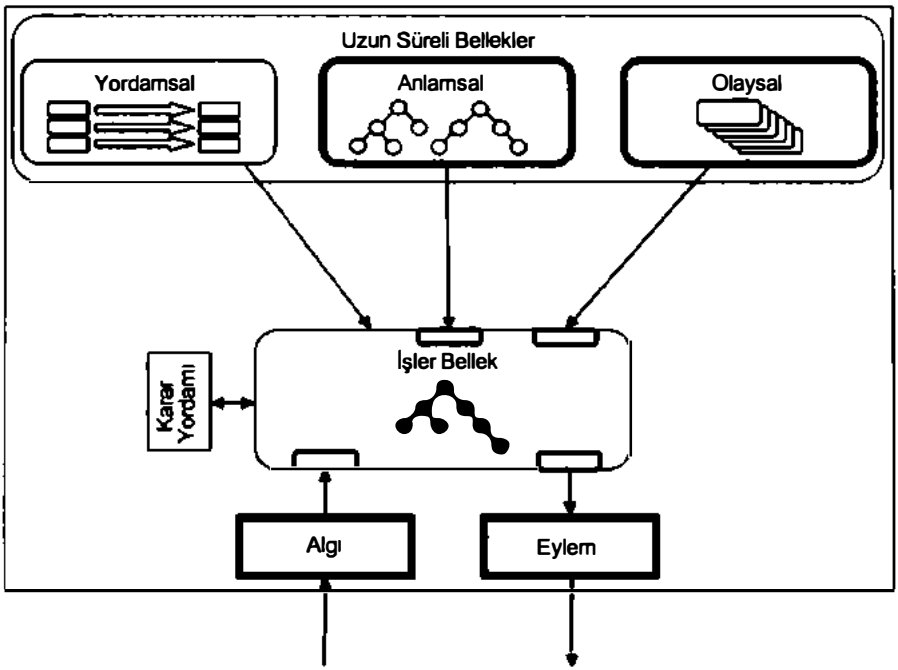
Sıkça karşılaşılan uzun süreli ve kısa süreli yapılarıyla birlikte bir üretim sistemi, problem uzamı kurma amacıyla kullanılır. Uzun süreli bellek (USB), güncel durumdan bağımsız bilgi saklar. Kısa süreli ya da işler bellek (İB), güncel durumla en ilgili bilgiyi depolar. SOAR, kendi problem çözme deneyimlerinin sonuçlarını, hem USB'sindeki ürünler olarak hem de gelecekteki durumlarda yararlı olabilecek, önceki durumlarla ilgili İB'sindeki genel olgular olarak önbelleğe alıp bir şeyler öğrenir. Eskiden deneyimlenmiş problem çözme izlerinden oluşan öğrenme dizilerini bir araya getirmeye "yığınlama" denir; bu terim bazen insanlardaki benzer öğrenme sürecini tanımlarken de kullanılır.

Bu bellek yapıları arasındaki ilişki (SOAR'ın yeni bir sürümünde), Resim 31.14'te gösteriliyor.

Üç tür USB yapısı bulunur: yordamsal, anlamsal ve olaysal. SOAR makalelerinden biri, bunları şöyle anlatıyor:[58]

> Yordamsal bilgi, şeylerin nasıl ve ne zaman yapıldığı hakkındadır; bisiklet nasıl sürülür, cebir problemi nasıl çözülür, ya da tarif nasıl okunur da bununla kek pişirilir gibi. Anlamsal bilgi, dünyaya dair olgulardan meydana gelir, yani genelde doğru olduğuna inandığınız şeylerden; hep "bildiğiniz" şeylerdir bunlar, örneğin bisikletlerin iki tekerleği olması, beyzbol oyununda dokuz vuruş sırası olması ve vuruş sırasının, üç ıskayla geçmesi gibi. Olaysal bilgi, "hatırladığınız" şeylerden meydana

58 Jill Fain Lehman, John Laird ve Paul Rosenbloom, "A Gentle Introduction to SOAR: An Architecture for Human Cognition: 2006 Update"; şu internet adresinde: http://ai.eecs.umich.edu/soar/sitemaker/docs/misc/Gentle-Introduction-2006.pdf.



Resim 31.14. SOAR'da bellek yapıları. (Çizim, John Laird ile Paul Rosenbloom'un izniyle kullanılmıştır.)

gelir; örneğin bisikletten düşüp dirseğinizin sıyrılması gibi yaşamış olduğunuz belirli durumlardan. USB'ye doğrudan ulaşılamaz, fakat güncel durumla ilintili olanı bulmak için "aranmalıdır."

Biraz daha sezgilere hitap ederek anlatmak gerekirse, USB, pek çok farklı durumla ilişkili şeyin açıkça hatırlanmasını içerirken, İB, modelin o an içinde bulunduğu durumla ilintili olanın ne olduğunu düşünmeyi içerir. İB ile USB arasındaki kilit ayrımlardan biri, işler bellekteki bilginin, USB'deki başka bilgileri getirmek üzere kullanılmasının gerekmesidir, oysa USB, öncelikle İB'ye getirilmelidir. Bilgi, USB'den İB'ye, ilintili USB yapılarının hem otomatik hem kasıtlı getirilmesiyle geçer.

SOAR mimarisi (çeşitli sürümleriyle), bütün dünyada çeşitli görevlerde araştırmacılar tarafından kullanılmıştır. SOAR üzerinde çalışan insanların niyeti, şunları yaptırmaktır:[59]

- Zeki etmenlerden beklenen, son derece sıradan görevlerden hayli zorlu, açık uçlu problemlere kadar, çeşit çeşit görev üzerinde çalışmak,

59 SOAR internet sitesinden: http://sitemaker.umich.edu/soar/home.

- Uygun bilgi biçimlerini temsil edip kullanmak, örneğin yordamsal, bildirimsel, olaysal ve büyük olasılıkla ikonik bilgi biçimleri,
- Çeşitli problem çözme yöntemleri devreye sokmak,
- Dış dünyayla etkileşime girmek,
- Görevlerin tüm yönlerini ve SOAR'ın bunlar üzerindeki performansını öğrenmek.

Fakat bunu gerçekleştirmek çok zor, o yüzden izlemeye devam edelim. İlginç bir SOAR uygulaması alanı, taklit eğitim araştırmalarında insanların dublörü olarak otomatik etmenler programlamak olmuştur. Örneğin TacAir-SOAR, avcı uçağı pilotlarının zeki davranışlarını taklit eder.[60] 1998'de Laird, SOAR Technology şirketini kurdu; Ann Arbor (Michigan) kökenli bu şirket, SOAR mimarisi kullanarak özerk YZ birimleri yaratmada uzmanlaşmıştır.

ACT-R ve SOAR, büyük olasılıkla en önde gelen bilişsel mimarilerdir, fakat başka mimariler de var. Pat Langley, John Laird ve Seth Rogers'ın birlikte yazdıkları makale, bu alanın gayet güzel bir yazın taramasını veriyor.[61]

60 M. Jones ve diğ., "Automated Intelligent Pilots for Combat Flight Simulation", *AI Magazine*, c. 20, no: 1, 1999, s. 27-41.

61 Pat Langley, John E. Laird ve Seth Rogers, "Cognitive Architectures: Research Issues and Challenges", 2006; şu internet adresinde: http://cll.stanford.edu/~langley/papers/final.arch.pdf.

SEKİZİNCİ KISIM

MODERN YZ: BUGÜNÜ VE YARINI

YZ teknolojisi, son yirmi yılda kayda değer oranda olgunlaşmıştır ve günümüzde, bir dizi güçlü hesaplama aracı içermektedir. Görece ucuz bilgisayarların gitgide güçlenmesi, büyük veritabanlarının ortaya çıkışı ve İnternetin genişlemesi sayesinde bu araçlar daha verimli şekilde yürürlüğe sokuluyor. Günümüzün YZ programları, insanların pek çok bilişsel yeteneğini kestirebiliyor, bazılarını bütünüyle otomatik hale getirebiliyor, hatta kimi becerilerde insanları aşıyorlar. Günümüzün YZ'si birçok gerçek dünya probleminin çözümüne katkı verebileceği için, YZ araştırmalarında uzmanlaşmış pek çok üniversite mezunu, akademik YZ araştırma yerine şirketlerde çalışmayı ya da kendi şirketlerini açmayı seçiyor. Sadece iki örnek vermek gerekirse, Google ve Microsoft, bünyelerine pek çok üniversite mezunu katmıştır.

Başka mühendislik dalları nasıl yavaş yavaş alt uzmanlık alanları geliştirmişse, YZ de geliştirmiştir. Örneğin, Temmuz 2009'da düzenlenen Uluslararası Birleşik Yapay Zekâ Konferansı'nda (IJCAI), şu "konularda" makaleler sunulmuştur: etmen tabanlı ve çok etmenli sistemler; kısıt, sağlanabilirlik ve arama; bilgi gösterimi, uslamlama ve mantık; yapay öğrenme; çok disiplinli konular ve uygulamalar; doğal dil işleme; planlama ve çizelgeleme; robotik ve görü; YZ'de belirsizlik; ağ ve bilgi tabanlı bilişim sistemleri. (Bu alanların pek çoğunun, en az iki geniş konu başlığını birleştirdiğine dikkat edin; bu konu başlıkları, bildiri çağrısında daha da ayrıntılandırılmıştır.[1]) Elbette, bu alt uzmanlık alanlarının birçoğu birbirinden güç alır, böylece alan bir bütün olarak bağlı kalır.

Her ne kadar benden yetkin tarihçiler, şimdiki zamana fazlasıyla yakınlaşan anlatılardan kaçınarak akıllıca davransa da (geleceği tahmin etmek, daha bile tehlikelidir), kitabın bu son kısmında, YZ arayışının nasıl seyrettiğine bakıp gelecek hakkında fikir yürüteceğim. Son birkaç yıl, yeni YZ teknolojilerinin ve uygulamalarının çağladığına tanıklık ettiği için, YZ programlarının günümüzde yaptıklarına ve henüz yapabileceklerine dair temsili örneklerden ancak sınırlı sayıda sunacak kadar yerim var.

1 Bkz. http://ijcai-09.org/fcfp.html.

32
SIRADIŞI UYGULAMALAR

YZ'nin yeni uygulamalarından bazıları, YZ'nin ilerlemesinde sıradışı kilometre taşları olarak duruyor; kimi uygulamaları gündelik işlerimize öyle sızmışlardır ki neredeyse görünmez olmuşlardır. Bu iki aşırı uç arasında, YZ programları, bilimde ve ticarette önemli araçlar haline gelmiştir. Bu üç kategori, günümüz YZ'sinin durumunu örgütlemede faydalı bir yöntem sağlıyor. Öncelikle, 21. yüzyılın hemen öncesinde ve sonrasında manşetlere taşınmış bazı sistemlere göz atacağım ve işe, oyun oynayan YZ programlarıyla başlıyorum.

32.1 OYUNLAR

Bazılarına göre, bilgisayarların satranç ve dama gibi oyunlarda ustalaştırılması daha ciddi çalışmalardan anlamsız bir sapış olsa da, oyun oynayan bilgisayarlar, yeni YZ tekniklerinin keşfedilmesinde laboratuvar olarak hizmet etmiştir; özellikle sezgisel aramada ve öğrenmede. Örneğin, önceki bir bölümde, pekiştirmeli öğrenme yöntemlerinin, şampiyon düzeyinde tavla oynayan programların geliştirilmesinde nasıl kullanıldığını açıklamıştım. YZ'nin ilk günlerinden bu yana insanlar, satranç ve dama oynayan programlar üzerinde çalışmıştır ve günümüzde bilgisayarlar, esasen muazzam miktarda sezgisel yöntem kılavuzlu hesaplama kullanarak, bu ve benzeri oyunları insanlardan daha iyi oynuyor.

32.1.1 Satranç

1997'nin çarpıcı haberlerinden biri, dünya satranç şampiyonu Garry Kasparov'un, IBM'in "Deep Blue" adlı satranç bilgisayarına yenilmesiydi (bkz. Resim 32.1). Şubat 1996'da ilk karşılaşmalarında, Deep Blue ilk oyunu almış olsa da maçı Kasparov'a kaybetmişti. Fakat 11 Mayıs 1997'de, bilgisayarın güçlendirilmiş donanımlı 1997 sürümü (gayriresmi olarak "Deeper Blue" denir), altı oyunluk maçı (normal satranç turnuvası zaman kısıtlamalarıyla) kazanırken iki oyunu almış, bir oyunu vermiş, üçünde de berabere kalmıştı. (Bilgisayar Ta-



Resim 32.1. Garry Kasparov, altı oyunluk tekrar maçının ikinci oyununda Deep Blue'ya karşı satranç oynuyor. (Fotoğraf, AP/Wide World Photos © izniyle kullanılmıştır.)

rihi Müzesi'nin *Endgame: Challenging the Chess Master* başlıklı filmi için şu internet adresini ziyaret ediniz: http://www.youtube.com/watch?v=5hRNlfAUeEE. 1997'deki maçın oyun hamlelerinin kayıtları için bkz. http://www.research.ibm.com/deepblue/watch/html/c.shtml.)

1997 model Deep Blue, IBM RS/600 SP2 süperbilgisayarında çalışan, özel amaçlı bir donanım ve yazılım birleşimiydi. Özelliklerinden bazıları, "1996 yılında Kasparov'a karşı oynanmış oyunlarda gözlemlenen özgül problemlere yanıt olarak" gündeme getirilmiş iyileştirmeleri içeriyordu.[1]

Kasparov, yenilgiden sonra *New York Times*'a "hiç oyun oynayacak ruh halinde değildim", demiştir.[2] *Times*'da çıkan makale şöyle devam ediyordu: "Cumartesi günkü beşinci oyundan sonra, Kasparov'un keyfi o kadar kaçmıştı ki maçın çoktan bittiğini düşünüyordu. Nedeni sorulunca şöyle konuştu: 'İnsanım ben. Anlayışımın çok ötesinde bir

1 Murray Campbell ve diğ., "Deep Blue", şu internet adresinde: http://sjeng.org/ftp/deepblue.pdf.

2 Bruce Webber, "Swift and Slashing, Computer Topples Kasparov", *New York Times*, 12 Mayıs 1997.

şey gördüğümde korkarım."' Bazı internet sitelerine göre, yenilgiden sonra Kasparov, makinenin hamlelerinde zaman zaman derin zekâ ve yaratıcılık gördüğünü söyledi. Fakat bu ifadeyle, maçın ikinci oyununda insan satranç oyuncularının işe karışmış olabileceğini ima ediyordu. Kasparov tekrar maç yapmak istedi ama IBM makineyi sökmüştü ve bir daha maç yapılmadı.

Fakat, Kasparov'u savunanlar da olmuştur. *Times*'da çıkan makale, eski ABD şampiyonu Lev Albert'ın şu sözlerini aktarır: "Bu bir gösteriydi. Eğer onlar [IBM], bunun bir gösteriden fazlası olduğunu kanıtlamak istiyorsa, Garry dışında başka biriyle oynatsınlar bilgisayarı. Örneğin Büyük Usta Boris Gulko'ya karşı oynasa, ki kendisi ilk ellide bile değil, bilgisayarın maçı kaybedeceğine 10.000 dolar bahse girerim."[3]

Deep Blue'nun tarihi, Carnegie Mellon Üniversitesi'nde doktora öğrencisi Feng-Hsiung Hsu'nun geliştirdiği satranç programı "Deep Thought"la başlamıştı. IBM'in Deep Blue hakkındaki internet sitesine (http://www.research.ibm.com/deepblue/) göre, "IBM Deep Blue projesi, Hsu ile Murray Campbell [Hsu'nun Carnegie Mellon'dan sınıf arkadaşı] IBM Research'e 1989'da katılıp" paralel işlemci sistemleri üzerinde çalışmaya başladıklarında doğdu. Kasparov maçını kazanan Deep Blue sürümü, 256 "satranç işlemcisi" içeriyordu; bu işlemciler hep birlikte, saniyede yaklaşık 200 milyon satranç konumunu inceleyebiliyordu. (Kıyaslamak için, 1970'lerde Northwestern Üniversitesi'nin CHESS 3.0 programının, saniyede sadece 100 konum değerlendirdiğini belirtelim.)

Deep Blue, satranç konumlarını hem "hızlı" hem de "yavaş" değerlendirme işlevi kullanarak değerlendiriyordu. Bu işlevlerde kullanılan yaklaşık 8.000 özelliğin değerleri, özel bir donanım tarafından hesaplanıyordu. Deep Blue hakkında yazılmış bir makaleye göre:[4]

> [Hem yavaş hem hızlı işlevlerin] bu kullanımı, kestirimin yeterince iyi olduğu durumlarda, masraflı tam değerlendirmeyi atlamak amacıyla yararlanılan standart bir tekniktir. Tek saat çevrimi içinde belli bir satranç konumu için puan hesaplayan hızlı değerlendirme işlevi, yüksek değerli olup kolay hesaplanan başlıca tüm değerlendirme koşullarını içerir. Hızlı değerlendirmenin en önemli parçası, "taş yerleştirme" değeridir, yani temel taş değerlerinin, bulundukları karelere göre yapılmış ayarlamalarıyla toplamı. Örneğin "piyon kaçabilir" gibi hızlıca hesaplanabilen konumsal özellikler de hızlı değerlendirmenin parçasıdır. Yavaş değerlendirme işle-

3 http://www.nytimes.com/library/cyber/week/051297weber.html.

4 Murray Campbell ve diğ., "Deep Blue", şu internet adresinde: http://sjeng.org/ftp/deepblue.pdf.

> vi, tahtada tek tek her sırayı tarar, kare denetimi, açmaza alma, çapraz hamleler, şah güvenliği, piyon dizilişi, kaybedilmiş piyonlar, olası hamle denetimi, ileri karakol, piyon egemenliği, rok, abluka, kısıtlama, renk stratejileri, tuzağa düşürülmüş taşlar, gelişim, ve benzeri gibi satranç kavramları için değerler hesaplar. Hem yavaş hem hızlı değerlendirme işlevlerinde tanınan özelliklerin, programlanabilir ağırlıkları bulunur, böylece göreli önemleri kolaylıkla ayarlanabilir.

Sezgisel arama (buna, Deep Blue'nun değerlendirme işlevleri kılavuzluk eder), altı ila on altı kat derinlikten, bazı durumlarda en fazla kırk kat derinliğe kadar aramaya olanak tanır. Aramaya ek olarak Deep Blue, 4.000'i aşkın konum içeren standart "kayıtlı hamleleri" temel alabilir. Programın oyununu, 700.000 büyük usta oyunundan oluşan veritabanı da etkiler. Aynı zamanda, "tahtada en fazla beş taşın olduğu tüm satranç konumlarını ve bloke edilmiş piyonların eşliğinde altı taşlı seçili konumları" içeren oyun sonu veritabanlarından da faydalanır.

Deep Blue, bazı bilgisayar bilimcilerinin "kaba kuvvet" yöntemi dediği yöntemler sayesinde kazandığı için, bu zaferin bir "YZ başarısı" olduğu söylenebilir mi? IBM'in bu konuyla ilgili görüşü şöyle:[5]

> Deep Blue, yapay zekâ kullanıyor mu? Kısa yanıt "hayır"dır. İnsan düşünüşünü taklit etmeye çalışan eski bilgisayar tasarımları, bu işte pek iyi değildi. Sezgilere karşılık gelen bir formül yok... Deep Blue, daha çok hesaplama gücüne, ayrıca basit arama ve değerlendirme işlevlerine dayanır.
>
> Uzun bir yanıt vermek gerekirse, yanıt yine "hayır" olur. "Yapay Zekâ", dünyamızdan ziyade bilimkurguda başarılı; ayrıca, çok da iyi anlamadığımız bir süreci taklit eden makine tasarlamanın neden zor olduğunu bilmek için Isaac Asimov olmak şart değil. Nasıl düşündüğümüz sorusu, yanıtı olmayan bir sorudur. Deep Blue, uğraşsa bile asla bir HAL-9000 olamaz. Hatta Deep Blue, "uğraşmak" ne demek onu da bilmez.

IBM, Kasparov ve Deep Blue'nun satranç oynama yaklaşımları arasındaki farkları sıralarken şu maddelerden bahseder:[6]

- Deep Blue, saniyede 200.000.000'a kadar satranç konumunu inceleyip değerlendirebilir; Garry Kasparov ise saniyede en fazla üç satranç konumu inceleyip değerlendirir.
- Deep Blue'nun satranç bilgisi kıt, hesaplama becerisi muazzamdır; Garry Kasparov'un satranç bilgisi müthiş, hesaplama becerisi bir parça düşüktür.

5 http://www.research.ibm.com/deepblue/meet/html/d.3.3a.shtml#ai.

6 http://www.research.ibm.com/deepblue/meet/html/d.2.shtml.

- Garry Kasparov, dünya şampiyonu kalibresinde satranç oynamak için inanılmaz hislerini ve sezgisini kullanır; Deep Blue, histen ya da sezgiden yoksun bir makinedir.
- Garry Kasparov, kendi başarılarından ve hatalarından ders çıkarıp bunu oyununa çabucak uyarlayabilir; bugünkü haliyle Deep Blue, bir "öğrenme sistemi" değildir. Dolayısıyla, rakibinden bir şeyler öğrenecek ya da satranç tahtasının o anki konumu hakkında "kafa yoracak" yapay zekâdan faydalanamaz.
- Deep Blue'nun satranç oynama tarzındaki herhangi bir değişiklik, oyunlar arasında geliştirme ekibinin üyeleri tarafından yapılmak zorundadır; Garry Kasparov, oyun tarzını herhangi bir anda değiştirebilir, örneğin oyun esnasında, öncesinde ve sonrasında.

Fakat YZ'ye dair görüşüm daha geniş kapsamlı. Her ne kadar Deep Blue, (örneğin) kural tabanlı uslamlama yerine kaba kuvvet yöntemlerine dayansa da, YZ'nin temel tekniklerinden biri olan sezgisel aramadan faydalanır. Kasparov ile Deep Blue arasındaki farklar, satranç programlarının, kaba kuvvete ek olarak insan satranç oynama bilgisini ve becerisini (bunlar, programlayacak kadar iyi bilindiğinde), ayrıca yapay öğrenme yöntemlerini devreye soksalar ne kadar başarılı olacaklarını gösteriyor.

John McCarthy, benzer görüşlerini dile getirmiştir. Yakın tarihli bir makalede şöyle yazmıştır:[7]

> Bununla birlikte, şampiyon düzeyindeki bu oyunun, insan satranç oyuncularına kıyasla milyon kat hesaplama gerektirmesi, yapay zekâ ilkelerine dair anlayışımızın ne kadar kısıtlı olduğunun ölçüsüdür.
>
> ... Şampiyon düzeyinde oyun oynamak, Deep Blue'ya ve yeni rakiplerine kıyasla çok daha az hesaplama gücüyle de mümkün.

McCarthy devam edip şöyle der: "Bilgisayar satranç turnuvaları, "hesaplama gücü cidden sınırlandırılmış programları kabul etmeli sadece. Bu şart, dikkati bilimsel ilerlemelere yoğunlaştırmayı getirir."

Bilgisayarlar ile insanlar arasında, ayrıca bilgisayarların kendi arasındaki maçlar oynanmaya devam ediyor. 25 Kasım ve 5 Aralık 2006 tarihlerinde, Almanya'nın Bonn kentinde oynanan maçta Dünya Şampiyonu Vladimir Kramnik, Almanya'da Frans Morsch ile Mathias

7 John McCarthy, "AI as Sport", kitap incelemesi, *Science*, c. 276, no: 5318, 6 Haziran 1997, s. 1518-1519.

Feist'ın geliştirmiş olduğu satranç programı Deep Fritz'e karşı oynadı. Maçtaki altı oyunun ikisini Deep Fritz kazanırken, dördü berabere bitti. Kramnik şunu demiştir: "Deep Fritz 8 [ucuz sürümü], Deep Blue'dan güçlü". Programın en yeni sürümü Deep Fritz 11'dir. (Deep Fritz 8, şu internet adresinden satın alınabilir: http://www.chesscentral.com/software/deep-frtiz-8.htm.)

Bilgisayar satranç programlarına ve bunlar arasındaki maçlara adanmış birtakım internet siteleri vardır.

32.1.2. Dama

Profesör Jonathan Schaeffer (1957- ; Resim 32.2) ve Kanada'nın Edmonton şehrinde Alberta Üniversitesi'ndeki ekibi, 2007 yılının eylül ayında, "Dama Çözüldü" başlıklı bir makale yayımladı; "iki tarafın da kusursuz oyununun beraberlikle sonuçlacağını" ilan etmişlerdi.[8] Schaeffer ve çalışma arkadaşları, 1989'dan beri damayı çözmeye uğraşıyordu. Gerekli hesaplamaların o zamandan bu yana neredeyse kesintisiz sürdüğünü söylemişlerdir. Damanın beraberlikle sonuçlandığına dair nihayet tamamlanan kanıt, "asla kaybetmeyen açık bir stratejiden oluşur; program, ister siyah ister beyaz taşlarla oynasın, herhangi bir rakibe karşı en kötü beraberlik alıyor". Dama ekibi bu sonucun şanını, "ileri YZ programlarına ve geliştirilmiş donanıma (daha hızlı işlemciler, daha büyük bellekler ve daha büyük diskler)" veriyor. Bu çabanın, on sekiz yıllık vasıflı bir uğraşı ve muazzam miktarda hesaplamayı gerektirmesi hiç şaşırtıcı değil, çünkü dama oyununda 500.995.484.682.338.672.639 farklı konum vardır!

Kanıta giden yolda bu ekip, "CHINOOK" adlı, sürekli gelişen mükemmel bir dama programı geliştirmiştir. (Chinook, batıdan gelip Kanada Rocky Dağlarından aşağı Büyük Bozkıra doğru esen ılık kış rüzgarıdır.) 1992'nin ilk insan-makine dünya satranç şampiyonasında, dama şampiyonu Marion Tinsley, CHINOOK'u, otuz üç beraberliğin yanında dörde karşı iki galibiyetle yenmişti.[9]

8 Jonathan Schaeffer ve diğ., "Checkers is Solved", *Science*, c. 317, 14 Eylül 2007, s. 1518-1522.

9 Jonathan Schaeffer, "Man versus Machine: The Silicon Graphics World Checkers Championship", TR 92-19, Bilgisayar Bilimi Bölümü, Alberta Üniversitesi, 1992. Şu internet adresinde: http://www.cs.ualberta.ca/~jonathan/Papers/Papers/TR92-19.ps. Ayrıca bkz. Jonathan Schaeffer ve diğ., "Man versus Machine for the World Checkers Championship", *AI Magazine*, c. 14, no: 2, 1993, s. 28-35.

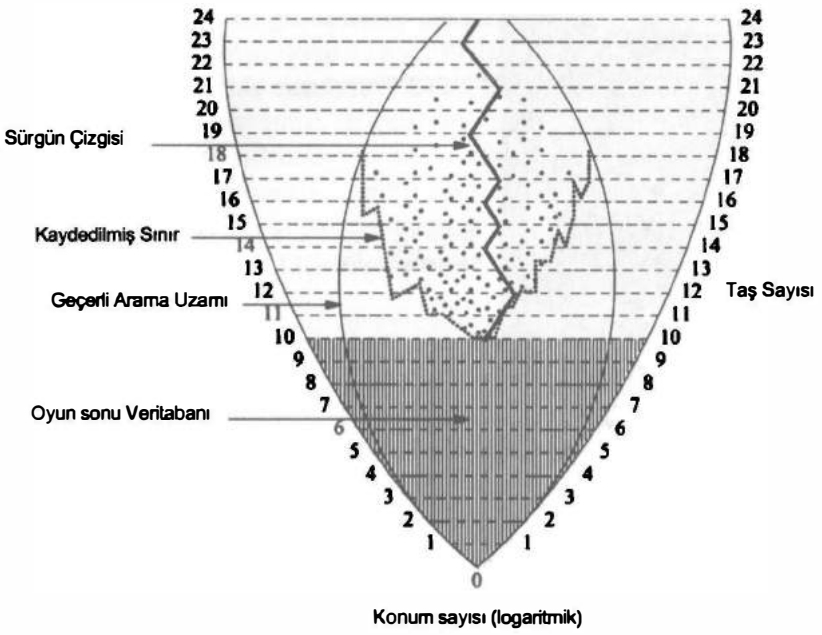


Resim 32.2. Jonathan Schaeffer (Fotoğraf, Jonathan Schaeffer'in izniyle.)

Çok daha gelişmiş bir CHINOOK'la yapılan rövanş maçı, 1994'te Boston Bilgisayar Müzesi'nde düzenlendi. CHINOOK ekibine göre:[10] "1994 model CHINOOK, [CHINOOK 1992'ye kıyasla] daha iyi ve derin arama yapıyordu, konumları daha iyi değerlendiriyordu, daha çok ve daha nitelikli oyun sonu veritabanlarına erişebiliyordu ve 12 kat daha fazla (ve daha nitelikli) açılış bilgisine ulaşabliyordu". Rövanş karşılaşmasının ilk altı oyunu berabere sonuçlandı. Yedinci oyundan önce, Tinsley sağlık sebepleri ileri sürerek maçtan çekildi. Kurallar uyarınca, CHINOOK İnsan-Makine Dünya Şampiyonu ilan edildi. (Kısa süre sonra Tinsley'e pankreas kanseri tanısı konulacaktı.) Maç organizatörleri, oyunun Boston'da devam edebilmesi için Büyük Usta Don Lafferty'yle CHINOOK arasında maç ayarladı. Bu karşılaşmada CHINOOK unvanını korudu, çünkü iki taraf birer galibiyet almış, on sekiz oyunda yenişememişlerdi. Tinsley, sağlığı yeterince iyileşince unvanını tekrar kazanmak için maç yapmayı teklif etti, fakat maç yapılamadan 1995'te vefat etmiştir.

Damada en az berabere kalınacağının her zaman kesin olduğunun kanıtı, bol miktarda hesaplama ve çok büyük veritabanları içerir. Kanıt ekibi, "dama, bugüne kadar çözülmüş oyunlar arasında hesap bakımından en zorlayıcısıdır", diye yazmıştır. Kanıtın yapısının genel fikrini anlatacağım. Resim 32.3, tahtada kalan taşların sayısı ile bu taşların düzenlenebileceği dama konumları arasındaki ilişkiyi gösteren bir diyagram görülüyor. (Dikey eksen, taş sayısını gösteriyor;

10 Jonathan Schaeffer ve diğ., "CHINOOK: The Man-Machine World Checkers Champion", *AI Magazine*, c. 17, no: 1, 1996, s. 21-29.



Resim 32.3. Dama kanıtının şeması. (Çizim, Jonathan Schaeffer'ın izniyle kullanılmıştır.)

yatay eksen, konum sayısının logaritmasıdır.) Diyagramın dibindeki gölgeli alan, en fazla on taşın tüm olası dizilişlerini temsil ediyor. Bunun gibi 39.271.258.813.439 konum vardır ve Scahefer ile ekibi, her oyuncu için bu konumların galibiyetle mi, mağlubiyetle mi, beraberlikle mi sonuçlandığını hesaplayıp oyun sonu veritabanlarına işlemiştir. Diyagramdaki açık renkli küçük çemberler, bir değer (galibiyet, mağlubiyet ya da beraberlik) belirlenmiş konumları (en az on taş) temsil eder. Optimum oyun, başlangıç konumundan itibaren en azından beraberliğin kesin olduğu gölgeli alandaki bir konuma ulaşan bir çizgi bulmak amacıyla sezgisel arama kullanımı içerir. Örnek bir yol, diyagramda "sürgün" diye işaretlenmiş kalın çizgidir. (Bu diyagramın başka özellikleri, kanıtı ortaya koyan makalede açıklanıyor.) Kanıtın çeşitli konumları nasıl değerlendirdiğini CHINOOK internet sayfalarını ziyaret ederek inceleyebilirsiniz: http://www.cs.ualberta.ca/~chinook/. Bu internet sayfalarında, CHINOOK'a karşı dama da oynayabilirsiniz.

Bana kalırsa, damayla ilgili bu sonuç harika bir YZ performansıdır. Makale yazarlarının sonuç bölümünde yazdıkları gibi:

> Dama hesaplamaları, arama yoğunluklu algoritmalarla elde edilebileceklerin sınırlarını zorluyor. Yapay zekâda sınırlı bilgi yaklaşımlarının

gücünü gösteren ikna edici kanıtlar sağlıyor. Derin arama, bilgiyi örtük şekilde sunar. Dahası, arama algoritmaları, çok çekirdekli hesaplamanın kısa süre içinde gündeme getireceği işlemci paralelliği artışından faydalanmaya gayet uygundur. YZ'de arama yoğunluklu yaklaşımlar, bu alanın evriminde gitgide daha önemli rol oynayacak.

O halde, yoğun arama yöntemleri, s. 262'de betimlemiş olduğum "bilgide güç yatar" öğretisine meydan okumak üzere yeniden boy gösteriyor. Bu başarıya ulaşmak için onca kişisel çabanın sarf edildiğini belirtmeden geçmeyelim. Jonathan Schaefer'in karısı Stephanie'nin şunu söylediği aktarılır: "18 yıl oldu!.. saplantılı-zorlanımlı bir davranış bu... hiç normal değil... başka işin yok mu Jonathan."[11]

Peki, çok ama çok daha büyük bir arama uzayı olan satranca ne demeli; bu oyunda en azından 10^{40} konum vardır. Damada ise sadece 500×10^{18} konum var, yani satrancınkinin neredeyse kare kökü kadar. Makalenin yazarları, satrancın kısa sürede çözülmesinin pek olası olmadığını düşünüyor. Şöyle yazmışlardır: "Damayı çözmenin gerektirdiği çabaya bakılınca, yeni bir teknoloji icat edilmezse, satrancın uzun süre çözülmeyeceğini söyleyebiliriz.[12]

32.1.3 Başka Oyunlar

Günümüzde bilgisayarların çok iyi oynadığı başka oyunlar da var. Çok ilgi çeken örneklerden biri pokerdir. Alberta Üniversitesi'nin, AAAI'nın Bilgisayar Pokeri Yarışmasına adanmış bir internet sayfası var: http://www.cs.ualberta.ca/~pokert/indiex.html. Örneğin, 2008 yarışmalarında hem "limitli teksas pokerine" hem de "limitsiz teksas pokerine" dokuzar yarışmacı, "altı oyunculu limitli teksas pokerine" altı oyuncu katıldı.[13] Şu internet adresinde, Alberta Üniversitesi'nin bazı poker programlarına karşı oynayabilirsiniz: http://poker.cs.ualberta.ca/.

Matt Ginsberg (1955-), briç oynayan GIB adlı (Goren in a Box ya da Ginsberg's Intelligent Bridgeplayer) bir program geliştirmiştir.[14] Oyu-

11 Schaefer'in CHINOOK'la ilgili slaytlardan biri şu internet adresinde: http://www.cs.ualberta.ca/~chinook/news/ChinookTalk.pdf.

12 Schaeffer ve dama hakkında daha fazlası için bkz. Jonathan Schaeffer, *One Jump Ahead: Computer Perfection at Checkers*, 2. baskı, New York: Springer-Verlag, 2008.

13 Sonuçlar için bkz. http://www.cs.ualberta.ca/~pokert/2008/results/.

14 Matthew L. Ginsberg, "GIB: Imperfect Information in a Computationally Challenging Game", *Journal of Artificial Intelligence Research*, c. 14, 2001, s. 303-358. Şu internet adresinde: http://www.cs.cmu.edu/afs/cs/project/jair/pub/volumel4/ginsberg01a.pdf.

nun herhangi bir aşamasında, GIB kendi elindeki kartları, oyun eşinin elindeki kartları ve o ana dek oynanmış kartları bilir. Böylece, kalan kartları rakiplere rasgele dağıtır ve bu rasgele dağıtıma göre oynanacak en iyi kartı hesaplar. Gerçekten kart oynamaya başlayana dek bu süreci binlerce kez tekrar eder. Bu "Monte Carlo" yaklaşımıyla topladığı istatistiklere dayanarak, en iyi olduğunu düşündüğü kartı seçip oynar.

14 Mart 2008 tarihli bir *New York Times* makalesinde briç yazarı Philip Alder, birbirine karşı oynayan iki insanın oyununu (her birinin eşi GIB'ti) incelemişti. Alder, GIB'in oyununu özetlerken şunu demiştir: "Bu robotlar, bu tür programlara bazen denildiği gibi, bir tuhaf. Arasıra iyi deklarasyon yapıp güzel oynuyorlar, ama çoğunlukla tuhaf kararlar veriyorlar."[15] Şu internet adresinde, (GIB'le birlikte ya da ona karşı) briç oynayabilirsiniz: http://www.bridgebase.com; ayrıca, GIB kopyası satın alabileceğiniz bir GIB internet sitesi de var: http://www.gibware.com. Ginsberg, GIB üzerinde çalışmayı bırakmıştır ve muhtemelen daha güçlü programlar vardır. Jonathan Schaeffer bana, briç hakkında şu yorumu yazmıştı: "Güçlü bir oyuna ulaşmada bize ayak bağı olan şey hâlâ oyunun deklarasyon kısmı. Deklarasyon âdetleri, insanların tasarlamış olduğu âdetler ve bu 'deklarasyon dilini' kullanarak oynamayı geliştirmiş insan oyuncuların tüm kurallarını kapmak zor."[16]

Büyük olasılıkla, bilgisayarlar için en zor oyunlardan biri Go'dur. 2008'de, Fransa'da INRIA ve Hollanda'da Maastricht Üniversitesi'nin geliştirmiş oldukları MoGo Titan adlı program,[17] Hollandalı süperbilgisayar Huygens'e dokuz taş avantaj verildiği bir oyunda meslekten bir Go oyuncusunu yendi. Bilgisayar Go programları gelişmeyi sürdürüyor. Artık en iyileri, hamle seçimi yaparken Monte Carlo yöntemlerinden faydalanıyor. Şu internet sayfası, insan-bilgisayar Go oyunlarının sonuçlarını sıralıyor: http://www.computer-go.info/h-c/index.html.

Elbette son olarak Scrabble® oyununa bakalım; bu oyun, büyük sözlüklere ulaşma ve muazzam aramalar yapma becerileriyle tam da bilgisayarlara göre. Scrabble programları artık en iyi insan uzmanları düzenli olarak yeniyor.[18]

15 16 Phillip Alder, "BRIDGE; A Case Study of the Mysteries of the Robot Thought Process", *New York Times*, 14 Nisan 2008.

16 E-posta, 14 Nisan 2008.

17 Bkz. http://www.nwo.nl/nwohome.nsf/pages/NWOA_7HHBNS.

18 Bkz. Brian Sheppard, "World-Championship-Caliber Scrabble", *Artificial Intelligence*, c. 134, no: 1-2, Ocak 2002, s. 241-275.

Bilgisayarların oyun oynaması hakkında çok daha fazlası yazılabilirdi. Pek çok programı satın almak ya da internetten indirmek artık mümkün. Gözde oyununuzu seçip kendi becerilerinizi YZ'nin başardıklarıyla boy ölçüştürebilirsiniz! Uluslararası Bilgisayar Oyunları Birliğinin (ICGA), her türlü bilgisayar oyunu turnuvası hakkında bilgi veren bir internet sitesi var.[19] Fakat şimdi, kimilerinin boş laf saydığı (ama zorlu olan) bu konudan muhtemelen daha heyecan verici olan robotlar konusuna geçelim.

32.2 ROBOT SİSTEMLERİ

Robotlar her yerde! Mars'a, okyanus derinliklerine, hatta volkanların içine gittiler. Tarım robotları, fabrika robotları, cerrahi robotları, depo robotları var. Robot inşa etme işi, özellikle Pennsylvania eyaletinin Pittsburgh kentinde, yani Carnegie Mellon Üniversitesi'nin ve bunun Robotik Enstitüsü'nün yuvasında palazlanıyor. *The Christian Science Monitor* adlı dergide 16 Kasım 2008'de basılmış bir makaleye göre, "bugün Pittsburgh'da otuzu aşkın robotik şirketi var."[20] Günümüz "robotlarının" bazıları, aslında insanların yönlendirdiği uzaktan denetimli mekanik cihazlardır, fakat gitgide daha fazlası otomatik hale gelip kendi başına zekâsıyla hareket edebiliyor. Bazı örneklerden bahsedelim.

32.2.1 Deep Space 1'de Remote Agent

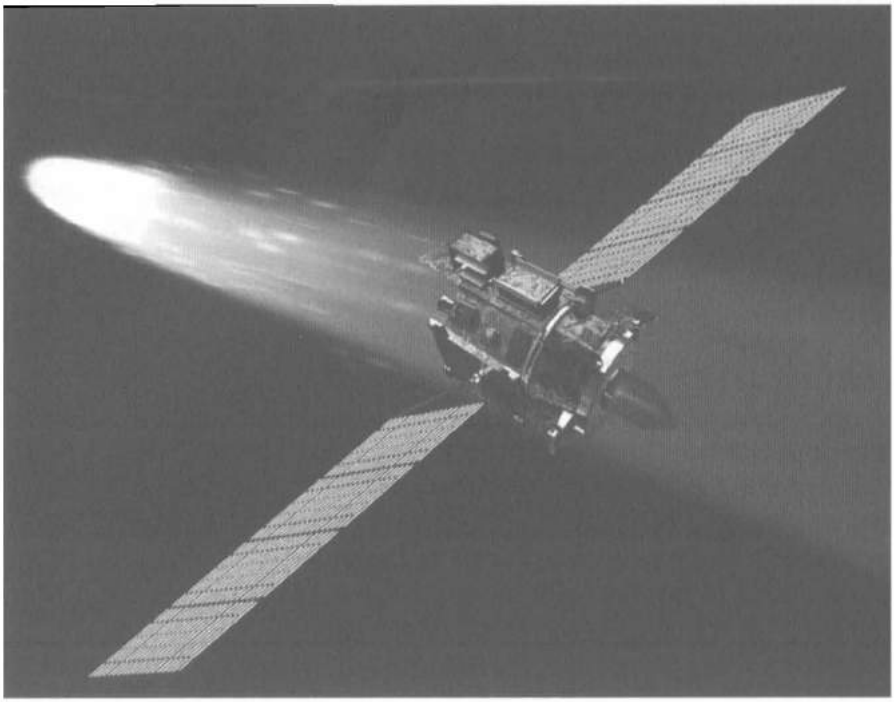
24 Ekim 1998 tarihinde NASA, Deep Space 1 (DS1) adlı uzay gemisini fırlattı; bu geminin görevi, on iki ileri teknolojinin uzayda işe yarayıp yaramayacağını değerlendirmekti. Bu teknolojilerden biri, "Remote Agent"tı (RA); bu robot sistemi, uzaygemisinin eylemlerini planlayıp gerçekleştiriyordu.[21] Resim 32.4'te, bir çizerin, 19P/Borelly adlı kuyrukyıldıza yaklaşan DS1 çizimi gösteriliyor.

Remote Agent, her ne kadar HAL 9000'in (*2001: Uzay Yolculuğu* filmi ve romanında uzay taşıtını denetleyen zeki sistem) tüm becerilerini barındırmasa da, büyük bir YZ başarısıydı. Bu adın verilmesinin sebebi, dünyadaki operatörler ile gemideki sensörler ve icracılar ara-

19 http://www.icga.org/.

20 Tom A. Peter, "Pittsburgh Is Robot Country", *The Christian Science Monitor*, 16 Temmuz 2008.

21 Nicola Muscettola ve diğ., "Remote Agent: To Boldly Go Where no: AI System Has Gone Before", *Artificial Intelligence*, c. 103, 1998, s. 5-47. Şu internet adresinde: http://groups.csail.mit.edu/mers/papers/aij98.pdf.



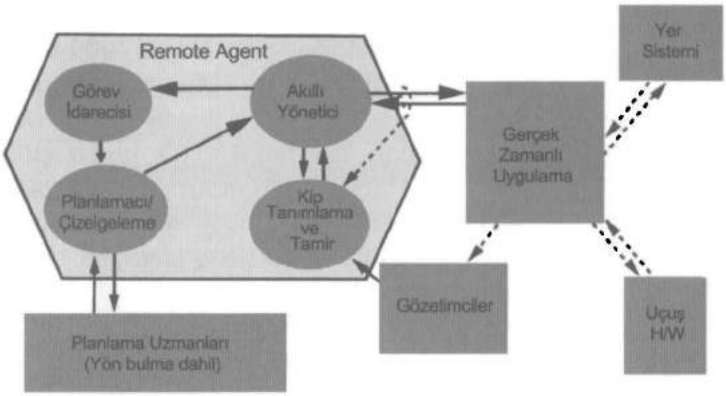
Resim 32.4. Gökyüzüne yaklaşan DS1'in sanatçı tarafından çizimi.

sında aracı olarak, yani DS1'de zeki bir etmen olarak iş görmesiydi. Bu projeyle ilgili NASA raporlarından birine göre:[22]

> RA'nın en eşsiz özelliklerinden ve geleneksel uzay gemisi komutasıyla başlıca farklarından biri, yerdeki operatörlerin RA'yla, zamanı belirtilmiş ayrıntılı komutlar yerine hedefler (örneğin, "önümüzdeki hafta boyunca şu göktaşlarının fotoğraflarını çek ve zamanın %90'ında itiş gücü ver") üzerinden iletişim kurmasıydı. RA, o hedeflere ulaşmak amacıyla bir eylem planı belirler ve uzay taşıtına komutlar vererek planı yürürlüğe sokar.

RA'yı, Pasedana'daki Jet Tahrik Laboratuvarı'nın ve California Mountain View'daki Ames Araştırma Merkezi'nin mühendisleri geliştirmişti. Sıkı sıkıya bütünleştirilmiş üç ana altsistemden oluşuyordu. Planlamacı ve Çizelgeleme birim, görev hedeflerine ulaşacak planları yaratmak için sezgisel aramadan faydalanır; Yönetici sistem, bu planların belirlediği etkinlikleri yürütür; Kip Tanımlanması ve Tamir Sistemi,

22 Bkz. http://nmp-techval-reports.jpl.nasa.gov/DS1/Remote_Integrated_Report.pdf.



Resim 32.5. Uzaktan denetimli etmen mimarisi. (P. Pandurang Nayak'ın izniyle kullanılmıştır.)

uzay taşıtının durumunu izleyip hataları keşfetmeye, saptamaya, düzeltmeye çalışır. Sistemin mimarisi, Resim 32.5'te gösteriliyor.

RA'nın Akıllı Yöneticisi (EXEC), uzay taşıtını ve bileşenlerini denetleyen Gerçek Zamanlı Yönetici sistemine komutlar verir. Bu komutlar, iyon itiş sisteminin açılıp kapanmasını, uzay taşıtının yönünün ayarlanmasını, kamerayı konumlandırmayı, ve benzeri içerir. RA'nın Kip Tanımlama sistemi, bir dizi sensör Gözlemci aracılığıyla uzay taşıtının durumu hakkında bilgi toplar. Görev İdarecisi (Gİ), görevin hedeflerini ve uzay taşıtının durumuyla ilgili EXEC'in öngörülerini temel alarak, Planlamacı/Çizelgeleme (PÇ) birimi için bir planlama problemi formüle eder. PÇ de, EXEC'in gerçekleştireceği eylemlerin zaman çizelgesinden oluşan bir plan kurar. Planlama Uzmanları, yön bulma gibi uzmanlık gerektiren işlerde planlama sürecine katılır. EXEC, PÇ'nin oluşturduğu üst düzey etkinlikler çizelgesini, motorlara, vanalara ve başka yöneticilere gidecek komutlara dönüştürür ve o komutların yerine getirilmesini takip eder. Eğer verilen işlerden biri yerine getirilmezse, EXEC, ya alternatif bir komut vermeyi dener ya da Kip Tanımlama ve Tamir sisteminden problemin saptanıp çözülmesini ister. Eğer bu eylemlerden hiçbiri işe yaramazsa, EXEC, güncel planını iptal eder, uzay taşıtını "güvenli duruma" getirir ve Gİ için yeni bir plan talep eder. Bu esnada, Kip Tanımlama sistemi, komut veren EXEC'i gözlemler, gerçekleşen olaylar hakkında duyu Gözlemcilerinden veri alır ve uzay taşıtı bileşenlerinin durumunu çıkarsayıp EXEC'e iletmek için bu bileşenlerle ilgili modellerinden yararlanır.

RA, DS1 uçuşunda kullanılmasına izin verilmeden önce NASA tarafından tam bir sınamaya ve değerlendirmeye tabi tutulmuştu. Uzay-

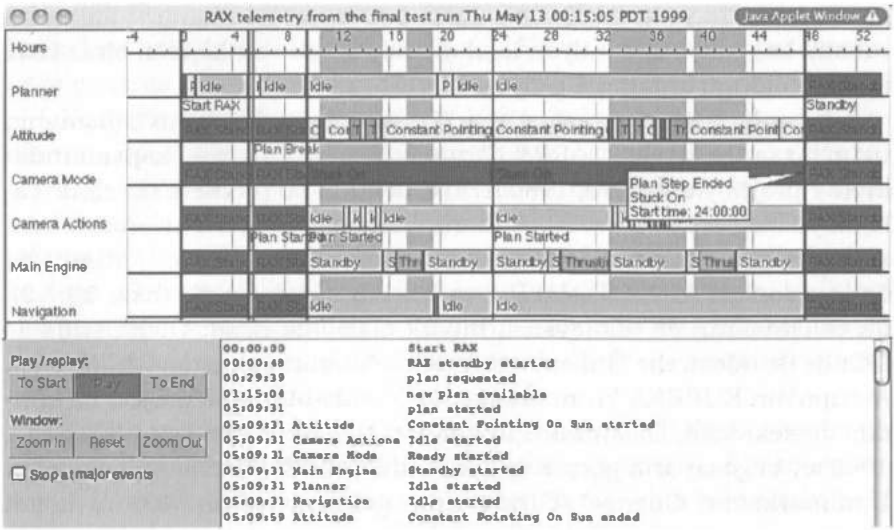
da sınanmış sürümünün adı RAX'tı (eklenen "X" harfi, deneyleri ifade eder). Uzay taşıtının denetimi, 17 Mayıs ve 21 Mayıs 1999 tarihlerinde RAX'a verildi. (İlk olarak, 17 Mayıs 1999 Batı Amerika saatiyle sabah beşten 19 Mayıs 1999 saat akşam 7'ye kadar devreye girdi. Daha sonra, 21 Mayıs 1999'da saat sabah 7:15'ten 13:30'a kadar çalıştı.) Bu deneylerde "RAX İdarecisi" (DS1 uçuş yazılımının parçası) RAX'ın uzay taşıtı denetimi gözetmiştir. RAX'la ilgili sonuç raporuna göre: "RAX idarecisi sayesinde RAX etkinliğini sıkı sıkıya denetleme becerisi, yer denetçilerinin uzay taşıtının denetimini kolaylıkla RAX'ın elinden alabileceğine, DS1 projesini ikna etmenin önemli bir etkeniydi."[23] Denetim RAX'tayken, o da DS1 uçusu sırasında sınanan İyon İtiş Sistemine (İMS), Bütünleşik Minyatür Kameraya ve Spektrometreye, Özerk Yön Bulma sistemine, uzay taşıtı Yönelim Denetim Sistemine ve bir dizi güç anahtarına komut verebiliyordu. Sonuç raporunda betimlenildiği gibi: "Ana senaryonun hedefleri, bir İMS itiş eğrisi gerçekleştirmek, otomatik yön bulucunun istediği optik yön bulma görüntülerini elde etmek ve birtakım aksaklık benzetimlerine tepki vermekti. Bu hatalar arasında, güncel planı bozmadan tepki verilebilecek ufak aksaklıklar ve kalan hedeflere ulaşmak için yeni bir plan yaratmayı gerektiren daha ciddi aksaklıklar vardı". Rapor şu sonuca varır: "Uçuş deneyi esnasında ortaya çıkan birkaç kusura rağmen, RA, uçuş geçerleme hedeflerinin %100'ünü başarıyla yerine getirmiştir."[24]

Resim 32.6, son test uçuşunda saatler boyunca RAX'ın ne işle uğraştığını gösteren telemetre diyagramıdır. Etkileşimli bir internet uygulamasından alınmıştır; http://ti.arc.nasa.gov/ptroject/remote-agent/ adresinden bu uygulamaya erişip çalıştırabilirsiniz. (Bu internet sitesi aynı zamanda RA ve RAX hakkında ayrıntı, makale, basın açıklamaları ve başka malzemelere bağlantılar içeriyor.)

İlginç bir yan unsur olarak, RA yazılımının LISP Works'la programlandığını belirtelim; bu, aslen Harlequin Group Ltd.'nin pazarladığı bir

23 Douglas E. Bernard ve diğ., "Remote Agent Experiment DS1 Technology Validation Report", Jet Propulsion Laboratory, California Institute of Technology and NASA Ames Research Center, Moffett Field. Şu internet adresinde: http://nmp-techval-reports.jpl.nasa.gov/DS1/Remote_Integrated_Report.pdf.

24 "Kusurların" tam bir betimlemesi için bkz. P. Pandurang Nayak ve diğ., "Validating the DS1 Remote Agent Experiment", Artificial Intelligence, Robotics and Automation in Space, *Proceedings of the Fifth International Symposium, ISAIRAS '99,* 1-3 Haziran, 1999, şu yayında: ESTEC, Noordwijk, the Netherlands, M. Perry (ed.), ESA SP-440, Paris: European Space Agency, 1999, s. 349-356. Şu internet adresinde http://citeseerx.ist.psu.edu/viewdoc/download?doi= 10.1.1.30.2688&rep=repl&type=pdf.



Resim 32.6. RAX etkinlikleri çizimi. (Çizim, NASA'dan Mark Shirley'nin izniyle.)

LISP ürünüydü. Bildiğim kadarıyla, RAX, uzayda çalışan LISP programlarının ilk örneğiydi. NASA'da çalışırken RA'yı geliştirenlerden biri olan MIT profesörü Brian C. Williams, MIT'deki "Model-tabanlı Gömülü ve Robotik Sistemler" ekibinde planlama, gerçekleştirme ve model-tabanlı uslamlamayla ilintili çalışmalarını sürdürüyor.[25] Williams, bana şunu yazmıştı: "uzaktan denetimli etmen mimarisi, yörünge cihazları, hava taşıtları, tekneler, denizaltılar ve kara taşıtları dahil çok çeşitli robotik sistemlerce yaygınlıkla taklit edilip devreye sokulmuştur."[26]

32.2.2 Sürücüsüz Otomobiller

Çok farklı koşullarda otomobil sürmek, belki de uzay taşıtı denetlemekten zordur. Uzay taşıtındaki pek çok kontrolle başa çıkma hızı, trafikte beceriyle ilerlemenin gerektirdiği hızlı planlama ve tepki vermenin yanında yavaş kalır. Dolayısıyla, araba sürmenin gerektirdiği algı ve eylem hesaplamalarının, Remote Agent hesaplamalarından oldukça farklı olmasını bekleriz.

İnsanlar, güneşli ve fırtınalı günlerde, geceleri, şehir sokaklarında, otobanlarda, çöl yollarında ya da arazide az çok iyi araba sürer. Evet, kazalar da oluyor. Amerika Birleşik Devletlerinde 2007'de, yolcu-araba

25 mers.csail.mit.edu.

26 E-posta, 25 Ocak 2009.

kazalarında 28.933 kişi öldü ve 2.221.000 insan yaralandı.[27] Fakat bu rakam, kat edilen 100 milyon taşıt milinde sadece bir kişiden biraz fazla insanın öldüğü anlamına geliyor.

İnsanlar kadar iyi taşıt süren YZ sistemleri yaratmaya çalışmanın uzun bir tarihçesi var. DARPA, Stratejik Bilişim Programı kapsamındaki ALV projesiyle birlikte, 1980'lerin ortasında bu problem üzerinde çalışmaya başlamıştı (bkz. 23. Bölüm). Carnegie Mellon Üniversitesi'nin, sürücüsüz taşıtlar üzerine süregiden programları vardı; bunların en önde gelenleri ALVINN, RALPH ve Navlab sistemleridir (bkz. 29.4.3). İlk çalışmaların en etkileyicisi, büyük olasılıkla Ernst Dickmanns ile Münih Bundeswehr Üniversitesi'ndeki ekibinin çalışmasıydı. Kısmen Avrupa'nın EUREKA Prometheus sürücüsüz-otomobil projesi tarafından desteklenen, Dickmanns'ın VaMP taşıtı (Mercedes-Benz 500 SEL), 1995'te, bilgasayarla görme ve radar kullanarak Münih'ten yola çıkıp Danimarka'nın Odense şehrine gidip gelmişti (yolun %95'ini kendi başına kat etmişti).[28] Başka sürücüsüz-taşıt projelerine Japonya'da (Tsukuba Makine Mühendisliği Laboratuvarı), Hollanda'da (2getthere) ve İtalya'da (ARGO projesi) başlanmıştır.

DARPA, 30 Temmuz 2002'de "Büyük İddia"sını duyurarak sürücüsüz araba problemiyle boğuşmaya geri dönmüş oldu. İddia (sadece ABD kökenli ekiplerin katılabildiği bir yarışma), 13 Mart 2004'te California'nın Barstow kentinden yola çıkıp Nevada'nın Primm kentine, çöl yolları ve arazide kendi başına ilerleyecek bir taşıt yapmaktı. 142 millik yolu, on saatlik sınırın altında en hızlı tamamlayan ekip, 1.000.000 dolarlık para ödülünü kazanacaktı. DARPA'nın bu yarışma anısına hazırlanan internet sitesinde belirtildiği gibi: "Yarışmacı taşıtlar, insansız kara araçları olmalıdır ve uzaktan sürülmeleri yasaktır. Gidilecek yolu sınırlar belirler ve bu sınırların dışına taşanlar yarışmadan atılacaktır. Her taşıtın arkasında, çarpışma ya da başka tehlikeli durumların önüne geçmek için bir acil durum durdurma sistemiyle donanmış olan insanlı bir denetim aracı seyredecektir."[29]

Bu yarışmayı gündeme getiren şey, FY2001 Savunma Bakanlığı Yetki Ek Yasası talimatnamesiydi (H.R. 4205/ P.L. 106-398, 30 Ekim 2001); bu talimatnameye göre, "uzaktan denetlenen insansız teknolojiye ulaşmak Silahlı Kuvvetlerin hedefi olmalı, öyle ki (1) 2010 yılına

27 Bu veriler, "Trafik Güvenliği Olguları"ndan, şu internet adresinde: http://www-nrdnhtsa.dot.gov/Pubs/811017.PDF.

28 Bkz. http://ijcai.Org/Past%20Proceedings/IJCAI-97-VOL2/PDF/117.pdf.

29 http://www.darpa.mil/grandchallenge04/program.pdf.

gelindiğinde operasyonel vurucu hava taşıtı filosunun üçte biri bu tür hava taşıtlarından oluşmalı ve (2) 2015'e gelindiğinde, operasyonel kara çatışma taşıtlarının üçte biri insansız olmalı."

Yarışmaya katılmak için aslen 106 ekip başvurmuş olsa da, California, Bastow'daki başlangıç çizgisinden ayrılan sadece on beş insansız taşıt olmuştur. İrili ufaklı şirketlerin, üniversitelerin ve bireysel mucitlerin özel olarak topladığı ekiplerin takımları yarışıyordu.[30] Her takıma verilen CD'de yaklaşık 200 "ara nokta" koordinatı belirtilmişti, böylece takımlar, GPÇ (Genel Problem Çözücü) sistemlerini kullanarak kendilerine rota çizmişti. Engellerin yanından geçmek, belirlenen yoldan çıkmamak ve şarampollerden uzak durmak, araçtaki sensörlerin ve planlama mekanizmalarının işiydi.

Ancak taşıtlardan hiçbiri yolu tamamlayamadı.[31] Sekizi, henüz bir mil gitmeden yarışı bıraktı ve öbürleri başlangıçtan kısa süre sonra bozuldu. (Yarışmanın kısa filmi için şu internet adresine bakınız: http://www.youtube.com/watch?v=wTDG5gjwPGo.) En uzağa giden taşıt, yedi buçuk mil kat etmişti. CNN'de de duyurulduğu gibi, DARPA'nın son verilerine göre: "Taşıt 22 Kırmızı Takım (Carnegie Mellon): Yolun 7,4'üncü milinde, dağlık kısmın dönemeçlerinde, taşıt ["Sandstorm" adı verilmişti] yoldan çıktı, bir bankete takıldı ve ön tekerlerin kauçuğu alevlendi, ancak yangın hemen söndürülmüştür. Taşıt, komut alamaz duruma gelmişti."[32] Büyük İddia projesinin müdür yardımcısı Tom Strat, yarışmanın sonuçları hakkında yorum yaparken şunu demiştir:[33]

> Taşıtlardan bazıları, GPS ara noktalarını çok doğru izleyebilmişti; fakat önlerine çıkan engelleri fark edememiş, deneme turlarında kaza geçirmişlerdir... Engelleri fark etmede çok başarılı olan taşıtlar da vardı, ama ara noktaları izlemede sorun yaşıyorlardı ya da kendi gölgelerinden korkuyor, aslında var olmasa da önlerinde birtakım engeller varmış gibi davranıyorlardı.

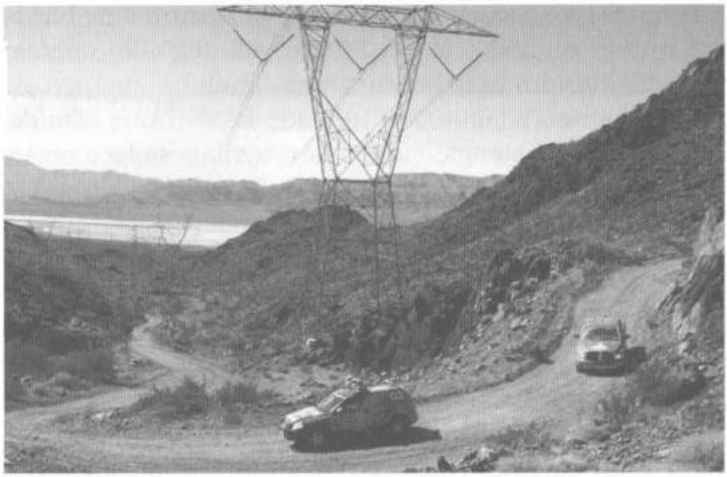
Her ne kadar 2004 Büyük İddia'da herhangi bir "sıradışı başarı" kazanılmış olmasa da, belki de arkadan gelenlerin olmazsa olmaz bir öncülüydü bu. 2004'teki yarışmanın üç ay sonrasında, DARPA, 8 Ekim 2005'te, bu sefer 2 milyon dolar ödülle, "ikinci bir Otomatik Robotik Kara Ta-

30 Betimlemeler için bkz. http://www.darpa.mil/grandchallenge04/program.pdf.

31 Yolculuklarını görmek için bkz. http://www.msnbc.msn.com/id/4517001/.

32 Bkz. http://www.cnn.com/2004/TECH/ptech/03/14/darpa.race/index.html.

33 Bu alıntı, Marsha Walton'ın CNN için yaptığı haberden: http://wwwcnn.com/2004/TECH/ptech/03/14/darpa.race/index.html.



Resim 32.7. Beer Bottle Geçinde ilerleyen Stanley'i DARPA takip aracı izliyor. (Fotoğraf, DARPA'nın izniyle.)

şıtları Büyük İddiası" düzenleneceğini duyurdu.[34] 2005'teki yarışmaya yirmi üç finalist katıldı ve biri hariç hepsi, 2004'te Sandstorm'un kat ettiği mesafeyi geçti. Taşıtların beşi, Nevada'da Primm kentinde başlayıp biten 132 millik çöl yolunu tamamladı:

1. Stanley (tadil edilmiş 2004 Volkswagen Touareg R5 TDI; Resim 32.7), Stanford Üniversitesi'nden "Stanford Yarış Takımı"yla girdi. Stanley, altı saat elli beş dakikayla yolu birinci tamamladı.
2. Sandstorm (tadil edilmiş 1986 AM General M998 HMMWV; Resim 32.8), Carnegie Mellon Üniversitesi'nden "Kırmızı Takım"la girdi. Sandstorm (2004 modelinin güncellenmiş haliydi), yedi saat beş dakikayla ikinci geldi.
3. H1ghlander (tadil edilmiş 199 AM General H1 Hummer Sport Kamyonet), Carnegie Mellon Üniversitesi'nden "Kırmızı Takım İki"yle girdi. H1ghlander, yedi saat on dört dakikayla üçüncü geldi. ("Kültürel bir not" olarak belirtmek gerekirse, *Deja Vu* filminin internet sitesi, "Humvee zaman makinesinin aslında, 2005 DARPA Büyük İddiası için Carnegie Mellon Kırmızı Takımının inşa ettiği gerçek robot H1ghlander olduğunu", bildirir.[35])

34 Bkz. http://www.darpa.mil/grandchallenge05/InitialPressRelease.pdf.

35 Bkz. http://www.imdb.com/title/tt0453467/trivia.



Resim 32.8. Sandstorm, Beer Bottle Geçidinde. (Fotoğraf, DARPA'nın izniyle.)

4. Kat-5 (tadil edilmiş bir Fod Escape Hybrid), Louisiana Metairie'den (New Orleans'ın bir banliyösü) Gray Sigorta Şirketinin "Gray Takımı"yla girdi. Kat-5, yedi saat otuz dakikayla dördüncü geldi. (Kat-5'in yapımına iki kasırga sekte vurmuştu, özellikle de Ağustos 2005'in ikinci yarısında New Orleans'ı vurmuş olan Katrina kasırgası. Takım üyelerinden Paul Trepagnier, bana şöyle yazmıştır: "Çoğumuz, Büyük İddianın hemen öncesinde evimizi kaybettik. [Kat-5 ismi], Katrina şerefine koyulmuş değil, ki pek çok insan böyle demiş olsa da yanılıyorlar; aslında, bu isim bir başkaldırı hareketidir. Sen, elinden geleni ardına koymadın, buna rağmen hâlâ yıkılmadık ayaktayız, anlamına geliyor. Bizi yere serebilirsin ama buna dayanır, eskisinden de güçlü olarak ayağa dikiliriz. Aynı zamanda, Cat-5 kablolarına da bir cinas söz konusuydu."[36])
5. TerraMax (tadil edilmiş bir nakliye aracı), Oshkosh Kamyon Şirketinden "TerraMax Takımı"yla girdi. TerraMax, on saatlik zaman sınırını aşarak, toplamda on iki saat elli bir dakikayla (bir gece mola vermek zorunda kalmıştı) beşinci geldi.

2005 rotası, bazı bakımlardan 2004 rotasına kıyasla daha kolaydı. Yollar daha genişti ve viraj sayısı azdı. En asap bozucu kısmı, dik yokuşlar

36 E-postayla iletişim, 1 Aralık 2008.

içeren Beer Bottle [Bira Şişesi] Geçidiydi. O kısımda yanlış bir manevra, taşıtın şarampole yuvarlanmasına yol açabilirdi. Resim 32.7'de, Stanley'i yolun bu kısmını aşarken görüyoruz. H1ghlander, yarışın büyük bölümünde önde gidiyordu ama 100 mili henüz geçmişken sorun yaşadı ve Stanley onu geçip hem yarışın önderliğini hem galibiyeti hem de 2 milyon dolarlık ödülü kaptı. (CMU ve Stanford yarışmacılarının videoları şu internet adreslerinde bulunabilir: http://redteamracing.org/ ve http://robots.stanford.edu/talks/stanley/RaceDay.wmv. Yarış hakkında bir PBS *NOVA* programı şu internet adresinde: http://www.pbs.org/wgbh/nova/darpa/.)

2005 Büyük İddiasına giren yarışmacılar, bilgisayarla görme teknolojisinin yanı sıra menzil belirleme ve optik sensörler kullanıyor, böylece engellerden sakınıp araba sürülebilecek yolu, arabanın giremeyeceği araziden ayırt edebiliyorlardı. Aynı zamanda hızı ve gidiş yönünü denetlemek amacıyla planlar kurabiliyorlardı, gerçi bu planların dayandığı algısal bilgi muğlaktı. Yer kıtlığından ötürü tüm bu taşıtları anlatmama olanak yok ama yarışın galibi Stanley'nin kullandığı bazı teknolojileri betimleyeceğim.[37]

Stanley, Intel'in temin ettiği altı işlemcili hesaplama platformu, birtakım sensörler ve bilgisayara bağlı elektronik kontrol sistemiyle donatılmıştı. Sensörlere, beş adet lazerli menzil-belirleme birimi, bir video kamera, GPS sistemi, jiroskoplar ve ivmeölçerler dahildi.[38] Stanford profesörü Sebastian Thrun (1967- ; Resim 32.9), projenin genel yöneticisiyken, Stanford'un kıdemli araştırma mühendislerinden Michael Montemerlo (1975- ; Resim 32.9) yazılım tasarım ekibinin başındaydı. Aracın kendi başına yön bulmasının öncelikle bir yazılım sorunu olduğuna, yazılımı ve mimarisini tasarlamanın başarıda kilit unsur olacağına daha en başında karar vermişlerdi. Uzun menzilli arazi algılama, gerçek zamanlı çarpışmadan uzak durma, kaygan ve engebeli arazide sabit araç denetimi gibi sorunlarla başa çıkmak için yeni yöntemler geliştirildi ve var olan yöntemlerin kapsamı genişletildi. Özellikle, üç önemli yazılım yeniliği, Stanley'in performansında büyük rol oynamıştır.

37 Daha derine inmek isteyen okurlar, internetteki çeşitli makaleleri inceleyebilir, örneğin şu adresteki makale Sandstorm'u anlatıyor: http://www.darpa.mil/grandchallenge05/TechPapers/RedTeam.pdf; ve şu adresteki makale H1ghlander'ı anlatıyor: www.darpa.mil/grandchallenge05/TechPapers/RedTeamToo.pdf.

38 Stanley'i anlatan bir inceleme makalesi için bkz. Sebastian Thrun ve diğ., "Stanley. The Robot That Won The DARPA Grand Challenge", *Journal of Field Robotics*, c. 23, no: 9, Haziran 2006, s. 661-692. Şu internet adresinde: http://robots.stanford.edu/papers/thrun.stanley05.pdf.



Resim 32.9. Sebastian Thrun (solda) ve Michael Montemerlo (sağda). (Fotoğraf, John Markoff'un izniyle.)

Bu yeniliklerden biri, olasılıksal arazi çözümlemesi (OAT) algoritmasının geliştirilmesi olmuştur. Bu algoritma, tek taramalı lazerden zamanla gelen menzil ölçümlerini bütünleştirmek amacıyla olasılık tekniklerinden faydalanıyordu. Ardından, araç sürülebilir araziyi sürülemez olandan ayırt etmek için etkili istatistik testleri devreye sokuyordu. Büyük İddia esnasında, OAT algoritması, sensörlerin ciddi hatalarıyla başa çıkabilmiş, saatte 35 mile kadar çıkan hızla giderken yol üzerindeki engelleri neredeyse hatasız olarak tanımlayabilmiştir.[39]

Başka bir yenilik de, bilgisayarla görme alanındaydı. Öncelikle, OAT algoritması tarafından, araç sürülmesine uygun olarak tanımlanan yakındaki arazi parçaları, görsel alana yerleştiriliyordu. Böylece bu arazi parçası, araç sürülmesine uygun yüzeylerin nasıl göründüğünün örneği olarak kullanılıyor ve bundan, bilgisayarla görme algoritmasının öğrenme verileri olarak faydalanıyordu. Bu tekniği anlatan bir makaleye göre, "görme algoritması, kameranın gördüğü bütün alanı sınıflandırır [lazer tarayıcının ötesine geçerek] ve araba sürmeye uygun arazinin 70 metreye kadar haritasını çıkarır. Toplu sensörler [lazer artı

39 Daha fazla bilgi için bkz. Sebastian Thrun, Michael Montemerlo ve Andrei Aron, "Probabilistic Terrain Analysis for High-Speed Desert Driving", *Proceedings of the Robotics Science and Systems Conference, II* içinde, G. Sukhatme ve diğ. (ed.), Philadelphia, PA, 2006. Şu internet adresinde: http://robots.stanford.edu/papers/thrun.mapping-Stanley.pdf.

bilgisayarla görü] bütünleştirilip bir sürüş stratejisi oluşturulur, bu da epey gürbüz ve uzun menzilli bir duyumsama sağlar."[40] Stanley'nin görme sisteminin, tüm nesnelerin türlerini ve yerlerini belirlemek için bütün manzarayı çözümleyen türde (örneğin David Marr'ın savunduğu gibi) görme sistemi olmadığını belirtelim. Bunun yerine, özel amaçlı alt-programlardan yararlanıyordu ve bunların tek işi, nerede güvenli araba sürüleceğine Stanley'nin karar vermesini kolaylaştırmaktı.

Son olarak, çevrimiçi hız denetimi için gözetimli bir öğrenme sistemi geliştirilmişti. Hız denetimi, yarışı en hızlı bitiren Stanley'nin başarısında önemli bir unsurdu. Taşıt, tehlikeli arazide yavaşlamalı, güvenli arazide hızlanmalıydı. Hem tehlikeli hem de güvenli arazilerde insanların sürüş tarzına uyan bir gözetimli öğrenme kullanan bu sistem, tehlike ve sürüş hızı dengesini sağlayan sürüş hızlarını seçmeyi öğrenmişti.[41]

Wired dergisi, 2006'nın Ocak ayında, Stanley'yi "Gelmiş Geçmiş En İyi Robot" seçti.[42] (Shakey, iki Mars robotu Spirit ve Opportunity ile iki kurmaca robotun ardından en iyi beşinci robot seçilmişti.) Şu an Stanley, Smithsonian Ulusal Amerikan Tarihi Müzesi'nde tutuluyor.

DARPA, 3 Kasım 2007'de başka bir otomatik sürüş yarışı düzenledi. "Kent Yarışı" denilen bu etkinlik, bir zamanlar George Hava Kuvvetleri Üssü olan mevkide, California eyaletinin Victorville şehri civarında düzenlendi. Yarışmacılar, taklit bir şehir ortamında yer alan altmış millik rotada, altı saat içinde belli "kontrol noktalarına" uğramak zorundaydı. Parkuru başarıyla tamamlamak, trafik içinde yol alıp California sürüş kurallarına tamamen uyarak bir dizi karmaşık manevrayı başarmayı gerektiriyordu, örneğin kavşaktan geçme, sollama, park etme, şerit değiştirme gibi. Toplamda, hem insanlı hem otomatik elliyi aşkın taşıt, son yarışmada cadde sokaklarında aynı anda seyrediyordu. (Yarışmanın videoları ve fotoğrafları şu internet adresinde: http://www.darpa.mil/grandchallenge/gallery.asp.)[43]

40 Hendrik Dahlkamp ve diğ., "Self-supervised Monocular Road Detection in Desert Terrain", *Proceedings of the Robotics Science and Systems Conference, II* içinde, G. Sukhatme ve diğ. (ed.), Philadelphia, PA, 2006. Şu internet adresinde: http://robots.stanford.edu/papers/dahlkamp.adaptvision06.pdf.

41 Daha fazla bilgi için bkz. David Stavens, Gabriel Hoffmann ve Sebastian Thrun, "Online Speed Adaptation Using Supervised Learning for High-Speed, Off-Road Autonomous Driving", *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007, s. 2218-2224. Şu internet adresinde: http://robots.stanford.edu/papers/stavens_hoffmann_thrun_ijcai07.pdf.

42 Bkz. http://www.wired.com/wired/archive/14.01/robots.html?pg=l&topic =robots &topic_set=.

43 Yarışmanın resmî internet sitesi: http://www.darpa.mil/GRANDCHALLENGE/index.asp.

Bir dizi eleme yordamı ve araç testi sonrasında DARPA, en baştaki doksan sekiz başvuruyu önce yirmi iki eyaletten gelen otuz beş takıma, sonra da on bir takıma indirdi. Yarış günü bu takımların altısı parkuru başarıyla tamamlamıştır. DARPA, yarışı bitirenleri ve bitiriş zamanları şu şekilde açıklamıştı:

1. Carnegie Mellon Üniversitesi'nden Tartan Racing'in Boss (2007 Chevy Tahoe) adlı aracı (bkz. Resim 32.10). Boss, 4:10:20 zamanıyla (ve ortalama saatte 14 mil hızla) birinciliği kazanıp 2 milyon dolarlık ödülü aldı.
2. Stanford Üniversitesi'nden Stanford Racing Team'in Junior (2006 Volkswagen Passat Wagon) adlı aracı. Junior, 4:29:28 zamanıyla ikinci olup 1 milyon dolar ödül kazandı.
3. Virginia Tech'den Victor Tango takımının Odin (2005 Ford Hybrid Escape) adlı aracı. Odin, 4:36:38 zamanıyla üçüncü olup 500.000 dolar ödül kazandı.
4. Massachusetts Teknoloji Enstitüsü'nden MIT Takımının Talos (Land Rover LR3) adlı aracı. Talos, yaklaşık altı saatle yarışı dördüncü bitirdi.

(Burada belirtilen zamanların, yarış sırasında taşıtların kendi hataları dışında durdurulduğu zaman hesaba katılarak ayarlandığını belirtelim.) Yarışı iki takım daha bitirmişti; bir tanesi, Pennsylvania Üniversitesi ile Lehigh Üniversitesi'nin ortak takımıydı, öbürü Cornell Üniversitesi'nin takımıydı.

Kent Yarışı, algılama ve planlama teknolojilerinden yeni ve güçlü taleplerde bulunmuş oldu. Örneğin, taşıtların yapması gereken şeylerden bazılarına göz atalım: yolun kurallarına uymak, uzun mesafelerde başka taşıtları saptamak ve takip etmek, uygun yer bulup arabayı oraya park etmek, kavşak geçiş kurallarına uymak, öteki taşıtları güvenli mesafeden izlemek, kapalı yollar ya da bozulmuş arabalar gibi dinamik koşullara tepki vermek. Yarışı bitirenlerin tümü, bu maharetleri sergilemiştir; konuyla ilgilenen okurlar, bu işi nasıl başardıklarının ayrıntıları için yarışmacılarla ilgili makale ve internet sitelerine bakabilir. Boss'un internet sitesinden özet bir betimlemeyi aktaralım:[44]

> Boss, trafik hakkında uslamlama yapmak ve hedefine güvenle ilerlerken uygun eylemleri gerçekleştirmek için algı, planlama ve davranış yazılımları kullanır.

44 http://www.tartanracing.org/tech.html.



Resim 32.10. Tartan Racing Team'in takım kaptanı William (Red) Whittaker ve Boss, birincilik ödülüyle poz veriyorlar. (Fotoğraf, Carnegie Mellon Üniversitesi'nin izniyle.)

> Boss, dünyayı görebilmek amacıyla bir düzineyi aşkın lazer, kamera ve radarlarla donatılmıştır. Yüksek düzey rota planlama, yol ağı içinde en iyi yolu belirler. Devinim planlama, algı sisteminin saptadığı durağan ve dinamik engelleri, ayrıca şerit ve yol sınırlarını, park yeri sınırlarını, dur çizgilerini, hız sınırlarını ve benzer şartları dikkate almayı gerektirir. Boss, kırmızı ışıkta geçen ya da aniden duran veya direksiyon kıran taşıtlar gibi sürprizlerle de başa çıkabilir. Boss'un savunma sürüşü becerileri, kazalardan uzak durmasını mümkün kılar.

Çeşitli DARPA Yarışmalarında taşıtların başarı göstermesi üzerine, otomobil sanayisi, sürücüsüz arabalara daha büyük ilgi gösterir oldu. (En azından 2008-2009 mali sorunlarından önce.) Kasım 2007'de, Volkswagen'ın Amerika şubesi, Stanford Üniversitesi'ne, Volkswagen Otomotiv İcat Laboratuvarı'nın (VAIL) yer alacağı binayı inşa etmesi için 2 milyon dolar ve otomotiv teknolojisine yönelik beş yıllık araştırma için yılda 750.000 dolar vereceğini duyurdu. Volkswagen Amerika Elektronik Araştırmalar Laboratuvarı'nın yetkili müdürü Dr. Burkhard Huhnke, şunu demişti: "VAIL, Volkswagen araştırmacıları ile Stanfordlu bilimcilerin, otomotiv teknolojisini keşfetmenin yeni yollarını bulacağı sağlam bir zemin olacak. VAIL'de gerçekleştirilecek çalışmalar, taşıtların ve DARPA Büyük İddiasında Stanley ve Junior adlı taşıtlarla ortaklığımızı başlattığımız otomatik sürüşün geleceğini daha da geliştirecektir."[45]

45 Bkz. http://www.vw.com/vwbuzz/browse/en/us/detail/Volkswagen_to_contribute

Haziran 2008'de General Motors, otomatik sürüş teknolojileri araştırmaları kapsamında Carnegie Mellon Üniversitesi'ni beş yıl boyunca (muhtemelen daha da uzun sürecekti) yılda bir milyon dolarla destekleyeceği programı duyurdu. GM'nin Ar-Ge yetkili müdürü Alan Taub, basın açıklamasında şunu demişti: "General Motors'da bizler, otomatik sürüşün önümüzdeki on yıl içinde piyasaya girmeye başlayacak ve 2020'ye gelindiğinde kesinlikle piyasada olacak kadar olgunlaştığına inanıyoruz."[46] GM'nin kısa süre önce iflas edişinin bu vaadi ne kadar etkilediğini merak ediyorsunuzdur. CMU Şirket İlişkileri Müdürü William H. Swisher bana, aslında bir milyon doların verildiğini ama "beş yıllık anlaşmayı tamamlayıp tamamlamayacaklarının belli olmadığını" söyledi.[47]

Sebastian Thrun bana, kendi araştırma taşıtlarının "yaklaşık Haziran 2010'a gelindiğinde... San Francisco'dan Los Angeles'a [özerk olarak] muntazaman gidip gelmesini" beklediğini söylemiştir.[48] Aynı zamanda, "2030'a gelindiğinde otoyol millerimizin yarısının, insan müdahalesi olmadan otomatik olarak kat edileceğini" tahmin ediyor.[49] Ancak, sürücüsüz arabaların kısa süre içinde galerilerde boy göstereceğinden kuşku duyuyor. Çeşitli toplumsal ve yasal sorunlara işaret ediyor, örneğin taşıt yasalarının güncellenmesi, kaza mesuliyetinin belirlenmesi ve insanların dizginleri elde tutma isteği gibi. Bunun yerine, önümüzdeki birkaç sene içinde sürücüsüz otomobil araştırmalarından doğacak başlıca faydanın, insan şoförlere sağlanacak otomatik yardımlar olacağını düşünüyor. Bunlar sayesinde, otoyollar daha verimli kullanılacak, trafikteki yaralanmalar ve ölümler azaltılacaktır. Bu yardımlara dair örnekler arasında, her türlü çarpışmaya karşı uyarı sistemlerini, radar tabanlı seyir denetimini, şerit değiştirme uyarı cihazlarını, elektronik kararlılık denetimini, uydu küresel konumlama sistemlerini ve sayısal haritaları sayabiliriz.

YZ teknolojisi yakın gelecekte araba sürmemize yardımcı olacağı gibi başka ufak tefek YZ teknolojileri de gündelik hayatımıza çoktan girmiştir; bir sonraki konum bunlar olacak.

_5_75_million_to_Stanford_University/180.

46 Associated Press'in haberi. Bkz. http://www.mercurynews.com/nationworld/ci.9664682.

47 E-posta, 21 Haziran 2009.

48 E-posta, 17 Ekim 2008.

49 E-posta, 14 Ocak 2009.

33
YAPAY ZEKÂ HER YERDE

Günümüz dünyasında, YZ'nin büyüsü her yere sızmış durumda; belki "tam teşekküllü bir YZ" söz konusu değil ama önemli kısımları vardır. Allen Newell'e göre bu YZ kırıntıları, "büyülü ülkenin" bir parçasıdır. 1976'da yaptığı bir konuşmada, bilgisayarlar için "büyü teknolojisi" demişti. Bunları büyülü kılan iki bileşen sayar:[1]

> Birincisi, bilgisayar belli hedeflere ulaşmak için bilgiyi eyleme dökmenin teknolojisidir... Algoritmalar ve programlar bu işe yarar; eylemi dondurup gereksinim halinde çözerler.
>
> İkinci bileşen, zeki eylem yeteneği taşıyan fiziksel sistemlerin küçültülmesidir.
>
> Dolayısıyla, bilgisayar teknolojisi, büyülü bir dünya yaratma yeteneğiyle tüm öteki teknolojilerden ayrılıyor:
>
> Çünkü, ıslak yolda nasıl duracağını bilen frenler sağlıyor.
>
> Kullanıcılarıyla sohbet eden aygıtlar sağlıyor.
>
> Üzerinden geçenlerin güvenliğini gözeten köprüleri mümkün kılıyor.
>
> Altlarında duranları umursayan, yolu bilen, böylece kimsenin kaybolmasına izin vermeyen sokak lambaları yapıyor.
>
> Sizin için gelir vergisi hesabını çıkaran küçük kutular temin ediyor.
>
> Kısacası, bilgisayar teknolojisi, dünyamızın tüm girdisine çıktısına zeki davranış yerleştirme olasılığı sunuyor. Bu teknoloji sayesinde, büyülü bir ülke inşa edebiliriz.

O halde, YZ'nin "dünyamızın tüm girdisini çıktısını" nasıl mesken tuttuğuna dair birkaç örneğe bakalım. İşe, evlerimizle ve evlerin içindeki, dışındaki bazı nesnelerle başlayacağım.

1 Konuşmasının yayımlanmış bir versiyonu için bkz. Allen Newell, "Fairy Tales", *AI Magazine*, c. 13 no: 4, 1992. Şu internet adresinde: http://www.aaai.org/ojs/mdex.php/aimagazine/article/download/1020/938.

33.1 HANELERİMİZDE YZ

Evler ve içindekiler, gitgide daha zekileşiyor. Günümüzde (ya da yakın gelecekte) modern bir evi gezerken bulabileceklerinizin kısmi bir listesi şöyle:

- Sıcaklık değişikliklerini ve ev sakinlerinin gereksinimi öngören, öteki ev cihazlarıyla iletişim kuran, uygun eylemleri önceden gerçekleştiren ısı ve klima sistemlerinin termostatları;
- Yiyecek paketlerinin barkodunu okuyarak ne kadar süre pişirmesi gerektiğini belirleyen mikrodalga fırınlar;
- Koşucunun boyunu ve adım uzunluğunu hissedip minyatür bir tazyik ve kablo sistemi aracılığıyla, taban desteğiyle değişikliklere göre sporcu sağlığını koruyan bilgisayar çipli akıllı koşu ayakkabıları;
- Giysileri daha iyi yıkamak amacıyla farklı koşullara uyum sağlayan çamaşır makineleri;
- İçindekilerin dökümünü otomatik olarak çıkarıp ihtiyaçları sahibine bildiren buzdolapları;
- Yüzleri tanıyan ve odaklanma, öne çıkarma ve çerçeve içine almayı denetleyen bilgisayarla görme sistemi barındıran kameralar;
- Ortamdaki ses düzeylerine uyum gösteren ve "uğultuyu" engelleyen işitme cihazları;
- İnsanlarla etkileşime giren robot ev "hayvanları" ve oyuncaklar;
- Yerleri temizleyen ve süpüren robotlar;
- Yaşlılarla veya hastalarla ilgilenen bakıcı robotlar.

Bu liste büyümeye devam ediyor. Bazı YZ araştırmacıları, "çevresel zekâ" denilen bir alandan bahseder; bu alanın "ana fikri, bir mekânı (örneğin oda, ev, bina, otobüs durağı, hastanede önemli bir yer gibi), zeki yazılımla eşlenmiş sensörlerle donatmaktır, böylece o mekânı kullanan insanlar, tepki veren, hatta akıllı bir ortamın yararını görür."[2] Çevresel zekânın nüfuz ettiği bir ortamın bileşenleri aynı zamanda ağ üzerinden birbirlerine bağlanır ki birbirleriyle iletişim kurabilsinler ya

2 Şu makaleye bakınız: Juan Carlos Augusto ve Daniel Shapiro, "The First Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI '06)", *AI Magazine*, c. 28, no: 1, Bahar 2007, s. 86-87. Ayrıca bkz. http://www.infj.ulst.ac.uk/~jcaug/aitami07.htm.

da insanlar bu bileşenlerle gündelik konuşma aracılığıyla iletişim kurabilsin. Vlingo, Nuance ve Yap, sesi metne dönüştüren cep telefonu (örneğin iPhone) ürünleri satan üç şirkettir. Örneğin Vlingo, kendi internet sitesinde şunu iddia ediyor: Teleonunuza "Pittsburgh'daki pizzacılar" dediğinizde, "ne istediğinizi anlar, bulur ve nasıl ulaşacağınızı size gösterir. Telefona dokunmaya, parmağınızla yönlendirmeye falan gerek yok, bildiğimiz konuşma yeterli."[3] Büyük ihtimalle benzer bir teknoloji sayesinde akıllı uygulamalara, sırf konuşarak komut vermek ve sorgulamak mümkün olacaktır.

33.2 İLERİ SÜRÜCÜ YARDIM SİSTEMLERİ

Bilgisayarların ve yapay zekânın belki de en çok noktasına sızdığı alan, günümüzün yolcu otomobilidir. Günümüz arabalarında elli kadar mikroişlemcili denetim olabiliyor, örneğin birkaçını saymak gerekirse, otomatik vites, benzin enjeksiyon sistemi, kilitlemesiz frenler, hava yastıkları, güvenlik sistemleri, seyir denetim sistemleri gibi. Her ne kadar arabalar henüz tamamen otomatik olmasa da, gitgide daha fazla otomobil, "ileri şoför yardımı sistemleri" (ADAS) denilen güvenlik özellikleriyle donatılıyor.

Günümüzde var olan ya da bazı otomobil üreticilerinin tasarladığı birkaç ADAS özelliği şunlardır:

- Uyumsal seyir denetimi (ACC), hızın zeki denetimini getiriyor, radarla ya da lazer sensörlerle algılanan trafik koşullarına göre taşıtı yavaşlatıyor ya da hızlandırıyor;
- Zeki hız uyarlanımı (ISA), yerel hız sınırlarını gözler, hız sınırı olan bölgeye girildiğinde taşıtı yavaşlatır (ya da sürücüyü uyarır);
- Şerit denetimi sistemleri, yan şeritlerde taşıt ya da engel var mı yok mu diye bakar, sürücünün yan şeride ya da yol dışına taşıp taşmadığını yoklar;
- Otomatik park sistemleri, paralel park manevrası yapan sürücüye yardımcı olur;
- Trafik tabelası tanıma sistemleri;
- Uyuklayan sürücüyü tespit eden sistemler;
- Teker basıncını denetleyen zeki sistemler.

3 Bkz. http://www.vlingo.com/.

Tüm araba bilgisayarları YZ teknolojisi barındırmasa da, en iddialı ADAS özellikleri, bilgisayarla görme, planlama yöntemleri, olasılıksal çıkarın ve yapay öğrenmeden yararlanır. ADAS kullanmanın itici gücü, otomobil kazası yaralanmalarını ve ölümlerini ortadan kaldırma arzusudur. Örneğin 1997'de İsveç Parlementosu, "Yol Trafiği Güvenlik Tasarısını" meclisten geçirdi; "Sıfır Vizyonu" adlı bu tasarının hedefi, "yol taşıma sistemi içinde seyreden hiç kimsenin ölmemesi ya da ciddi yaralanmamasıydı."[4]

33.3 HARİTALARDA ROTA BULMAK

Hazır otomobil konusunu ele alıyorken, sürücülerin sık sık gereksinim duyduğu şeylerden birinin, bir yerden bir yere nasıl gidileceğini öğrenmek olduğunu belirtelim. Pek çok otomobil, sizinle "konuşan" cihazlar barındırır; bu cihazlar, araç GPS sistemlerini, harita veritabanlarını ve konuşma sentezini kullanır. Harita veritabanları, konum "düğüm noktalarından" ve aktarma yolu "bağlantılarından" oluşan grafikler sayılabilir. Dolayısıyla, seyir tavsiyesini ister yola çıkmadan önce evinizdeki bilgisayar (örneğin Google Maps aracılığıyla) ister araç seyir sistemi versin, bu öneri, örneğin *A* düğüm noktasından *B* düğüm noktasına giden yolu bulma amacıyla grafiği arama işlemiyle oluşturulur.

En sık kullanılan grafik arama yönteminin A* olduğunu hatırlarsınız; bu sezgisel arama yöntemi, hem o ana dek kat edilmiş mesafeyi hem de hedefe kalan mesafeye dair tahminleri hesaba katar (bkz. s. 168). Peki A*, rota bulma programlarında kullanılıyor mu? Pekâlâ, örneğin Google, "rota ağlarında en kısa yolu milisaniyeler içinde hesaplayan son derece gelişmiş hiyerarşik grafik algoritmaları kullandığını" söyleyecektir.[5] Büyük ihtimalle bu algoritmalar ve benzeri başka rota bulucular, A*'nın kullandıklarına benzeyen fakat iki boyutlu harita aramada uzmanlaşmış sezgisel teknikler kullanıyordur. Örneğin, aramalar hiyerarşik olarak örgütlenmiştir. Yani, uzak bir hedefe yönelik yolculuklar için, sadece ana yolları ve otobanları gösteren büyük ölçekli haritalar kullanılır. Ardından, başlangıç noktasından yola çıkıp hedefe giden ana yola ulaşmak için, daha az yolculuk edilen yolların ayrıntılı haritası kullanılır. Hiyerarşik arama, optimal yolun biraz altında bir sonuç verebilir, fakat yine de gayet kabul edilebilir yollar gösterir. Buna ek olarak, milyarlarca sorguya yanıt verilirken, her

4 İsveç Ulaştırma ve İletişim Bakanlığı, 1997.

5 Google A.Ş. Araştırma Müdürü Peter Norvig'in e-postası, 9 Eylül 2008.

sorgu için tekrarlanması gereken hesaplamalar, tekrarlanmak yerine sorgular arasında paylaşılabilir.

Çoğu rota bulma programı, örneğin tahmini yolculuk süresi gibi uzaklık harici ölçütleri de hesaba katabilir (ve katıyor). Örneğin, MicroSoft'un ClearFlow adlı programı, trafik yoğunluğunu tahmin etmek amacıyla trafiği gözleyen sensörlerin bilgi akışı sağladığı Bayes ağları kullanır ve rota tavsiyelerini ya en kısa süreye ya en kısa uzaklığa ya da o anki trafik koşullarına dayandırabilir. (Şu internet adresinde kendiniz deneyebilirsiniz: http://maps.live.com/.)

33.4 BELKİ BUNU DA BEĞENİRSİNİZ

Amazon.com'un internet sitesine girdiğimde, "Merhaba Nils J. Nilsson, senin için önerilerimiz var", diyor. Sonra, beğenebileceğimi tahmin ettiği bazı kalemleri sıralıyor, örneğin Alexander McCall Smith'in *Portekizcede Düzensiz Fiiller* kitabı (bunu tavsiye etmesinin nedeni, daha önce aynı yazarın başka bir kitabını satın almış olmam) ve daha önce *BBC Shakespeare: Othello* videosunu indirdiğim için Brooke Shields'ın oynadığı *Lipstick Jungle* videosu (?!) gibi. Sonra, bazı değişiklikler yapmama izin veriyor. *Lipstick Jungle*'la ilgilenmediğimi söylediğimde, *How I Met Your Mother* videosuyla ilgilenir miyim diye soruyor. Belki de yeterli sayıda video ısmarlamadığım için neyle ilgilendiğim konusunda görüş oluşturmuş değil.

Amazon'un önerileri, toplumsal ya da işbirlikçi diye nitelenen süzgeçleri temel alır. Her kullanıcı için bir seçimler veritabanı (kitap, film, vesaire) tutulur. Eğer B kullanıcısının seçimleri A kullanıcısının seçimleriyle yeterince bağıntı sergiliyorsa, işbirlikçi bir süzme sistemi, *B*'nin satın aldığı kalemlerden bazılarını (*A*'nın henüz satın almadıklarını) *A* kullanıcısına önerecektir. Kullanıcılar arasındaki seçim bağıntılarını bulmak için, görece kolay olan yapay öğrenme teknikleri kullanılır. iTunes, TiVo, Netflix gibi başka internet siteleri de önerilerini işbirlikçi filtrelemeye dayandırır.

Başka bir tavsiye sistemiyse, içerik tabanlı denen filtrelemeyi kullanır; bu sistemde kullanıcının kitap, film, belge, vesaire tercihleri, aynı türde başka *öğelerle* benzerlikler bulmak amacıyla çözümlenir (aynı tercihlere sahip başka kullanıcılarla benzerlik kurma yerine). Böylece en benzeşen kalemler önerilir. Örneğin belgeler için kıyaslamalar, 27.3'te bahsettiğim vektör gösterimleri kullanılarak yapılır. İçerik tabanlı süzme, istenmeyen e-postaları (toplu e-postalar) ve İnternet sitelerini (por-

nografi gibi) engellerken yaygınlıkla kullanılır. Ayrıca, kişiselleştirilmiş internet aramaları için de kullanılır; örneğin, belirli konular hakkında haber toplamak için gelen haberlerin niteliğinin belirlenmesinde. İşbirlikçi ve içerik tabanlı yöntemleri birleştiren tavsiye sistemleri de geliştirilmiştir. İlgilenen okurlar, *AI Communications* adlı derginin "Tavsiye Sistemleri" konulu özel sayısına göz atmak isteyebilir.[6]

33.5 BİLGİSAYAR OYUNLARI

Satranç ve dama gibi oyunlar oynamaya yarayan YZ tekniklerinin kullanımına ek olarak, YZ, insan kullanıcıların benzetimi yapılan bir dünyada yapay karakterlerle etkileşime girdiği türde bilgisayar oyunlarında kullanılır oldu. Her ne kadar bu oyunlarda zengin ve gerçekçi grafiklere ağırlık verilse de, YZ tekniklerinin kullanımı, bu oyunları daha da cazip ve zorlu kılabilir. Bu oyunların geliştiricileri ve meraklıları, bilgisayar oyunlarındaki YZ'yi akademik YZ ya da "Ar-Ge YZ"si dedikleri şeyden ayırt etmek amacıyla "Oyun YZ'si" terimini kullanır. Pek çok oyunda gereken şeyin sadece, zekâ yanılsaması ya da görüntüsü olduğuna işaret ederler; tıpkı ELIZA programının sohbet ediyormuş gibi görünmesi ama aslında neden bahsettiğini bilmeyişi gibi.

Bununla birlikte, bilgisayar oyunları, YZ kullanımı olasılıkları bakımından zengindir. Normal düzende, insan oyuncu, oyundaki yapay etmenlerle temas edip rekabete girer; bu etmenlere, oyuncu olmayan karakter denir (NPC). NPC'ler, başka işlerinin yanı sıra, kendi dünyalarında A noktasından B noktasına, engellere ya da birbirlerine toslamaksızın gidebilmelidir; dolayısıyla pek çok oyun A*'dan faydalanır. Fakat aynı zamanda NPC'ler kendi temsili ortamlarını algılayabilmeli, plan yapıp icra edebilmeli ve öğrenebilmeli; tıpkı gerçek ortamlarında robotların yaptığı gibi. Bu işleri ne kadar zekice becerebilirlerse, oyunun insan oyuncusuna (ve müşterisine) o kadar gerçekçi görünürler. Bu yetenekleri sergileyen oyunlara iki örnekten bahsedeceğim.[7]

Lionhead Studios'un geliştirdiği ve Electronic Arts'ın pazarladığı *Black and White 2* adlı oyun, sinir ağları ve karar ağaçlarının bir birleşiminden faydalanır. Oyunla ilgili internet sitelerinden birine göre, NPC'ler (kötücül ve hayırsever ilahlar) "strateji öğrenebiliyor, yeni ye-

6 *AI Communications*, Tavsiye Sistemleri Özel Sayısı, Dietmar Jannach, Markus Zanker ve Joseph Konstan (misafir editörler), c. 21, no: 2-3, 2008.

7 Bana bu oyunlardan bahsettiği için John Laird'e teşekkür ediyorum (e-posta, 10 Eylül 2008).

teneklerde ve becerilerde ustalaşıyor, çarpışmalarda ordulara önderlik ediyor... Oyunda yaptığınız her seçimin bir etkisi oluyor. Her eylem ve eylemsizlik, binalar, hayvan ve bitki örtüsünde bariz değişiklikler tetikliyor, bunların hepsi kişiliğinizi yansıtacak biçime kavuşuyor."[8]

Jeff Orkin'in ve Monolith Productions'ın oyunu *F.E.A.R.* (*First Encounter Assault Recon*), A*'nın yol bulmadaki olağan rolüne ek olarak, NPC eylemlerinin planlanması için A* kullanıyor.[9]

Black and White ve *F.E.A.R.*, oyunlarda YZ kullanımıyla ilgili bir internet sitesi olan http://AIGameDev.com tarafından "En Etkili 10 YZ Oyunu" listesinde sırasıyla 1 ve 2 numara olarak gösteriliyor. Oyunlarda YZ kullanımına ayrılmış birtakım internet siteleri ve konferanslar vardır.[10]

Kimi YZ araştırmacıları, bilgisayar oyunlarından, zeki etmenler için yeni fikirler geliştirmenin kullanışlı bir arenası olarak faydalanmayı savunuyor. Örneğin, Michigan Üniversitesi profesörlerinden John Laird şöyle yazmıştır: "[çünkü] robotik alanındaki araştırmalar, alt düzey duyumsama ve gerçek dünyada eylemekle ilgili, zorlu pek çok problemi çözmeyi gerektiriyor ve bunlar, zekânın bilişsel yönlerinden çok uzak... dolayısıyla bilgisayar oyunları, araştırma için bize kendi sanal ortamlarımızı yaratabileceğimiz ucuz, güvenilir ve esnek bir teknoloji temin ediyor."[11]

Burada bahsetmiş olduğumuz YZ parçaları, ister evde, otomobilde, ister bilgisayar oyunlarında olsun, çoğunlukla "tepkisel" ve "davranış tabanlı" cinstendir. Koşullar duyumsanır, ve duyumsanan şeye göre bir eylem hayata geçirilir. Eldeki duruma uygun tepkiler verebilen sistemler kurmak, YZ araştırmalarının önemli bir dalı olagelmiştir. İster temsili ister gerçek olsun, dinamik ortamlara yerleşmiş YZ etmenleri, ne zaman tepki vereceğine ve ne zaman düşünüp taşınacağına da karar verebilmeli. Albert Lewis, Oakland Raiders Amerikan Futbolu takımının köşe-savunma oyuncusuyken, ne zaman tepki verilip ne zaman düşünüldüğü hakkında şunu söylemişti:[12]

8 http://www.lionhead.com/bw2/Default.aspx.

9 Bkz. Jeff Orkin, "Three States and a Plan: The A.I. of F.E.A.R."; şu internet adresinde: http://web.media.mit.edu/~jorkin/gdc2006_orkin_jeff_fear.doc.

10 Örneğin bkz. http://www.gameai.com/, http://www.igda.org/ai/, http://www.aiwisdom.com/bygame.html ve http://aigamedev.com/.

11 John E. Laird, "Research in Human-Level AI Using Computer Games", *Communications of the ACM*, c. 45, no: 1, 2002, s. 32-35.

12 Aktaran: Sam Farmer, *San Jose Mercury News*, s. 1D, 30 Ağustos 1996.

> Sahada düşünürseniz, o oyunu otomatik olarak kaybedersiniz. Düşünmeniz gereken zaman, hafta içinde antrenman yaptığınız zamanlardır. Kafanızda o zaman ışık yanmalı. Maça girdiğinizde, her şey gördüklerinize tepki vermekten ibarettir.

Nitekim, *Madden NFL 09* gibi oyunlarda temsili futbolcuların (NPC'ler), yeri geldiğinde, üst düzey uslamlamanın yardımcı olduğu karmaşık tepki stratejileri kullanacağını düşünebiliriz.

34
AKILLI ALETLER

Manşetlere çıkan kilometre taşı nitelikli YZ başarıları ile her yerde rastladığımız daha ufak hesaplamalı zekâ kırıntıları arasında bir yerde, gündelik işlerinde kendilerine yardımcı olsun (bazen de bu işleri otomatikleştirsin) diye doktorlar, bilimciler, mühendisler ve işadamlarının yararlandığı etkileyici YZ programları var. Bunlara, YZ'nin "akıllı aletleri" diyorum. Bunlar bazen tek başına çalışıyor, fakat çoğunlukla daha büyük bir bilgisayar çerçevesi ya da donanım cihazlarıyla bütünleşmiş haldeler. Bazıları, ancak belirli bir problemi çözmek üzere başvurulduğunda işe yarıyor, örneğin hastalığa tanı koymak gibi. Bazılarıysa daima etkindir, örneğin çevrimiçi hisse senedi alım-satım sistemleri. Tek tek hepsinden bahsedemeyeceğim, çünkü çok fazlalar; bazılarını da sadece bunların şirketlerdeki ve devletteki kullanıcıları biliyor. Fakat birkaç örnek, bunların faydalarını ve çeşitliliğini gözler önüne serecektir.

34.1 TIPTA

YZ'nin, tıbbi klinik uygulamalarında nasıl kullanıldığıyla başlayalım. Daha 1980'lerden itibaren, YZ teknolojisi, tıbbi sistemlerin ve cihazların önemli bir parçası olmuştur. 2000 yılının Mart ayında, *Medical Device & Diagnostic Industry* adlı aylık dergide çıkan bir makalede şu görüş ileri sürülmüştü: "Tıbbi cihaz sanayisi, bilgisayar tabanlı zeki karar destek sistemlerinin (DSS) ve uzman sistemlerin ortaya çıkışına şahit oluyor; bunların güncel başarısı, yapay zekâ teknolojisinin olgunlaşmasını yansıtıyor."[1] Bu makale, YZ-sızmış birkaç cihazdan da bahsetmişti, örneğin "Agilent Akut Kardiyak Kansızlığı Zaman-Duyarsız Öngörü Cihazı... zeki bir elektrokardiyografi (ECG) cihazıdır ve kalp krizinin yaygın bir biçimi olan akut kardiyak kansızlık olasılığını

1 Ralph J. Begley ve diğ., "Adding Artificial Intelligence to Medical Devices", *Medical Device & Diagnostic Industry*, Mart 2000, s. 150vd. Şu internet adresinde: http://www.devicelink.com/mddi/archive/00/03/014.html.

tahmin eder", ayrıca General Electric "MAC 5000 Durağan Durum Test Sistemi, Marquette 12SL ECG çözümleme programıyla birlikte; bu bütünleşik DSS, ECG dalgabiçimlerini yorumlayıp sınıflandırmak amacıyla yeni geliştirilmiş sayısal işleme yöntemleri ve tanı programı algoritmaları kullanıyor."

Clinical Window'da (GE Healthcare'in hamilik ettiği dergi) çıkmış, Cornell Üniversitesi Kardiyoloji Kürsüsünden Dr. Paul Kligfield'ın ECG'ler hakkında kaleme aldığı, Eylül 2005 tarihli çevrimiçi incelemede,[2] "günümüzde tüm büyük üreticilerin sayısal elektrokardiyografileri, doktorlara yardımcı olan otomatik tanılar koyabiliyor", denmiştir. Bununla birlikte, bu tanıların "doktoru yanıltabileceği" örneklerden de bahseder. Belirli bir cihazı değerlendirmeye alan Dr. Kligfield, kalp pili olmayan 3.954 hastada, kalp ritmi yorumlarının %7,8'inin, iki uzman kardiyolog tarafından gözden geçirilip düzeltimesi gerektiğini belirtmiştir.

OpenClinical,[3] güncel kullanımda olan bir dizi karar destek sistemi sıralayan internet siteleri hazırlıyor. Bunlar arasında Athena DSS (yüksek tansiyon idaresi için), Gideon (bulaşıcı hastalıklar için), Illiad (dahiliye için), TherapyEdge HIV (HIV hastalarının idaresi için) ve birtakım başka sistemler var. Sıralanan sistemlerden bazılarının kökü MYCIN, INTERNIST-1, PUFF ve Bayes ağlarına dayanan tanı sistemlerine uzanır. Bu ve başka sistemler hakkında daha ayrıntılı bilgilere OpenClinical internet sayfalarından ulaşabilirsiniz. Başka bir bilgi kaynağı da Elsevier dergisi *Artificial Intelligence in Medicine*'dir.

Bir iki temsili örnekten bahsedelim. ATHENA DSS, ABD Tıp Enstitüsü'nün tanımlamış olduğu rehberlerle tutarlı şekilde yüksek tansiyonu ele alma konusunda doktorlara tavsiyeler veren bir sistemdir. Bu sistemi Stanford Tıbbi Enformatik, ABD Gaziler İdaresi Palo Alto Sağlık Sistemi ve Stanford Temel Bakım ve Sonuçları Merkezi ortaklaşa geliştirdi. ATHENA, hastanın klinik verilerini, kendi bilgi tabanındaki yüksek tansiyon idaresi bilgileriyle kıyaslayıp klinik muayene sırasında tedavi için hastaya özgü tavsiyeler yaratır. Bu sistem şu an kullanımdadır ve ABD Gaziler İdaresi'nin kimi tıp merkezlerinde değerlendirilip güncellenmeye devam ediliyor. ATHENA-HTN adlı yeni bir sürümü de değerlendirmeye alınmıştır. ATHENA teknolojisinin kökü, daha önce Stanford'da geliştirilmiş olan tıbbi kural tabanlı sistemle-

2 Bkz. http://www.clinicalwindow.net/cw_issue_20_articlel.htm.

3 Bkz. http://www.openclinical.org/aisinpracticeDSS.html. OpenClinical, kâr amacı gütmeyen bir kurumdur, Birleşik Krallık Kanser Araştırmalarının desteğiyle uluslararası bir teknik danışma kurulunun gözetimi altında kurulup sürdürülmektedir.

re uzanır. Stanford Biyomedikal Enformatik Kürsüsü'nün başı Mark Musen'a göre, "ATHENA, protokol tabanlı tıbbi bakıma yardımcı olmak amacıyla göreve özgü EON mimarisinden faydalanır; bu mimariyse, 1980'lerin ikinci yarısında yaptığım tez çalışmalarından, o çalışmalar ONCOCIN'den [onkoloji protokollerinin idaresine yardımcı olan bir program], o program da MYCIN'den tomurcuklanmıştır..."[4]

Başka bir sistem, Gideon, ülkelere özgü hastalıkları tanı ve tedavi etme konusunda doktorlara yardımcı olan bir programdır. Gideon, tanılarını, hastalıklardan, belirtilerden, işaretlerden, laboratuvar bulgularından ve ülkelerden oluşan büyük bir veritabanına dayandırır. Belirli bir hastayla ilgili veriler ışığında, hastalık olasılığının hesaplanmasında Bayesçi çözümlemeden yararlanılır. Gideon'un asıl sürümünü geliştirenler, Tel Aviv Sourasky Tıp Merkezi'nden Stephen A. Berger ile Tel Aviv Üniversitesi'nden Uri Blackman olmuştur. Günümüzde Blackman, Los Angeles'daki Gideon Informatics A.Ş.'nin icra kurulu başkanıdır.[5] Bu programın, cep telefonlarıyla ya da cep bilgisayarlarıyla erişilebilecek sürümleri de vardır.

Gideon internet sitesine göre:[6]

> Gideon, dört modülden meydana geliyor: Tanı, Epidemiyoloji, Tedavi ve Mikrobiyoloji. Sürekli güncellenen veritabanı, 337 hastalık, 224 ülke, 1.147 mikrop taksonu ve 306 antibakteriyel (ya da mantar, asalak ve virüs karşıtı) etmen ve aşı içerir. Gideon'un dünya çapındaki veri kaynakları esasen bütün dünya yazınını kapsar ve Bulgu Tabanlı Tıbbın standartlarına bağlıdır. 10.000'i aşkın not, her ülkedeki özgül enfeksiyonların statüsünü sıralıyor. Ayrıca, 20.000'i aşkın görüntü, grafik, etkileşimli harita ve kaynak sunuluyor.
>
> ...
>
> 495 hastadan oluşan çok merkezli kör bir saha araştırmasında, örneklerin en az %94'ünde doğru tanı konulmuş, en az %75'inde doğru teşhis birinci sırada gösterilmiştir.

Journal of the American Medical Association'da çıkmış bir incelemede şöyle yazılmıştı: "Gideon: İlk Küresel Bulaşıcı Hastalık ve Epidemiyoloji Ağı, dünyanın herhangi bir ülkesinde (224 ülke içinden) herhangi bir bulaşıcı hastalığı (tanımlanmış 337 hastalık içinden) doktorların tanı

4 E-postayla iletişim, 25 Eylül 2008.

5 Bkz. Stephen A. Berger ve Uri Blackman, "A Computer Program for Diagnosing and Teaching Geographic Medicine", *Journal of Travel Medicine*, c. 2, no: 3, s. 199-203. Şu internet adresinde: http://www.gideononline.com/reviews/JTM1995.pdf.

6 http://www.gideononline.com/index.htm.

koymasına yardımcı olmak amacıyla yaratılmış, harika tasarlanmış bir uzman sistemdir... Bu tanı sistemi, kolay kullanımı, ölçeğinin genişliği ve bilgi derinliği bakımından çarpıcıdır. Bir programdan umduğumuz kadar kullanışlıdır bu sistem."[7]

Gideon benzeri tanı sistemlerinin başarısına rağmen, tıptaki çoğu YZ uygulaması, doktorların kaynakça olarak faydalandığı DSS'ler içerir. Tıp enformatiği konusunda uzman olan Thomas Rindfleisch şunu söylüyor: "DSS adı önemli çünkü yazılımı FDA kurallarına göre düzenlemekten kaçınmak için hastayla ilgili son kararlardan her zaman doktor sorumlu olmalı". Ayrıca Rindfleisch, DSS'lerin çoğu doktora faydalı olması için, elektronik tıp ve sağlık kaydı sistemleriyle bütünleştirilmeleri gerektiğini belirtir (böylece doktorlar, hastayla ilgili bütün geçmiş bilgisini tuşlamak zorunda kalmaz).[8] Bununla birlikte, 2.758 Amerikalı doktorla yürütülnüş bir araştırma (2007'nin ikinci yarısında ve 2008'in ilk yarısında), bunların sadece %17'sinin elektronik kayıt sistemlerine erişimi olduğunu ortaya koymuştur.[9]

34.2 ÇİZELGELEMEDE

Zeki çizelgeleme sistemleri, YZ tekniklerinin kullanıldığı başka bir alandır. "Geleneksel yaklaşımların kullanıldığı çözümleri aşan problemleri çözme amacıyla yapay zekâ ve ileri yazılım teknolojileri" uygulamada uzmanlaşmış olan Stottler Henke Associates, A.Ş adlı şirketin pazarladığı AURORA™ sistemi, bunun örneklerinden biridir. AURORA, Boeing Dreamliner™'ın çizelgelemesi ve inşasında yardımcı olsun diye Boeing Şirketi tarafından kullanılmıştır. Stottler Henke şunu söylüyor: "AURORA, bir çizelge oluşturduğunda, sergilediği bir dizi grafikli imge sayesinde kullanıcının, çizelgelenmiş etkinlikleri, kaynak tahsisini ve etkinlikler arasındaki zamansal ilişkileri görmesine olanak tanır."[10]

United Space Alliance LLC'nin geliştirmiş olduğu TEMPORIS™, gelecekteki uzay uçuşlarında gemi mürettebatının kullanabileceği zeki bir uzay uçuşu idaresi ve planlaması aracıdır. TEMPORIS'in yardımı

7 Vincent J. Felitti, MD, Reviewer, *Journal of the American Medical Association*, c. 293, s. 1674-1675, 2005.

8 Thomas Rindfleisch, e-posta, 20 Kasım 2008.

9 Catherine M. DesRoches ve diğ., "Electronic Health Records in Ambulatory Care – A National Survey of Physicians", *The New England Journal of Medicine*, c. 359, no: 1, 3 Temmuz 2008, s. 50-60.

10 Bkz. http://www.stottlerhenke.com/news/pr_aurora_boeing.htm.

sayesinde mürettebat, uçuş sırasında geçirecekleri zamanlarını tüm yönleriyle bir çizelgeye yerleştirir, örneğin sıradan gündelik etkinlikler, uzay gemisinin temizliği ve bakımı, gemide deneyler yürütülmesi gibi. Fakat kabul edilebilir çizelgeler oluşturma, "uzay uçuşu kısıtlamalarını, uçuş kurallarını, bağımlılıkları, dizgeleri, tıbbi rehberleri ve güvenlik gereksinimlerini" bilmeyi gerektirir. Şirketin basın açıklamasına göre bu unsurlar, "TEMPORIS'in zekâsına etkin şekilde iliştirilebilir. Örnekle açıklamak gerekirse: Uluslararası Uzay İstasyonundaki tek günün etkinliklerini çizelgelemek için, 50 görev planlamacısının iki hafta boyunca aralıksız çalışması gerekir. TEMPORIS, bu iki haftalık işi, bilgisayar faresine tıklanmasıyla birkaç saniyeye indiriyor."[11] Stottler Henke'nin AURORA yazılımı, TEMPORIS'in dahilî bir parçasıdır.

34.3 OTOMATİK ALIŞVERİŞTE

Hisse senedi, meta ve para alım satımına dair otomatik kararlar verirken gerçek zamanlı alışveriş verileri ve haberlerin çözümlenmesinde YZ veri madenciliği, metin işleme ve karar yöntemleri kullanılır. En yeni haberleri veren kaynaklar sayısal biçimde zaten bulunur. Reuters'in "HaberÖlçeği Arşivi"[12] ve Dow Jones'un "Temel Haberleri"[13], otomatik alışveriş ve çözümleme için kullanılan haber kaynaklarıdır. Örneğin Reuters, şunu ileri sürüyor:[14]

> Haberlere dayanan programlı alım-satım stratejileri kurmak isteyen müşterilere, Reuters'in bütün dünyadan topladığı haberlerin ayrıntılı ve bilgisayarlarca okunabilir arşivi temin ediliyor. Olaylar gerçekleşir gerçekleşmez piyasalara sunulur; her bilgi açıklamasının zamanı, milisaniyesine kadar belirtilir ve makinelerin kolayca algılayabilmesi için üst-veri sınıflarıyla işaretlenirler.

New York Times'ta çıkmış, otomatik alım-satımla ilgili bir makalaye göre,[15] MIT Mali Mühendislik Laboratuvarı Müdürü Profesör Andrew Lo ve çalışma arkadaşları, "kaygı", "buhran" ve iflas" gibi belirli sözcükle-

11 Bkz. http://www.unitedspacealliance.com/news/press/2006/060404.pdf.

12 Bkz. http://about.reuters.com/productinfo/newsscopearchive/.

13 Bkz. http://www.djnewswires.com/us/djenf.htm.

14 http://about.reuters.com/productinfo/newsscopearchive/.

15 Tim Arango, "I Got the News Instantaneously, Oh Boy", *New York Times*, s. WK3, 14 Eylül 2008. Şu internet adresinde: http://www.nytimes.com/2008/09/14/weekinreview/14arango.html?partner=rssnyt&emc=rss.

rin haberlerde boy gösterme sıklığı ile S&P'nin borsa endeksi arasında bir bağıntı olduğunu keşfettiler. Hisse alım-satım algoritmaları, başka bileşenlerin yanı sıra bu bağıntılardan, hisse alışverişi başlatmak amacıyla yararlanabilir.

Algoritmalara dayalı alım-satım hizmeti sunan şirketlerden biri olan Vhayu Technologies, "en tepedeki 10 mali kurumdan 8'inin, milisaniyeler içinde fırsatları fark edebilmek amacıyla kendi ürünlerinden birini", yani Vhayu Velocity™'yi kullandığını iddia ediyor.[16] Başka bir firma, Streamhouse Systems, ürünlerinden biri sayesinde, "önde gelen alım-satım kuruluşlarının, birden fazla piyasada önemli piyasa koşullarını izleyebildiğini ve kısa vadeli alım-satım fırsatlarını yakalamak amacıyla gelişmiş stratejileri hayata geçirebildiklerini", söylüyor.[17] Elbette, anı anına gerçekleşen bu bilgi akışı sayesinde, bir sat emrinin başka sat emirlerini tetiklemesi, böylece piyasada hızlı düşüşlerin (ya da çıkışların) yaşanması potansiyeli de var.

34.4 İŞ UYGULAMALARINDA

İş Dünyası Kuralları İdare Sistemleri (BRMS), 1980'lerin kural tabanlı uzman sistemlerin mirasçılarıdır. Örnekleri arasında, Fair Isaac'ın "BLAZE ADVISOR 6.1", ILOG'un "JRules 6.0" ve Information Builders'ın "WebFOCUS" sistemlerini sayabiliriz. İş dünyası kuralları, bir işletmenin nasıl çalıştığı, politikaları ve kısıtlamaları hakkında bilgi ifade eder. Tüm şirketlerin bu tür kuralları bulunur. Fair Isaac'ın belirttiği gibi, bu kurallar çoğunlukla "sohbet, yazılı metin ve yazılımda ifade edilir; [örneğin] 'eğer krediye başvuran kişinin kredi sicili yeterli değilse, o zaman borç bürosundan rapor al' gibi 'eğer-ise' nitelikli önermeler benzeri"dir. BRMS'lerde bu kurallar çoğunlukla İngilizce benzeri, bilgisayarların okuyabileceği bir sözdizimiyle kodlanır. Bazı uzman sistemlerde kullanılan kuralların aksine, bunlara olasılık ya da kesinlik etkeni atanmış değildir, fakat bunlar, iş uygulamalarıyla ilgili kesin ifadelerdir. İş dünyası kuralının ifade ettiği bilgiler zamanla değiştiği için, güncel siyasetleri yansıtmak amacıyla kuralların korunması önemlidir.

Kural motorları, bir kurallar ağında hem ileri hem geri çıkarsama yapmak üzere kullanılır. s. 310'da bahsettiğimiz Rete algoritmasının mirasçıları, örneğin BLAZE ADVISOR'daki çıkarsama motorunda kullanılıyor. Politikanın, en son iş fırsatlarının ve kadro ile öbür kişiler için

16 Bkz. http://www.vhayu.com/.

17 Bkz. http://www.streambase.com/algorithmic-trading.htm.

gerekli eylemlerin iletiminde, bu çıkarsamaların sonuçlarından faydalanılır. Bazı örneklerde bu sonuçlar, ısmarlama, e-posta gönderme, vesaire gibi otomatik eylemler doğurur. Örneğin, Information Builders reklamlarında, ellerindeki olay idaresi çözümleri sayesinde, "otomatikleştirilmiş süreç, önceden belirlenmiş kararları bizzat alabilir ve ona verilmiş iş dünyası zekâsı kapsamındaki eşikleri temel alarak çeşitli eylemlerde bulunabilir", deniliyor.[18] WebFocus hakkındaki bir makale şu örneği veriyor: "[Bir alım talebi], bir dizi yanıtı ve kararı tetikleyebilir; örneğin, işlemde gömülü olan WebFocus çözümlemeleri temel alınarak, alım talebinin fiziksel boyutu belirlenir. Eğer sipariş, deponun alamayacağı kadar büyükse, WebFOCUS çözümlemecileri işlemde değişiklik yapıp, siparişi, gerekli boyuta sahip depoya gönderir ve depo çalışanlarını uyarıp beklenenden daha büyük sipariş geldiğini haber verir."[19]

34.5 DİLLERİN ÇEVİRİSİNDE

Günümüzde, ticari olan birtakım doğal dil çeviri sistemleri var. IBM'in konuşmadan-konuşmaya çeviri programı (MASTOR), serbest biçim İngilizce konuşmaları Mandarin diline (ve tersine) çevirebiliyor (kendi İnternet sitesine göre).[20] BBN Technologies, pek çok konuşma işleme sistemi geliştirmiştir. Bunlardan biri, "Yayın Takibi Sistemi"dir; (kendi internet sitesine göre) bu sistem, "uluslararası televizyon yayınlarının kesintisiz ve içinde arama yapmaya uygun bir arşivini yaratır". İnternet sitesinde açıklandığı üzere, "Sistem, gerçek zamanlı işitsel bilgiyi otomatik olarak metin halinde kaydeder ve İngilizceye çevirir. Hem metin hem de çeviri içinde arama yapmak mümkündür, ayrıca metin ve çeviri, videoyla eş zamanlı hale getirilir; bu iş, konuşma içeriği temel alınarak videoya etkin şekilde erişim ve videonun hatasız biçimde tekrar oynatılması için çok güçlü beceriler temin eder. Bu devrimci sistemle birlikte kullanıcılar, başka dillerdeki haberlerin büyük miktarda derlemelerine hızlı ve etkin biçimde ulaşabilir."[21] SRI International'ın IraqComm çeviri sistemi, konuşma İngilizcesini Irak Arapçasının gündelik konuşma diline (ve tersine) çevrilebilir. Şu an

18 http://www.informationbuilders.com/products/webfocus/intelligent_processes.html.

19 Şu dergiden: *Expert Systems*, c. 23, no: 4, Eylül 2006, s. 245.

20 kz. http://domino.research.ibm.com/comm/research.nsf/pages/r.uit.innovation.html.

21 http://www.bbn.com/products_and_services/bbn_broadcast_monitoring_system/.

(internet sitesine göre), "kuvvet koruma, güvenlik ve temel tıbbi hizmetler gibi konulardaki sohbetleri çevirmek üzere şekillendiriliyor ve gerektiği anda başka konuları da kapsayacak biçimde uyarlanabilir."[22]

34.6 İCATLARI OTOMATİKLEŞTİRMEDE

Evrim sürecini taklit etmeyi temel alan bir arama yöntemi olan Genetik Programlamanın (GP) mucidi John Koza, bizzat GP'nin bir "icat makinesi" olduğunu iddia eder. (Aynı zamanda, GP'nin aşağı yukarı YZ kapsadığını da iddia ediyor, çünkü YZ'nin hedefi, zeki programlar yaratmaktır ve GP tam da bu işi yapıyor.) Örneğin, Koza ve meslektaşları, optimal bir anten sistemi evriltmek (otuz bir kuşakta) amacıyla GP'den faydalanmıştır.[23] Ayrıca, optik mercek tasarımları da evriltmişlerdir (yüzlerce kuşakta). Makalelerinden birinde belirtildiği gibi: "Genetik yolla evrilmiş mercek sistemlerinden biri, daha önce alınmış bir patenti ihlal ederken, patentlerde geçen performans tanımlarını tekrarlayan (ya da iyileştiren) ve patentleri ihlal etmeyen yeni tasarımlar da yaratılmıştır."[24] Bu şekilde evriltilmiş tasarımlar arasında elektrik devresi, regülatör, mekanik sistem ve çeşitli cihaz tasarımları da vardı.[25] Bu ekibin hedefi, "insanlarla yarışan tasarımlar" yaratmaktı, yani özellikleri, en başarılı insan tasarımcılarının eserlerini yakalayan ya da aşan tasarımlar oluşturmaktı. Bilgisayarların gücünün artmasıyla birlikte, GP ve GP benzeri arama yöntemleri kullanımının, birtakım ilginç icatlar üreteceğine inanıyorum.

22 Bkz. http://www.iraqcomm.com/.

23 John R. Koza ve diğ., "Automated Synthesis of a Fixed-Length Loaded Symmetric Dipole Antenna Whose Gain Exceeds That of a Commercial Antenna and Matches the Theoretical Maximum", *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, New York: Association for Computing Machinery, 2007, s. 2074-2081.

24 John R. Koza, Sameer H. Al-Sakrana ve Lee W. Jones, "Automated *ab Initio* Synthesis of Complete Designs of Four Patented Optical Lens Systems by Means of Genetic Programming", *AIEDAM: Artificial Intelligence for Engineering, Design, and Manufacturing*, c. 22, Cambridge: Cambridge University Press, 2008, s. 249-273.

25 Bkz. John R. Koza, Sameer H. Al-Sakran ve Lee W. Jones, "Multi-Domain Observations Concerning the Use of Genetic Programming to Automatically Synthesize Human-Competitive Designs for Analog Circuits, Optical Lens Systems, Controllers, Antennas, Mechanical Systems, and Quantum Computing Circuits", *Genetic Programming Theory and Practice IV* içinde, Rick Riolo, Terence Soule ve Bill Worzel (ed.), New York: Springer-Verlag, 2007, s. 131-147.

34.7 YÜZ TANIMADA

İnsanlar, ister "canlı" ister fotoğrafta olsun bildik yüzleri tanımada gayet başarılıdır. Bu işi, duruş, ölçek, yüz ifadesi ya da ışıklandırma koşulu ne olursa olsun becerirler. (İlginçtir, eğer yüz fotoğrafı tersyüz edilip sunulursa o kadar başarılı olamıyorlar.) Bilgisayarlar ise bu işte gitgide daha becerikli oluyor, öyle ki Yapay Zekâ Her Yerde başlıklı bölüme, bilgisayarla yüz tanımayı dahil edebilirdim. Yüz tanıma sistemleri, henüz her yerde olmasa da, havaalanlarında, bankalarda ve kişisel kimliğin doğrulanması ya da saptanması gereken yerlerde yaygınlaşıyor. Kişisel mahremiyet konusunda endişe duyan kimi insanlara göre, bu uygulama fazlasıyla yaygın.

Bilgisayarla yüz tanıma çalışmaları ilk günlerinden itibaren devam ediyor. Peki, ne kadar ilerleme kaydedildi? Ulusal Standartlar ve Teknoloji Kurumunun 2007 yılında yüz tanıma testleriyle ilgili yayımlanmış raporunda (başka görüşlerin yanı sıra) şu da ileri sürülüyordu: "Sonuçlar gösteriyor ki, insanların düşük hata oranlarıyla kıyaslandığında, farklı ışıklandırma koşullarında yüz tanıma konusunda yedi otomatik yüz tanıma algoritması, insanlarınkiyle benzer ya da daha iyi randıman vermiştir. Dahası, bu yedi algoritmanın üçü, ölçülen tüm hata oranlarında insanlara yakın, hatta onlardan iyiydi."[26] En başarılı yöntemler, çok büyük veri kümeleri üzerinde çalışan yapay öğrenme algoritmaları kullanıyordu.[27]

Çeşit çeşit algoritma geliştirilmiştir. Bazıları, bir yüz imgesinin özelliklerini toplayıp buna en yakın eşleşmeyi bulmak amacıyla, bunları, tanımlanmış yüzlerin büyük kütüphanesindeki yüz özellikleriyle kıyaslayan, iyi bilinen örüntü tanıma tekniklerini temel alır. Bazı algoritmalarsa Bayesçi tekniklerden ve SMM'lerden faydalanır. Bu yöntemlerin pek çoğu, yüz imgesinin yüksek boyutlu vektör gösterimini daha az boyutlu altuzamdaki bir vektöre uyarlamak için matematik teknikleri kullanır. Bu yöntemlerden biri, koordinatları, *eigenfaces* [özyüz] denilen bir dizi indirgenmiş imgeden oluşan daha az boyutlu uzamlar kullanır; bu indirgenmiş imgelerin, veritabanındaki herhan-

26 P. Jonathon Phillips ve diğ., "FRVT 2006 and ICE 2006 Large-Scale Results", NISTIR7408, Mart 2007. Şu internet adresinde: http://www.frvt.org/FRVT2006/docs/FRVT2006andICE2006LargeScaleReport.pdf.

27 Araştırmacıların kullandığı bazı veri kümeleri şu internet adresinde listelenmiştir: http://www.face-rec.org/databases/. Ayrıca bkz. Ralph Gross, "Face Databases", *Handbook of Face Recognition* içinde, Stan Z. Li ve Anil K. Jain (ed.), New York: Springer-Verlag, 2005. Şu internet adresinde: http://www.ri.cmu.edu/pub.files/pub4/gross_ralph_2005-1/gross_ralph_2005_1.pdf.

gi bir yüzün kestirimini verecek şekilde birleştirilme özelliği bulunur (farklı frekanslı tek tek ses tonlarının, herhangi bir sese benzeyecek şekilde birleştirilmesi gibi). Araştırma makalelerine, kitaplara, algoritmalara ve satış bayilerine linkler içeren ve yüz tanıma hakkında bilgi sunan bir internet sayfası için bkz. http://www.face-rec.org/.

Üstelik hâlâ yeni yaklaşımlar geliştiriliyor. Yöntemlerden biri, "görüntü ortalaması almanın" (yani aynı yüzün farklı görüntülerini birleştirip tek imge oluşturma), "[ticari FaceVACS] otomatik yüz tanıma sisteminin performansını muazzam iyileştirdiğini" gösterdiğini iddia ediyor.[28] *Wired* dergisinde çıkmış bir makale, Berkeley California Üniversitesi ve Urbana-Champaign Illinois Üniversitesi araştırmacılarının geliştirdiği bir yöntemden bahsediyor. Bu makaleye göre, Berkeley California Üniversitesi'nin Mühendislik Fakültesi Dekanı Shankar Sastry, bu yeni yöntemin, "bu alandaki onca yıllık araştırmaları hükümsüz kıldığını" belirtmiş.[29]

Yüz tanıma ve yüz saptama yazılımı ve teçhizatı satan kimi ticari şirketler şimdiden var. Örneğin, Oki Electric Industry Co. Ltd., FSE (Face Sensing Engine; Yüz Duyumsama Motoru) adlı bir ürün satıyor. Bu program, pek çok uygulamasıyla böbürlenir, örneğin kameralı cep telefonlarındaki ve başka cihazlardaki bilgiye erişimi denetlemek, yüz tanımaya bağlı olarak fotoğrafları sıralamak ve kameranın görüş alanındaki yüzleri tespit etmek gibi. Alman şirketi Cognitec Systems Ltd., yukarıda bahsetmiş olduğumuz FaceVACS sistemini pazarlıyor.[30]

Akıllı aletlerle ilgili bu bölümü kapatmadan önce, YZ aletlerinin, insan verimini artırdığı başka alanlar da olduğunu belirtelim. Örneğin, film animasyonu işlemlerine, yazı yazan ve hataları düzelten bilgisayar programlarına, sanayi işlemleri denetimine, devre ve program doğrulanmasına, anlamsal Ağı güçlendirip aramaya yardımcı olan (ve otomatikleştiren) araçlardan bahsedebilirdim. YZ tekniklerinin can verdiği aletler, insanın bilişsel, devimsel ve algısal yetilerine gitgide daha fazla yardımcı olup bunları güçlendirecektir (zaman zaman yerlerini de alacaktır). Bekleyip görün!

Son birkaç bölümde, yapay zekâ arayışının bazı yönlerinin oldukça başarılı olduğunu gösterdiğimi umuyorum. YZ, hayatımızda ve

28 R. Jenkins ve A. M. Burton, "100% Accuracy in Automatic Face Recognition", *Science*, c. 319, 25 Ocak 2008, s. 435.

29 Bryan Gardiner, "Engineers Test Highly Accurate Face Recognition", *Wired*, 24 Mart 2008. Şu internet adresinde: http://www.wired.com/science/discoveries/news/2008/03/new_face_recognition.

30 http://www.cognitec-systems.de/index.html.

uzmanların hayatında gitgide daha fazla yer kaplar oldu. Fakat bu arayışın ana hedefine henüz ulaşılmış değil (en azından bazılarımız için), yani insan eseri nesneleri, tam teşekküllü insan (ve üstün insan) lisanı, algısı, mantığı ve öğrenme yetisiyle donatma amacına henüz erişilmedi. O halde, bu arayışın bizi nerelere sürükleyebileceğine göz atalım artık.

35
ARAYIŞ SÜRÜYOR

YZ arayışı bundan sonra nereye varacak? Şu anki eğilimlere bakarak, yakın gelecekte olacaklar hakkında fikir edinebiliriz. Muhtemelen, kilometre taşı nitelikli yeni başarılar elde edilecek. Hiç şüphesiz, evlerimizde, arabalarımızda ve etkinliklerimizde YZ teknolojisi yaygınlaşacak ve uzmanların akıllı araçları hem daha da akıllı hem de daha sık rastlanır olacaktır.

Fakat YZ'nin şu anki seyrinin ötesini tahmin etmek sorunludur. Geçmişte yapılmış öngörülerin bazılarının nasıl sonuçlandığına bakalım. 1957'de Simon'un, on sene içinde satranç şampiyonu bilgisayar yapılacağı fikri fazlasıyla iyimserdi. 1973'te, Oscar Firschein'ın başını çektiği SRI mühendisleri, belirli "ürünlerin" ne zaman hayata geçirileceğini kimi YZ "uzmanlarına" tekrar tekrar sormuştur. Tahmin edilen tarihlerin ortanca (*median*) ve menzilleri bu programlara bildirilmiş, böylece onlara, sonuçlar yerli yerine oturana dek, tahminlerini değiştirme olanağı tanınmıştır. (Bu öngörüde bulunma yöntemine *Delphi yöntemi* denir.[1]) Aşağıdaki tablo, son tahminlerden birkaçını gösteriyor:[2]

Ürün	ortanca prototip tarihi	ortanca ticari ürün tarihi
Otomatik tıbbi tanı koyucu	1976	1980
Tüm ev işlerini yapan robot hizmetçi	2000	2010
Sesle çalışan daktilo	1985	1992
Yüksek nitelikli otomatik çevirmen	1987	1995
Şehir sokaklarında ve kırsalda araba kullanan robot sürücü	1992	2000

1 Örneğin bkz. http://www.iit.edu/~it/delphi.html.

2 Alıntılandığı yer: Oscar Firschein ve diğ., "Forecasting and Assessing the Impact of Artificial Intelligence on Society", *Proceedings of the IJCAI*, 1973, s. 105-120.

"Robot hizmetçi" ve "robot sürücü" hâlâ çok uzak görünüyor ama öbürleri belki de sadece birazcık iyimser tahminlermiş. (1973'ten bakıldığında 2000'e daha çok varınış gibi görünüyordu.)

Bu öngörü başarıları ve başarısızlıkları üzerine, görece bariz görünenler dışında öngörüde bulunmaya tereddüt ediyorum. Ancak, *günün birinde*, insan zekâsını (ve tüm göstergelerini) yakalayıp geçen zekâ düzeyine sahip insan yapımı nesnelerin var olacağını öngörebilirim. Bu öngörüyü yapmamın sebebi, biz insanların makine olduğuna (çünkü makine değilsek başka ne olabiliriz ki?) ve bizlerin yaptığı her şeyi yapabilecek makineler inşa edebileceğimize, çünkü bunun hem iktisadi hem de bilimsel sebeplerinin var olduğuna inanıyor oluşumdur.

"İnsan düzeyinde yapay zekâ" hakkında daha söyleyeceklerim olacak, fakat önce, bu yüzyılın başlarında YZ laboratuvarlarında yürürlükte olan kimi araştırma projelerine göz atalım ki gelecek hakkında bize içgörü kazandırsınlar.

35.1 LABORATUVARLARDA

Günümüzde, yapay zekâ araştırmaları yürüten ve sanayi, devlet ve akademiye bağlı büyük olasılıkla yüzlerce laboratuvar var. Buralarda olup bitenlerin ufak bir kısmını bile betimlememe imkân yok ama her durumda projelerin biri başlıyor biri bitiyor. Güncel olayları gazetelere ve başka yayınlara bırakan bir tarihçi olarak, YZ araştırmalarının güncelini izlemek isteyen okurlara, YZ laboratuvarlarının, YZ cemiyetlerinin, YZ araştırmalarını destekleyen devlet kurumlarının, uzmanlaşmış konferansların ve çalıştayların hazırladığı internet sitelerini ziyaret etmelerini tavsiye ediyorum.[3] Güncel araştırmaların genişliği hakkında fikir vermek amacıyla, bu yüzyılın hemen başında gerçekleştirilmiş birkaç projeye değineceğim. Elbette, bu araştırma projelerinden bazılarının, geleceğe izlerini bırakması olası ama kesin değil.

3 Şu kurumların internet siteleri başlamak için iyi birer yerdir: *Association for the Advancement of Artificial Intelligence*'in (AAAI), http://www.aaai.org/home.html; Avrupa Yapay Zekâ Eşgüdüm Komitesi (ECCAI), ttp://www.eccai.org/; Japonya Yapay Zekâ Derneği (JSAI), ttp://www.ai-gakkai.or.jp/jsai/english.html (İngilizce versiyonu); Çin Yapay Zekâ Derneği, http://caai.cn/; DARPA Enformasyon İşleme Teknikleri Bürosu, ttp://www.darpa.mil/ipto/; Almanya Yapay Zekâ Merkezi (DFKI), ttp://www.dfki.de/web/welcome?set_language=en&cl=en; ve INRIA, ttp://www.inria.fr/index.en.

35.1.1 Özelleşmiş Sistemler

Özelleşmiş alanlarda işe yarayacak akıllı aletler yapmak, YZ araştırmalarının hâlâ büyük bölümünü oluşturur. Bununla birlikte, bu aletler üzerindeki çalışmalar YZ merkezli olmaktan uzaklaşıyor ve bu çabaların bel bağladığı teknolojilerin mensup olduğu disiplinlerle kaynaşıyorlar, örneğin istatistik, kontrol mühendisliği, görüntü işleme ve dilbilim bunlardan sadece bazılarıdır. Elbette, en uzmanlaşmış niş sistemlerin inşası kapsamında icat edilen ve kullanılan yeni teknikler, aslında, YZ'yi ilgilendiren başka alanlarda da uygulanabilir.

A. İçerik Tabanlı Görüntü Erişimi

Günümüzde, örneğin "motorsiklet" gibi sözcüklerden oluşan sorgulara yanıt veren görsel ve video arama motorları, bir görsel ya da video dosyası yanında (jpeg, quicktime ya da başka bir format) "motorsiklet" sözcüğünü içeren internet sitelerine bakarak bu işi yapıyor. Maalesef, bu yöntemlerle erişilen bazı sitelerdeki görsel ya da video dosyalarında hiç motorsiklet de olmayabilir. Görsel ve video aramasını, sadece görüntülerin yanındaki metnin yanı sıra görüntünün içeriğine de dayandıran araştırmalar yürütülmektedir.

Bu tür bir projeyi, Oxford Üniversitesi ve Microsoft Şirketi araştırmacıları üstlenmiştir.[4] Kamuya açık bir internet sitesindeki (Flickr) görüntüleri kullanan "sorgu", aranan nesnenin içerdiği belli bir görüntü parçasının taslağını çizerek (dikdörtgenle) oluşturulur. Ardından, görüntünün bu parçası, nesne görüntüsünün kilit özelliklerini koruyan bir vektör gösterimine dönüştürülür. Büyük bir görüntü veritabanındaki görsellerin vektör gösterimleri, benzer vektörlerden meydana gelen "öbekler" oluşturmak üzere çözümlenir. Buradaki fikir, her öbeğin, benzer nesnelerin görüntüleriyle ilgili olmasıdır. Böylece, sorgunun vektör gösterimi, görüntü veritabanı vektörleriyle eşleştirilir ki en benzer görüntülerin öbekleri bulunabilsin. Ardından bunlar sıralanır ve hedef bölgeleri işaretlenmiş en yüksek dereceli görüntüler, sorgunun yanıtı olarak döndürülür. Sorgu görüntüleri ve yanıt görüntülerine dair birtakım örnekler Resim 35.1'de gösterilmiştir. Sorgu bölgesinin dikdörtgen içine alındığı sorgu görüntüsü, resmin sol tarafındadır ve buna yanıt olarak getirilen yedi görüntü resmin sağ ta-

4 Bkz. James Philbin ve diğ., "Object Retrieval with Large Vocabularies and Fast Spatial Matching", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. Şu internet adresinde: http://www.robots.ox.ac.uk/~vgg/publications/papers/philbin07.pdf.



Resim 35.1. Görsellerde nesne arama. (Şu makaleden: James Philbin ve arkadaşları, "Object Retrieval with Large Vocabularies and Fast Spatial Matching", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.)

rafında duruyor (hedef bölgeler dikdörtgen içine alınmış). Elbette, her arama işlemindeki gibi, yanlış görüntüler de getirilecektir, fakat bu örneklerde yanlış görüntüler, yanıt görüntülerinin gösterildiği listenin sonuna doğru boy gösteriyor.

Bu projeyi yürütenler, çalışmalarını şu şekilde değerlendiriyor:

> Bu sistem, sorgu görüntüsü ile getirilen görseller arasında önceki düzeyde ışık, perspektif, nitelik ve farklılıkları olmasına ve örtülmelere rağmen, sorgulanan nesneyi de içeren bir görüntü tabanından fotoğraflar getirir.
>
> Biz bunu, milyarlarca görüntüyü tarayacak olan internet ölçekli bir bulma sistemi hedefine doğru atılmış bir adım olarak görüyoruz. Dolayısıyla, ölçeklendirilebilecek algoritmalara yoğunlaştık ve bu algoritmaları bir dizi bilgisayara dağıtmanın kolay olduğuna inanıyoruz.

Bu araştırmacılar, "dizine uzamsal bilgi dahil etmenin etkin yollarını bulmak [görüntüler sorgulanmadan önce hesaplanır] ve uzamsal eşleştirmenin bir kısım yükünü sıralama aşamasından süzme aşamasına geçirmek için" daha fazla çalışma yapılması gerektiğini kabul ediyor.

İçerik-tabanlı video bulmayı içeren projeler de var. Örneğin, kullanıcılar Youtube'a video gönderdiğinde, bunu betimleyen "etiket" sözcükler de gönderir. Plaj, otostop, futbol, kedi, konser, vesaire, etiket sözcüklere örnek olarak sayılabilir. Videoları etiketlemek belli bir çaba gerektirir, dolayısıyla bazı araştırmacılar, istatistiksel yapay öğrenme yöntemleriyle bu süreci otomatik hale getirmeye uğraşıyor. Alman araştırmacılardan oluşan bir ekip, önceden etiketlenmiş videoların veritabanından yola çıkarak, başka videolar için etiket sözcük önermek için görüntü bilgisi çıkaran bir prototip sistem geliştirmiştir.[5]

5 Bkz. Adrian Ulges ve diğ., "Content-based Video Tagging for Online Video Portals", *Third MUSCLE/ImageCLEF Workshop on Image and Video Retrieval Evaluation,*

Onlarınki gibi sistemler, en sonunda büyük video görüntü tabanlarını etiketlemek amacıyla kullanılabilir. Böyle bir görüntü tabanı, etiketlendiğinde daha rahat aranır hale gelecektir.

Bu yönde ilerleyen iki farklı şirket, VideoSurf ve Digitalsmiths, içerik tabanlı arama yapmaya olanak tanıyan ürünlerini duyurmuştur. VideoSurf'ün internet sitesinde (http://www.videosurf.com/about) şöyle söyleniyor: "VideoSurf, yeni bilgisayarla görme tekniklerinin ve hızlı bilişim yöntemlerinin eşsiz bir birleşimi kullanarak, bilgisayarlara, videoların içini 'görüp' hızlı, verimli ve ölçeklenebilir bir biçimde içerik bulmayı öğretiyor."

B. Anlam Tabanlı Ağ Arama

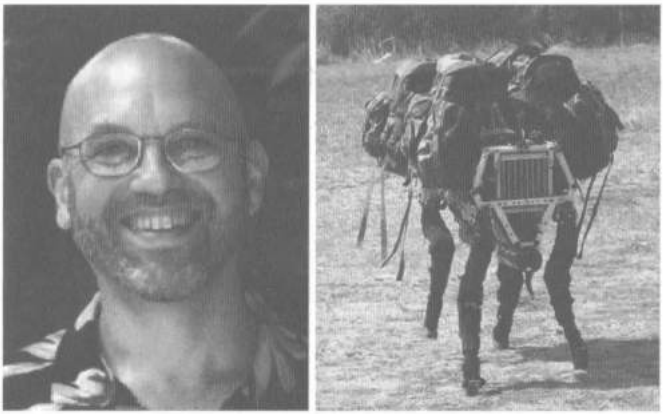
Şimdi de Powerset adlı ticari oluşumdaki bir YZ projesinden bahsedeceğim. Powerset, San Francisco'da kurulmuş bir şirkettir ve en başta, doğal dil işleme tekniklerini kullanan bir internet arama motoru geliştiriyordu. (Şirketi, 2008'de Microsoft devralmıştır.) Powerset, "sorgunuzun anlamıyla ilişkili yazıları, hatta bazen sorunuzun doğrudan yanıtlarını", bulabileceğini iddia ediyor.

Sorgu, sıradan doğal dille ifade edilen bir önerme ya da soru olabilir. Böylece, bir dilbilgisi ve anlambilim işlemcisi, sorguyu çözümler; yani, cümleyi özgün cümlenin anlamını ifade eden bir gösterime dönüştürür. Bu sistemin prototip sürümü, Wikipedia makalelerini aramakla kısıtlıydı, ayrıca bu makaleleri işleyip anlamlarını çıkarabiliyordu. Ardından, anlamları bakımından sorguyla en ilişkili olan makaleleri getiren bir eşleştirme yöntemi kullanılıyordu.

Bu sistemin geliştiricileri, teknolojilerinin, Wikipedia'nın ötesine geçip internetin daha büyük kısmıyla başeder hale geleceğini düşünüyor. Powerset'in kullandığı doğal dil işleme teknolojisi, aslen Joan Bresnan ile Ronald Kaplan'ın geliştirmiş oldukları Sözel İşlevsel Dilbilgisini temel alır. PARC'nin (Palo Alto Araştırma Merkezi) eski araştırmacılarından olan Kaplan, günümüzde bir Powerset çalışanıdır. Onların internet sitesi üzerinden yapılan bir Powerset aramasının örneği aşağıdaki gibidir.[6] "Powerset, hangi teknolojinin lisansını almıştır", sorgusunu yazdım. Yanıtı şu oldu: "Bu şirket, eskiden Xerox Palo Alto Araştırma Merkezi olarak bilinen PARC'ın doğal dil işleme teknolojsinin lisansı-

2007. Şu internet adresinde: http://demo.iupr.org/videotagging/youtube.pdf. Sistemin bir gösterimi için ayrıca bkz. http://demo.iupr.org/videotagging/tagging-description.html.

6 http://www.powerset.com/.



Resim 35.2. Marc Raibert (solda) ve BigDog (sağda). (Fotoğraflar, Boston Dynamics © 2008'in izniyle).

nı almıştır..."; ayrıca bu yanıtı bulduğu Wikipedia sayfasına atıfta da bulundu, yani http://en.wikipedia.org/wiki/Powerset adresindeki "Powerset (şirket)" maddesini getirdi. Eğer, Powerset çalışanlarının umduğu gibi, anlam tabanlı arama tekniklerinin ölçeği genişlerse, İnternet aramasının niteliği epey gelişmiş olur.

C. Ayaklı Robotlar

Caltech'de Marc Raibert, yürüyen, koşan, zıplayan robotlar üzerine araştırmalar yapmıştır. Bu konuyla ilintili çalışmalarını önce CMU profesörü olarak, sonra da MIT'de sürdürdü. 1992'de, Boston Dynamics adlı bir şirket kurdu; kendi internet sitesine göre bu şirket, "robotik ve insan benzetimi alanlarında uzmanlaşmıştır". Şirketin prototip ürünlerinden birinin adı BigDog'dur (Resim 35.2); yaklaşık Danua cinsi köpek boyutunda olup yürüyen bu dört ayaklı robotun, "dünya üzerindeki en ileri dört ayaklı robot" olduğu iddia edilmiştir. BigDog, son derece dayanıklıdır. Yürüyebilir, koşabilir, engebeli arazide tırmanabilir. 340 librelik [yaklaşık 155 kg] yük taşıyabildiği söyleniyor. Hidrolik hareket sistemini çalıştıran gazolinli bir motorla hareket eder. BigDog'un sensör donanımı, bir lazer giroskopu, çiftli bir görüş sistemi, eklem konumlarını ve kuvvetini, ayrıca motor sıcaklığı vesaire gibi iç unsurları duyumsayan cihazlar içerir. Denetim bütünüyle robot üzerindeki yerleşik bir bilgisayarla sağlanır. İşleyişiyle ilgili bir filmde, robotun ormanda yürümesi, bir tepeye tırmanması, yan tarafına gelen güçlü bir darbeden sonra kendine gelmesi ve buzlu bir park yerine kurulması gösteriliyor; bunların hepsini düşmeden başarıyor. (En azından benim seyrettiğim

filmde düşmüyor. Bu film şu internet adresinde: http://www.bostondynamics.com/dist/BigDog.wmv.) BigDog projesi, DARPA tarafından desteklenmektedir.

BigDog'la ilgili bir makaleye göre,[7] ekip, BigDog düştüğünde kendi kendine doğrulması gibi problemler ve yönünü kendi kendine bulma üzerine çalışmayı sürdürüyor. (Günümüzde, buna kılavuzluk eden insan operatöre bel bağlıyor.)

Yüzlerce meslek için "akıllı aletler" geliştiren laboratuvarlarda yüzlerce proje daha var. Söz konusu çeşitlilik inanılmazdır. İnsanlar, animasyon film işlemi,[8] bilişimsel genomik,[9] robot cerrahisi, iş istihbaratı[10] ve çok daha fazlasının otomasyonu üzerinde çalışıyor. Bunların çoğu, "niş" sistemleri diyebileceğim sistemlerdir, çünkü genel kapsamlı işlerden çok özgül işleri yapmaya odaklanmışlardır. Bir sonraki kısımda, ya çok amaçlı sistemler kurmayı ya da geniş bir yelpazede uygulanabilecek teknolojiler geliştirmeyi hedefleyen bazı laboratuvar çalışmalarını anlatıyorum.

35.1.2 Geniş Uygulama Alanı Olan Sistemler

A. Robotik

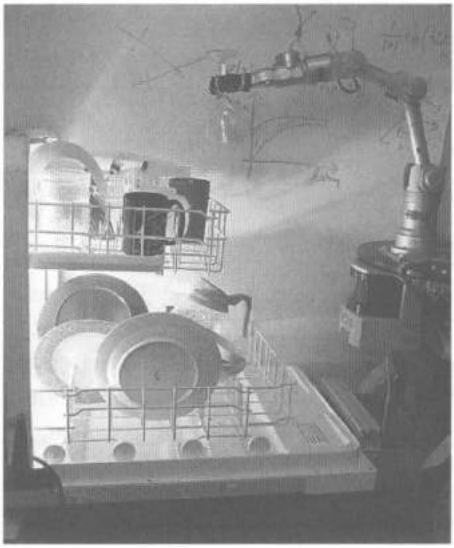
Robot Shakey'nin zamanından bu yana, YZ araştırmacıları, robotlardan, zekî davranışın pek çok yönünü bütünleştiren, dolayısıyla genel bir işe yararlık hedefleyen YZ sistemleri geliştirmenin düzlemi olarak faydalanmıştır. Çok amaçlı robot sistemleri alanındaki çalışmalar önce yürürlüğe girmiş, sonra da bunların modası geçmiştir ve robotik araştırmacıları, özgül işlere odaklanmak zorunda kalmıştır, örneğin sanayide işlerin otomasyonu gibi. 1970 ve 1980'lerde çok amaçlı robotlarla ilgili araştırma tekliflerime yanıt olarak, potansiyel destekçilerim, "robotunuz tam olarak ne yapacak?" diye sorardı bazen. Bu robotların "çok amaçlı" olacağını söylediğimde çözülecek özgül sorunları olan destekçilerim pek tatmin olmuyordu. Fakat günümüzde, pek çok şeyi yapabilen robotlara bir dönüş varmış gibi görünüyor; yani sırf engelli koşan ya da sırıkla yüksek atlayan robotlar yerine dekatlon

7 Marc Raibert ve diğ., "BigDog, the Rough-Terrain Quaduped Robot", *Proceedings of the 17th World Congress of The International Federation of Automatic Control*, Seul, Kore, 6-11 Temmuz 2008. Şu internet adresinde: http://www.nt.ntnu.no/users/skoge/prost/proceedings/ifac2008/data/papers/4278.pdf.

8 Örneğin bkz. http://graphics.cs.cmu.edu/projects/behavior_planning/.

9 Örneğin bkz. http://www.psrg.csail.mit.edu/.

10 Örneğin bkz. http://www.dfki.de/pas/f2w.cgi?ltp/musing-e.



Resim 35.3. STAIR, Stanford YZ laboratuvarında bulaşık makinesi boşaltıyor. (Fotoğraf, Andrew Ng ve Ashutosh Saxena'nın izniyle.)

yapan robotlar ilgi görüyor.

Başkanlığını Stanford profesörlerinden Andrew Ng'nin yaptığı, STAIR (STanford AI Robot) adlı robotla birlikte yürütülen çalışma bunun örneklerinden biridir. STAIR, bir tür "kahya" olarak tasarlanmıştır, yani ev ve büro ortamlarında yolunu bulmak, nesneleri ve aletleri kavrayıp bunlarla iş görmek, bu ortamlardaki insanlarla zekice konuşup onlara yardımcı olmak dahil elinden birçok iş gelen bir robottur (bkz. Resim 35.3). Kendi internet sitesine göre,[11] STAIR, "yapay öğrenme, görme, yön bulma, elle idare, planlama, uslamlama ve konuşma/doğal dil işleme dahil YZ'nin pek çok alanından gelen yöntemleri bütünleştiriyor. Bu yaklaşım, 30 yıldır var olan, YZ'nin alt alanları üzerinde ayrı ayrı çalışma eğilimine belirgin olarak zıttır ve araştırmaları, gerçekten bütünleşmiş bir YZ'ye doğru yönlendirmenin aracı olacaktır."

STAIR, daha önce hiç görmediği nesneleri bile kavramayı öğrenir. STAIR'in, yapay öğrenme yöntemlerinden faydalanan algı sistemi, evde ya da büroda göründükleri halleriyle pek çok yaygın nesnenin en az biner fotoğrafından oluşan bir veritabanıyla eğitilmiştir. Her görüntü, o nesneye özgü uygun kavrama konumuyla işaretlenmiştir. Seçilen cisimler arasında, zımba (2001 örnek fotoğraf), kahve fincanı (2001 örnek fotoğraf), yemek kasesi (1001 örnek fotoğraf) ve başka nesneler var. Eğitim amaçlı fotoğraflarda nesneler, farklı ışık koşulları, farklı

11 http://stair.stanford.edu/index.php

kamera açıları ve farklı yönlenimlerde gösteriliyor. (Eğitim amaçlı verileri toplama ve etiketleme işini kolaylaştırmak için, sentetik bilgisayar grafik görüntüleri kullanılmıştır.) Eğitim sürecinin ardından STAIR, birtakım yeni nesneler için en iyi kavrama açısını tahmin edip, kolu ve kavrayıcıyı yönlendirme planı oluşturabiliyordu. Bu sistem, yeni nesnelerden oluşan büyük bir kümeyle sınanmıştır, ki bunlar içinde robotun daha önce hiç görmediği nesneler de vardı. Bu kümeye, "bantlar, ispirtolu kalemler, şeffaf bir kutu, kesici aletler, cep telefonları, tükenmez kalemler, anahtarlar, tornavidalar, zımbalar, diş fırçaları, kalın bir kablo rulosu [ve] tuhaf şekilli bir boynuz" dahildi. STAIR, denemelerinin ortalama %87,8'inde bu nesneleri kavrayabildi. (Başarılı kavrama, "robotun nesneyi tutup, yaklaşık 30 cm havaya kaldırması ve otuz saniye boyunca düşürmemesi" anlamına geliyordu.[12])

STAIR'i, kapı açmak, ofisin içinde zımba kapıp getirmek ve bulaşık makinesi boşaltmak gibi çeşitli işleri yaparken gösteren videolar, ekibin çok-ortamlı internet sitesinde izlenebilir: http://stair.stanford.edu/multimedia.php.

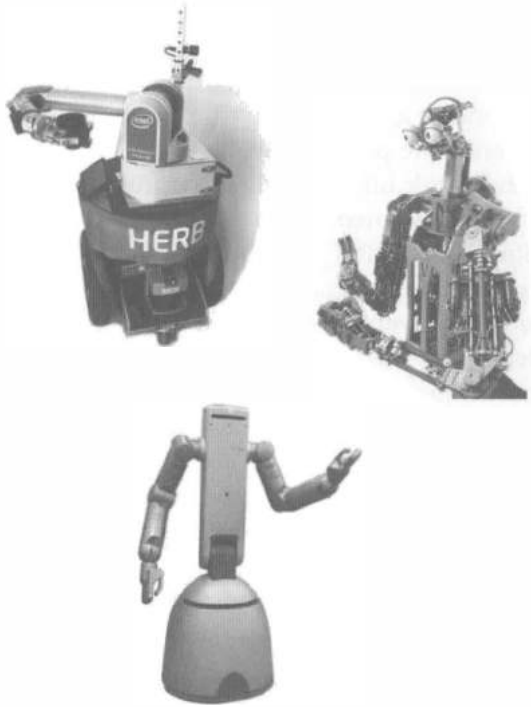
Ng, çalışma arkadaşları ve öğrencileri, şu aşağıdakiler gibi işleri yapabilen robotlar öngörüyor:

- Evde ya da büroda çeşitli malzemeleri getirip götürecek,
- Odayı derleyip toplayacak, çöpü alıp atacak, bulaşık makinesini çalıştıracak,
- Normal bir mutfakta öğün hazırlayacak,
- Kitap rafı montajlarken alet kullanacak.

En sonunda robotlar, "ev ve büro otomasyonunda devrim yapacak ve ev hizmetlerinden yaşlıların bakımına kadar önemli robot uygulamaları olacaktır", deniliyor.

Ng, bu tür robotik araştırmaları yapan tek kişi değil. Pittsburgh Intel Research ve Carnegie Mellon Üniversitesi Robotik Enstitüsü araştırmacılarından oluşan bir ekip, HERB (Home Exploring Robotic Butler) adlı bir robot geliştiriyorlar (bkz. Resim 35.4). HERB, lazerli menzil bulucudan, Segway RMP hareketli gövde, WAM kolu, Barrett Eli ve iki video kameradan meydana geliyor. HERB'i tanıtan bir makaleye göre, nesneleri arayabilir, dağınık ve değişken kapalı mekânlarda

12 Bkz. Ashutosh Saxena, Justin Driemeyer ve Andrew Y. Ng, "Robotic Grasping of Novel Objects Using Vision", *International Journal of Robotics Research (IJRR)*, c. 27, no: 2, Şubat 2008, s. 157-173. Şu internet adresinde: http://ai.Stanford.edu/~asaxena/learninggrasp/IJRR_saxena_etal_roboticgraspingofnovelobjects.pdf.



Resim 35.4 HERB (üstte solda), DOMO (üstte sağda) ve SMARTPAL V (altta). (HERB fotoğrafı, Siddhartha Srinivasa'nın izniyle; DOMO fotoğrafı Aaron Edsinger'in izniyle; SmartPal fotoğrafı, Yaskawa Electric'in izniyle kullanılmıştır.)

yolunu bulmayı öğrenebilir, görme yetisini kullanarak nesneleri tanıyabilir, kapıları ve başka kısıtlı nesneleri elle idare edebilir.[13] HERB'i bu tür işler yaparken gösteren bir video şu internet adresinde: http://pittsburgh.intel-research.net/~ssrin10/HERB09/HERB.wmv.

Başka bir örnek de DOMO'dur (Resim 35.4.); davranış tabanlı, çok eklemli bu robotu, MIT doktora öğrencisi Aaron Edsinger, "insanların ortamlarında elle yapılan faydalı işleri başarmak amacıyla" geliştirmiştir.[14]

13 Siddhartha Srinvasa ve diğ., "HERB: A Home Exploring Robotic Butler"; şu internet adresinde: http://pittsburgh.intel-research.net/~ssrinlO/HERB09/HERB09.pdf.

14 Bkz. DOMO İnternet sayfası: http://people.csail.mit.edu/edsinger/domo.htm. Edsinger'in 2007 doktora tezi, "Robot Manipulation in Human Environments", bu sayfadan indirilebilir.

Japonya'da Yaskawa Electric Şirketi, ev ve büro ortamında iş gören, SMARTPAL V adlı bir hizmet robotu geliştirdi[15] (bkz. Resim 35.4). Ev robotları yapan başka Japon şirketleri de var. Çok amaçlı robot yapma akımına, AAAI, IJCAI ve başka toplulukların düzenlediği yarışmalar yardımcı oluyor.[16]

Önceki bölümlerde, özel işler için tasarlanmış uzman robot sistemlerinden bahsetmiştim, örneğin futbol oynamak ve özerk otomobil sürme gibi. Fakat bunlar bile, YZ algısı, öğrenmesi, planlaması ve plan gerçekleme tekniklerini ileri taşıyan bütünleşik sistemlerdir ve çok faydalı olacaklardır.

B. Zeki Yardımcılar

Şimdi de robotları bırakıp insanlara hareket etmeksizin yardımcı olabilen gövdesiz "etmenlere" göz atalım. Bunlar insanlara, veritabanları, iletişim, internet erişimi ve görev gerçekleştirmede yardımcı olurlar. Bu tür etmenler geliştirmeye çalışan projelerin temsilcisi sayabileceğimiz birkaç projeden bahsedeceğim.[17]

DARPA'nın "PAL Programı", bu çalışmalara hep destek olmuştur. PAL, *Personalized Assistant that Learns*'ın (Öğrenen Kişisel Yardımcı) kısaltmasıdır. Programın internet sitesine göre:[18]

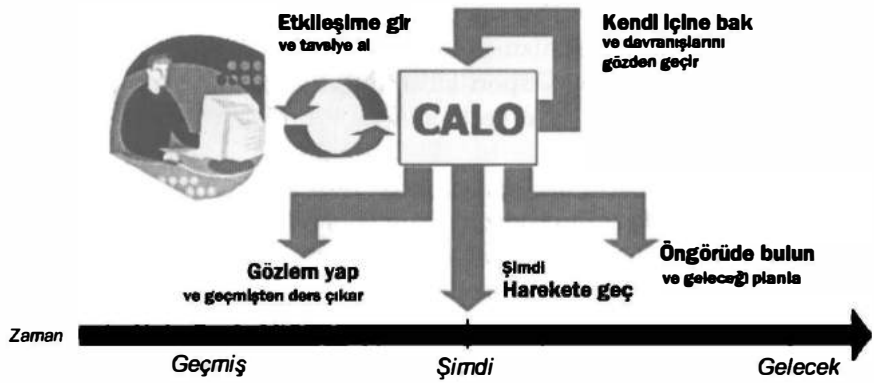
> PAL programının görevi, bilgisayarların insanlara yardımcı olma şeklini kökten iyileştirmektir; bu amaçla, bilişsel sistemleri mümkün kılmaya çalışır, yani akıl yürüten, deneyimlerinden bir şeyler öğrenen, komut alan, yaptıklarını açıklayan, deneyimleri hakkında kafa yoran, sürprizlere güçlü bir şekilde tepki veren bilgisayar sistemleri yaratmak.
>
> ...
>
> Bu program, 1980'lerde DARPA'nın ödenek çıkarttığı Stratejik Bilişim İnisiyatifinden bu yana, bilişsel sistemlerdeki ilk geniş tabanlı araştırma programıdır. O zamandan beri, bilişsel sistemleri mümkün kılacak teknolojilerde önemli gelişmeler kaydedilmiştir, örneğin yapay öğrenme, uslamlama, algı ve çok kipli etkileşim gibi. İşlemciler, bellek, sensörler ve ağlardaki iyileşmeler de, bilişsel sistem araştırmalarının bağlamını kö-

15 Bkz. http://www.yaskawa.co.jp/en/newsrelease/2007/04.htm.

16 "Robot Yarışmalarıyla İlgili Sık Sorulan Sorular" listesi için bkz. http://robots.net/rcfaq.html.

17 *AI Magazine* dergisinin "Karma İnisiyatif Yardımcıları" başlıklı özel sayısında daha fazla projeden bahsediliyor, c. 28, no: 2, Yaz 2007.

18 http://www.darpa.mil/ipto/programs/pal/pal.asp. PAL'la ilgili bir DARPA broşürü şu internet adresinde: http://caloproject.sri.com/PALbrochure.pdf, ve videosu da şu internet adresinde: http://www.darpa.mil/ipto/programs/pal/docs/PAL.wmv.



Resim 35.5 CALO'nun işlevleri. (Karen Myers ve Thomas Garvey'in izniyle kullanılmıştır.)

günden değiştirmiştir. Şimdi, çeşitli alanları, bir araya toplanıp yaygın bir Öğrenen Kişisel Yardımcı uygulama problemine odaklanmayı yüreklendirmenin zamanıdır.

PAL Programının genel şemsiyesi altında geliştirilen sistemlerden biri CALO'dur (Cognitive Assistant that Learns and Organizes [Öğrenen ve Düzenleyen Bilişsel Yardımcı]).[19] CALO, SRI International tarafından yönetiliyor ve Amerikan üniversitelerinden ve şirketlerinden otuzu aşkın katılımcısı var. Bu proje, yapay öğrenme, doğal dil işleme, bilgi gösterimi, insan-bilgisayar etkileşimi, esnek planlama ve davranış araştırmaları alanlarından uzmanları bir araya getiriyor. CALO yazılımı, kullanıcılarıyla etkileşim kurarak ve onlara danışarak bir şeyler öğrenir ve amacı, askerî karar verme sürecinde kullanıcılara yardımcı olmaktır. (CALO ismi, Latince "calonis" sözcüğünden esinlenilmiştir, ki bunun anlamı "askerin hizmetçisi"dir.)

CALO'nun öğrenme ve yardım etme şekli Resim 35.5'te betimleniyor.

CALO projesi kapsamında, üzerinde çalışılmış iki bileşen şunlardır:

- OPI (Organize and Prepare Information [Bilgiyi Düzenle ve Hazırla]), karar verme sürecini ve durum farkındalığını iyileştirmek amacıyla kullanıcı bilgisini düzenler. (OPI hakkında bir video şu internet adresinde: http://caloproject.sri.com/videos/OrganizeInformation.mov.)

19 Daha fazla bilgi için bkz. http://caloproject.sri.com/. CALO projesinin ilk üç yılı, 2009'un ilk yarısında sona erdi. Teknolojiyi gerçek uygulamalara aktarmak için üç yıllık bir proje daha gerektiği öngörülüyor.

- PEXA (Project Execution Assistant [Proje Uygulama Yardımcısı]), belirlenen görevleri yerine getirir, gereksinimleri ve fırsatları öngörür, kullanıcının zamanını ve sorumluluklarını idare eder.[20] (Bununla ilgili bir video şu internet adresinde: http://caloproject.sri.com/videos/TaskMgtMov2.mov.)

Toplu CALO sisteminin tek tek bileşenlerinden birçoğu, genel olarak zeki sistemler araştırmalarına önemli katkılardır. Bu bileşenlerden biri, Oregon State Üniversitesi'nde Profesör Thomas Dietterich ile Xinlong Bao'nun geliştirdikleri Marjinal Olasılık Mimarisidir.[21] Bu mimari, çoklu öğrenme bileşenini bütünleştirip CALO sistemini oluşturmak için birinci dereceden mantık ile olasılıksal grafik modelleri birleştiren bir gösterim olan "Markov Mantığını" kullanır.[22]

CALO'nun başka bir bileşeni de kullanıcı arayüzü ile IRIS (Integrate. Relate. Infer. Share.) denilen bilgi ambarıdır. IRIS'in sürümlerinden biri, açık kaynak uygulaması olarak sunulmuştur. Kendi internet sitesine göre,[23] "IRIS, kullanıcıların, ofisle ilgili bilgiler barındıran nesnelere dair 'kişisel harita' çıkarmasını mümkün kılan, anlamsal bir masaüstü uygulama çerçevesidir. IRIS, bu sürecin otomasyonuna yardımcı olan bir yapay öğrenme düzlemi içerir. 'Pano' görüntüsü, bağlamsal yön bulma ve takvim, internet ve dosya penceresi, e-posta alıcısı ve hazır mesajlaşma alıcısı gibi kapsamlı bir ofis uygulamaları kümesinde ilişki tabanlı yapı sağlar". IRIS'in bu sürümünü, internet sitesinden indirebilirsiniz.

Şimdi de Atlas okyanusunun karşı tarafına geçip başka bir zeki yardımcı projesine göz atalım. Birleşik Krallık'ta Sheffield Üniversitesi Bilgisayar Bilimi Bölümü profesörlerinden Yorick Wilks (1939-), "COMPANIONS" adlı projenin koordinatörüdür. Bu projeye Avrupa Komisyonu destek sağlıyor ve hem Avrupa'dan hem de ABD'den katılımcıları var. Kendi internet sitesine göre bu proje:[24]

20 PEXA'nın teknik bir betimlemesi için bkz. Karen Myers ve diğ., "An Intelligent Personal Assistant for Task and Time Management", *AI Magazine*, c. 28, no: 2, s. 47-61, Yaz 2007.

21 Thomas G. Dietterich ve Xinlong Bao, "Integrating Multiple Learning Components through Markov Logic", *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, s. 622-627. Yeniden baskısı şu internet adresinde: http://web.engr.oregonstate.edu/~tgd/publications/aaai08-ilr.pdf.

22 Matthew Richardson ve Pedro Domingos, "Markov Logic Networks", *Machine Learning*, c. 62, 2006, s. 107-136. Şu internet adresinde: http://www.es.Washington.edu/homes/pedrod/papers/mlj05.pdf.

23 http://www.openiris.org/.

24 http://www.companions-project.org/.

> İnsanların bilgisayarlar ve internetle ilişkileri hakkındaki düşünme şeklini, sanal sohbet edebilecekleri bir "Companion" [yoldaş] yaratarak değiştirmeyi hedefliyor.
>
> Bu, kullanıcıyla uzun süre birlikte kalacak, onunla ilişki kurup seçimlerini ve isteklerini tanıyacak bir etmen ya da "varlık"tır. Kullanıcıyla iletişim kurarken öncelikle konuşmadan faydalanacak ve konuşmaları anlayacak, fakat dokunmaya duyarlı ekran ve sensörler gibi başka teknolojilerden de yararlanacak.

İki prototip sistemin video gösterimleri YouTube'da izlenebilir. COMPANION'lardan biri, kullanıcıyla "sağlık ve zindelik" hakkında sohbet ediyor. Öbürüyse, yaşlı bir vatandaşa sayısal albümünden fotoğraflar gösterip bunlar hakkında onunla sohbet ediyor (bkz. http://www.youtube.com/watch?v=KQSiigSEYhU ve http://www.youtube.com/watch?v=s33_UcGyFSE).

Çok yönlü sohbet etmenleri, insan ilişkilerinin yerini alarak kimi insanlara faydalı olabilir. Fakat, Theodore Roszak ve başkalarının dile getirdiği gibi, bunların hatalı ve aşırı kullanımına karşı temkinli olmak gerekiyor. Birleşik Krallık'ta Birmingham Üniversitesi Bilgisayar Bilimleri Fakültesi'nde Aaron Sloman, sayısal arkadaş geliştirmenin kimi zorlukları hakkında insanı düşündüren bir makale yazmıştır.[25]

C. Okuyarak Öğrenme

Tüm YZ araştırmacıları, zekâ için hem sağduyu bilgisinin hem de uzmanlık gerektiren bilginin kilit olduğunda anlaşır. Bu bilgileri, bilgisayar programlarının kullanabileceği biçimde toplayıp örgütlemek amacıyla çeşitli yaklaşımlar denenmiştir. Cyc gibi projeler, milyonlarca küçük bilgi parçasını mantık cümlesi olarak elle kodlamayı içerir. Yapay öğrenme araştırmaları, bilgi elde etmek için büyük veritabanlarında "madencilik yaparken" istatistik yöntemlerinin kullanılabileceğini göstermiştir.

Üçüncü bir yaklaşım, doğal dil metinleri okuyabilen (ve anlayabilen) programlar inşa etmektir. En sonunda, bu yaklaşımın yandaşlarının iddia ettiği gibi, dünya bilgiyle dolup taşıyor: kitaplar, haberler, internet bilgiden geçilmiyor. (Elbette, pek çok saçmalık da var, fakat

25 Aaron Sloman, "Requirements for Digital Companions: It's Harder Than You Think", Toplumda Yapay Yoldaşlar: Günümüze ve Geleceğe Dair Bakış Açıları Çalıştayı Görüş Makalesi, Düzenleyen Companions Project, Oxford Internet Enstitüsü, 25-26 Ekim 2007. Şu internet adresinde: http://www.cs.bham.ac.uk/research/projects/cogaff/sloman-oii-2007.pdf.

akıllı programlar en sonunda güvenilir kısımları öne çıkarabilir.) Daha önce gördüğümüz üzere, doğal dil metinlerini anlayan bilgisayar, hem sağduyu bilgisine hem de metnin konusuyla ilgili arkaplan bilgisine gereksinim duyar. Dolayısıyla, bu üçüncü yaklaşımın adı olan "Okuyarak Öğrenme" (OÖ), bilgi gösterimi ve uslamlama, ayrıca doğal dil işleme teknolojilerine yaslanıyor.

İnternetten bilgi çıkarmaya yönelik ilk girişimlerden biri, Carnegie Mellon Üniversitesi'nde Profesör Tom M. Mitchell'ın (1951-) önderlik ettiği ekibe aittir. 1999 tarihli bir makalede, "içeriği, Dünya Çapında Ağın içeriğini yansıtan, *bilgisayarlarca anlaşılabilir* bir bilgi tabanını otomatik olarak yaratacaklarını", ileri sürmüşlerdi.[26] Daha sonra, DARPA, okuyarak öğrenme sisteminin verimini belirlemek amacıyla, "Möbius Projesi" denilen iki senelik bir çalışmanın masraflarını üstlendi. Projenin, bir OÖ sistemi prototipini betimleyip değerlendiren sonuç raporu şu sonuca varmıştı: "Okuyarak Öğrenme, hedeflenen bilginin problem çözme yeteneklerinde [yani soru yanıtlamada] istatistiksel olarak anlamlı bir iyileşme sağlamıştır ve büyük bir araştırma çabasıyla birlikte, OÖ'nün genel uygulanışında muzzam ilerleme kaydedilebilir."[27] Bu satırların yazıldığı sırada, Möbius Projesi raporunun sonuçlarına da bakınca, DARPA'nın kısa süre içinde OÖ araştırmalarını desteklemeye başlayacağı anlaşılıyor.[28]

Günümüzde, okuyarak öğrenme ve benzer sorunlar üzerinde çalışan araştırmacılar var. (Bu alanda yapılmış çalışmaların bir örnek kümesi için, Mart 2007'de düzenlenmiş AAAI Bahar Sempozyumundaki konuşmaların çizelgesine bakılabilir; şu internet adresinde: http://www.cs.washington.edu/homes/pjallen/aaaiss07/schedule.htm.) Bu ekiplerden birine, Washington Üniversitesi Turing Merkezi Müdürü Profesör Oren Etzioni (1964-) başkanlık ediyor.[29] Bu merkezin "KnowItAll Projesi",[30] "TextRunner" adlı arama programını geliştirmiştir. Bu program, başka işlerinin yanı sıra, metinden man-

26 Mark Craven ve diğ., "Learning to Construct Knowledge Bases from the World Wide Web", *Artificial Intelligence*, c. 118, no: 1, Nisan 2000., s. 69-113 Ön baskısı için bkz. http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb/overview-aij99.ps.gz.

27 Noah S. Friedland ve diğ., "Project Möbius: A study on the Feasibility of Learning by Reading"; bkz. http://www.noahfriedland.com/uploads/Moebius_Final_Report.pdf.

28 Örneğin bkz. http://www.darpa.mil/ipto/solicit/baa/RFI-08-11.pdf.

29 http://turing.cs.washington.edu/index.htm.

30 Bkz. http://www.cs.washington.edu/research/knowitall/.

tıksal ilişkiler çıkarmaya çalışır ki bunlar, bilgisayarın erişebileceği bilgi tabanını doldurmak (ya da üzerine eklemek) için kullanılabilsin. Etzioni'nin verdiği örneklerden birinde, TextRunner, "EBay aslen Pierre Omidyar tarafından kurulmuştur", cümlesinden, (`Ebay, Kurulmuştur, Pierre-Omidyar`) listesini çıkarmıştır. Bu liste, `Kurulmuştur` yüklemi ile `Ebay` ve `Pierre-Omidyar` konularını barındıran bir mantıksal bağıntıyı yazmanın yoludur. Daha önce bahsettiğim gibi, bildirimsel bilgi gösterimi için kullanılan şemaların çoğunda mantıksal bağıntı kümelerinden faydalanılır. Kendi internet sitesine göre,[31] "TextRunner, 500 milyon yüksek nitelikli internet sayfasından çıkarılmış yüz milyonlarca değerlendirme içinde arama yapar". TextRunner internet sayfası, bu değerlendirmelerden oluşan veritabanını, kaynak gösterdiği internet sayfalarıyla birikte yanıt alacak şekilde sorgulamanıza olanak tanır.[32]

35.2 İNSAN DÜZEYİNDE YAPAY ZEKÂYA DOĞRU

35.2.1 Ödüle Göz Koymak

1995'te yazılmış "Ödüle Göz Koymak" başlıklı makalede, YZ araştırmacılarının, önceki bölümde bahsettiğim türde akıllı alet çalışmalarına ek olarak genel zekâ barındıran sistemler geliştirmek için daha fazla çaba sarf etmeleri gerektiğini söylemiştim. YZ'nin, "hünerli" dediğim sistemleri kurmaya odaklanması gerektiğini belirtmiştim; bu sistemler, tıpkı insanlar gibi, o akıllı aletleri kullanmayı öğrenebilir. Daha genel olarak, bana daima öyle gelmiştir ki, YZ'nin başlıca hedefi, yani aradığımız büyük "ödül", insanların yapabildiği şeylerin çoğunu yapan insan eseri nesneler olmalı; özellikle de "zekâ" gerektiren şeyleri yapabilmeliler. Bu sistemler, kimi YZ araştırmacılarının "İnsan Düzeyinde Yapay Zekâ" (İDYZ) dediği şeyi barındırmalı.

İDYZ, bu alanın kurucularının ve ilk YZ araştırmacılarının pek çoğunun hedefiydi. John McCarthy, 20 Şubat 1947'de Londra Matematik Derneği'nde verdiği seminerde, "insan düzeyinde makine zekâsından bilimsel bağlamda ilk bahseden kişinin Alan Turing ol-

31 http://www.cs.washington.edu/research/textrunner/.

32 TextRunner'la ilgili teknik ayrıntılar için bkz. Michele Banko ve Oren Etzioni, "The Trade-offs between Open and Traditional Relation Extraction", *Proceedings of ACL-08: Human Language Technologies*, Association for Computational Linguistics, Haziran 2008, s. 28-36. Şu internet adresinde: http://www.aclweb.org/anthology-new/P/P08/P08-1004.pdf.

duğunu" söylemiştir.[33] Turing, 1950 tarihli "Hesaplama Makineleri ve Zekâ" başlıklı makalesinde, insan düzeyinde zekânın makineleşmesini açık bir hedef olarak ortaya koymuştu. İleride, 1956 Dartmouth Yaz Araştırmaları için sunduğu öneride John McCarthy şöyle yazacaktı: "Araştırmalar, öğrenme ya da herhangi bir zekâ özelliğinin, prensipte, bir makinenin taklit etmesine imkân tanıyacak ölçüde betimlenebileceği varsayımı temelinde ilerleyecektir."

Newell ile Simon'un "Genel Problem Çözücü" (GPÇ), özellikle İDYZ'ye ulaşmayı hedeflemişti, üstelik zeki davranışı hem anlamayı hem de mekanikleştirmeyi ilgilendiren problemler üzerinde çalışmayı sürdürmüşlerdir. 1957'deki bir konuşmada Simon, "artık dünyada düşünen, öğrenen, yaratan makineler bulunuyor; dahası, bu işleri yapma becerileri hızla artacak ta ki görünür bir gelecekte, başa çıkabildikleri sorunların yelpazesi, insan zihninin uygulanabildiği sorunların yelpazesiyle denk olana kadar". Marvin Minsky, 1961 tarihli "Yapay Zekâya Doğru" başlıklı makalesinde, "problem çözen zeki makinelerin kuvvetle etkileyeceği, hatta büyük olasılıkla hüküm süreceği bir çağın eşiğindeyiz", diye yazmıştır. Edward Feigenbaum, 2003 tarihli makalesinde şu sonuca varıyordu: "Hesaplamalı Zekâ, bilgisayar bilimimizin açık kaderi, amacı, hedefi, son sınırıdır. Başka bilim dallarına kıyasla, bilgisayar bilimimizin kavramları ve yöntemleri, varoluşumuzun en büyük sırlarından birini, zekânın doğasını çözme ve anlama arayışında merkezî konumdadır. Geleceğin bilgisayar bilimcisi kuşakları, bu büyük maceranın doğurduğu zorluklardan ve meydan okumalardan ilham almalı."[34]

Kimi insanlar, İDYZ'nin üstün insan düzeyinde zekâya karşılık geldiğini belirtir. Henüz 1965'te İngiliz istatistikçi (ve Alan Turing'in çalışma arkadaşı) I. J. Good şöyle yazmıştı:[35]

> Aşırı-zeki makineleri, ne kadar zeki olursa olsun herhangi bir insanın düşünsel etkinliklerini uzak ara geçen makineler olarak tanımlayalım. Makine tasarımı, bu düşünsel etkinliklerden biri olduğu için, aşırı-zeki bir makine daha da iyi makineler tasarlayabilir; hiç şüphesiz bir "zekâ

33 John McCarthy, "From Here to Human-Level AI", *Artificial Intelligence,* c. 171, no: 18, Aralık 2007, s. 1174-1182. Ön baskısı şu internet adresinde: http://www-formal.stanford.edu/jmc/human/human.html.

34 Edward A. Feigenbaum, "Some Challenges and Grand Challenges for Computational Intelligence", *Journal of the ACM,* c. 50, no: 1, Ocak 2003, s. 32-40.

35 Irving J. Good, "Speculations Concerning the First Ultraintelligent Machine", *Advances in Computing* içinde, Franz L. Alt ve Morris Rubinoff (ed.), c. 6, 1965, s. 31-88.

> patlaması" söz konusu olacaktır ve insan zekâsı çok geride kalacaktır. Dolayısıyla, ilk aşırı-zeki makine, insanın yapma gereği duyacağı son icat olacak, elbette makinenin, kendisini nasıl denetim altında tutacağımızı bize söyleyecek kadar uysal olacağını varsayıyoruz.

Bir zamanlar YZ'yi eleştiren Jack Schwartz, 1987'de şöyle yazmıştır:[36]

> Eğer yapay zekâlar yaratılabilirse, ilk başarıların önemli matematiksel, bilimsel ya da mühendislik alternatiflerini, insanların becerisini çok aşan hızlarla keşfedebilecek ya da aynı muazzam hızla planlar yaratıp bunları yürürlüğe sokabilecek yapay zekâların gerçekleşmesine hemen yol açamayacağına inanmak için fazla sebep yok. İnsanoğlunun, tüm yüksek zekâ biçimleri üzerinde aşağı yukarı kurduğu tekel, bu gezegenin geçmişinde insan varlığının en temel olgularından biri olagelmiştir, dolayısıyla bu tür gelişmeler, açıkça yeni bir iktisat, yeni bir toplumbilim, yeni bir tarih yaratacaktır.

Makinelerin gitgide zekileşmesi fikri, matematikçi, bilgisayar bilimcisi ve bilim kurgu yazarı Vernor Vinge'in (1944-), 2030'a gelindiğinde üstün bir bilgisayar zekâsının ortaya çıkmış olacağını tahmin etmeye yöneltmişti. Vinge, bu olaya "tekillik" demiştir, yani zamanın bir noktasında, teknolojik ilerlemenin sürati hayal edilemez ölçüde artar. Tekillik hakkındaki bir makalede Vinge şöyle yazmıştır: "İnsan zekâsını aşan bir zekâ, ilerlemelere yön verdiğinde, bu ilerleme çok daha hızlı olacaktır. Aslında, bizzat ilerlemenin, daha da kısa bir zaman ölçeğinde, daha zeki varlıklar yaratmayı içermemesi için hiç bir sebep yok."[37] Vinge, "büyük bilgisayar ağlarının (ve bunların kullanıcılarının), insandan üstün zeki bir varlık olarak 'uyanacağı'" günü öngörmüştür.

Mucit ve YZ araştırmacısı Ray Kurzweil (1948-), *The Singularity is Near* [Tekillik Yakın] başlıklı kitabıyla bu tekillik fikrini halka tanıtıp yaygınlaştırmış oldu.[38] Çoğunlukla, teknolojik ilerleme *hızının* geomet-

36 Jacob Schwartz, "Limits of Artificial Intelligence", *Encyclopedia of Artificial Intelligence* içinde, Stuart C. Shapiro ve David Eckroth (ed.), c. 1, New York: John Wiley and Sons, Inc., 1987, s. 488-503.

37 Vernor Vinge, "The Coming Technological Singularity: How to Survive in the Post-Human Era"; şu internet adresinde: http://www-rohan.sdsu.edu/faculty/vinge/misc/singularity.html. Bu denemenin özgün sürümü, NASA Lewis Araştırma Merkezi ile Ohio Havacılık-Uzay Enstitüsü'nün hamilik ettiği, 30-31 Mart 1993 tarihli VISION-21 Sempozyumunda sunulmuştur. Hafif değiştirilmiş bir sürümü, *Whole Earth Review* adlı derginin 1993 Kış sayısında basılmıştır. O sürüm (yazarın birtakım açıklamalarıyla birlikte) şu internet adresinde: http://wholeearth.com/ArticleBin/111-3.pdf.

38 Ray Kurzweil, *The Singularity Is Near: When Humans Transcend Biology*, New York:

rik olarak arttığı (ve artmayı sürdüreceği) fikrine yaslanan Kurzweil, 2010'dan itibaren 2045'teki "Tekilliğe" ve ötesine varana dek çeşitli on yıllık dilimler içinde teknolojinin üreteceği şeyler hakkında pek çok tahminde bulunmuştur. Tahminlerinden bazıları görece tuhaf görünür ve tekillik fikrinin bütünü epey alaya maruz kalmıştır, fakat bu konuda kendi kararınızı kendiniz verin isterim. (Tahminlerinden biri şöyle: 2045'e gelindiğinde, "makineler, kendi kendilerini geliştirdikleri bir "kaçış tepkimesi"ne girecek ve her yeni YZ nesli daha hızlı ortaya çıkacak. Bu noktadan itibaren, teknolojik ilerleme bir patlama halini alır, makinelerin denetimi altına girer, dolayısıyla doğru olarak tahmin edilemez.") http://singularity.com/themovie adresindeki internet sayfası, *The Singularity is Near* kitabının filminin 2009'un ikinci yarısında gösterime gireceğini söylüyor (Bu filmde, Kurzweil ve başka fütüristler ile bilgisayar bilimcileri boy gösterecek). Tekillik hakkında kimi yorum yapanlar, *IEEE Spectrum* dergisinin Haziran 2008 tarihli özel sayısına makaleler yazmıştır. Bu derginin internet sayfası, makalelere ve yardımcı metinlere link de veriyor.[39]

2004'te, "bu acil zorlukla boğuşmak ve hem fırsatı hem de tehlikeyi göğüslemek amacıyla" Yapay Zekâ Tekillik Enstitüsü (SIAI) kuruldu.[40] Enstitünün Araştırma Müdürü Ben Goertzel, aynı zamanda "Genel Yapay Zekâ Araştırma Enstitüsü" (AGIRI) adlı kurumun da başkanıdır; bu kurumun "görevi, hem güçlü hem de ahlaken olumlu Genel Yapay Zekâ yaratılmasını teşvik etmektir."[41] AGIRI, konferanslar ve çalıştaylar tertipler, bazı açık kaynak projelerini yönetir. AGIRI, "genel yapay zekâ" (GYZ) terimini, benim İDYZ'yi kullandığım anlama az çok benzer şekilde kullanır. AGIRI'nin internet sitelerinden birine göre, "bu terim, araştırılan sistemlerde arzulanan becerilerin 'genel' doğasını vurgulamak amacıyla kullanılır; epey uzmanlaşmış 'zeki' beceriler barındıran sistemlere odaklanan ana akım Yapay Zekâ (YZ) çalışmalarından farkı budur."

Bu bölümü ve kitabı, İDYZ'den, bunun ne olduğundan, bunu savunan ya da buna karşı çıkan savlardan, olası sonuçlarından, buna ulaşmak üzere ileri sürülmüş yöntemlerden bahsederek bitireceğim.

Viking Press, 2005.

39 Bkz. http://spectrum.ieee.org/singularity.

40 Bkz. http://singinst.org/.

41 http://www.agiri.org/wiki/Artificial_General_Intelligence_Research_Institute.

35.2.2 Anlaşmazlıklar

İDYZ (ve ötesi), artık her ne halt ise, hâlâ pek çok YZ araştırmacısının hedefidir, gerçi buna ulaşmamıza henüz çok var. Aslında, İDYZ'nin ne olduğu konusunda da anlaşmazlıklar sürüyor. Örneğin Turing testi gibi çeşitli testleri geçen programlar elde ettiğimizde buna ulaşmış olacak mıyız? Florida'nın Pensacola kentindeki İnsan ve Makine Bilişsel Yetileri Enstitüsü'nden Pat Hayes ve Ken Ford, YZ'nin ilerleyişinin ölçüsü olarak Turing testi kullanmaya, çeşitli uygulamalar ve yöntemler zeminine dayanarak karşı çıkanlar arasındadır.[42]

İDYZ'yi tanımlamaya yardımcı olabilecek başka testler de önerilmiştir. Örneğin, 2005 tarihli bir makalede, "istihdam testi" dediğim bir test önermiştim. O makalede şöyle yazmıştım: Testi geçmek için, "programlar, insanların yaptığı sıradan işleri yapabilmeli. Böylece, insan düzeyinde YZ'ye doğru ilerleyiş, makinelerin makul oranda yapabildiği işlerin kesitiyle ölçülebilir". Aklımda, insanların yaptığı her türlü iş vardı; vasıflı işlerden vasıfsız işlere, idari işlere ve büro işlerine kadar.[43] Bana göre, İDYZ'ye ulaşmak, (en azından) insanlara yapmaları için para ödediğimiz işleri yapabilecek insan eseri nesnelerin nasıl inşa edileceğini bilmeyi gerektirir.

Hayes ve Ford, İDYZ'nin, YZ araştırmalarında hedef sayılması fikrine karşı çıkıyor.[44] Öncelikle, YZ programlarının şimdiden pek çok alanda insanların performansını aştığı gerçeğine işaret ediyorlar. Dolayısıyla, İDYZ'ye ulaşmak, bazı bakımlardan fazlasıyla alçak gönüllü bir hedef olabilir. Dahası, nasıl uçaklar kuşları taklit etmeksizin uçuyorsa, YZ'nin de insanları taklit etmesi şart değil. "İnsan kıtlığı olmadığına ve daha fazla insan üretmenin ispatlanmış yollarını bildiğimize göre", diyen Hayes ve Ford, "insani bilişsel yetiler bakımından da kıtlık çekilmediğini", belirtirler; bana kalırsa, insanın tüm bilişsel yetilerini otomatik hale getirmeye lüzum olmadığını ima ediyorlar.

Evet, bazı işleri insanlardan daha iyi beceren YZ sistemleri şimdiden var olsa da, nasıl otomatik hale getireceğimizi bilmediğimiz çok

42 Patrick Hayes ve Kenneth Ford, "Turing Test Considered Harmful", *Proceedings of IJCAI-95*, c. 1, s. 972-997, 1995. Şu internet adresinde: http://dli.iiit.ac.in/ijcai/IJCAI-95-VOL%201/pdf/125.pdf.

43 Nils J. Nilsson, "Human-Level Artificial Intelligence? Be Serious!", *AI Magazine*, c. 26, no: 4, Kış 2005, s. 68-75. Şu internet adresinde: http://ai.Stanford.edu/~nilsson/OnlinePubs-Nils/GeneralEssays/AIMag26-04-HLAI.pdf.

44 Örneğin bkz. Kenneth Ford ve Patrick Hayes, "On Computational Wings: Rethinking the Goals of Artificial Intelligence", *Scientific American Presents*, c. 9, no: 4, 1998, s. 78-83.

ama pek çok insani bilişsel yeti var. Bence, YZ araştırmacılarının bu yetileri otomatikleştirme mücadelesi iki sebepten ötürü sürecek. Bir tanesi iktisadidir: Günün birinde, (insanlar ne kadar kalabalık olursa olsun) daha pahalıya gelecek insan çalışanlar yerine YZ sistemlerini çalıştırma beklentisi, insanlara para ödeyerek yaptırılan işlerin altından kalkacak varlıklar inşa etmeye yönelik güçlü ve dayanılmaz dürtüyü besleyecektir. Öteki sebep bilimseldir: İnsan beyninin nasıl işlediğini anlama çabası içinde insanlar, pek çok beyin işlevinin hesaplamalı modellerini inşa etmeyi sürdürecektir. Dolayısıyla, İDYZ'nin kimi sürümlerine ulaşmanın, YZ'nin uzun vadeli hedeflerinden biri olmayı sürdüreceğini sanıyorum.

İDYZ'yi tanımlamak zor olsa da, buna benzer bir şeye ulaşacağımız öngörüsüne bağlı bir para ödülü söz konusu. 2002'de Mitchell Kapor, "2029'a gelindiğinde hiçbir makinenin ya da "makine zekâsının" Turing testini geçmiş olmayacağına" dair 20.000 dolarlık bir bahis ortaya koydu. Kapor, Lotus 1-2-3'ün tasarımcısı, Lotus Development Corporation'ın kurucusu ve Elektronik Cephe Vakfı'nın eş-kurucusudur. Long Now Vakfı, bu bahsi "uzun süreli bahisler" listesine koymuştur: http://www.longbets.org./ Ray Kurzweil, bu bahse girmeyi kabul etti. Hem Kapor hem de Kurzweil, kendi görüşlerine dair savlarını belirtmiştir ve bu savlar, bahsin ayrıntılı koşullarıyla birlikte şu internet adresinde bulunabilir: http://www.longbets.org/1#terms.

35.2.3 Buna Nasıl Ulaşırız?

İDYZ'nin, YZ'nin hedeflerinden biri olduğunu varsayarsak buna nasıl ulaşırız? Şimdiye kadar geliştirilmiş teknolojik araçlar doğru bir şekilde kullanılıp birleştirilse insan düzeyinde zekâ yaratılabilir mi? Yapay öğrenme, sinir ağları, grafik modeller, benzetimsel evrim, bilgi gösterimi, uslamlama, sezgisel arama, doğal dil işleme, davranışsal mekanizmalar ve algılama (özellikle görü) alanlarında araştırmaları sürdürmek bizi önünde sonunda hedefe ulaştırır mı? Yoksa bütünüyle farklı bir şeye mi gereksinim var?

John McCarthy, İDYZ'ye ulaşma yolunda iki yaklaşımdan bahseder. Bir tanesi, insan zekâsının işleyişini taklit etmektir; McCarthy'nin yazdığı gibi: "İnsan beynini, işlevini taklit edecek kadar iyi anlamak... psikolojide ve sinir fizyolojisinde kuramsal ve deneysel başarı ister". Öbür yaklaşım, insanın zeki davranışlarını taklit eden programlar yazmaktır, ki YZ araştırmacıları büyük oranda bunu yapmaya çalışmıştır. McCarthy, "bu, içlerinden birine karar verme meselesi değildir,

zira ikisi de en sonunda başarıya erecektir, dolayısıyla bu daha çok bir yarışa benzer", demiştir.[45]

Peki, yarışmacılar hedefe hemen varmaya mı çalışmalı, yoksa aşama aşama mı ilerlemeli? Bence ikinci yaklaşım doğru. Marvin Minsky, 1961'de yayımlanmış "Yapay Zekâya Doğru" başlıklı makalesinde şöyle yazmıştı: "İnancım odur ki hiçbir öğrenme ya da örüntü tanıma şemasının çok genel bir faydası olamaz, meğer ki geçmiş sonuçların özçağrılı ya da en azından hiyerarşik bir kullanımı temin edilmiş olsun". Öğrenmenin ve örüntü tanımanın yanına, zekânın başka yönlerini de gönül rahatlığıyla ekleyebilirdi. Şunu da söylemiştir:[46]

> Bir öğrenme sistemini, makul bir dereceyle gitgide zorlaşan bir dizi problemle hazırlamadığımız takdirde, çok zorlu problemlerle başa çıkmasını bekleyemeyiz. İlk problem, başlangıçtaki kaynaklarla makul sürede çözülebilecek nitelikte olmalı. İkinci problem, ilk problemde geliştirilmiş olan yöntemlerin basit ve erişilebilir birleşimlerinin akılcı şekilde kullanılmasıyla yine makul sürede ulaşılacak bir çözüm içermeli, vesaire. Yeterli "alıştırma dizisi" kullanımını içeren bu yaklaşım seçenekleri, 1) ya başlangıçta verilen ileri kaynaklardır ya da 2) belki de organik evrim geçmişinde bulunabilecek harika keşif süreçleridir.

Bence Minsky tamamen haklıydı. "Başlangıçta verilen ileri kaynak" seçeneğini deneyip ve "organik evrimi" taklit edip durduk. Bu yaklaşımlar, akıllı araçları ve başka yararlı programları yarattı ama henüz İDYZ'ye ulaşamadı. Peki, "gitgide zorlaşan bir dizi problem" yaklaşımına ne demeli? Bu strateji de önerilmiştir ve ciddi olarak gözden geçirilmeyi hak ediyor.

Alan Turing, 1950 tarihli makalesinde, "insan zihnini taklit etmeye çalışan bir program üretmek yerine, çocuk zihnini taklit eden bir program neden yaratmayalım? Eğer bu, doğru düzgün bir eğitime bağlı tutulursa, insan beyni elde edilir", yazmıştır. Burada bahsedilen "doğru düzgün eğitim", Misky'nin "dereceli problem dizilerinin" muadili oluyor.

Aşamalı yaklaşım aynı zamanda, Rodney Brooks'un hayata geçirildiğini görmek istediği YZ becerileri listesinde de yansıtılıyor. Brooks, bu becerilerin, "evrimsel bir süreç aracılığıyla insanlarda daha yüksek

45 Alıntılar, John McCarthy'nin eserinden, *a.g.e.*

46 Marvin Minsky, "Steps toward Artificial Intelligence", *Proceedings of the IRE,* c. 49, no: 1, Ocak 1961, s. 8-30. İnternet versiyonu şu adreste: http://web.media.mit.edu/~minsky/papers/steps.html.

zekâ düzeylerinin ortaya çıkışının temeli" olabileceğini söyler. Söz konusu liste aşağıdaki gibidir:[47]

- İki yaşındaki çocuğun nesne tanıma becerileri
- Dört yaşındaki çocuğun dil becerileri
- Altı yaşındaki çocuğun el becerileri
- Sekiz yaşındaki çocuğun toplumsal anlayışı

Brooks, bilgisayar sistemlerinde henüz bu becerilerin olmadığına işaret eder. Bence, bu becerileri elde etmeye yönelik çalışmalar, İDYZ'ye doğru önemli, belki de olmazsa olmaz adımlar oluşturacaktır. Bu beceriler sayesinde YZ, makineleri, insanları eğittiğimiz biçimde eğitmeyi salık veren Turing stratejisini hayata geçirebilir. Fakat bu becerilere ulaşmak çok zor olacaktır.

Bana kalırsa Brooks'un aklında, insan davranışlarını, yani bu örnekte çocuk davranışlarını taklit ederek bu becerilere ulaşmak vardı; bu, McCarthy'nin önerdikleri arasında ikinci yaklaşımdır.

Öteki yaklaşıma, yani "insan beynini, işlevini taklit edecek kadar iyi anlama" yaklaşımına gelince, bu problem üzerine de çalışmalar gerçekleştirilmiştir. Bazı bilgisayar bilimcileri, beyni açıklamak amacıyla, YZ araştırmacılarının aşina olduğu kavramları kullanmaya kalkışır. Mumford, Hinton, Hawkins, Dean ve bunların çeşitli çalışma arkadaşlarının önermiş olduğu, hiyerarşik korteks modellerinden daha önce bahsetmiştim. Örneğin piksel dizilişlerinden gelen ilkel duyu girdilerinin özellikleri üzerine kurulan ve daha da soyut temeller üzerinde yükselen bu modeller, görüntüleri, boyutundan, çevirisinden ve yönleniminden bağımsız olarak sınıflandırmayı öğrenebilir. Ancak bildiğim kadarıyla, bu modelleri, algı çalışmalarının ötesinde kullanan olmamıştır. Acaba bunlar, anlamayı, uslamlamayı, planlamayı ve eylem seçmeyi öğrenebilir mi? Benim "üçlü kule mimarimin" (bkz. s. 578) bakış açısından, beynin, algılama ve modelleme kulelerini nasıl hayata geçirebileceği sorunuyla boğuşulmuştur, ama eylem kulesiyle henüz bir şey denenmiş değil. Bu korteks modellerinden biri, örneğin robot denetleyebilir mi?

California, San Jose'deki IBM Almaden Araştırma Merkezi'nin bilişsel hesaplama müdürü Dharmendra Modha (1969-),[48] "aşağıdan yukarı" yönelen bir yaklaşım benimseyenler arasındadır. 2007'de

47 Rodney Brooks, "I, Rodney Brooks, Am a Robot", *IEEE Spectrum*, c. 45, no: 6, Haziran 2008, s. 62-67. Şu internet adresinde: http://spectrum.ieee.org/jun08/6307.

48 http://www.almaden.ibm.com/cs/people/dmodha/.

Modha'nın ekibi, tarihsel bir deney yaptı; bu deneyde, 32.768 işlemcili ve 8 TB ana bellekli BlueGene süperbilgisayarını kullanarak neredeyse gerçek zamanlı işlev gören, korteksin sıçan ölçeğindeki modelinin bilgisayar benzetimini oluşturdular (sinyal gönderen 55 milyon sinir hücresiyle ve 448 milyon sinapsla, ayrıca sinyal gönderme zamanına bağlı esneklikle birlikte). Modha, DARPA'nın "Sinir Hücresi Biçimli, Uyumsal, Esnek, Ölçeklenebilir Elektronik Sistemler" (SyNAPSE) geliştirme projesinin baş araştırmacısıdır, yani sade bir dille söylemek gerekirse, bu projenin hedefi, 100 milyon sinir hücresinin eylemlerini taklit edecek bir makine yapmaktır. Bu rakam, sıçan beynindeki sinir hücresi sayısının iki katı ama insan beynindeki sinir hücresi sayısının %0,25'i. Yukarıdan aşağıya ve aşağıdan yukarıya yaklaşımlarının ortada bir yerde buluşacağını umabiliriz.

Bu bölümde daha önce bahsetmiş olduğum pek çok laboratuvar uğraşına, "İDYZ-dostu" diyorum; yani bu uğraşların, İDYZ sistemlerinin gereksinim duyacağı teknolojileri geliştirmesi olası. Bunlardan biri, STAIR projesi, henüz 1996'da ortaya koyduğum zorlu bir problemi ele alıyor:[49]

> ... tipik bir iş merkezinde kahyalık ve getir götür işleri yapacak bir robot üretmek; özellikle robotlara uygun olarak donatılmamış bir iş merkezinde hem de... Robot, var olan icracı/sensör donanımıyla insanların ondan yapmasını "makul olarak" beklediği her işin altından kalkabilmeli (ya da komutlarla ve eğitimle yapmayı öğrenebilmeli ama fabrika çıkışından sonra bilgisayarının açıkça programlamasıyla değil).
>
> ...
>
> Bu iddianın ikinci kısmı, robotun, yeniden programlama için fabrikaya geri gönderilmeksizin bir sene boyunca iş başında ve işlevsel kalması zorunluluğudur.
>
> ...
>
> Robotu, ona verilebilecek her türlü görevi öngören programlarla donatıp iş merkezine öyle göndermek, bana kalırsa robotu yapanların üstesinden gelemeyeceği bir iştir. Bence robot, plan yapabilmeli ve programcılarının öngöremediği ama (robotun sadece sensörlerini ve icracı birimlerini bilen) iş merkezi sakinlerinin ondan yapmasını bekleyebileceği işleri yapmayı öğrenebilmeli.

49 Bart Selman ve diğ., "Challenge Problems for Artificial Intelligence", *Proceedings of AAAI-96,* Thirteenth National Conference on Artificial Intelligence, Menlo Park, CA: AAAI Press, 1996, s. 1340-1345. Şu internet adresinde: ftp://ftp.research.microsoft.com/pub/ejh/selman.ps.

İDYZ'nin bir hedef olarak (hatta tanımının) uygunluğuna dair çeşitli endişelerden bağımsız olarak, aslında buna ulaşacağımızı sanıyorum. Ne zaman olacağını tahmin etmeyeceğim ama bu yüzyıl içinde olacaktır. Peki, buna ulaştığımızda ne olacak? Şimdi de bu konuya değiniyorum.

35.2.4 İDYZ'nin Kimi Olası Sonuçları

Diyelim ki günün birinde üstün akıllı makineler toplumumuzun bir parçası oldu ve bize yardımcı oluyor, bizi eğlendiriyor, işlerimizin çoğunu bizim yerimize yapıyorlar. Çeşitli biçimlere girebiliyorlar: insansı robotlar ya da başka robot çeşitleri, Dünya Çapında Ağ'da "varlık"lar, evimizde ve dizüstü bilgisayarlarımızda yazılım ve büyük olasılıkla, kendi düşünsel işlevlerimize yardımcı olacak özel implantlar. O zaman geldiğinde, bu makineleri bizlerle ve birbirleriyle savaşmasını olanaksız kılacak bir toplumsal düzen kurmuş olacak mıyız? Hatta makineler için, sadece toplumsal açıdan kabul gören hedeflere hizmet etmenin ne anlama geldiğini tanımlamış olacak mıyız?

Bu endişeleri hem bilgisayar bilimcileri hem de hümanistler taşıyor. Jack Schwartz, 1987'de şöyle yazmıştı:

> ... Pek çok hümanist düşünür, kamunun genişlemesinden duyduğu biçimsiz huzursuzluğu ifade ediyor. Zaman zaman korkunç kırılgan görünen insan toplumu dokusunun, başlıca meyvelerini vermeye başlamış olan YZ araştırmalarının devreye sokacağı muazzam hızlı teknolojik değişiklikler yüzünden dağılacağından korkuyorlar. Örneğin, müstakbel diktatörlerin, küçük merkezî oligarkların ya da saldırgan ulusların, toplum üzerinde iktidarlarını kurmak amacıyla bu teknolojiden faydalanıp, az insanın yer aldığı ve tüm geleneksel insani ya da insancıl kaygılara bütünüyle kayıtsız kalan robot ordular ve kolluk kuvvetleri kurduğunu düşünmek mümkün. Bu kabusu bir yana bıraksak bile, daha incelikli birtakım zararlı etkilerinden korkulabilir, örneğin her açıdan yetkin başa çıkılmaz mekanizmaların varlığı yüzünden tüm çalışma ve uğraşıp didinme baskıları, hatta sebepleri ya da fırsatları rafa kalktığında insan toplumunun hızla çöküp kendi kendini yok eden katışıksız bir hazcılığa gömülmesi gibi. İnsanın, eşsizliğine dair hissi de kesinlikle yara alacaktır ve kendi gözüne, ilkel bir hayvandan biraz fazlası, sadece geçici keyifler alan bir varlık olarak görünecektir.[50]

YZ araştırmacılarından Stephen M. Ormohundro (1959-), bu korkuları yatıştırmak amacıyla, Öz-Farkındalık Sistemlerini kurdu; bu

50 Jacob Schwartz, *a.g.e.*

kurum, "belirmekte olan teknolojilere bilgelik katmaya adanmıştı."[51] Omohundro, YZ sistemleri geliştirirken "çok dikkatli olmak zorundayız", der. Bunun sebebi, YZ sistemlerinin, tasarım gereği çeşitli hedeflerinin ve güdülerinin olmasıdır. Bunlar arasında, kendini geliştirme ve akılcı olma hedefleri de bulunur. YZ sistemleri, bu hedeflere ve insanların belirlediği hedeflere en etkin şekilde ulaşmaya kalkışacaktır. Azami verimi sağlamak için, kendilerini koruma ve kaynakları elde etme dürtüleri olacak. Bu özellikler bize, *2001: Uzay Yolculuğu* adlı kitaptaki ve filmdeki uzay gemisi robotu HAL-9000'i anımsatıyor. Omohundro, teknolojilerimize sadece zekâ değil "bilgelik" de katmamızı istiyor. Bununla kastettiği şey, teknolojilere "insani değerleri yerleştirmektir, örneğin insan haklarına ve mülkiyet haklarına saygı göstermek ve başka varlıklara şefkat göstermek gibi". Şöyle der: "Bu tür değerleri en başta yerleştirmemiz mutlak önem taşıyor, yoksa çok güçlü olan ama bizim değerlerimizi umursamayan sistemler vücut bulur."[52] Bence Omohundro geçerli kaygılar gündeme getiriyor, fakat YZ sistemlerine onun bahsettiği gibi insani değerler yerleştireceksek, önce "insani değerler" lafıyla ne kastedildiği üzerinde anlaşmamız gerekiyor. Değerler hakkındaki farklı görüşlerimizin çoğunlukla savaş doğurduğuna bakarsak, bu oldukça zor bir iş.

Georgia Teknoloji Enstitüsü'nden Profesör Ronald Arkin (1949-), hareketli robotlar konusunda onca yıl çalıştıktan sonra, askerî robotların kullanımıyla ilgili etik meselelere eğilmiştir. *Governing Lethal Behavior in Autonomous Robots* [Özerk Robotlarda Ölümcül Davranışlara Hükmetmek] başlıklı kitabı, robotlarda "yapay vicdan" programlamayı anlatır.[53] Bu tür robotların, günümüzde insanlara kıyasla, çarpışma alanlarında daha ahlaklı davranacağını savunur. Elbette pek çok kişi, insanlar için de robotlar için de, savaş alanında *olmanın* bile gayriahlaki olduğunu düşünüyor.

İDYZ olasılığı, başka ilginç sorular da doğuruyor. Bunların "hakları" olacak mı? Kendi mülkleri olabilir mi? Medenî hukuk ya da suç hukuku işlemlerinde katılımcı olabilirler mi? Edebiyat, müzik, sanat eserleri yaratabilecekler mi? Duyguları olacak mı ya da acı veya neşe

51 http://selfawaresystems.com/.

52 Stephen M. Omohundro, "Self-Improving AI and the Future of Computation", 1 Kasım 2007'de Stanford'da verilen seminer; metin hali şu internet adresinde: http://selfawaresystems.com/2007/11/01/standford-computer-systems-colloquium-self-improving-ai-and-the-future-of-computing/.

53 Ronald C. Arkin, *Governing Lethal Behavior in Autonomous Robots*, Boca Raton, FL: CRC Press, 2008.

hissedecekler mi? Bilinçleri olacak mı? İnsanlar, bunların kimi sürümlerine duygusal olarak bağlanacak mı (ya da tam tersi)? Bunların hepsi heyecan verici sorulardır ve pek çok insan bu sorular hakkında yazılar yazmıştır.

Toplumun, zeki makinelere haklar ve başka yasal güçler vermesi biz insanlara bağlı olacak. En sonunda, insan dışı kimi varlıkların (şirketler gibi) belirli hakları ve yükümlülükleri olmasına ve bazı insanların (çocuklar gibi) belirli hakları olmamasına bizler karar verdik. Zeki makinelerle ilgili de benzer kararlar vermek zorunda kalacağız.

Peki, yaratıcılık meselesine ne demeli? İşte size bir örnek. Santa Cruz'daki California Üniversitesi'nin fahri profesörlerinden David Cope, "Müzikal Zekâ Deneyleri" dediği bir dizi program geliştirmiştir.[54] Bu programlar, bir bestecinin tarzını çözümleyip aynı tarza sahip bütünüyle yeni besteler yaratmak amacıyla özel "yeniden birleştirme" yöntemlerinden faydalanır. Cope, yüzlerce bestecinin tarzında besteler yapmak için bu yazılımı kullanmıştır. Cope'un internet sitelerinden birinden, bilgisayarla yaratılmış farklı Bach tarzı korallerin 5.000 MIDI dosyasından herhangi birini indirebilirsiniz.[55] Caz müziği (özellikle Scott Joplin tarzı) dinlemeye daha eğilimli olanlar şu internet adresine bakabilir: ftp://arts.ucsc.edu/pub/cope/joplin.mp3. Cope'a göre, "doğal bir evrimsel ve ve yaratıcı süreç olarak yeniden birleştirmeler her yerde karşımıza çıkıyor."[56] Bunun kimi sürümleri, ister edebiyatta, ister sanatta, ister müzikte olsun, bütün yaratıcılığın temelini oluşturabilir.

Marvin Minsky, yapay zekâ alanındaki onca yıllık çalışmasını temel alarak, duygular ve başka zihinsel görüngüler hakkında mükemmel bir kitap kaleme almıştır.[57] Minsky, kitabın Sunuş kısmında, "başlıca dört 'duygusal durumumuzdan' her biri, [beynin] belirli kısımlarının etkinleşmesi ve başka kısımlarının etkinliğini kesmesiyle, dolayısıyla beynimizin davranış şeklinin kısmen değişmesiyle oluşur", der. Kitap, Minsky'nin "kaynaklar" dediği bu beyin kısımlarından bazılarının ne iş yaptığını ve örtüşen yığınlarının nasıl etkinleştiğini ya da durulduğunu betimliyor; böylece, sırf duygusal durumları değil, "'düşünme' dediğimiz süreci de" doğururlar.

54 Bkz. http://arts.ucsc.edu/faculty/cope/experiments.htm.

55 Bkz. http://arts.ucsc.edu/faculty/cope/5000.html.

56 Alıntı şu internet adresinden: http://arts.ucsc.edu/faculty/cope/experiments.htm.

57 Marvin Minsky, *The Emotion Machine: Commonsensc Thinking, Artificial Intelligence, and the Future of the Human Mind,* New York: Simon and Schuster, 2006. Taslağı, Minsky'nin internet sayfasında bulunabilir: http://web.media.mit.edu/~minsky/.

Minsky'nin kitabında, bilinçlilik hakkında da bölüm var. Bilinçliliğin "bavul benzeri bir sözcük olduğunu, hepimizin bu sözcüğün içini, tek ortak davası olamayacak şekilde doldurduğumuzu", ileri sürer. Bir şeyin "bilincinde" olmanın, bir düzine zihinsel etkinlik gerektirdiğini söyler; bu derlemeler, farklı koşullar altında farklı farklıdır. Dahası, sinirbilimin henüz tüm bu zihinsel etkinliklerin nasıl işlediğine dair doğru düzgün bir bilimsel görüşü bulunmuyor. Minsky, şu alıntıyı yaptığı filozof Aaron Sloman'la aynı fikirde:[58]

> [Bilinçlilik] düşüncesi, temel bir yanlış anlama üzerine kuruludur, çünkü cins isim "bilinçlilik", bunun manyetizma, elektrik, basınç ya da sıcaklık gibi bir "şey" olduğunu akla getirir ve o şeyin bağıntılarını aramaya değer.
>
> ...
>
> Bağıntı kurulacak sadece tek şey değil farklı farklı şeylerin çok büyük bir derlemesi.

İnanıyorum ki günün birinde, sinirbilimden yardım gören (ve onlara yardım eden) YZ araştırmacıları, beyin "kaynaklarının" nasıl işlediğini yeterince bilip, bilinçli olduklarına herkesi inandıracak makineler inşa edebilecek. Günü geldiğinde, bu makinelerin bilinçli olma iddiasını onaylayacağım, tıpkı sizlerin bilinçli olma iddianıza rıza gösterdiğim gibi.

Ancak, bu işi henüz başarmış değiliz, dolayısıyla tartışma kapanmış değil. Bilgisayar bilimcisi ve Yale profesörü David Gelernter (1955-), "olanaksız değilse de yazılım üzerinden bilinçli bir zihin oluşturmak çok uzak bir olanak", demiştir. Bununla birlikte, "bilinçsiz benzetimle bir zekânın belki yazılımla oluşturulabileceğini ve faydalı olabileceğini" düşünür. [Belki mi?!] "Ne yazık ki, YZ, bilişsel bilim ve akıl felsefesi, nasıl zekâ oluşturulacağını bilmenin yakınında bile değil". Gelernter, YZ'nin çabalarını, kendisinin "düşünce mekanizmaları (ya da algoritmaları)" dediği şeye odaklayacağını düşünür. O zamana dek, "YZ'nin ormanda yolunu kaybettiğinden" sızlanır.[59]

Yolunu mu kaybetmiş? Hiç sanmıyorum. Gidecek çok mu yolumuz var? Büyük olasılıkla, ama hedefe ulaşana kadar bunu bilemeyeceğiz. Ama bu işi başarana dek, alanın hem içinden hem dışından

58 Bu alıntının geldiği not için bkz. http://www.cs.bham.ac.uk/~axs/misc/consciousness/consciousness.june.18.96.txt.

59 David Gelernter, "Artificial Intelligence Is Lost in the Woods", *Technology Review*, MIT, Temmuz/Ağustos 2007; şu internet adresinde: http://www.technologyreview.com/Infotech/18867/?a=f.

çeşitli "odaklanma" çağrıları duyacağız. Her durumda, en sağlam yargılarımızın kılavuzluk ettiği pek çok farklı yaklaşımı denemeyi sürdüreceğiz. Bu, tıpkı ancak ulaştığımızda tanıdığımız bir hedefe yönelik sezgisel aramaya benziyor.

Bizler, bu hedefe ulaşmak üzere çalışmayı sürdürürken, İDYZ'nin kimi sonuçları kademe kademe belirecek. İlk önce, akıllı makinelere gitgide daha fazla bağımlı olacağız. Tıpkı geçmiş bir buçuk asırda arabalara ve başka icatlara bağımlı oluşumuz gibi, toplumumuz pek çok bakımdan internete, gelişmiş alım-satım ve mezat programlarına, çizelge programlarına, hava tahmini modellerine ve bilgisayarla ilişkili birtakım başka teknolojilere çoktan bel bağlamış durumda. Dahası, otomobil ve radyo örneklerinin aksine, gitgide daha az insan bu teknolojileri anlıyor, dolayısıyla bu durum, söz konusu teknolojilere güvenme konusunda bizi nazik bir konuma itiyor.

Gitgide daha fazla "iş", ucuz donanım-yazılım birleşimleriyle halledildikçe, bu işler için para alan insanlar, ya başka işler bulacak ya da işsiz kalma tehlikesiyle karşı karşıya kalacak. Bu, toplam gayrisafi yurtiçi hasılanın azalacağı anlamına gelmez; aslında, büyük olasılıkla yükselecektir. Toplum, makinelerin yaratacağı zenginlikten üyelerinin eşit pay almasını sağlamanın yollarını bulmak zorunda kalacak. Bu meselelerle ilgili özenli bir makalede,[60] iktisatçı Robin Hanson, insan işçilerin yerini makinelerin almasını, merhametsizce yükselen deniz düzeyine benzetir...

> ... deniz yükselirken, yüksek yerlerde "insana en özgü" işler var [insanların makinelere iktisadi üstünlüğü olduğu işler]. Bu noktada, en iyi insanların elinden gelen işleri buluyoruz, örneğin özel yemekler pişirme ya da seçkin saç kesimi gibi. Ardından, insanların ve makinelerin eşit beceriyle yaptığı işlerden oluşan "kıyı" geliyor; bunun ötesinde ise, makinelerin daha iyi yaptığı işlerden oluşan "okyanus" var. Makineler, ucuzladıkça ya da akıllandıkça ya da hem ucuzlayıp hem akıllandıkça, su düzeyi yükselir ve kıyı, iç bölgelere ulaşır.
>
> Bu engin değişikliğin iki etkisi olur. Birincisi, makineler, yeni "su basmış" işleri devralarak insanların yerine geçer. İkincisi, makinelerin işlerini insanların işlerini tamamlar hale getirmek, bu işleri iyi yapmanın değerini yükseltir.

Hanson, "makine okyanusunun" en sonunda "İş-Ülkesini" basmasının ve bunun sonucunda "maaşların düşüp çoğu insanın salt eme-

60 Robin Hanson, "Economics of the Singularity", *IEEE Spectrum*, c. 45, no: 6, Haziran 2008, s. 45-50. Şu internet adresinde: http://spectrum.ieee.org/jun08/6274.

ğiyle geçinememesinin ama başka sebeperden ötürü çalışmak istemelerinin" olası olduğuna inanıyor. (Elbette otomasyon, İş-Ülkesinde yeni ve daha yüksek zirveler yaratıp yükselen okyanusun etkilerini yavaşlatır.) Bununla birlikte Hanson, makinelerin yarattığı, epey artan zenginliğin ufak bir parçasının, "insanların rahat yaşamasına olanak tanıması" gerektiğini düşler.

35.3 ÖZETLERKEN

Bu iyimser notla birlikte, yapay zekâ arayışına dair öykümün sonuna geliyorum; bu arayış henüz tamamlanmış değil. YZ, çeşitli yolları keşfetmiş, bunu yaparken birtakım başarılar kaydedip pek çok güçlü bilişim aracı meydana getirmiştir. Bahsettiğim fikirleri ve başarıları özetlemenin yollarından biri, bunları dört ana sınıfa ayırmaktır: *tam YZ sistemleri* (iş yapan sistemler), *mimariler* (YZ sistemleri için düzenleniş ilkeleri), *süreçler* (iş yapan altprogramlar) ve *gösterimler* (*süreçlerin* yarattığı, değiştirdiği ve eriştiği yapılar). Eksiksiz anlatmaya çalışmaksızın, bu sınıflardan her birine ait başlıca YZ uygulamaları olduğunu düşündüğüm kimi gelişmeleri sıralayacağım.

- *Tam YZ sistemleri:* LT, Heuristic DENDRAL, Shakey, uzman sistemleri (MYCIN ve PROSPECTOR gibi), MSYS, konuşma tanıma sistemleri (HARPY, DRAGON, HEARSAY II gibi), Genghis, sürücüsüz arabalar (tüm sınıfları), Deep Blue ve oyun oynayan öbür makineler, RAX ve CALO.
- *Mimariler:* Pandemonium, üretim sistemleri, üç düzeyli mimariler, Karatahta mimarileri, İAN mimarileri (PRS gibi), davranış tabanlı mimariler, SOAR, ACT-R ve korteks modelleri.
- *Süreçler:* kenar çizgisi ve alan bulan filtreler (Laplace ve Gauss filtreleri dahil), etkinlik yayılması, çözümleme, kesin çözüm, A* ve bunun soyundan gelenler, ışınlı arama, Rete algoritması, STRIPS ve başka planlama sistemleri, örnek-tabanlı uslamlama, öbekleme (*k*-merkezli ve AutoClass gibi), kısıtlama yayılması, Bayes kuralı, genetik algoritmalar ve genetik programlama, GSAT ve DPLL-tabanlı yöntemler, bilgi kaynakları (karahtalarda ve içkin imgelerin hesaplanmasında kullanılanlar gibi), geri yayılım, çerçeveleme, gizli anlamsal çözümleme, Q-öğrenme ve öncelikli tarama, parçacık süzgeçleri, çekirdek hesaplaması (destek vektör makineleri

için) ve beklentiyi büyütme (BAÇ).

- *Gösterimler:* durum (problem) uzamları, vektörler, mantıksal ifadeler (EĞER-İSE kuralları dahil), anlamsal ağlar, programlar (veri yapıları olarak), karatahtalar, grafik modeller (Bayes ağları ve SMM'ler dahil), dilbilgileri, sinir ağları, karar ağaçları, komut dosyaları, çerçeveler ve genişletişmiş geçiş ağları.

[Üzerinde çalıştıkları *süreçler* ve *gösterimler* çoğunlukla yakından bağlantılıdır. Örneğin, GSAT bir önermeler (koşul biçiminde) kümesi üzerinde çalışır, Karatahta bilgi kaynakları, Karatahta kalemlerine tepki verip bunları değiştirir ve genetik programlama, LISP programı gösterimleri üzerinde iş görür.]

YZ uygulamalarına birçok disiplin katkıda bulunmuştur. Bu kitabın başında yazmış olduğum gibi, ilk YZ öncüleri, nasıl ilerleneceği konusunda pek çok ipucundan faydalanmıştı; matematikten ve mantıktan, sinirbilimden, dilbilimden, istatistikten ve olasılık kuramından, kontrol mühendisliğinden, psikolojiden ve bilgisayar biliminden gelen ipuçlarını kullanmışlardı. İşin doğrusu, YZ arayışında bugüne dek kaydedilmiş muazzam ilerlemenin sebebi, tüm o disiplinlerden gelen fikirlerin kullanılmasıdır. Şimdilik, her şeyi kapsayan bir YZ kuramı ortaya çıkmamıştır ve bence pek olası da değil.

Bu arayış sürecek. Peki, geleceğin zeki sistemlerinde, YZ'yi destekleyen disiplinlerin payanda olduğu hangi YZ yöntemi birleşimleri kullanılacak? Kimse gerçekten biliyor değil, dolayısıyla YZ'nin "araştırma cephesinde" tüm yöntemlerin etkin olmasını sağlamalıyız. Arayışın erken aşamalarında geliştirilmiş (belki artık unutulmuş) olan kimi yöntemler, daha iyi bir teknolojiyle faydalı olabilir. Bu arayışın peşinden koşmak isteyen araştırmacılar, tüm YZ yöntemi çeşitlerine, katkı veren disiplinlere ve (evet) YZ tarihine aşina olmalı.

Gelecekte yazarlar, bu maceranın öyküsünü anlatmayı hiç şüphesiz sürdürecek. Günün birinde bu yazarlardan biri, Aristoteles'in düşüncelerinden yaklaşık iki bin beş yüz yıl sonra, "ya bizim emrimizde ya da ihtiyac halinde kendi işini icra eden" araçlarımızın olduğunu bildirebilecek.

DİZİN

2½-B eskiz (iki buçuk boyutlu) 333, 334, 340, 344
3-LISP 582

A* 221, 224, 225, 284, 528, 548, 631, 633, 634, 676
Abelson, Robert 211, 212, 213, 422
ABE (uzman sistem seti) 379, 380
ABSTRIPS 236
AC-3 (Arc Consistency Algorithm, no. 3) 463
ACLS 515
ACM (Association for Computing Machinery [Bilgi İşleyen Makineler Birliği]) 66, 69, 256, 263, 318, 363, 436
ACRONYM 342, 344, 346
ACT-R (Adaptive Control of Thought; uyarlanımsal düşünce denetimi- rasyonel) 592, 593, 594, 595, 599, 676
Adaboost 537
ADALINE (adaptive linear network, uyarlanımsal-doğrusal ağ) 101
Adams, James 238
Adelson-Velskiy, Georgi 255
AgentSpeak 589
Agin, Gerald 341
Aizerman, M. A. 108
akıllı aletler 649, 653
A kutusu 451
Albert, Lev 605
Albus, James 575, 576, 577, 578
Alder, Phillip 612
alfa-beta yöntemi 254
Allen, Paul 16, 42, 69, 70, 78, 79, 81, 84, 124, 235, 274, 277, 286, 301, 590, 595, 628
Alman Yapay Zekâ Araştırma Enstitüsü (DFKI) 362
Aloimonos, Yiannis 342
ALPAC (Otomatik Dil İşleme Danışma Komitesi) 152, 240, 404
Alvey, John 360, 362
Alvey Programı 359, 360
ALVINN (Autonomous Land Vehicle in a Neural Network [Yapay Sinir Ağlı Özerk Kara Aracı]) 520, 521, 522, 618
amaçlı görü 550
Amarel, Saul 120
Amerikan Yapay Zekâ Birliği (AAAI) 347, 348, 349, 416, 417, 611, 657, 661
Ames Araştırma Merkezi 614
Anderson, Alan Ross 385
Anderson, John R. 592
Andreae, John 527
anlambilim 148, 241, 245, 651
anlamsal ağlar 303, 449, 450, 451, 452, 677
anlamsal bilgi 137, 209
anlamsal çözümleme 144, 233, 676
anlamsal gösterimler 134
ARGO projesi 618
Aristoteles 19, 27, 139, 201, 677
Arkin, Ronald 672
ARPA, *bkz.* DARPA
Arpanet 176, 347, 363
Ars Magna 20, 21
artı geribildirim 50
Ashby, Ross 52, 53, 54, 56, 86
Asimov, Isaac 25, 26
ASK 323
aşikâr kümelenme 395
Athena DSS 637
Atkeson, Christopher 503
Atkin, Larry 256
AURORA 639, 640
Austin, John 155, 203, 208, 209, 237, 359, 415, 448, 455, 457, 587
AutoClass 525, 526, 676

Automata Studies 82
Axline, Stanton 298
ayırıcı sınırlar 108
ayrıştırma ağaçları 147, 252
ay taşları 248, 318
azami-olabilirlik 55

Babbage, Charles 58, 59, 68, 127, 163
Bach, Michael 172, 673
Backus, John 86
bağdaşım birimleri 539
Baker, James 277, 281, 288
Baker, Janet 281, 288
Bao, Xinlong 659
Bar-Hillel, Yehoshua 150, 151, 152
Barlow, Horace 174, 199, 334, 335, 355
Barnett, Jeffrey 274
Barricelli, Nils 44
Barrow, Harry 335, 336, 337, 338, 562
Barto, Andrew 526, 534
Basamak Hesaplayıcısı 58
BASEBALL 153
Bayes ağları 304, 421, 485, 486, 487, 488, 489, 491, 494, 496, 497, 498, 499, 500, 632, 677
Bayes kuralı 54, 55, 56, 107, 279, 281, 303, 304, 486, 557, 676
Bayes, Thomas 54
BBN (Bolt, Beranek, Newman) 159, 160, 164, 165, 166, 246, 247, 250, 274, 276, 287, 288, 298, 317, 324, 363, 370, 376, 378, 379, 380, 450, 588, 642
BBN-LISP 298
BBN Technologies 642
"Beast" 216
Beer, Randall 432, 622
Begriffsschrift 33, 34
Beklentileri Büyütme (BAÇ) algoritması 547, 677
belirlenimci olmayan polinom (NP) 409, 410, 466, 471
belit modeli 226, 227
Bell, Anthony J. 174
Bell Laboratuvarları 272, 275
Bellman, Richard 527
Benzerlik problemleri 134
Berliner, Hans 257, 259
Bernstein, Alex 82
Beşinci Nesil Bilgisayar Sistemleri (BNBS) 353, 354, 356, 357, 358, 359, 360, 361, 362, 363, 366
Bethe, Hans 485
betimleme mantığı 450
BigDog 652, 653
Bigelow, Julian 51
bilgi ilkesi 262
bilgisayarla görme 113, 171, 173, 174, 178, 179, 190, 269, 317, 318, 331, 332, 337, 343, 344, 346, 374, 376, 437, 439, 461, 475, 523, 549, 550, 551, 561, 570, 622, 623, 624, 629, 631, 651
bilgisayar oyunları 224, 437, 633, 634
bilgi tabanı 300, 307, 444, 454, 457, 458
Bilişimsel Dilbilim Derneği (ACL) 153, 589
bilişsel bilim 75, 126, 214, 264, 505, 572, 674
bilişsel mimariler 590, 592, 594, 595, 599
Binford-Horn çizgi bulucu 191
Binford, Thomas 191, 340, 341
birleşimsel patlama 266, 407, 411, 412
Bisson, Charles 175
Black and White 2 (bilgisayar oyunu) 633, 634
Blackbox 472, 473
Black, Fisher 202
Blaise Pascal 57
Blake, Andrew 556, 557
BLAZE ADVISOR 6.1 641
Bledsoe, Woodrow 45, 100, 175, 203, 359
Blei, David 480

Block, H. David 99, 100
Bobrow, Daniel 156, 158, 215, 249, 250, 253
Boden, Margaret 241
Bod, Rens 548
Boeree, C. George 39
Bolles, Robert 195
Boole cebiri 32
Boole, George 31, 32, 33, 56, 201, 203
Booth, Taylor 545
Boston Dynamics 652
Bower, Gordon 208
Boyer, Robert 163
Brachman, Ronald 449, 450, 451
Brain, Alfred E. (Ted) 102, 103
Bratley, Paul 249
Braverman, E. M. 108
Breiman, Leo 515, 536
Bremermann, Hans 45
Bresnan, Joan 651
Brice, Claude 230, 361
Brooks, Rodney 341, 342, 423, 424, 425, 426, 574, 668, 669
Browning, Iben 100
Bryson, Arthur 518
Buchanan, Bruce 260, 296, 298, 301
BUILD 446
bulanık mantık 420, 421
Burstall, Rod 162
Butterfly Multiprocessor 376

C4.5 514, 515
C5.0 515
Cajal, Santiago Ramón y 34
calculus ratiocinator (uslamlama kalkülüsü) 29
CALO (Cognitive Assistant that Learns and Organizes [Öğrenen ve Düzenleyen Bilişsel Yardımcı]) 658, 659, 676
Caltech (California Teknoloji Enstitüsü) 75, 80, 234, 323, 652
Campbell, Alan 302, 306
Campbell, Murray 127, 605
Canny, John 183
Čapek, Karel 23, 24, 25
Carlstrom, David 343, 344
CART (Classification and Regression Trees) 511, 514, 515
CASES (Yetenek Değerlendirme Uzman Sistemi) 370, 371
CASNET (Causal-ASsociational NETwork) 308, 309
CCH-ES uzman sistemi 311
Centre National de la Recherche Scientifique (Ulusal Bilimsel Araştırma Merkezi) 206
Champernowne, D. G. 127
Charniak, Eugene 545
CHAT-80 321, 322, 323
Cheeseman, Peter 525
CHESS, Northwestern Üniversite programları 256, 257, 414, 605
CHINOOK 608, 609, 610
Chomsky, Noam 40, 41, 42, 144, 145, 147, 248
Church, Alonzo 59
Churchland, Patricia 343
CIA (Merkezî İstihbarat Teşkilatı) 153, 175, 359, 376
Clark, Wesley 76, 78, 79, 95
CLASSIC 451
ClearFlow 632
Clowes, Max 188
CMU 159, 161, 166, 194, 216, 274, 276, 277, 281, 284, 287, 288, 309, 312, 344, 374, 376, 377, 417, 520, 543, 622, 627, 652
Cocke-Younger-Kasami (CYK) algoritması 543
Cognitec Systems Ltd. 645
Cohen, Philip 588
Cohen, Stanley 296, 298
Coles, Stephen 233, 234
Collins, John 162, 265
Colmerauer, Alain 206, 207, 321
Colorado Üniversitesi 476
COMPANIONS 659

CONDENSATION (Conditional Density Propagation [Koşullu Yoğunluk Yayılımı]) algoritması 556
Contract Net 585
Cook, Stephen 409
Cooper, Greg 489
Cooper, Robert 365, 366, 375, 382
Cope, David 673
Corbató, Fernando 165
Cornell Havacılık Laboratuvarı 95
Cover, Thomas 108
CSAIL (Bilgisayar Bilimi ve Yapay Zekâ Laboratuvarı) 165
Cuyler, Margery 441
Cyc 455, 456, 457, 458, 459, 460, 541, 660
CycL 457, 458, 459
Cycorp 457, 458
çalışma istasyonu 347
çekirdek makineleri 539, 540
çıkarsama motoru 300, 458
çizelgeleme sistemleri 639
Çoklu Etmen Hesaplama Ortamı (MACE) 585
çoklu etmen sistemleri 584, 589

Dağıtık Taşıt Gözetimi Test Cihazı (DVMT) 585
dairesel olmayan yönlü grafikler (DYG) 486
dama 64, 73, 81, 119, 120, 128, 130, 131, 133, 134, 159, 163, 254, 406, 603, 608, 609, 610, 633
DAML-ONT 452
DARPA (İleri Savunma Araştırmaları Projeleri Dairesi) 15, 166, 219, 235, 236, 237, 265, 266, 269, 272, 274, 275, 277, 281, 284, 286, 287, 288, 290, 295, 296, 302, 314, 315, 316, 317, 318, 320, 321, 324, 327, 343, 344, 345, 346, 349, 362, 363, 365, 366, 367, 368, 369, 370, 371, 372, 374, 375, 376, 377, 378, 379, 380, 382, 410, 411, 417, 588, 618, 619, 620, 624, 625, 626, 653, 657, 661, 670
DART (Dynamic Analysis and Replanning Tool) 380
Da Vinci, Leonardo 20
Davis, Martin 467
Davis, Ruth 217, 219, 256
DDR&E (Savunma Araştırmaları ve Mühendisliği Dairesi) 217
Dean, Thomas 499, 520, 567, 669
Dechter, Rina 463
Deep Blue 603, 604, 605, 606, 607, 608, 676
Deep Fritz 608
Deep Thought 605
de Kleer, Johan 446
Delphi yöntemi 647
Dempster-Shafer (D-S) kuramı 419
DENDRAL 261, 262, 269, 296, 300, 326, 327, 676
Deneysel Programlama Birimi 162
Denicoff, Marvin 164
Dennett, Daniel 213
derin inanç ağı 563
destek vektör makinesi (DVM) 112, 538, 539, 540, 676
de Vaucanson, Jacques 22, 49
Devol, George 191
Dewar, Hamis 249
DIAGRAM 294, 324, 542
DIALOGIC 324
DIAMOND 324
DISTRIBUTED HEARSAY-II 584, 585
Dickmanns, Ernst 558, 574, 618
Diebenkorn, Richard 568
Dietterich, Thomas 659
Digitalsmiths 651
Dijkstra, Edsger 405
Dilbilim Veri Konsorsiyumu 544
dinamik Bayes ağları (DBA) 499, 500, 556
dinamik görü 558
dinamik programlama 105, 527, 543
dinamik süreçler 431
Dinneen, Gerald 77, 78, 79, 91, 93,

94, 179
Djerassi, Carl 262
doğal dil işleme 143, 152, 153, 155, 233, 240, 248, 269, 303, 320, 331, 363, 375, 422, 437, 439, 450, 454, 458, 503, 549, 601, 651, 654, 658, 661, 667
DOMO 656
Donahue, Christopher 536
Doran, J. 223
Doyle, Lauren 156
Doyle, Worthie 94
DPLL 467, 468, 472, 676
DRAGON 277, 278, 279, 280, 281, 282, 283, 288, 676
Dragon Systems 281, 287, 288
Dresden Teknoloji Üniversitesi 445
Dreyfus, Hubert 167, 256, 393, 394, 395, 396, 397, 398, 411, 418
Druffel, Lawrence 345, 346
Duda, Richard 104, 105, 109, 225, 230, 302, 312, 526
Dumais, Susan 480
durum kalkülüsü 205
durum tahtası 327
Dünya Bilgisayar Satrancı turnuvaları 257
Dünya Çapında Ağ 15, 411, 473, 671
düzleştirme 499
Dyson, George 22

Earnest, Lester 160, 161, 166, 193, 238
Eckert, J. Presper 63
edimbilim 144, 148
Efros, Alexei 551
EĞER-İSE kuralları 286, 296, 297, 298, 300, 302, 304, 309, 313, 328, 513, 514, 515, 590, 677
eigenfaces (özyüz) 644
Einaudi, Marco 302
Ejiri, Masakazu 196
eksi geribildirim 50
ELIZA 66, 67, 153, 400, 633
Elias, Peter 178
Elschlager, Robert 338, 339
EMYCIN 300, 301, 312, 379
enerji tabanlı modeller (ETM) 568
Engelberger, Joseph F. 191
en yakın komşu yöntemi 108, 176
EON 638
EPAM (Elementary Perceiver and Memorizer [Temel Alıcı ve Ezberleyici]) 508, 509, 510, 511
Erman, Lee 277, 286, 584
Ernst, George 158
Ernst, Heinrich 190
eski toprak YZ (ETYZ) 395, 397, 418, 431
ESPRIT (European Strategic Program of Research in Information Technology [Bilgi Teknolojisi Avrupa Stratejik Araştırma Programı]) 349, 358, 361, 362, 540
eş küme 453, 454
Etzioni, Oren 661, 662
EUFID 323
Evans, Thomas 134, 135, 136, 137, 158
evrişim 180

FaceVACS (yüz tanıma sistemi) 645
FACTory (Cyc oyunu) 460
Fahlman, Scott 446
Falk, Gilbert 189
Farley, Belmont 76, 78, 79, 95
F.E.A.R. (bilgisayar oyunu) 634
Feigenbaum, Edward 83, 131, 257, 258, 259, 260, 262, 264, 282, 296, 318, 326, 327, 366, 379, 404, 508, 509, 510, 663
Feist, Mathias 608
Feldman, Gary 181
Feldman, Jerome 192, 431
Feldman, Julian 264
Fennema, Claude 230
Fikes, Richard 227, 228, 451
Firby, R. James 574

Firschein, Oscar 647
Fischler, Martin 338, 339, 376
Fisher, Ronald A. 56
Fix, E. 108
fiziksel simge sistemi hipotezi 70
Fletcher, Tristan 539
FLP (liste işleme operasyonlarıyla güçlendirilmiş FORTRAN) 158
Fogel, Lawrence 45
FOIL 516
FOL 582
Forbus, Kenneth 448
Ford, Kenneth 109, 113, 194, 625, 666
Forgy, Charles 309, 312, 313
FORTRAN 86, 104, 158, 254, 281, 355
Fox, Dieter 558
Franz A.Ş. 348
FranzLISP 348
Fraser, Bruce 249
FREDDY 197, 198, 199, 266
Fredkin, Edward 159
Frege, Gottlob 33, 34, 56, 201
FRESH (Güç Gereksinimleri Uzman Sistemi) 370, 371
Freuder, Eugene 191
Freud, Sigmund 39
Freund, Yoav 537
Friedberg, R. N. 44, 45
Friedman, Jerome 515
Friedman, Nir 493
FRL (Frame Representation Language [Çerçeve Gösterimin Dili]) 214
Fuçi, Kazuhiro 354
Furukava, Koiçi 355, 357

Galatea 19
Garvey, Thomas 337
Gaschnig, John 302
Gasser, Les 585
Gat, Erann 574
Gates, Bill 234
Gauss, Carl Friedrich 182
Gausvari Laplaceçı (LoG) 182, 183
geçiş ağı dilbilgisi 252
Gelernter, David 674
Gelernter, Herb 121, 122, 123, 124, 134, 158, 163, 394
Genel Problem Çözücü (GPÇ) 124, 125, 126, 158, 228, 590, 595, 619, 663
Genel Yapay Zekâ Araştırma Enstitüsü (AGIRI) 665
genel yapay zekâ (GYZ) 665
General Electric 93, 310, 379, 637
Genesereth, Michael 583, 589
genetik algoritmalar 676
genetik programlama 676, 677
Genghis 423, 424, 676
George, Dileep 565, 567
Georgeff, Michael 579, 580, 581
Georgia Teknoloji Enstitüsü 505, 672
gerçek zamanlı denetim sistemleri (GDS) 575, 576, 577, 578
geribildirimli denetim 50
geri izlemeli arama 468
Getoor, Lise 493
GIB (Goren in a Box ya da Ginsberg's Intelligent Bridgeplayer) 611, 612
Gibson, William 52
Gilmore, Paul 201
Ginsberg, Matt 611, 612
Gizli Anlamsal Çözümleme (GAÇ) 476, 477, 478, 479, 480
Gizli Anlamsal Dizinleme (GAD) 479
Glennie, Alick 127
Glucksman, H. 539
Goertzel, Ben 665
Goldstein, Ira 215
Golgi, Camillo 34
Good, I. J. 152, 484, 663
Google 280, 398, 476, 497, 549, 552, 601, 631
Gorlen, Keith 256
Gödel, Kurt 386, 387
gözetimsiz öğrenme 523, 563
Grape, Gunnar 189
Green, Bert 154

Greenblatt, Richard 255, 256
Green, Cordell 87, 202, 204, 205, 206, 274, 314
Grenander, Ulf 561
Grosz, Barbara 294, 295
GSAT 469, 470, 471, 472, 676, 677
Gulko, Boris 605
GUS (Genial Understander System) 250, 252, 253
Guzman, Adolfo 184, 185, 186, 188, 189

H1ghlander entrant 620, 622
HAL 398, 606, 613, 672
HAM 208, 592
Hanson, Alan 376
Hanson, Robin 675, 676
Harbaugh, John 302
Harley, Thomas 31, 109, 111, 112, 113
Harmon, Paul 311
HARPY 276, 277, 281, 282, 283, 284, 285, 287, 676
Harrison, John 51
Hart, Peter 104, 105, 108, 109, 175, 223, 224, 228, 230, 302, 312, 451, 526
HASP (Heuristic Adaptive Surveillance Program) 318, 326, 327, 328, 329, 579
Haugeland, John 395, 418
Hawkins, Jeff 109, 564, 565, 567, 669
Hayes-Roth, Frederick 277, 286, 380
Hear Here 194
HEARSAY-I 277, 278, 281, 282, 284
HEARSAY-II 276, 277, 284, 285, 286, 287, 327, 328, 330, 579, 584, 585
Hebb, Donald 37, 76, 95
Hebb kuralı 37
hedonist sinir hücreleri 527
Heidegger, Martin 397
Heilmeier, George 316, 317, 318, 363
Hendrix, Gary 303, 318, 320
Hephaistos 19
HERB (Home Exploring Robotic Butler) 655, 656
Herskovits, Edward 489
HeurIstIc Dendral 261, 296
Hewitt, Carl 207, 208, 227, 245, 441, 583
HIVIP 196
Hildreth, Ellen 182, 183, 184
Hinton, Geoffrey 430, 517, 563, 669
hiyerarşik planlama 236
HMMWV (High-Mobility Multipurpose Wheeled Vehicle) 375, 620
Hobbes, Thomas 20
Hodges, Andrew 65
Hodges, J. L. 108
Hoff, Marcian (Ted) 101
Hoiem, Derek 551
Holland, John 45, 433, 435
Hollerith, Herman 49
Hollister, Floyd 317
Hollister, Victor 306
homeostazi 225
Honeywell 199, 344, 359
Hopfield ağları 431
Hopfield, John 431
Horn, Berthold 191, 331, 332, 333
Horvitz, Eric 583
Hough dönüşümü 230
Hovland, Carl 511
Hsu, Feng-Hsuing 605
Hubel, David 173, 174
Huffman, David 186, 187, 188
Huffman kodlaması 186
hukuksal uslamlama 356
Hunt, Earl 511, 512
Hurd, Cuthbert 151, 347
Huskey, Harry 149
Hutchins, W. John 240
HWIM (Hear What I Mean) 276, 277, 287
hypernym 453, 454
hyponym 453, 454

IBM (International Business Machines) 44, 80, 81, 82, 102, 104, 114, 121, 127, 128, 151, 159, 160, 163, 201, 202, 254, 287, 288, 311, 347, 349, 377, 464, 533, 603, 604, 605, 606, 642, 669
ICOT (Yeni Nesil Bilgisayar Teknolojisi Enstitüsü) 354, 355, 356, 357, 358
ID3 (Iterative Dichotomizer [Yineleyen İkiye Ayırıcı]) 511, 512, 513, 514, 515
IDA (Intelligent Data Access) 320
Iida, Fumiya 397
ILOG 464, 641
Inman, Bobby Ray 359, 455
IntelliCorp 312, 379
IntelliGenetics 379
INTERLISP 298, 318, 347
INTERNIST-1 303, 308, 309, 489, 637
INTERPLAN 237
IPL (information processing language [bilgi işleme dili]) 153, 154, 158, 167, 509, 510
IPTO (Bilgi İşleme Teknikleri Ofisi) 164, 165, 274, 294, 295, 296, 314, 315, 316, 317, 318, 326, 363, 375
IraqComm 642
IRIS (Integrate. Relate. Infer. Share.) 659
IRUS 324, 378
Isard, Michael 556, 557
ışınlı arama 284, 676

İAN mimarisi 579, 580, 676
ilişkisel veri madenciliği 516
İlyada 19
İnsan Düzeyinde Yapay Zekâ (İDYZ) 662, 663, 665, 666, 667, 668, 669, 670, 671, 672, 675
insanın çağrışımlı belleği (İÇB) 592
İskendireyeli Pappus 122
istatistiksel regresyon 503
iş âlemi kuralları idaresi sistemleri (BRMS) 313, 641
işler bellek (İB) 590, 597, 598

Jacobstein, Neil 380
Jacquard, Joseph-Marie 20, 49, 50
James, William 38, 39, 51, 126, 238, 249, 264, 266, 277, 288, 407, 429, 542, 574, 575
JANUS 288, 378
Jelinek, Frederick 288
Jeolojik Tetkik Arama 302, 307
JETA (Jet Engine Troubleshooting Assistant) 310
Jochem, Todd 523
John Anderson 208
JOHNNIAC bilgisayarı 115, 158
Johnson, George 349
Jordan, Michael 480
JRules 6.0 641

Kahn, Robert 363, 364, 365, 366, 382
Kanade, Takeo 176, 376
Kanal, Laveen 109, 110, 112, 113, 518
Kanazawa, Keiji 499
Kandel, Eric 37
Kaplan, Ronald 247, 250, 252, 548, 651
Kapor, Mitchell 667
kapsama mimarileri 574
karar ağaçları 513, 514, 677
karmaşıklık kuramı 408
kartografi 331, 376
Kasparov, Garry 603, 604, 605, 606, 607
Kat-5 621
Kautz, Henry 472
kaydırmalı blok bulmaca 116
Kay, Martin 543
KBMS (Knowledge Based Management System) 320
KEE (Knowledge Engineering Environment [Bilgi

Mühendisliği Ortamı]) 379, 380
Kelly, Michael 175
Kesin Koşullu Dilbilgileri (KKD) 321, 542
KIF (Knowledge Interchange Format) 588, 589
kısıt sağlama problemleri 461
kısmen gözlemlenir Markov karar süreçleri (KGMKS) 532
Kirsch, Joan 568
Kirsch, Russell 39, 91, 92, 179, 568
Klatt, Dennis 287
Klein, Dan 548
KL-ONE 450
Klopf, A. Harry 527
KnowItAll Projesi 661
Knuth, Donald 404
Koller, Daphne 493, 494
Kolodner, Janet 505, 506
komut dosyaları 213
Konolige, Kurt 302, 305
Konuşma Anlama Çalışma Ekibi 274
konuşma dalgabiçimi 278, 281, 283, 539
konuşma fiilleri 588
Korf, Richard 224, 412, 413
Kornfeld, William 583
koşullu bağımsızlık 485
koşullu olasılık tablosu (KOT) 486, 491, 492, 496, 497
koşut çıkarsama makinesi (PIM) 356, 357
Kotok, Alan 254, 255
Kowalski, Robert 163, 206, 207
Koza, John 435, 436, 643
KQML (Knowledge Query Manipulation Language) 588, 589
Kramnik, Vladimir 607, 608
KRL (Knowledge Representation Language) 215
Kronrod, Alexander 128, 255
KRYPTON 451
Kuehner, Donald 206
Kuipers, Benjamin 415, 448
Kulikowski, Casimir 308
Kumar, Vipin 463
Kunze, Fritz 348
Kuramsal ve Deneysel Fizik Enstitüsü 255
Kurzweil, Ray 664, 665, 667
kütle spektrogramı 259

LADDER (Language Access to Distributed Data with Error Recovery) 320, 321
Lafferty, Don 609
Laird, John 595, 599, 634
Landauer, Thomas 476, 478, 479, 480
Langley, Patrick 599
Laplace, Marquis de 54
Lashley, Karl 75, 234
LCS (Bilgisayar Bilimi Laboratuvarı) 164
LDC-1 324
Lebowitz, Michael 505
LeCun, Yann 568
Lederberg, Joshua 258, 259, 260, 261, 262, 296
Lee, Kai-Fu 288, 377
Lee, Tai Sing 561
Leibniz, Gottfried 28, 29, 31, 56, 58
Lenat, Douglas 455, 456, 457, 459, 460
Lesser, Victor 277, 286, 584, 585
Lettvin, Jerome 173
Levesque, Hector 450, 451, 469
Leviathan 20
Levin, Leonid 409
Levy, David 256, 414
LIFER (Language Interface Facility with Elliptical and Recursive Features) 318, 319, 320
LISP (List Processing [Liste İşleme]) 134, 138, 158, 159, 167, 185, 204, 207, 210, 216, 228, 245, 246, 298, 299, 347, 348, 405, 435, 571, 616, 617, 677
Licklider, J. C R. 164, 165, 166, 294, 295, 296, 315, 316, 317,

318, 344
Lighthill, Sör James 266, 267, 347, 360, 407, 408
Lincoln Laboratuvarı 77, 78, 85, 274, 276
Lindsay, Robert 154
Lisp Machines A.Ş. 347
Lisp makineleri 347
Llull, Ramon 20, 21, 31
Lo, Andrew 640
Loebner, Hugh 66
Long Now Vakfı 667
Lovelace, Ada 59, 68
Lowerre, Bruce 277, 281
LT (Logic Theorist [Mantık Kuramcısı]) 83, 84, 115, 116, 118, 158, 391, 676
Lucas, John 386, 387, 399
Lukasik, Stephen 315, 316
LUNAR 247, 248, 269, 276, 318
Lungarella, Max 397

MAC HACK VI 255, 256, 414
Machina speculatrix (tosbağa) 46, 47
Mackworth, Alan 189, 463
MAC Projesi 164, 165
MADALINE (çoklu ADALINE) 101, 516, 537
Malik, Jitendra 559
Manning, Christopher 544, 548, 549
Mansfield Ek Yasası 265, 269, 314, 347
mantık programlama 163, 206
Marin, Janet 511
Markov, Andrey Andreyevich 279
Markov karar süreci (MKS) 532
Markov tabloları 225, 229
Markov varsayımı 279, 280
Marr, David 182, 183, 184, 333, 334, 340, 342, 343, 550, 551, 576, 624
Masterman, Margaret 140
MASTOR (IBM'in konuşmadan konuşmaya çeviri programı) 642
Maturana, H. R. 173
Mauchly, John W. 63
McCammon, Richard 307
McCarthy, John 80, 81, 82, 83, 84, 85, 86, 87, 119, 120, 128, 132, 133, 134, 152, 158, 159, 160, 162, 163, 165, 166, 192, 200, 201, 202, 203, 205, 234, 245, 254, 255, 256, 282, 414, 415, 418, 422, 444, 447, 607, 662, 663, 667, 669
McClelland, James 429, 430
MCC (Microelectronics and Computer Technology Şirketi) 358, 359, 455, 457
McCorduck, Pamela 82, 84, 366, 396
McCulloch-Pitts sinir hücresi 36
McCulloch, Warren 35, 36, 53, 56, 71, 75, 86, 96, 173
McCune, William 203
McDermott, Drew 417, 418, 419
McDermott, John 309
MEDIPHOR 296
Meltzer, Bernard 65, 163, 203
memistor (ayarlanabilir ağırlık) 101
MENACE (Matchbox Educable Noughts and Crosses Engine [Eğitilebilir Kibrit Kutusu SOS Motoru]) 162, 527
MENS (MEmory Net Structure) 209
MENTAL (MEmory Net That Answers and Learns [Yanıtlayıp öğrenen bellek ağı]) 209
Merigan, Thomas 298
meta-enformasyon 449
Meter, David Van 56
MH-1 190, 191
MINOS 101, 102, 103, 104, 516
MIT 42, 76, 77, 80, 81, 84, 85, 86, 101, 114, 126, 134, 140, 150, 156, 158, 159, 160, 161, 164, 165, 166, 167, 169, 173, 176, 177, 178, 179, 185, 186, 188, 189, 190, 191, 192, 199, 207, 208, 216, 218, 227, 241, 242,

254, 255, 256, 264, 273, 274, 287, 288, 318, 331, 332, 341, 344, 347, 363, 377, 393, 418, 423, 425, 426, 431, 446, 460, 550, 567, 617, 625, 640, 652, 656
Michie, Donald 65, 161, 162, 163, 197, 223, 256, 265, 266, 512, 514, 515, 527, 540
Microsoft 58, 234, 235, 288, 359, 548, 601, 649, 651
Middleton, David 56
Miller, George 41, 42, 43, 225, 452
Miller, Randolph 308
Minsky, Marvin 39, 44, 45, 80, 81, 83, 84, 85, 86, 120, 121, 122, 123, 126, 128, 134, 135, 137, 138, 139, 140, 156, 158, 159, 160, 165, 167, 169, 178, 184, 214, 215, 217, 253, 264, 265, 386, 421, 422, 449, 481, 505, 583, 663, 668, 673, 674
Mitchell, David 469
Mitchell, Thomas 661
Modha, Dharmendra 669, 670
modus ponens (doğrulama kipi) 201, 202, 203
MoGo 612
Montemerlo, Michael 622
Moore, Andrew 503
Moore, J. Strother 163
Moravec, Hans 47, 216, 239
morfoloji 398
Morsch, Frans 607
Möbius Projesi 661
MSYS 338, 562, 676
Mumford, David 561, 562, 569, 669
Munson, John 104, 105, 228
Musen, Mark 638
MYCIN 296, 298, 299, 300, 301, 302, 304, 308, 415, 419, 481, 637, 638, 676
Myers, Jack 308
Myhrvold, Nathan 58

Nagy, George 100, 109
NASA (National Aeronautics and Space Administration) 15, 246, 247, 369, 525, 577, 613, 614, 615, 617
NASREM (NASA/NBS Standard Reference Model) 577
Navlab 520, 521, 523, 618
Nealey, Robert 133
Neocognitron 560
NETtalk 518, 519, 551
Nevatia, Ramakant 341
Newell, Allen 70, 115, 117, 118, 119, 125, 126, 127, 132
New Generation Computing 357
NEWTON 446, 447
Newton, Isaac 28
Neyman, Jerzey 56
Ng, Andrew 480, 535, 553, 555, 654, 655
Nilsson, Nils 15, 104, 632
nitel fizik 447
nitel uslamlama 446, 447
NLP-DBAP 324
NOAH (Nets of Action Hierarchies [Eylem Hiyerarşi Ağları]) 236, 237, 291, 292
NONLIN 237
Norris, William 359
Norvig, Peter 287, 301, 463, 484, 497, 499, 540, 579
Novak, Gordon 155
Novikoff, Albert 99
NP-tam 466, 471
NSS (Newell, Shaw, Simon) 79, 158
Numenta 567

Oakley, Brian 360
Office of Naval Research (ONR) 15, 164, 166
Ohlander, Ronald 346, 375, 379
Okuyarak Öğrenme (OÖ) 660, 661
olasılık kuramı 304, 483
olasılıksal arazi çözümlemesi (OAT) algoritması 623
olasılıksal bağlamdan yoksun dilbilgisi (OBBD) 545, 546,

547, 548
olasılıksal çıkarsama 486, 487, 496
olasılıksal grafik modelleri 500, 659
Olasılıksal İlişki Modeli (OİM) 494, 495, 496
Oldburry, Derek 133
Olsen, Kenneth 78
Omohundro, Stephen 672
ONCOCIN (onkoloji protokollerin idaresine yardımcı olan bir program) 638
OpenCyc 459
OPI (Organize and Prepare Information [Bilgiyi Düzenle ve Hazırla]) 658
O-PLAN (The Open Planning Architecture) 237
Orkin, Jeff 634
Ormohundro, Stephen M. 671
Osindero, Simon 563
otomatlar 19, 22, 49, 82, 266, 545
Ovidius 19
Owen, Kenneth 360
OWL 452
Oxford Üniversitesi 128, 556, 649

öbek çözümlemesi 526
öbek yapısı dilbilgisi 144
öncelikli tarama 532, 676
önermeli mantık 33, 42, 391, 465
örüntü tanıma 76, 85, 86, 87, 91, 99, 101, 107, 108, 109, 113, 160, 177, 216, 257, 394, 644, 668
özçağrı 125, 126
özçağrılı geçiş ağları 249, 250
Özerk Kara Taşıtı (ALV) 371, 372, 373, 374, 375, 376, 381, 618
özerk taşıtlar 372, 374, 375, 381, 523

Pandemonium 87, 88, 89, 90, 94, 96, 112, 132, 399, 560, 572, 676
Panoramic Research 45
Papert, Seymour 73, 160, 178, 179, 215, 242, 256, 264, 265, 269
PARC (Palo Alto Araştırma Merkezi) 250, 253, 451, 651
parçacık süzgeci 499, 558, 562
Park, Charles 101, 161, 302
PASCAL 514
Pascal, Blaise 57, 58
Paul, Richard 193
PDP (parallel distributed processing [koşut dağıtık işleme]) 159, 160, 161, 185, 192, 193, 219, 228, 243, 255, 284, 298, 429, 430, 560
Pearl, Judea 304, 483, 484, 485, 486
Pearson, Karl 56
pekiştirmeli öğrenme 162, 526, 527, 529, 532, 534, 535, 536, 603
Penrose, Sir Roger 188, 387, 388, 418
perceptronlar 100
Pereira, Fernando 321, 323
Perrault, C. Raymond 588
Pfeffer, Avi 493
Pfeifer, Rolf 397
Philco 109, 110, 111, 112, 113, 160
Piaget, Jean 47, 160
Pierce, John 152, 275, 404
Pingle, Karl 192, 193
Pitts, Walter 35, 36, 71, 79, 95, 96, 173
Plankalkül 63, 127
Poggio, Tomaso 567
Pomerleau, Dean 523
Poole, David 494
POP-2 162, 167, 199
Pople, Harry 308
Popplestone, Robin 162
Porphyrios 139
Powerset 548, 651, 652
Prawitz, Dag 202
PROLOG 206, 207, 321, 347, 353, 354, 355, 356, 442, 516
PROSPECTOR (maden yataklarına yönelik uzman sistemi) 302, 303, 304, 305, 306, 307, 312, 419, 481, 676

PROTEUS 378, 379
Protosynthex 156
PUFF 637
PUNDIT 378
Putnam, Hilary 385, 467

QA3 202, 204, 205, 206, 226, 227, 228
Q-öğrenme 529, 532, 676
Quillian, Ross 140, 141, 142, 143, 158, 200, 208, 392
Quinlan, J. Ross 511, 512, 513, 514, 515, 516
Quintus, A.Ş. 347

R1 309
Rabinow, J. 92
RADC (Rome Havacılık Geliştirme Merkezi) 218, 219
RADIUS (Research And Development for Image Understanding Systems) 376
Raibert, Marc 652
RALPH (Rapidly Adapting Lateral Position Handler) 523, 618
Ramachandran, V. S. 343
Randall, Neil 109, 110, 432
Raphael, Bertram 137, 138, 139, 143, 153, 158, 200, 204, 215, 216, 218, 223, 224, 264, 442
RA (Remote Agent) 613, 614, 615, 616, 617
RAX 616, 617, 676
RCA 273, 316, 359
Reboh, Rene 302, 312
Rechenberg, Ingo 45
Reddy, Raj 181, 194, 277, 286
referans model mimarisi 575
Reis, Victor 380
Reiter, John 302
Reiter, Raymond 443
ResearchCyc 459
Rete algoritması 676
Ridgway, William 101, 516
Rindfleisch, Thomas 639
Risch, Tore 302, 312
Riseman, Edward 376
Rissland, Edwina 506
Robbins cebiri 203
Robbins, Herbert 203
Roberts, Bruce 215
Roberts Geçişi 177, 178
Roberts, Larry 178
Roberts, Lawrence 176, 177, 178, 314
Robinson, Alan 202, 203, 206
RoboCup 586
robot 25
Rochester, Nathaniel 80, 81, 82, 120, 163, 269, 344, 431
Rogers, Seth 599
Roland, Alex 364, 365, 366, 374, 377, 378, 379, 381, 382, 417
Rosenberg, Charles 518
Rosenblatt, Frank 95, 96, 98, 99, 100, 101, 102, 103, 109, 264, 265, 430, 516, 539
Rosenbloom, Paul 595
Rosenblueth, Arturo 51
Rosen, Charles 15, 101, 102, 216, 347, 517
Rosenfeld, Azriel 376
Rosenschein, Jeff 589
Roszak, Theodore 402, 403, 660
Roussel, Philippe 206, 207
Rozonoer, L. I. 108
RulesPower, A.Ş. 313
Rumelhart, David 429, 430, 517, 518
Russell, David 295, 317
Russell, Stuart 583

Sacerdoti, Earl 236, 237, 291, 347
SAD SAM (Sentence Appraiser and Diagrammer and Semantic Analyzing Machine [Cümle Değerlendiricisi ve Diyagramcısı ve Anlamsal Çözümleme Makinesi]) 154, 155

SAINT 126
saklı düğüm noktaları 493
Samuel, Arthur 81, 82, 128, 129, 130, 132, 133, 159, 254, 406, 527
Sandstorm (tadil edilmiş 1986 AM General M998 HMMWV) 619, 620
Sapir, Edward 544
Sastry, Shankar 645
SATPLAN (Planning as Satisfiability) 472, 473
satranç 14, 73, 78, 79, 82, 89, 118, 119, 127, 128, 158, 163, 167, 254, 255, 256, 257, 278, 279, 414, 425, 461, 514, 526, 603, 604, 605, 606, 607, 608, 647
Saunders, Rin 369, 370, 371
Savunma Bakanlığı Harita Birimi 345
Saxena, Ashutosh 555
Schaeffer, Jonathan 130, 608, 612
Schaeffer, Stephanie 611
Schank, Roger 209, 210, 211, 212, 213, 214, 422, 505
Schapire, Robert 537
Scheinman, Victor 194
Schickard, Wilhelm 57
Schmidt, Rodney 238
Schuler, Karin 455
Schultz, Alan 594
Schütze, Hinrich 544, 549
Schwartz, Jacob 382, 410, 411, 412, 413, 664, 671
SCORPIUS 376
Scrabble® (RRRRR işaretine bakkkkk) 612
SDC (System Development Şirketi) 208, 274, 276, 323, 378
SDS (Scientific Data Systems) 104, 155, 204, 274
Searle, John 388, 389, 390, 391, 392, 393, 418, 587
Sebestyen, George 109
SEE 185, 186
See5 515
Sejnowski, Terrence 174, 343, 518, 551
Selfridge, Oliver 77, 78, 79, 81, 85, 86, 87, 88, 89, 90, 91, 93, 94, 96, 106, 160, 572
Selman, Bart 466, 469, 472, 473
Seo, Hyojung 536
sesbilgisi 143, 279
sesbilim 143
sezgisel arama 84, 118, 167, 205, 223, 282, 284, 395, 412, 413, 610, 631, 667
Shafer, Glenn 304, 419
Shakey 43, 216, 219, 220, 221, 222, 223, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 239, 290, 342, 398, 425, 550, 573, 580, 624, 653, 676
Shanahan, Murray 444, 445
Shannon, Claude 62, 81, 82, 83, 118, 119, 127, 190, 513
Shapiro, Stuart 139, 209
Shaw, Cliff 81
Shawe-Taylor, John 539
Shaw, J. C. 124, 127
Shepherd, Roger 172
Shortliffe, Edward (Ted) 296, 298, 299, 300, 301, 302
SHRDLU 242, 243, 244, 245, 246
Shriai, Yoshiaki 189
SIAP (Surveillance Integration Automation Program) 326, 327, 328, 329, 330, 579
Sıfır Vizyonu 631
SIGART (Special Interest Group for ARTificial intelligence) 263, 264, 318
SIPE-2 237
SIR (Semantic Information Retrieveal) 137, 138, 139, 143, 153, 204, 215, 216, 442
sibernetik 52, 53
simge yapıları 115, 116, 125, 126, 134, 136, 392
Simmons, Robert 155, 156, 208
Simon, Herbert 42, 69, 70, 78, 79, 81, 82, 83, 84, 85, 115, 117,

118, 119, 124, 125, 126, 127, 132, 140, 153, 155, 157, 158, 159, 165, 167, 169, 182, 228, 233, 286, 391, 392, 394, 418, 430, 508, 510, 563, 590, 591, 595, 647, 663
Simpson, Robert 346, 375
sinaps 35, 37, 391, 399
Singer, Jonathan 192
sinir hücresi doktrini 34
Sinirsel Enformasyon İşleme Sistemleri (NIPS) Konferansları 540
Skinner, B. F. 39, 40, 41
Slagle, James 126, 158, 264
Slate, David 256
SL-kesin çözüm 206, 207
Sloman, Aaron 660, 674
SMARTPAL V 657
Smith, Brian 582
Smith, Edwin 297
Smith, Jack 442
Smith, Reid 585
SMM (Saklı Markov Modelleri) 281, 288, 497, 498, 499, 644, 677
SNePS (Semantic NEtwork Processing System) 209
SOAR (Sate, Operator And Result) 592, 595, 596, 597, 598, 599, 676
Sobel, Irwin 51, 180, 181
Sobel Operatörü 181
sombrero fonksiyonu 183
SOS 161, 162
Southern California Üniversitesi 269, 585
Sowa, John 139
sözcük ağları 454
sözcüksel işlevsel dilbilgileri (LFG) 542
sözdizim 154, 241, 249, 252
SPEECHLIS 276, 277
SPHINX 288, 377
Spiegelhalter, D. 540
Sputnik (Sovyet uydusu) 164
SRI (Stanford Araştırma Enstitüsü) 15, 43, 93, 99, 101, 102, 103, 104, 165, 169, 179, 202, 204, 213, 216, 217, 218, 219, 223, 227, 235, 236, 237, 274, 276, 287, 288, 290, 292, 293, 294, 295, 296, 298, 302, 305, 315, 318, 320, 324, 334, 344, 348, 355, 371, 375, 376, 378, 405, 442, 451, 516, 539, 642, 647, 658
STAIR (STanford AI Robot) 654, 655, 670
Stanford Arabası 238, 239
Stanford YZ Laboratuvarı (SAIL) 160, 161, 192, 193, 194
Stanhope, Charles 29, 30, 31
Stanhope Göstericisi 31
StatLog 540
Stefik, Mark 367
STeLLA 527
Stone, Charles 515
Stone, Philip 511
Stork, David 526
Stottler Henke 639, 640
Strachey, Christopher 128
Strat, Thomas 619
STRIPS (Stanford Research Institute Problem Solver [Stanford Araştırma Enstitüsü Problem Çözücü]) 228, 229, 233, 234, 236, 245, 442, 473, 588, 676
Surat Duyumsama Motoru (FSE) 645
Sutherland, Georgia 290, 302
Sutherland, Ivan 218
Sutton, Richard 132, 254, 526, 534
Symantec 320
Symbolics 347, 370
Syntelligence 312
Systems Control Technology, A.Ş. 327
Systran (System Translator) 240, 241

TacAir-SOAR 599
tahminsel uslamlama 445
taksonomi hiyerarşileri 442

Tate, Austin 237
Taube, Mortimer 387
tavla 526, 533, 534, 535, 603
tavsiye alıcı 86
TD-GAMMON 533, 534, 535
TDUS (Task Dialog Understanding System) 292, 294, 296
TEAM (Transportable English Database Access Medium) 324, 325
Teh, Yee-Whye 563
tekdüze olmayan uslamlama 444
Tekil Değer Ayrışması (TDA) 477
tekillik 664, 665
Teknowledge A.Ş. 312, 379, 380
teleo-reaktif programlar 53
Tenenbaum, J. Martin 334, 335, 336, 337, 338, 343, 344, 562
tepe tırmanma 46, 89, 468, 470, 492, 597
Tepkisel Eylem Paketleri (TEP) 574
TerraMax (tadil edilmiş bir nakliye aracı) 621
Tesauro, Gerald 533
Tesibius 50
TextRunner 661, 662
TherapyEdge HIV (HIV hastalarının idaresi için) 637
Thielscher, Michael 444
Thomas, Lewis 54, 108, 109, 111, 112, 128, 134, 135, 191, 288, 298, 337, 340, 401, 476, 480, 499, 567, 639, 659
Thorndike, Edward 526
Thorne, James 249
Thorpe, Charles 376
Thrun, Sebastian 622, 627
Tick, Evan 358
Tilden, Mark 433
Tinsley, Marion 608, 609
T kutusu 451
Tolhurst, David 174
Toma, Petr 240
Toplumsal Sorumluluk Sahibi Bilişim Profesyonelleri (CPSR) 381, 399
Torrance, Mark 426
TOTE birimleri 225
Trafton, Greg 594
Turing, Alan 59, 60, 63, 65, 67, 68, 127, 161, 385, 662, 663, 668
Turing makinesi 60, 61, 70, 453
Turing Merkezi 661
Turing Ödülü 116
Turing testi 65, 66, 666

Uluslararası Bilgisayar Oyunları Birliği (ICGA) 613
Uluslararası Birleşik Yapay Zekâ Konferansı (IJCAI) 193, 263, 318, 348, 601, 657
United Space Alliance (LLC) 639
UTSB (Japonya'nın Uluslararası Ticaret ve Sanayi Bakanlığı) 353, 354, 358
uzman sistemler 300, 304, 312, 347, 375, 379, 437, 590, 676
uzun süreli bellek (USB) 590, 597, 598

üçgen tablo 228
üçlü kule mimarisi 578
üst-pikseller 549, 553

VaMoRs-P (Almanların şoförsüz "gören yolcu arabası") 574
VaMP taşıtı (Mercedes-Benz 500 SEL) 559, 618
van Gelder, Timothy 431
van Melle, William 300
varlıkbilim 449, 452, 454, 455
Vaucanson, Jacques 22, 49
Veloso, Manuela 586
VerbNet 455
Vhayu Technologies Şirketi 641
VideoSurf 651
Vidoni, Frederic 23
Vinge, Vernor 664
von Neumann, John 63, 75, 115, 571, 572

von Neumann mimarisi 63, 571

Waibel, Alex 288
Wald, Abraham 56
WALKSAT 469, 472
Walter, Grey 15, 35, 46, 47, 50, 53, 56, 79, 133, 173, 216, 423
Waltz, David 188, 189
Wang, Hao 202
Warren, David 35, 53, 75, 150, 173, 321, 347
Watkins, Christopher 529
Watt, James 51
Weaver, Warren 150
Webber, Bonnie 247
WebFOCUS 641, 642
Wefald, Eric 583
Weiss, Sholom 308
Weizenbaum, Joseph 66, 153, 400, 401
Werbos, Paul 518
Weyhrauch, Richard 582
Wichman, Bill 192
Widrow, Bernard 101, 162
Wiener, Norbert 51, 53, 56, 82, 150
Wiesel, Torsten 173, 174
Wilkins, David E. 237, 297
Wilks, Yorick 659
Williams, Brian 617
Williams, Ronald 430
Winograd, Terry 215, 241, 242, 243, 245, 246, 253, 541
Winston, Patrick 188, 191
Wolf, Helen (Chan) 175
Woods, William 246, 247, 248, 249, 250, 253, 317, 450
WordNet 452, 453, 454, 455
Wundt, Wilhelm 38, 39

XCON (eXpert CONfigurer) 309, 310
Xerox 250, 349, 451, 651

Yapay Öğrenme Alet Kutusu (YÖA) 362
Yapay Zekâ Tekillik Enstitüsü (SIAI) 665
Yapay Zekâ Uygulamaları Enstitüsü 237, 506
yapı-içi görüntüler 336, 344
Yaskawa Electric Şirketi 657
Yayın Takibi Sistemi 642
yordamsal bilgi 200, 245, 446
Yordamsal Uslamlama Sistemi (YUS) 580, 581, 582
YouTube 660
Yovits, Marshall 164
yumuşak bilişim 421
yüklemcil biçimsel dizge 201, 226, 234, 247, 580
yüz tanıma 175, 176, 567, 644, 645
YZ kışı 349, 385, 416, 417, 437
YZ ve bilişsel bilim 264

Z1 bilgisayarı 127
Z3 bilgisayarı 63, 127
Zadeh, Lotfi 420, 421
zChaff 468
Zeki Fiziksel Etmenler Kurumu (FIPA) 589
Zuse, Horst 63
Zuse, Konrad 63, 127

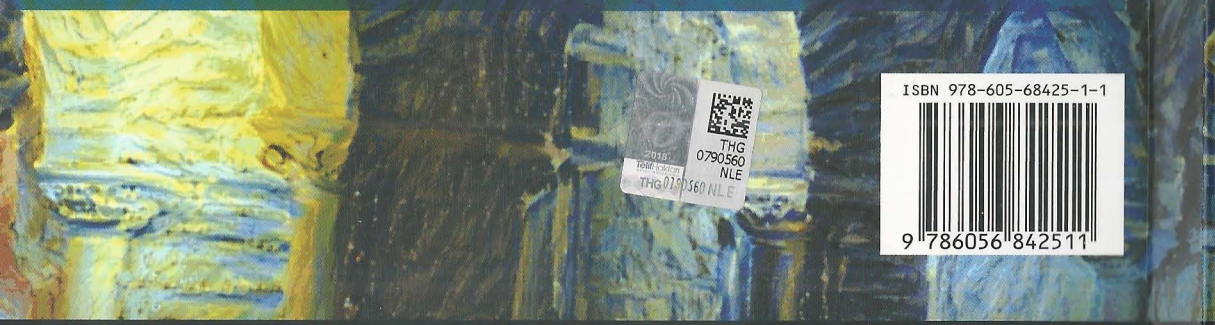
Yapay Zekâ (YZ) arayışında elli yıl boyunca yer aldım; bu elli yıl benim bütün meslek yaşantımı ve bu alanın bütün ömrünü kapsar. Bu arayışın başlangıcından bugününe kadar olan öyküsünü "içeriden birinin" anlatması bana iyi bir fikir gibi geldi. Aklımda üç tür okur var: Birincisi, bilimsel konulara meraklı ve YZ hakkında daha fazla bilgi edinmek isteyen okur topluluğu. İkinci grup ise, teknik ya da mesleki alanlarda çalışanlardan oluşuyor. Her iki okur zümresine de, kitapta karmaşık matematik formülleri ve bilgisayar jargonu olmayacağına, bol bol grafik sunacağıma, YZ programlarının ve tekniklerinin nasıl işlediğini berrak bir şekilde açıklamak için elimden geleni yapacağıma söz veriyorum. Üçüncü okur topluluğu, YZ araştırmacılarından, öğrencilerinden ve öğretmenlerinden oluşuyor. YZ'nin denediği, işe yaramış ya da yaramamış şeyleri bilmek, ayrıca tarihsel veya başka türlü bilgileri içeren sağlam bir kaynağa ulaşmak bu kesim için faydalı olacaktır. Bu alanın tarihini bilmek, bu alanda çalışanlar için önemli. Öncelikle, bir zamanlar keşfedilip sonra terk edilmiş olan pek çok fikir, teknolojik imkânların artmasıyla yaşam şansı bulabilir.

Elinizdeki kitap, YZ'nin tarihini aşağı yukarı belli bir zaman sırası takip ederek anlatıyor. Kimi aktörleri ve olayları içermiyor olabilir, ama umuyorum ki hikâyem, YZ'nin başlıca fikirlerini, anlaşmazlıklarını, uygulamalarını, sınırlarını makul biçimde yansıtabiliyordur. Bu işte yer almış şahsiyetlerden ziyade fikirlere ve bunların nasıl hayat bulduğuna odaklandım. Bana kalırsa, YZ tarihini anlamak için, YZ programlarının gerçekte nasıl çalıştığını en azından genel hatlarıyla kavramak önemli.

YZ'nin başlıca hedefi, yani aradığımız o büyük "ödül" nedir? Bana kalırsa Yapay Zekâ, insanların yapabildiği şeylerin çoğunu özellikle de "zekâ" gerektiren şeyleri yapan insan eseri araçlardır. YZ gelecekte pek çok alanda hayatımızda yer alacak gibi görünüyor. İnanıyorum ki günün birinde, YZ araştırmacıları bilinçli olduklarına herkesi inandıracak makineler inşa edebilecek. Bu hayalimizin gerçekleştiği zamanları hayal ederken bazı soruların zihnimizi kurcalamasına engel olamıyoruz: O günler geldiğinde, bu makinelerin bizlerle ve birbirleriyle savaşmasını olanaksız kılacak bir toplumsal düzen kurmayı başarabilecek miyiz? Hatta makineler için, sadece toplumsal açıdan kabul gören hedeflere hizmet etmenin ne anlama geldiğini tanımlamış olacak mıyız? Bu ve benzeri endişeleri hem bilgisayar bilimcileri hem de hümanistler taşıyor. Gelecekte yazarlar, bu maceranın öyküsünü anlatmayı hiç şüphesiz sürdürecek. Günün birinde bu yazarlardan biri, Aristoteles'in düşüncelerinden yaklaşık iki bin beş yüz yıl sonra, "ya bizim emrimizde ya da ihtiyaç halinde kendi işini icra eden" araçlarımızın olduğunu yazabilecek.

NILS JOHN NILSSON (1933) Amerikalı bilgisayar bilimci. Yapay Zekânın kurucu araştırmacılarından biridir. Stanford Üniversitesi'nde Bilgisayar Bilimi Bölümü'nde emekli profesör olarak ders vermektedir. Doktorasını 1958'de aynı üniversitenin Elektrik Mühendisliği Bölümü'nden almıştır. *Artificial Intelligence ve Journal of Artificial Intelligence Research* dergilerinin yayın kurullarında görev almıştır. *Journal of the Association for Computing Machinery* dergisinin uzman editörlüğünü yapmıştır.

Kapak Resmi: "Yıldızlı Stanford". Shubhang Desai'ın, Vincent van Gogh'un "Yıldızlı Gece" tablosunun üslubunu bir yapay zekâ uygulamasına öğreterek Stanford Üniversitesi'ndeki Hoover Kulesi'nin fotoğrafından oluşturduğu resim.



THG 0790560 NLE

ISBN 978-605-68425-1-1

9 786056 842511

BOĞAZİÇİ ÜNİVERSİTESİ YAYINEVİ