SIXTH EDITION

Artificial

Intelligence

Structures and

Strategies

for Complex

ACTE

Problem

Solving

George F Luger

SIXTH EDITION

*This page intentionally left blank*

Boston San Francisco New York London Toronto Sydney Tokyo Singapore Madrid Mexico City Munich Paris Cape Town Hong Kong Montreal

SIXTH EDITION

Structures and Strategies for Complex Problem Solving

George F **Luger**

*University of New Mexico*

Executive Editor Michael Hirsch Acquisitions Editor Matt Goldstein Editorial Assistant Sarah Milmore Associate Managing Editor Jeffrey Holcomb Digital Assets Manager Marianne Groth Senior Media Producer Bethany Tidd Marketing Manager Erin Davis Senior Author Support/

Technology Specialist Joe Vetere Senior Manufacturing Buyer Carol Melville Text Design, Composition, and Illustrations George F Luger Cover Design Barbara Atkinson Cover Image © Tom Barrow

For permission to use copyrighted material, grateful acknowledgment is made to the copyright holders listed on page xv, which is hereby made part of this copyright page.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Library of Congress Cataloging-in-Publication Data

Luger, George F.

Artificial intelligence : structures and strategies for complex problem solving / George F. Luger.-- 6th ed.

p. cm. Includes bibliographical references and index. ISBN-13: 978-0-321-54589-3 (alk. paper) 1. Artificial intelligence. 2. Knowledge representation (Information theory) 3. Problem solving. 4. PROLOG (Computer program language) 5. LISP (Computer program language) I. Title.

Q335.L84 2008 006.3--dc22

2007050376

Copyright © 2009 Pearson Education, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America. For information on obtaining permission for use of material in this work, please submit a written request to Pearson Education, Inc., Rights and Contracts Department, 501 Boylston Street, Suite 900, Boston, MA 02116, fax (617) 671-3447, or online at http://www.pearsoned.com/legal/permissions.htm.

ISBN-13: 978-0-321-54589-3 ISBN-10: 0-321-54589-3

1 2 3 4 5 6 7 8 9 10—CW—12 11 10 09 08

For my wife, Kathleen, and our children Sarah, David, and Peter.

*Si quid est in me ingenii, judices . . .*

Cicero, Pro Archia Poeta

GFL

*This page intentionally left blank*

PREFACE

*What we have to learn to do we learn by doing. . .*

—ARISTOTLE, *Ethics*

Welcome to the Sixth Edition!

I was very pleased to be asked to produce the sixth edition of my artificial intelligence book. It is a compliment to the earlier editions, started over twenty years ago, that our approach to AI has been so highly valued. It is also exciting that, as new development in the field emerges, we are able to present much of it in each new edition. We thank our many readers, colleagues, and students for keeping our topics relevant and our presenta- tion up to date.

Many sections of the earlier editions have endured remarkably well, including the presentation of logic, search algorithms, knowledge representation, production systems, machine learning, and, in the supplementary materials, the programming techniques developed in Lisp, Prolog, and with this edition, Java. These remain central to the practice of artificial intelligence, and a constant in this new edition.

This book remains accessible. We introduce key representation techniques including logic, semantic and connectionist networks, graphical models, and many more. Our search algorithms are presented clearly, first in pseudocode, and then in the supplementary mate- rials, many of them are implemented in Prolog, Lisp, and/or Java. It is expected that the motivated students can take our core implementations and extend them to new exciting applications.

We created, for the sixth edition, a new machine learning chapter based on stochastic methods (Chapter 13). We feel that the stochastic technology is having an increasingly larger impact on AI, especially in areas such as diagnostic and prognostic reasoning, natu- ral language analysis, robotics, and machine learning. To support these emerging technol- ogies we have expanded the presentation of Bayes' theorem, Markov models, Bayesian

PREFACE vii

belief networks, and related graphical models. Our expansion includes greater use of prob- abilistic finite state machines, hidden Markov models, and dynamic programming with the Earley parser and implementing the Viterbi algorithm. Other topics, such as emergent computation, ontologies, stochastic parsing algorithms, that were treated cursorily in ear- lier editions, have grown sufficiently in importance to merit a more complete discussion. The changes for the sixth edition reflect emerging artificial intelligence research questions and are evidence of the continued vitality of our field.

As the scope of our AI project grew, we have been sustained by the support of our publisher, editors, friends, colleagues, and, most of all, by our readers, who have given our work such a long and productive life. We remain excited at the writing opportunity we are afforded: Scientists are rarely encouraged to look up from their own, narrow research interests and chart the larger trajectories of their chosen field. Our readers have asked us to do just that. We are grateful to them for this opportunity. We are also encouraged that our earlier editions have been used in AI communities worldwide and translated into a number of languages including German, Polish, Portuguese, Russian, and two dialects of Chinese! Although artificial intelligence, like most engineering disciplines, must justify itself to the world of commerce by providing solutions to practical problems, we entered the field of AI for the same reasons as many of our colleagues and students: we want to under- stand and explore the mechanisms of mind that enable intelligent thought and action. We reject the rather provincial notion that intelligence is an exclusive ability of humans, and believe that we can effectively investigate the space of possible intelligences by designing and evaluating intelligent artifacts. Although the course of our careers has given us no cause to change these commitments, we have arrived at a greater appreciation for the scope, complexity, and audacity of this undertaking. In the preface to our earlier editions, we outlined three assertions that we believed distinguished our approach to teaching artifi- cial intelligence. It is reasonable, in writing a preface to the present edition, to return to these themes and see how they have endured as our field has grown.

The first of these goals was *to unify the diverse branches of AI through a detailed dis- cussion of its theoretical foundations*. At the time we first adopted that goal, it seemed that the main problem was in reconciling researchers who emphasized the careful statement and analysis of formal theories of intelligence (the *neats*) with those who believed that intelligence itself was some sort of grand hack that could be best approached in an appli- cation-driven, *ad hoc* manner (the *scruffies*). That dichotomy has proven far too simple.

In contemporary AI, debates between neats and scruffies have given way to dozens of other debates between proponents of physical symbol systems and students of neural net- works, between logicians and designers of artificial life forms that evolve in a most illogi- cal manner, between architects of expert systems and case-based reasoners, and finally, between those who believe artificial intelligence has already been achieved and those who believe it will never happen. Our original image of AI as frontier science where outlaws, prospectors, wild-eyed prairie prophets and other dreamers were being slowly tamed by the disciplines of formalism and empiricism has given way to a different metaphor: that of a large, chaotic but mostly peaceful city, where orderly bourgeois neighborhoods draw their vitality from diverse, chaotic, bohemian districts. Over the years that we have devoted to the different editions of this book, a compelling picture of the architecture of intelligence has started to emerge from this city's structure, art, and industry.

viii PREFACE

Intelligence is too complex to be described by any single theory; instead, researchers are constructing a hierarchy of theories that characterize it at multiple levels of abstrac- tion. At the lowest levels of this hierarchy, neural networks, genetic algorithms and other forms of emergent computation have enabled us to understand the processes of adaptation, perception, embodiment, and interaction with the physical world that must underlie any form of intelligent activity. Through some still partially understood resolution, this chaotic population of blind and primitive actors gives rise to the cooler patterns of logical infer- ence. Working at this higher level, logicians have built on Aristotle's gift, tracing the out- lines of deduction, abduction, induction, truth-maintenance, and countless other modes and manners of reason. At even higher levels of abstraction, designers of diagnostic sys- tems, intelligent agents, and natural language understanding programs have come to rec- ognize the role of social processes in creating, transmitting, and sustaining knowledge.

At this point in the AI enterprise it looks as though the extremes of rationalism and empiricism have only led to limited results. Both extremes suffer from limited applicabil- ity and generalization. The author takes a third view, that the empiricist's conditioning: semantic nets, scripts, subsumption architectures and the rationalist's clear and distinct ideas: predicate calculus, non-monotonic logics, automated reasoning - suggest a third viewpoint, the Bayesian. The experience of relational invariances conditions intelligent agents's expectations, and learning these invariances, in turn, bias future expectations. As philosophers we are charged to critique the epistemological validity of the AI enterprise. For this task, in Chapter 16 we discuss the rationalist project, the empiricists dilemma, and propose a Bayesian based constructivist rapprochement. In this sixth edition, we touch on all these levels in the presenting the AI enterprise.

The second commitment we made in earlier editions was to *the central position of advanced representational formalisms and search techniques in AI methodology*. This is, perhaps, the most controversial aspect of our previous editions and of much early work in AI, with many researchers in emergent computation questioning whether symbolic rea- soning and referential semantics have any role at all in intelligence. Although the idea of representation as giving names to things has been challenged by the implicit representa- tion provided by the emerging patterns of a neural network or an artificial life, we believe that an understanding of representation and search remains essential to any serious practi- tioner of artificial intelligence. We also feel that our Chapter 1 overview of the historical traditions and precursors of AI are critical components of AI education. Furthermore, these are invaluable tools for analyzing such aspects of non-symbolic AI as the expressive power of a neural network or the progression of candidate problem solutions through the fitness landscape of a genetic algorithm. Comparisons, contrasts, and a critique of modern AI are offered in Chapter 16.

Our third commitment was made at the beginning of this book's life cycle: to place artificial intelligence within the context of empirical science. In the spirit of the Newell and Simon (1976) Turing award lecture we quote from an earlier edition:

... AI is not some strange aberration from the scientific tradition, but . . . part of a general quest for knowledge about, and the understanding of, intelligence itself. Furthermore, our AI programming tools, along with the exploratory programming methodology . . . are ideal for exploring an environment. Our tools give us a medium for both understanding

PREFACE ix

and questions. We come to appreciate and know phenomena constructively, that is, by pro- gressive approximation.

Thus we see each design and program as an experiment with nature: we propose a representation, we generate a search algorithm, and we question the adequacy of our char- acterization to account for part of the phenomenon of intelligence. And the natural world gives a response to our query. Our experiment can be deconstructed, revised, extended, and run again. Our model can be refined, our understanding extended.

New with The Sixth Edition

The biggest change for the sixth edition is the extension of the stochastic approaches to AI. To accomplish this we revised Section 9.3 and added a new chapter (13) introducing probability-based machine learning. Our presentation of stochastic AI tools and their application to learning and natural language is now more comprehensive.

From probability theory's foundations in set theory and counting we develop the notions of probabilities, random variables, and independence. We present and use Bayes' theorem first with one symptom and one disease and then in its full general form. We examine the hypotheses that underlie the use of Bayes and then present the *argmax* and *naive Bayes* approaches. We present examples of stochastic reasoning, including the anal- ysis of language phenomena and the Vierbi algorithm. We also introduce the idea of condi- tional independence that leads to Bayesian belief networks, the BBN, in Chapter 9.

In Chapter 13 we introduce hidden Markov models, HMMs, and show their use in several examples. We also present several HMM variants, including the auto-regressive and hierarchical HMMs. We present dynamic Bayesian networks, DBNs, and demonstrate their use. We discuss parameter and structure learning and present the expectation maxi- mization algorithm and demonstrate its use with loopy belief propagation. Finally, we present Markov decision processes, the MDP, and partially observable Markov decision process, the POMDP, in the context of an extension to the earlier presentation of reinforce- ment learning.

We include several more examples of *probabilistic finite state machines* and *probabi- listic acceptors*, as well as the use of *dynamic programming*, especially with stochastic measures (the *Viterbi algorithm*). We added a stochastic English language parser (based on the work of Mark Steedman at the University of Edinburgh) as well as the use of dynamic programming with the Earley parser.

We made a major decision to remove the Prolog and Lisp chapters from the book. Part of the reason for this is that these were getting too large. We have also accumulated a number of AI algorithms written in Java. When we added the new Chapter 13 on stochas- tic approaches to machine learning, we determined that the book was getting too large/ cumbersome. Thus the sixth edition is more than 150 pages smaller than the fifth and the AI algorithms in Prolog, Lisp, and Java are being released as supplementary materials. From our earliest days in AI we have always felt that the way to understand the power (and limitations) of AI algorithms is constructively - that is, by building them! We encour- age our present generation of readers to do exactly this: to visit the supplementary materi- als: to build and experiment directly with the algorithms we present.

x PREFACE

Finally, we have done the usual updating of references and materials that a new edi- tion warrants. In a revised Chapter 16, we return to the deeper questions on the nature of intelligence and the possibility of creating intelligent machines.

Sixth Edition: The Contents

**Chapter 1 introduces artificial intelligence**, beginning with a brief history of attempts to understand mind and intelligence in philosophy, psychology, and other areas of research. In an important sense, AI is an old science, tracing its roots back at least to Aristotle. An appreciation of this background is essential for an understanding of the issues addressed in modern research. We also present an overview of some of the important application areas in AI. Our goal in Chapter 1 is to provide both background and a motivation for the theory and applications that follow.

**Chapters 2, 3, 4, 5, and 6 (Part II) introduce the research tools for AI problem solving**. These include, in Chapter 2, the predicate calculus presented both as a mathemat- ical system as well as a representation language to describe the essential features of a problem. Search, and the algorithms and data structures used to implement search, are introduced in Chapter 3, to organize the exploration of problem situations. In Chapter 4, we discuss the essential role of heuristics in focusing and constraining search-based prob- lem solving. In Chapter 5, we introduce the stochastic methodology, important technology for reasoning in situations of uncertainty. In Chapter 6, we present a number of software architectures, including the blackboard and production system, for implementing these search algorithms.

**Chapters 7, 8, and 9 make up Part III: representations for AI, knowledge-inten- sive problem solving, and reasoning in changing and ambiguous situations**. In Chap- ter 7 we present the evolving story of AI representational schemes. We begin with a discussion of association-based networks and extend this model to include conceptual dependency theory, frames, and scripts. We then present an in-depth examination of a par- ticular formalism, conceptual graphs, emphasizing the epistemological issues involved in representing knowledge and showing how these issues are addressed in a modern repre- sentation language. Expanding on this formalism in Chapter 14, we show how conceptual graphs can be used to implement a natural language database front end. We conclude Chapter 7 with more modern approaches to representation, including Copycat and agent- oriented architectures.

Chapter 8 presents the rule-based expert system along with case-based and model- based reasoning, including examples from the NASA space program. These approaches to problem solving are presented as a natural evolution of the material in Part II: using a pro- duction system of predicate calculus expressions to orchestrate a graph search. We end with an analysis of the strengths and weaknesses of each of these approaches to knowl- edge-intensive problem solving.

Chapter 9 presents models for reasoning with uncertainty as well as the use of unreli- able information. We introduce Bayesian models, belief networks, Dempster-Shafer, causal models, and the Stanford certainty algebra for reasoning in uncertain situations.

PREFACE xi

Chapter 9 also contains algorithms for truth maintenance, reasoning with minimum mod- els, logic-based abduction, and the clique-tree algorithm for Bayesian belief networks.

**Part IV, Chapters 10 through 13, is an extensive presentation of issues in machine learning**. In Chapter 10 we offer a detailed look at algorithms for symbol-based learning, a fruitful area of research spawning a number of different problems and solution approaches. These learning algorithms vary in their goals, the training data considered, their learning strategies, and the knowledge representations they employ. Symbol-based learning includes induction, concept learning, version-space search, and ID3. The role of inductive bias is considered, generalizations from patterns of data, as well as the effective use of knowledge to learn from a single example in explanation-based learning. Category learning, or conceptual clustering, is presented with unsupervised learning. Reinforcement learning, or the ability to integrate feedback from the environment into a policy for mak- ing new decisions concludes the chapter.

In Chapter 11 we present neural networks, often referred to as sub-symbolic or con- nectionist models of learning. In a neural net, information is implicit in the organization and weights on a set of connected processors, and learning involves a re-arrangement and modification of the overall weighting of nodes and structure of the system. We present a number of connectionist architectures, including perceptron learning, backpropagation, and counterpropagation. We demonstrate Kohonen, Grossberg, and Hebbian models. We present associative learning as well as attractor models, including examples of Hopfield networks.

Genetic algorithms and evolutionary approaches to learning are introduced in Chapter 12. On this viewpoint, learning is cast as an emerging and adaptive process. After several examples of problem solutions based on genetic algorithms, we introduce the application of genetic techniques to more general problem solvers. These include classifier systems and genetic programming. We then describe society-based learning with examples from artificial life, called *a-life*, research. We conclude the chapter with an example of emergent computation from research at the Santa Fe Institute.

Chapter 13 presents stochastic approaches to machine learning. We begin with a defi- nition of hidden markov models and then present several important variations including the auto-regressive and hierarchical HMM. We then present dynamic Bayesian networks, a generalization of the HMM, and also able to track systems across periods of time. These techniques are useful for modeling the changes in complex environments as is required for diagnostic and prognostic reasoning. Finally, we add a probabilistic component to rein- forcement learning first introduced in Chapter 10. This includes presentation of the Markov decision process (or MDP) and the partially observed Markov decision process (or POMDP).

**Part V, Chapters 14 and 15, presents automated reasoning and natural language understanding.** Theorem proving, often referred to as automated reasoning, is one of the oldest areas of AI research. In Chapter 14, we discuss the first programs in this area, including the Logic Theorist and the General Problem Solver. The primary focus of the chapter is binary resolution proof procedures, especially resolution refutations. More advanced inferencing with hyper-resolution and paramodulation is also presented. Finally, we describe the Prolog interpreter as a Horn clause and resolution-based inferencing sys- tem, and see Prolog computing as an instance of the logic programming paradigm.

xii PREFACE

Chapter 15 presents natural language understanding. Our traditional approach to lan- guage understanding, exemplified by many of the semantic structures presented in Chap- ter 7, is complemented with the stochastic approach. These include using Markov models, CART trees, CHART parsing (the Earley algorithm), mutual information clustering, and statistics-based parsing. The chapter concludes with examples applying natural language techniques to database query generation, a text summarization systems well as.the use of machine learning to generalize extracted results from the WWW.

**Finally, Chapter 16 serves as an epilogue for the book.** It addresses the issue of the possibility of a science of intelligent systems, and considers contemporary challenges to AI; it discusses AI's current limitations, and projects its exciting future. Using This Book

Artificial intelligence is a big field, and consequently, this is a large book. Although it would require more than a single semester to cover all of the material offered, we have designed our book so that a number of paths may be taken through the material. By select- ing subsets of the material, we have used this text for single semester and full year (two semester) courses.

We assume that most students will have had introductory courses in discrete mathe- matics, including predicate calculus, set theory, counting, and graph theory. If this is not true, the instructor should spend more time on these concepts in the “optional” sections at the beginning of the introductory chapters (2.1, 3.1, and 5.1). We also assume that students have had courses in data structures including trees, graphs, and recursion-based search, using stacks, queues, and priority queues. If they have not, then spend more time on the beginning sections of Chapters 3, 4, and 6.

In a one quarter or one semester course, we go quickly through the first two parts of the book. With this preparation, students are able to appreciate the material in Part III. We then consider the Prolog, Lisp, or the Java code in the supplementary materials for the book and require students to build many of the representation and search techniques of the second part of the book. Alternatively, one of the languages, Prolog, for example, can be introduced early in the course and be used to test out the data structures and search tech- niques as that are encountered. We feel the meta-interpreters presented in the language materials are very helpful for building rule-based and other knowledge-intensive problem solvers. Prolog, Lisp, and Java are excellent tools for building natural language under- standing and learning systems; these architectures are presented in Parts II and III and there are examples of them in the supplementary course materials.

In a two-semester or three-quarter course, we are able to cover the application areas of Parts IV and V, especially the machine learning chapters, in appropriate detail. We also expect a much more detailed programming project from students. We also think that it is very important in the second semester for students to revisit many of the primary sources of the AI literature. It is crucial for students to see both where we are in the evolution of the AI enterprise, as well as how we got here, and to have an appreciation of the future promises of artificial intelligence. We use materials from the WWW for this purpose or select a collected set of readings, such as, *Computation and Intelligence* (Luger 1995).

PREFACE xiii

The algorithms of our book are described using a Pascal-like pseudo-code. This nota- tion uses the control structures of Pascal along with English descriptions of the tests and operations. We have added two useful constructs to the Pascal control structures. The first is a modified case statement that, rather than comparing the value of a variable with con- stant case labels, as in standard Pascal, lets each item be labeled with an arbitrary boolean test. The case evaluates these tests in order until one of them is true and then performs the associated action; all other actions are ignored. Those familiar with Lisp will note that this has the same semantics as the Lisp cond statement.

The other addition to our pseudo code language is a return statement which takes one argument and can appear anywhere within a procedure or function. When the return is encountered, it causes the program to immediately exit the function, returning its argu- ment as a result. Other than these modifications we used Pascal structure, with a reliance on the English descriptions, to make the algorithms clear.

Supplemental Material Available

The sixth edition has an ***attached web site*** maintained by my graduate students. This site, built originally by two UNM students, Alejandro CdeBaca and Cheng Liu, includes sup- plementary ideas for most chapters, some sample problems with their solutions, and ideas for student projects. Besides the ***Prolog, Lisp, and Java programs*** in the supplementary materials for this book, we have included many other AI algorithms in Java and C++ on the web site. Students are welcome to use these and supplement them with their own com- ments, code, and critiques. The web url is www.cs.unm.edu/~luger/ai-final/.

The ***Prolog, Lisp, and Java programs*** implementing many of the AI data structures and search algorithms of the book are available through your Addison-Wesley Pearson Education representative. There is also an ***Instructor’s Guide*** available which has many of the book's exercises worked out, several practice tests with solutions, a sample syllabus, and ideas supporting teaching the material. There are also a full set of ***PowerPoint presen- tation materials*** for use by instructors adopting this book. Again, consult your local A-W Pearson representative for access and visit www.aw.com/luger.

My e-mail address is luger@cs.unm.edu, and I enjoy hearing from my readers.

Acknowledgements

Although I am the sole author of the sixth edition, this book has always been the product of my efforts as Professor of Computer Science, Psychology, and Linguistics at the Uni- versity of New Mexico along with my fellow faculty, professional colleagues, graduate students, and friends. The sixth edition is also the product of the many readers that have e- mailed comments, corrections, and suggestions. The book will continue this way, reflect- ing a *community* effort; consequently, I will continue using the prepositions *we, our,* and *us* when presenting material.

I thank Bill Stubblefield, the co-author for the first three editions, for more than twenty years of contributions, but even more importantly, for his friendship. I also thank

xiv PREFACE

the many reviewers that have helped develop this and earlier editions. These include Den- nis Bahler, Leonardo Bottaci, Skona Brittain, Philip Chan, Peter Collingwood, Mehdi Dastani, John Donald, Sarah Douglas, Christophe Giraud-Carrier, Andrew Kosoresow, Terran Lane, Chris Malcolm, Ray Mooney, Marek Perkowski, Barak Pearmutter, Dan Pless, Bruce Porter, Stuart Shapiro, Julian Richardson, Jude Shavlik, John Sheppard, Carl Stern, Leon van der Torre, Marco Valtorta, and Bob Veroff. We also appreciate the numer- ous suggestions and comments sent directly by e-mail by readers. Finally, Chris Malcolm, Brendan McGonnigle, and Akasha Tang, critiqued Chapter 16.

From our UNM colleagues, we thank Dan Pless, Nikita Sakhanenko, Roshan Ram- mohan, and Chayan Chakrabarti for their major role in developing materials for Chapters 5, 9, and 13; Joseph Lewis for his efforts on Chapters 9 and 16; Carl Stern for his help in developing Chapter 11 on connectionist learning; Bob Veroff for his critique of the auto- mated reasoning material in Chapter 14; and Jared Saia, Stan Lee, and Paul dePalma for helping with the stochastic approaches to natural language understanding of Chapter 15.

We thank Academic Press for permission to reprint much of the material of Chapter 11; this first appeared in the book *Cognitive Science: The Science of Intelligent Systems* (Luger 1994). Finally, we thank more than two decades of students who have used various versions of this book and software at UNM for their help in expanding our horizons, as well as in removing typos and bugs.

We thank our many friends at Benjamin-Cummings, Addison-Wesley-Longman, and Pearson Education for their support and encouragement in completing the writing task of our six editions, especially Alan Apt in helping us with the first edition, Lisa Moller and Mary Tudor for their help on the second, Victoria Henderson, Louise Wilson, and Karen Mosman for their assistance on the third, Keith Mansfield, Karen Sutherland, and Anita Atkinson for support on the fourth, Keith Mansfield, Owen Knight, Anita Atkinson, and Mary Lince for their help on the fifth edition, and Simon Plumtree, Matt Goldstein, Joe Vetere, and Sarah Milmore for their help on the sixth. Katherine Haratunian of Addison Wesley has had a huge role in seeing that Professors received their Instructor Guides and PowerPoint presentation materials (These are maintained by Addison-Wesley Pearson and available only through your local sales representative). Linda Cicarella of the University of New Mexico helped prepare many of the figures.

We thank Thomas Barrow, internationally recognized artist and University of New Mexico Professor of Art (emeritus), who created the photograms for this book.

Artificial intelligence is an exciting and rewarding discipline; may you enjoy your study as you come to appreciate its power and challenges.

George Luger 1 January 2008 Albuquerque

PREFACE xv

xvi

PUBLISHER’S ACKNOWLEDGEMENTS

We are grateful to the following for permission to reproduce copyright material:

Figures 4.8 and 4.9, Tables 5.2 and 5.3 adapted from Figure 5.6, p. 157, Figure 5.18, p. 178, Figure 5.20, p. 180 and data on p. 167, from *Speech and Language Processing: an introduction to natural language processing, computation linguistics, and speech recognition,* Prentice Hall, (Pearson Education, Inc.), (Jurafsky, D., and Martin, J. H., 2000); Figure 5.3 adapted from Figure 5.12, p. 170, from *Speech and Language Processing: an introduction to natural language processing, computational linguistics, and speech recognition,* Prentice Hall, (Pearson Education, Inc.), (Jurafsky, D., and Martin, J.H., 2000), which was itself adapted from a figure from *Artificial Intelligence: A Modern approach*, 1st Edition, Prentice Hall, (Pearson Education, Inc.), (Russell, S. J., and Norvig, P., 1995); Figure 7.1 from figure from *Expert Systems: Artificial Intelligence in Business*, by Harmon, P. and King, D., John Wiley & Sons, Inc., Copyright © 1985 Paul Harmon and David King. This material is used by permission of John Wiley & Sons, Inc.; Figures 7.6, 7.9, and 7.10 from figures from “Inference and the computer understanding of natural language,” in *Artificial Intelligence*, Vol. 5, No. 4, 1974, pp. 373–412, Copyright © 1974, reprinted with permission from Elsevier Science, (Schank, R. C., and Reiger, C. J., 1974); Figure 7.27 from figures from *Analogy-Making as Perception: A Computer Model*, The MIT Press, (Mitchell, M., 1993); Figure 9.2 from “an improved algorithm for non-monotonic dependency net update,” in *Technical Report LITH-MAT-R-82-83*, reprinted by permission of the author, (Goodwin, J., 1982); Figure10.21 adapted from figure from “models of incremental concept formation,” in *Artificial Intelligence*, Vol. 40, Nos. 1–3, 1989, pp. 11-62, Copyright © 1989, reprinted with permission from Elsevier Science, (Gennari, J. H., Langley, P., and Fisher, D., 1989); Figure 11.18 from part of Figure 6.2, p. 102, from *Introduction to Support Vector Machines: and other kernel-based learning methods*, Cambridge University Press, (Cristianini, N., and Shawe-Taylor, J., 2000).

Academic Press for Chapter 11, adapted from *Cognitive Science: The Science of Intelligent Systems*, (Luger, G. F., 1994). American Society for Public Administration for an abridged extract from “Decision-making and administrative organization,” in *Public Administration Review*, Vol. 4, Winter 1944, (Simon, H. A., 1944).

In some instances we have been unable to trace the owners of copyright materials, and we would appreciate any information that would enable us to do so.

BRIEF CONTENTS

Preface vii Publisher’s Acknowledgements xv

PART I ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE 1

1 AI: HISTORY AND APPLICATIONS 3

PART II ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 35

2 THE PREDICATE CALCULUS 45

3 STRUCTURES AND STRATEGIES FOR STATE SPACE SEARCH 79

4 HEURISTIC SEARCH 123

5 STOCHASTIC METHODS 165

6 CONTROL AND IMPLEMENTATION OF STATE SPACE SEARCH 193

PART III CAPTURING INTELLIGENCE: THE AI CHALLENGE 223

7 KNOWLEDGE REPRESENTATION 227

xvii

PART III (continued)

8 STRONG METHOD PROBLEM SOLVING 277

9 REASONING IN UNCERTAIN SITUATIONS 333

PART IV MACHINE LEARNING 385

10 MACHINE LEARNING: SYMBOL-BASED 387

11 MACHINE LEARNING: CONNECTIONIST 453

12 MACHINE LEARNING: GENETIC AND EMERGENT 507

13 MACHINE LEARNING: PROBABILISTIC 543

PART V ADVANCED TOPICS FOR AI PROBLEM SOLVING 573

14 AUTOMATED REASONING 575

15 UNDERSTANDING NATURAL LANGUAGE 619

PART VI EPILOGUE 671

16 ARTIFICIAL INTELLIGENCE AS EMPIRICAL ENQUIRY 673

Bibliography 705 Author Index 735 Subject Index 743

xviii BRIEF CONTENTS

CONTENTS

Preface vii Publisher’s Acknowledgements xv

PART I ARTIFICIAL INTELLIGENCE: ITS ROOTS

AND SCOPE 1

1 AI: HISTORY AND APPLICATIONS 3 1.1 From Eden to ENIAC: Attitudes toward Intelligence, Knowledge, and

Human Artifice 3 1.2 Overview of AI Application Areas 20 1.3 Artificial Intelligence—A Summary 30 1.4 Epilogue and References 31 1.5 Exercises 33

PART II ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 35

2 THE PREDICATE CALCULUS 45 2.0 Introduction 45 2.1 The Propositional Calculus 45 2.2 The Predicate Calculus 50 2.3 Using Inference Rules to Produce Predicate Calculus Expressions 62 2.4 Application: A Logic-Based Financial Advisor 73 2.5 Epilogue and References 77 2.6 Exercises 77

xix

3 STRUCTURES AND STRATEGIES FOR STATE SPACE SEARCH 79 3.0 Introduction 79 3.1 Graph Theory 82 3.2 Strategies for State Space Search 93 3.3 Using the State Space to Represent Reasoning with the Predicate Calculus 107 3.4 Epilogue and References 121 3.5 Exercises 121

4 HEURISTIC SEARCH 123 4.0 Introduction 123 4.1 Hill Climbing and Dynamic Programming 127 4.2 The Best-First Search Algorithm 133 4.3 Admissibility, Monotonicity, and Informedness 145 4.4 Using Heuristics in Games 150 4.5 Complexity Issues 157 4.6 Epilogue and References 161 4.7 Exercises 162

5 STOCHASTIC METHODS 165 5.0 Introduction 165 5.1 The Elements of Counting 167 5.2 Elements of Probability Theory 170 5.3 Applications of the Stochastic Methodology 182 5.4 Bayes’ Theorem 184 5.5 Epilogue and References 190 5.6 Exercises 191

6 CONTROL AND IMPLEMENTATION OF STATE SPACE SEARCH 193 6.0 Introduction 193 6.1 Recursion-Based Search 194 6.2 Production Systems 200 6.3 The Blackboard Architecture for Problem Solving 187 6.4 Epilogue and References 219 6.5 Exercises 220

PART III CAPTURING INTELLIGENCE: THE AI CHALLENGE 223

7 KNOWLEDGE REPRESENTATION 227 7.0 Issues in Knowledge Representation 227 7.1 A Brief History of AI Representational Systems 228

xx CONTENTS

7.2 Conceptual Graphs: A Network Language 248 7.3 Alternative Representations and Ontologies 258 7.4 Agent Based and Distributed Problem Solving 265 7.5 Epilogue and References 270 7.6 Exercises 273

8 STRONG METHOD PROBLEM SOLVING 277 8.0 Introduction 277 8.1 Overview of Expert System Technology 279 8.2 Rule-Based Expert Systems 286 8.3 Model-Based, Case Based, and Hybrid Systems 298 8.4 Planning 314 8.5 Epilogue and References 329 8.6 Exercises 331

9 REASONING IN UNCERTAIN SITUATIONS 333 9.0 Introduction 333 9.1 Logic-Based Abductive Inference 335 9.2 Abduction: Alternatives to Logic 350 9.3 The Stochastic Approach to Uncertainty 363 9.4 Epilogue and References 378 9.5 Exercises 380

PART IV MACHINE LEARNING 385

10 MACHINE LEARNING: SYMBOL-BASED 387 10.0 Introduction 387 10.1 A Framework for Symbol-based Learning 390 10.2 Version Space Search 396 10.3 The ID3 Decision Tree Induction Algorithm 408 10.4 Inductive Bias and Learnability 417 10.5 Knowledge and Learning 422 10.6 Unsupervised Learning 433 10.7 Reinforcement Learning 442 10.8 Epilogue and References 449 10.9 Exercises 450

11 MACHINE LEARNING: CONNECTIONIST 453 11.0 Introduction 453 11.1 Foundations for Connectionist Networks 455 11.2 Perceptron Learning 458 11.3 Backpropagation Learning 467 11.4 Competitive Learning 474

CONTENTS xxi

11.5 Hebbian Coincidence Learning 484 11.6 Attractor Networks or “Memories” 495 11.7 Epilogue and References 505 11.8 Exercises 506

12 MACHINE LEARNING: GENETIC AND EMERGENT 507 12.0 Genetic and Emergent Models of Learning 507 12.1 The Genetic Algorithm 509 12.2 Classifier Systems and Genetic Programming 519 12.3 Artificial Life and Society-Based Learning 530 12.4 Epilogue and References 541 12.5 Exercises 542

13 MACHINE LEARNING: PROBABILISTIC 543 13.0 Stochastic and Dynamic Models of Learning 543 13.1 Hidden Markov Models (HMMs) 544 13.2 Dynamic Bayesian Networks and Learning 554 13.3 Stochastic Extensions to Reinforcement Learning 564 13.4 Epilogue and References 568 13.5 Exercises 570

PART V ADVANCED TOPICS FOR AI PROBLEM SOLVING 573

14 AUTOMATED REASONING 575 14.0 Introduction to Weak Methods in Theorem Proving 575 14.1 The General Problem Solver and Difference Tables 576 14.2 Resolution Theorem Proving 582 14.3 PROLOG and Automated Reasoning 603 14.4 Further Issues in Automated Reasoning 609 14.5 Epilogue and References 666 14.6 Exercises 667

15 UNDERSTANDING NATURAL LANGUAGE 619 15.0 The Natural Language Understanding Problem 619 15.1 Deconstructing Language: An Analysis 622 15.2 Syntax 625 15.3 Transition Network Parsers and Semantics 633 15.4 Stochastic Tools for Language Understanding 649 15.5 Natural Language Applications 658 15.6 Epilogue and References 630 15.7 Exercises 632

xxii CONTENTS

PART VI EPILOGUE 671

16 ARTIFICIAL INTELLIGENCE AS EMPIRICAL ENQUIRY 673 16.0 Introduction 673 16.1 Artificial Intelligence: A Revised Definition 675 16.2 The Science of Intelligent Systems 688 16.3 AI: Current Challanges and Future Direstions 698 16.4 Epilogue and References 703

Bibliography 705 Author Index 735 Subject Index 743

CONTENTS xxiii

**з;332**

**Ем Арык**

**vггvИЕВпки avrГЕНДео МЕг. СнавЕЕ**

ре! **Горосы**

28

P A R T I

**ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE**

*Everything must have a beginning, to speak in Sanchean phrase; and that beginning must be linked to something that went before. Hindus give the world an elephant to support it, but they make the elephant stand upon a tortoise. Invention, it must be humbly admitted, does not consist in creating out of void, but out of chaos; the materials must, in the first place, be afforded. . . .*

—MARY SHELLEY, Frankenstein

Artificial Intelligence: An Attempted Definition

*Artificial intelligence (AI) may be defined as the branch of computer science that is concerned with the automation of intelligent behavior.* This definition is particularly appropriate to this book in that it emphasizes our conviction that AI is a part of computer science and, as such, must be based on sound theoretical and applied principles of that field. These principles include the data structures used in knowledge representation, the algorithms needed to apply that knowledge, and the languages and programming tech- niques used in their implementation.

However, this definition suffers from the fact that intelligence itself is not very well defined or understood. Although most of us are certain that we know intelligent behavior when we see it, it is doubtful that anyone could come close to defining intelligence in a way that would be specific enough to help in the evaluation of a supposedly intelligent computer program, while still capturing the vitality and complexity of the human mind.

As a result of the daunting task of building a general intelligence, AI researchers often assume the roles of engineers fashioning particular intelligent artifacts. These often come in the form of diagnostic, prognostic, or visualization tools that enable their human users to perform complex tasks. Examples of these tools include hidden Markov models for language understanding, automated reasoning systems for proving new theorems in math- ematics, dynammic Bayesian networks for tracking signals across cortical networks, and visualization of patterns of gene expression data, as seen in the applications of Section 1.2.

1

The problem of *defining* the full field of artificial intelligence becomes one of defining intelligence itself: is intelligence a single faculty, or is it just a name for a collection of dis- tinct and unrelated abilities? To what extent is intelligence learned as opposed to having an a priori existence? Exactly what does happen when learning occurs? What is creativity? What is intuition? Can intelligence be inferred from observable behavior, or does it require evidence of a particular internal mechanism? How is knowledge represented in the nerve tissue of a living being, and what lessons does this have for the design of intelligent machines? What is self-awareness; what role does it play in intelligence? Furthermore, is it necessary to pattern an intelligent computer program after what is known about human intelligence, or is a strict “engineering” approach to the problem sufficient? Is it even pos- sible to achieve intelligence on a computer, or does an intelligent entity require the rich- ness of sensation and experience that might be found only in a biological existence?

These are unanswered questions, and all of them have helped to shape the problems and solution methodologies that constitute the core of modern AI. In fact, part of the appeal of artificial intelligence is that it offers a unique and powerful tool for exploring exactly these questions. AI offers a medium and a test-bed for theories of intelligence: such theories may be stated in the language of computer programs and consequently tested and verified through the execution of these programs on an actual computer.

For these reasons, our initial definition of artificial intelligence falls short of unambig- uously defining the field. If anything, it has only led to further questions and the paradoxi- cal notion of a field of study whose major goals include its own definition. But this difficulty in arriving at a precise definition of AI is entirely appropriate. Artificial intelli- gence is still a young discipline, and its structure, concerns, and methods are less clearly defined than those of a more mature science such as physics.

Artificial intelligence has always been more concerned with expanding the capabili- ties of computer science than with defining its limits. Keeping this exploration grounded in sound theoretical principles is one of the challenges facing AI researchers in general and this book in particular.

Because of its scope and ambition, artificial intelligence defies simple definition. For the time being, we will simply define it as *the collection of problems and methodologies studied by artificial intelligence researchers.* This definition may seem silly and meaning- less, but it makes an important point: artificial intelligence, like every science, is a human endeavor, and perhaps, is best understood in that context.

There are reasons that any science, AI included, concerns itself with a certain set of problems and develops a particular body of techniques for approaching these problems. In Chapter 1, a short history of artificial intelligence and the people and assumptions that have shaped it will explain why certain sets of questions have come to dominate the field and why the methods discussed in this book have been taken for their solution.

2 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

1 AI: EARLY HISTORY AND APPLICATIONS

*All men by nature desire to know...*

—ARISTOTLE*,* Opening sentence of the *Metaphysics*

*Hear the rest, and you will marvel even more at the crafts and resources I have contrived. Greatest was this: in the former times if a man fell sick he had no defense against the sickness, neither healing food nor drink, nor unguent; but through the lack of drugs men wasted away, until I showed them the blending of mild simples wherewith they drive out all manner of diseases. . . .*

*It was I who made visible to men’s eyes the flaming signs of the sky that were before dim. So much for these. Beneath the earth, man’s hidden blessing, copper, iron, silver, and gold—will anyone claim to have discovered these before I did? No one, I am very sure, who wants to speak truly and to the purpose. One brief word will tell the whole story: all arts that mortals have come from Prometheus.*

—AESCHYLUS, *Prometheus Bound*

**1.1** From Eden to ENIAC: Attitudes toward

Intelligence, Knowledge, and Human Artifice

Prometheus speaks of the fruits of his transgression against the gods of Olympus: his purpose was not merely to steal fire for the human race but also to enlighten humanity through the gift of intelligence or *nous*: the *rational mind*. This intelligence forms the foundation for all of human technology and ultimately all human civilization. The work of Aeschylus, the classical Greek dramatist, illustrates a deep and ancient awareness of the extraordinary power of knowledge. Artificial intelligence, in its very direct concern for Prometheus’s gift, has been applied to all the areas of his legacy—medicine, psychology, biology, astronomy, geology—and many areas of scientific endeavor that Aeschylus could not have imagined.

3

Though Prometheus’s action freed humanity from the sickness of ignorance, it also earned him the wrath of Zeus. Outraged over this theft of knowledge that previously belonged only to the gods of Olympus, Zeus commanded that Prometheus be chained to a barren rock to suffer the ravages of the elements for eternity. The notion that human efforts to gain knowledge constitute a transgression against the laws of God or nature is deeply ingrained in Western thought. It is the basis of the story of Eden and appears in the work of Dante and Milton. Both Shakespeare and the ancient Greek tragedians portrayed intellectual ambition as the cause of disaster. The belief that the desire for knowledge must ultimately lead to disaster has persisted throughout history, enduring the Renaissance, the Age of Enlightenment, and even the scientific and philosophical advances of the nine- teenth and twentieth centuries. Thus, we should not be surprised that artificial intelligence inspires so much controversy in both academic and popular circles.

Indeed, rather than dispelling this ancient fear of the consequences of intellectual ambition, modern technology has only made those consequences seem likely, even imminent. The legends of Prometheus, Eve, and Faustus have been retold in the language of technological society. In her introduction to *Frankenstein,* subtitled, interestingly enough, *The Modern Prometheus*, Mary Shelley writes:

Many and long were the conversations between Lord Byron and Shelley to which I was a devout and silent listener. During one of these, various philosophical doctrines were discussed, and among others the nature of the principle of life, and whether there was any probability of its ever being discovered and communicated. They talked of the experiments of Dr. Darwin (I speak not of what the doctor really did or said that he did, but, as more to my purpose, of what was then spoken of as having been done by him), who preserved a piece of vermicelli in a glass case till by some extraordinary means it began to move with a voluntary motion. Not thus, after all, would life be given. Perhaps a corpse would be reanimated; galvanism had given token of such things: perhaps the component parts of a creature might be manufactured, brought together, and endued with vital warmth (Butler 1998).

Mary Shelley shows us the extent to which scientific advances such as the work of Darwin and the discovery of electricity had convinced even nonscientists that the work- ings of nature were not divine secrets, but could be broken down and understood system- atically. Frankenstein’s monster is not the product of shamanistic incantations or unspeakable transactions with the underworld: it is assembled from separately “manufac- tured” components and infused with the vital force of electricity. Although nineteenth-cen- tury science was inadequate to realize the goal of understanding and creating a fully intelligent agent, it affirmed the notion that the mysteries of life and intellect might be brought into the light of scientific analysis.

**1.1.1** A Brief History of the Foundations for AI

By the time Mary Shelley finally and perhaps irrevocably joined modern science with the Promethean myth, the philosophical foundations of modern work in artificial intelligence had been developing for several thousand years. Although the moral and cultural issues raised by artificial intelligence are both interesting and important, our introduction is more

4 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

properly concerned with AI’s intellectual heritage. The logical starting point for such a history is the genius of Aristotle, or as Dante in the *Divine Comedy* refers to him, “the master of them that know”. Aristotle wove together the insights, wonders, and fears of the early Greek tradition with the careful analysis and disciplined thought that were to become the standard for more modern science.

For Aristotle, the most fascinating aspect of nature was change. In his *Physics*, he defined his “philosophy of nature” as the “study of things that change”. He distinguished between the *matter* and *form* of things: a sculpture is fashioned from the *material* bronze and has the *form* of a human. Change occurs when the bronze is molded to a new form. The matter/form distinction provides a philosophical basis for modern notions such as symbolic computing and data abstraction. In computing (even with numbers) we are manipulating patterns that are the forms of electromagnetic material, with the changes of form of this material representing aspects of the solution process. Abstracting the form from the medium of its representation not only allows these forms to be manipulated com- putationally but also provides the promise of a theory of data structures, the heart of mod- ern computer science. It also supports the creation of an “artificial” intelligence.

In his *Metaphysics*, beginning with the words “All men by nature desire to know”, Aristotle developed a science of things that never change, including his cosmology and theology. More relevant to artificial intelligence, however, was Aristotle’s epistemology or analysis of how humans “know” their world, discussed in his *Logic*. Aristotle referred to logic as the “instrument” (*organon*), because he felt that the study of thought itself was at the basis of all knowledge. In his *Logic*, he investigated whether certain propositions can be said to be “true” because they are related to other things that are known to be “true”. Thus if we know that “all men are mortal” and that “Socrates is a man”, then we can con- clude that “Socrates is mortal”. This argument is an example of what Aristotle referred to as a syllogism using the deductive form *modus ponens*. Although the formal axiomatiza- tion of reasoning needed another two thousand years for its full flowering in the works of Gottlob Frege, Bertrand Russell, Kurt Gödel, Alan Turing, Alfred Tarski, and others, its roots may be traced to Aristotle.

Renaissance thought, building on the Greek tradition, initiated the evolution of a dif- ferent and powerful way of thinking about humanity and its relation to the natural world. Science began to replace mysticism as a means of understanding nature. Clocks and, even- tually, factory schedules superseded the rhythms of nature for thousands of city dwellers. Most of the modern social and physical sciences found their origin in the notion that pro- cesses, whether natural or artificial, could be mathematically analyzed and understood. In particular, scientists and philosophers realized that thought itself, the way that knowledge was represented and manipulated in the human mind, was a difficult but essential subject for scientific study.

Perhaps the major event in the development of the modern world view was the Copernican revolution, the replacement of the ancient Earth-centered model of the universe with the idea that the Earth and other planets are actually in orbits around the sun. After centuries of an “obvious” order, in which the scientific explanation of the nature of the cosmos was consistent with the teachings of religion and common sense, a drastically different and not at all obvious model was proposed to explain the motions of heavenly bodies. For perhaps the first time, *our ideas about the world were seen as fundamentally*

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 5

*distinct from that world’s appearance*. This split between the human mind and its sur- rounding reality, between ideas about things and things themselves, is essential to the modern study of the mind and its organization. This breach was widened by the writings of Galileo, whose scientific observations further contradicted the “obvious” truths about the natural world and whose development of mathematics as a tool for describing that world emphasized the distinction between the world and our ideas about it. It is out of this breach that the modern notion of the mind evolved: introspection became a common motif in literature, philosophers began to study epistemology and mathematics, and the system- atic application of the scientific method rivaled the senses as tools for understanding the world.In 1620, Francis Bacon’s *Novum Organun* offered a set of search techniques for this emerging scientific methodology. Based on the Aristotelian and Platonic idea that the “form” of an entity was equivalent to the sum of its necessary and sufficient “features”, Bacon articulated an algorithm for determining the essence of an entity. First, he made an organized collection of all instances of the entity, enumerating the features of each in a table. Then he collected a similar list of negative instances of the entity, focusing espe- cially on near instances of the entity, that is, those that deviated from the “form” of the entity by single features. Then Bacon attempts - this step is not totally clear - to make a systematic list of all the features essential to the entity, that is, those that are common to all positive instances of the entity and missing from the negative instances.

It is interesting to see a form of Francis Bacon’s approach to concept learning reflected in modern AI algorithms for Version Space Search, Chapter 10.2. An extension of Bacon’s algorithms was also part of an AI program for discovery learning, suitably called *Bacon* (Langley et al. 1981). This program was able to induce many physical laws from collections of data related to the phenomena. It is also interesting to note that the question of whether a general purpose algorithm was possible for producing scientific proofs awaited the challenges of the early twentieth century mathematician Hilbert (his *Entscheidungsproblem*) and the response of the modern genius of Alan Turing (his *Turing Machine* and proofs of *computability* and the *halting problem*); see Davis et al. (1976).

Although the first calculating machine, the abacus, was created by the Chinese in the twenty-sixth century BC, further mechanization of algebraic processes awaited the skills of the seventeenth century Europeans. In 1614, the Scots mathematician, John Napier, cre- ated logarithms, the mathematical transformations that allowed multiplication and the use of exponents to be reduced to addition and multiplication. Napier also created his *bones* that were used to represent overflow values for arithmetic operations. These bones were later used by Wilhelm Schickard (1592-1635), a German mathematician and clergyman of Tübingen, who in 1623 invented a *Calculating Clock* for performing addition and subtrac- tion. This machine recorded the overflow from its calculations by the chiming of a clock.

Another famous calculating machine was the *Pascaline* that Blaise Pascal, the French philosopher and mathematician, created in 1642. Although the mechanisms of Schickard and Pascal were limited to addition and subtraction - including carries and borrows - they showed that processes that previously were thought to require human thought and skill could be fully automated. As Pascal later stated in his *Pensees* (1670), “The arithmetical machine produces effects which approach nearer to thought than all the actions of animals”.

6 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

Pascal’s successes with calculating machines inspired Gottfried Wilhelm von Leibniz in 1694 to complete a working machine that become known as the *Leibniz Wheel*. It inte- grated a moveable carriage and hand crank to drive wheels and cylinders that performed the more complex operations of multiplication and division. Leibniz was also fascinated by the possibility of a automated logic for proofs of propositions. Returning to Bacon’s entity specification algorithm, where concepts were characterized as the collection of their necessary and sufficient features, Liebniz conjectured a machine that could calculate with these features to produce logically correct conclusions. Liebniz (1887) also envisioned a machine, reflecting modern ideas of deductive inference and proof, by which the produc- tion of scientific knowledge could become automated, a calculus for reasoning.

The seventeenth and eighteenth centuries also saw a great deal of discussion of episte- mological issues; perhaps the most influential was the work of René Descartes, a central figure in the development of the modern concepts of thought and theories of mind. In his *Meditations*, Descartes (1680) attempted to find a basis for reality purely through intro- spection. Systematically rejecting the input of his senses as untrustworthy, Descartes was forced to doubt even the existence of the physical world and was left with only the reality of thought; even his own existence had to be justified in terms of thought: “Cogito ergo sum” (I think, therefore I am). After he established his own existence purely as a thinking entity, Descartes inferred the existence of God as an essential creator and ultimately reas- serted the reality of the physical universe as the necessary creation of a benign God.

We can make two observations here: first, the schism between the mind and the phys- ical world had become so complete that the process of thinking could be discussed in iso- lation from any specific sensory input or worldly subject matter; second, the connection between mind and the physical world was so tenuous that it required the intervention of a benign God to support reliable knowledge of the physical world! This view of the duality between the mind and the physical world underlies all of Descartes’s thought, including his development of analytic geometry. How else could he have unified such a seemingly worldly branch of mathematics as geometry with such an abstract mathematical frame- work as algebra?

Why have we included this mind/body discussion in a book on artificial intelligence? There are two consequences of this analysis essential to the AI enterprise:

1. By attempting to separate the mind from the physical world, Descartes and related thinkers established that the structure of ideas about the world was not necessar- ily the same as the structure of their subject matter. This underlies the methodol- ogy of AI, along with the fields of epistemology, psychology, much of higher mathematics, and most of modern literature: mental processes have an existence of their own, obey their own laws, and can be studied in and of themselves.

2. Once the mind and the body are separated, philosophers found it necessary to find a way to reconnect the two, because interaction between Descartes mental, *res cogitans*, and physical, *res extensa*, is essential for human existence.

Although millions of words have been written on this *mind–body problem*, and numerous solutions proposed, no one has successfully explained the obvious interactions between mental states and physical actions while affirming a fundamental difference

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 7

between them. The most widely accepted response to this problem, and the one that provides an essential foundation for the study of AI, holds that the mind and the body are not fundamentally different entities at all. On this view, mental processes are indeed achieved by physical systems such as brains (or computers). Mental processes, like physi- cal processes, can ultimately be characterized through formal mathematics. Or, as acknowledged in his *Leviathan* by the 17th century English philosopher Thomas Hobbes (1651), “By ratiocination, I mean computation”.

**1.1.2** AI and the Rationalist and Empiricist Traditions

Modern research issues in artificial intelligence, as in other scientific disciplines, are formed and evolve through a combination of historical, social, and cultural pressures. Two of the most prominent pressures for the evolution of AI are the empiricist and rationalist traditions in philosophy.

The rationalist tradition, as seen in the previous section, had an early proponent in Plato, and was continued on through the writings of Pascal, Descartes, and Liebniz. For the rationalist, the external world is reconstructed through the clear and distinct ideas of a mathematics. A criticism of this dualistic approach is the forced disengagement of repre- sentational systems from their field of reference. The issue is whether the meaning attrib- uted to a representation can be defined independent of its application conditions. If the world is different from our beliefs about the world, can our created concepts and symbols still have meaning?

Many AI programs have very much of this rationalist flavor. Early robot planners, for example, would describe their application domain or “world” as sets of predicate calculus statements and then a “plan” for action would be created through proving theorems about this “world” (Fikes et al. 1972, see also Section 8.4). Newell and Simon’s *Physical Symbol System Hypothesis* (Introduction to Part II and Chapter 16) is seen by many as the arche- type of this approach in modern AI. Several critics have commented on this rationalist bias as part of the failure of AI at solving complex tasks such as understanding human lan- guages (Searle 1980, Winograd and Flores 1986, Brooks 1991a).

Rather than affirming as “real” the world of clear and distinct ideas, empiricists con- tinue to remind us that “nothing enters the mind except through the senses”. This con- straint leads to further questions of how the human can possibly perceive general concepts or the pure forms of Plato’s cave (Plato 1961). Aristotle was an early empiricist, emphasiz- ing in his *De Anima*, the limitations of the human perceptual system. More modern empir- icists, especially Hobbes, Locke, and Hume, emphasize that knowledge must be explained through an introspective but empirical psychology. They distinguish two types of mental phenomena perceptions on one hand and thought, memory, and imagination on the other. The Scots philosopher, David Hume, for example, distinguishes between *impressions* and *ideas*. Impressions are lively and vivid, reflecting the presence and existence of an exter- nal object and not subject to voluntary control, the *qualia* of Dennett (2005). Ideas on the other hand, are less vivid and detailed and more subject to the subject’s voluntary control.

Given this distinction between impressions and ideas, how can knowledge arise? For Hobbes, Locke, and Hume the fundamental explanatory mechanism is *association*.

8 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

Particular perceptual properties are associated through repeated experience. This repeated association creates a disposition in the mind to associate the corresponding ideas, a pre- curser of the behaviorist approach of the twentieth century. A fundamental property of this account is presented with Hume’s skepticism. Hume’s purely descriptive account of the origins of ideas cannot, he claims, support belief in causality. Even the use of logic and induction cannot be rationally supported in this radical empiricist epistemology.

In *An Inquiry Concerning Human Understanding* (1748), Hume’s skepticism extended to the analysis of miracles. Although Hume didn’t address the nature of miracles directly, he did question the testimony-based belief in the miraculous. This skepticism, of course, was seen as a direct threat by believers in the bible as well as many other purvey- ors of religious traditions. The Reverend Thomas Bayes was both a mathematician and a minister. One of his papers, called *Essay towards Solving a Problem in the Doctrine of Chances* (1763) addressed Hume’s questions mathematically. Bayes’ theorem demon- strates formally how, through learning the correlations of the effects of actions, we can determine the probability of their causes.

The associational account of knowledge plays a significant role in the development of AI representational structures and programs, for example, in memory organization with *semantic networks* and *MOPS* and work in natural language understanding (see Sections 7.0, 7.1, and Chapter 15). Associational accounts have important influences of machine learning, especially with connectionist networks (see Section 10.6, 10.7, and Chapter 11). Associationism also plays an important role in cognitive psychology including the *sche- mas* of Bartlett and Piaget as well as the entire thrust of the behaviorist tradition (Luger 1994). Finally, with AI tools for stochastic analysis, including the *Bayesian belief network* (BBN) and its current extensions to first-order Turing-complete systems for stochastic modeling, associational theories have found a sound mathematical basis and mature expressive power. Bayesian tools are important for research including diagnostics, machine learning, and natural language understanding (see Chapters 5 and 13).

Immanuel Kant, a German philosopher trained in the rationalist tradition, was strongly influenced by the writing of Hume. As a result, he began the modern synthesis of these two traditions. Knowledge for Kant contains two collaborating energies, an a priori component coming from the subject’s reason along with an a posteriori component com- ing from active experience. Experience is meaningful only through the contribution of the subject. Without an active organizing form proposed by the subject, the world would be nothing more than passing transitory sensations. Finally, at the level of judgement, Kant claims, passing images or representations are bound together by the active subject and taken as the diverse appearances of an identity, of an “object”. Kant’s realism began the modern enterprise of psychologists such as Bartlett, Brunner, and Piaget. Kant’s work influences the modern AI enterprise of machine learning (Section IV) as well as the con- tinuing development of a constructivist epistemology (see Chapter 16).

**1.1.3** The Development of Formal Logic

Once thinking had come to be regarded as a form of computation, its formalization and eventual mechanization were obvious next steps. As noted in Section 1.1.1,

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 9

Gottfried Wilhelm von Leibniz, with his *Calculus Philosophicus*, introduced the first sys- tem of formal logic as well as proposed a machine for automating its tasks (Leibniz 1887). Furthermore, the steps and stages of this mechanical solution can be represented as move- ment through the states of a tree or graph. Leonhard Euler, in the eighteenth century, with his analysis of the “connectedness” of the bridges joining the riverbanks and islands of the city of Königsberg (see the introduction to Chapter 3), introduced the study of representa- tions that can abstractly capture the structure of relationships in the world as well as the discrete steps within a computation about these relationships (Euler 1735).

The formalization of graph theory also afforded the possibility of *state space search*, a major conceptual tool of artificial intelligence. We can use graphs to model the deeper structure of a problem. The nodes of a *state space graph* represent possible stages of a problem solution; the arcs of the graph represent inferences, moves in a game, or other steps in a problem solution. Solving the problem is a process of searching the state space graph for a path to a solution (Introduction to II and Chapter 3). By describing the entire space of problem solutions, state space graphs provide a powerful tool for measuring the structure and complexity of problems and analyzing the efficiency, correctness, and gener- ality of solution strategies.

As one of the originators of the science of operations research, as well as the designer of the first programmable mechanical computing machines, Charles Babbage, a nine- teenth century mathematician, may also be considered an early practitioner of artificial intelligence (Morrison and Morrison 1961). Babbage’s *difference engine* was a special- purpose machine for computing the values of certain polynomial functions and was the forerunner of his *analytical engine*. The analytical engine, designed but not successfully constructed during his lifetime, was a general-purpose programmable computing machine that presaged many of the architectural assumptions underlying the modern computer.

In describing the analytical engine, Ada Lovelace (1961), Babbage’s friend, sup- porter, and collaborator, said:

We may say most aptly that the Analytical Engine weaves algebraical patterns just as the Jac- quard loom weaves flowers and leaves. Here, it seems to us, resides much more of originality than the difference engine can be fairly entitled to claim.

Babbage’s inspiration was his desire to apply the technology of his day to liberate humans from the drudgery of making arithmetic calculations. In this sentiment, as well as with his conception of computers as mechanical devices, Babbage was thinking in purely nineteenth century terms. His analytical engine, however, also included many modern notions, such as the separation of memory and processor, the *store* and the *mill* in Bab- bage’s terms, the concept of a digital rather than analog machine, and programmability based on the execution of a series of operations encoded on punched pasteboard cards. The most striking feature of Ada Lovelace’s description, and of Babbage’s work in gen- eral, is its treatment of the “patterns” of algebraic relationships as entities that may be studied, characterized, and finally implemented and manipulated mechanically without concern for the particular values that are finally passed through the mill of the calculating machine. This is an example implementation of the “abstraction and manipulation of form” first described by Aristotle and Liebniz.

10 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

The goal of creating a formal language for thought also appears in the work of George Boole, another nineteenth-century mathematician whose work must be included in any discussion of the roots of artificial intelligence (Boole 1847, 1854). Although he made contributions to a number of areas of mathematics, his best known work was in the mathematical formalization of the laws of logic, an accomplishment that forms the very heart of modern computer science. Though the role of Boolean algebra in the design of logic circuitry is well known, Boole’s own goals in developing his system seem closer to those of contemporary AI researchers. In the first chapter of *An Investigation of the Laws of Thought, on which are founded the Mathematical Theories of Logic and Probabilities*, Boole (1854) described his goals as

to investigate the fundamental laws of those operations of the mind by which reasoning is performed: to give expression to them in the symbolical language of a Calculus, and upon this foundation to establish the science of logic and instruct its method; ...and finally to collect from the various elements of truth brought to view in the course of these inquiries some proba- ble intimations concerning the nature and constitution of the human mind.

The importance of Boole’s accomplishment is in the extraordinary power and sim- plicity of the system he devised: three operations, “AND” (denoted by ∗ or ∧), “OR” (denoted by + or ∨), and “NOT” (denoted by ¬), formed the heart of his logical calculus. These operations have remained the basis for all subsequent developments in formal logic, including the design of modern computers. While keeping the meaning of these symbols nearly identical to the corresponding algebraic operations, Boole noted that “the Symbols of logic are further subject to a special law, to which the symbols of quantity, as such, are not subject”. This law states that for any X, an element in the algebra, X ∗X =X (or that once something is known to be true, repetition cannot augment that knowledge). This led to the characteristic restriction of Boolean values to the only two numbers that may satisfy this equation: 1 and 0. The standard definitions of Boolean multiplication (AND) and addition (OR) follow from this insight.

Boole’s system not only provided the basis of binary arithmetic but also demonstrated that an extremely simple formal system was adequate to capture the full power of logic. This assumption and the system Boole developed to demonstrate it form the basis of all modern efforts to formalize logic, from Russell and Whitehead’s *Principia Mathematica* (Whitehead and Russell 1950), through the work of Turing and Gödel, up to modern auto- mated reasoning systems.

Gottlob Frege, in his *Foundations of Arithmetic* (Frege 1879, 1884), created a mathematical specification language for describing the basis of arithmetic in a clear and precise fashion. With this language Frege formalized many of the issues first addressed by Aristotle’s *Logic*. Frege’s language, now called the *first-order predicate calculus*, offers a tool for describing the propositions and truth value assignments that make up the elements of mathematical reasoning and describes the axiomatic basis of “meaning” for these expressions. The formal system of the predicate calculus, which includes predicate sym- bols, a theory of functions, and quantified variables, was intended to be a language for describing mathematics and its philosophical foundations. It also plays a fundamental role in creating a theory of representation for artificial intelligence (Chapter 2). The first-order

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 11

predicate calculus offers the tools necessary for automating reasoning: a language for expressions, a theory for assumptions related to the meaning of expressions, and a logi- cally sound calculus for inferring new true expressions.

Whitehead and Russell’s (1950) work is particularly important to the foundations of AI, in that their stated goal was to derive the whole of mathematics through formal opera- tions on a collection of axioms. Although many mathematical systems have been con- structed from basic axioms, what is interesting is Russell and Whitehead’s commitment to mathematics as a purely formal system. This meant that axioms and theorems would be treated solely as strings of characters: proofs would proceed solely through the application of well-defined rules for manipulating these strings. There would be no reliance on intu- ition or the meaning of theorems as a basis for proofs. Every step of a proof followed from the strict application of formal (syntactic) rules to either axioms or previously proven the- orems, even where traditional proofs might regard such a step as “obvious”. What “mean- ing” the theorems and axioms of the system might have in relation to the world would be independent of their logical derivations. This treatment of mathematical reasoning in purely formal (and hence mechanical) terms provided an essential basis for its automation on physical computers. The logical syntax and formal rules of inference developed by Russell and Whitehead are still a basis for automatic theorem-proving systems, presented in Chapter 14, as well as for the theoretical foundations of artificial intelligence.

Alfred Tarski is another mathematician whose work is essential to the foundations of AI. Tarski created a *theory of reference* wherein the *well-formed formulae* of Frege or Russell and Whitehead can be said to refer, in a precise fashion, to the physical world (Tarski 1944, 1956; see Chapter 2). This insight underlies most theories of formal seman- tics. In his paper *The Semantic Conception of Truth and the Foundation of Semantics*, Tar- ski describes his theory of reference and truth value relationships. Modern computer scientists, especially Scott, Strachey, Burstall (Burstall and Darlington 1977), and Plotkin have related this theory to programming languages and other specifications for computing. Although in the eighteenth, nineteenth, and early twentieth centuries the formaliza- tion of science and mathematics created the intellectual prerequisite for the study of artifi- cial intelligence, it was not until the twentieth century and the introduction of the digital computer that AI became a viable scientific discipline. By the end of the 1940s electronic digital computers had demonstrated their potential to provide the memory and processing power required by intelligent programs. It was now possible to implement formal reason- ing systems on a computer and empirically test their sufficiency for exhibiting intelli- gence. An essential component of the science of artificial intelligence is this commitment to digital computers as the vehicle of choice for creating and testing theories of intelligence.

Digital computers are not merely a vehicle for testing theories of intelligence. Their architecture also suggests a specific paradigm for such theories: intelligence is a form of information processing. The notion of search as a problem-solving methodology, for example, owes more to the sequential nature of computer operation than it does to any biological model of intelligence. Most AI programs represent knowledge in some formal language that is then manipulated by algorithms, honoring the separation of data and program fundamental to the von Neumann style of computing. Formal logic has emerged as an important representational tool for AI research, just as graph theory plays an indis-

12 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

pensable role in the analysis of problem spaces as well as providing a basis for semantic networks and similar models of semantic meaning. These techniques and formalisms are discussed in detail throughout the body of this text; we mention them here to emphasize the symbiotic relationship between the digital computer and the theoretical underpinnings of artificial intelligence.

We often forget that the tools we create for our own purposes tend to shape our conception of the world through their structure and limitations. Although seemingly restrictive, this interaction is an essential aspect of the evolution of human knowledge: a tool (and scientific theories are ultimately only tools) is developed to solve a particular problem. As it is used and refined, the tool itself seems to suggest other applications, leading to new questions and, ultimately, the development of new tools.

**1.1.4** The Turing Test

One of the earliest papers to address the question of machine intelligence specifically in relation to the modern digital computer was written in 1950 by the British mathematician Alan Turing. *Computing Machinery and Intelligence* (Turing 1950) remains timely in both its assessment of the arguments against the possibility of creating an intelligent computing machine and its answers to those arguments. Turing, known mainly for his contributions to the theory of computability, considered the question of whether or not a machine could actually be made to think. Noting that the fundamental ambiguities in the question itself (what is thinking? what is a machine?) precluded any rational answer, he proposed that the question of intelligence be replaced by a more clearly defined empirical test.

The *Turing test* measures the performance of an allegedly intelligent machine against that of a human being, arguably the best and only standard for intelligent behavior. The test, which Turing called the *imitation game*, places the machine and a human counterpart in rooms apart from a second human being, referred to as the *interrogator* (Figure 1.1). The interrogator is not able to see or speak directly to either of them, does not know which entity is actually the machine, and may communicate with them solely by use of a textual device such as a terminal. The interrogator is asked to distinguish the computer from the human being solely on the basis of their answers to questions asked over this device. If the interrogator cannot distinguish the machine from the human, then, Turing argues, the machine may be assumed to be intelligent.

By isolating the interrogator from both the machine and the other human participant, the test ensures that the interrogator will not be biased by the appearance of the machine or any mechanical property of its voice. The interrogator is free, however, to ask any questions, no matter how devious or indirect, in an effort to uncover the computer’s identity. For example, the interrogator may ask both subjects to perform a rather involved arithmetic calculation, assuming that the computer will be more likely to get it correct than the human; to counter this strategy, the computer will need to know when it should fail to get a correct answer to such problems in order to seem like a human. To discover the human’s identity on the basis of emotional nature, the interrogator may ask both subjects to respond to a poem or work of art; this strategy will require that the computer have knowledge concerning the emotional makeup of human beings.

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 13

THE INTERROGATOR

Figure 1.1 The Turing test.

The important features of Turing’s test are:

1. It attempts to give an objective notion of intelligence, i.e., the behavior of a known intelligent being in response to a particular set of questions. This provides a standard for determining intelligence that avoids the inevitable debates over its “true” nature.

2. It prevents us from being sidetracked by such confusing and currently unanswerable questions as whether or not the computer uses the appropriate internal processes or whether or not the machine is actually conscious of its actions.

3. It eliminates any bias in favor of living organisms by forcing the interrogator to

focus solely on the content of the answers to questions.

Because of these advantages, the Turing test provides a basis for many of the schemes actually used to evaluate modern AI programs. A program that has potentially achieved intelligence in some area of expertise may be evaluated by comparing its performance on a given set of problems to that of a human expert. This evaluation technique is just a variation of the Turing test: a group of humans are asked to blindly compare the performance of a computer and a human being on a particular set of problems. As we will see, this methodology has become an essential tool in both the development and verification of modern expert systems.

The Turing test, in spite of its intuitive appeal, is vulnerable to a number of justifiable criticisms. One of the most important of these is aimed at its bias toward purely symbolic problem-solving tasks. It does not test abilities requiring perceptual skill or manual dexterity, even though these are important components of human intelligence. Conversely, it is sometimes suggested that the Turing test needlessly constrains machine intelligence to fit a human mold. Perhaps machine intelligence is simply different from human intelli- gence and trying to evaluate it in human terms is a fundamental mistake. Do we really wish a machine would do mathematics as slowly and inaccurately as a human? Shouldn’t an intelligent machine capitalize on its own assets, such as a large, fast, reliable memory,

14 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

rather than trying to emulate human cognition? In fact, a number of modern AI practitio- ners (e.g., Ford and Hayes 1995) see responding to the full challenge of Turing’s test as a mistake and a major distraction to the more important work at hand: developing general theories to explain the mechanisms of intelligence in humans and machines and applying those theories to the development of tools to solve specific, practical problems. Although we agree with the Ford and Hayes concerns in the large, we still see Turing’s test as an important component in the verification and validation of modern AI software.

Turing also addressed the very feasibility of constructing an intelligent program on a digital computer. By thinking in terms of a specific model of computation (an electronic discrete state computing machine), he made some well-founded conjectures concerning the storage capacity, program complexity, and basic design philosophy required for such a system. Finally, he addressed a number of moral, philosophical, and scientific objections to the possibility of constructing such a program in terms of an actual technology. The reader is referred to Turing’s article for a perceptive and still relevant summary of the debate over the possibility of intelligent machines.

Two of the objections cited by Turing are worth considering further. *Lady Lovelace’s Objection*, first stated by Ada Lovelace, argues that computers can only do as they are told and consequently cannot perform original (hence, intelligent) actions. This objection has become a reassuring if somewhat dubious part of contemporary technologi- cal folklore. Expert systems (Section 1.2.3 and Chapter 8), especially in the area of diag- nostic reasoning, have reached conclusions unanticipated by their designers. Indeed, a number of researchers feel that human creativity can be expressed in a computer program.

The other related objection, the *Argument from Informality of Behavior*, asserts the impossibility of creating a set of rules that will tell an individual exactly what to do under every possible set of circumstances. Certainly, the flexibility that enables a biological intelligence to respond to an almost infinite range of situations in a reasonable if not nec- essarily optimal fashion is a hallmark of intelligent behavior. While it is true that the con- trol structure used in most traditional computer programs does not demonstrate great flexibility or originality, it is not true that all programs must be written in this fashion. Indeed, much of the work in AI over the past 25 years has been to develop programming languages and models such as production systems, object-based systems, neural network representations, and others discussed in this book that attempt to overcome this deficiency.

Many modern AI programs consist of a collection of modular components, or rules of behavior, that do not execute in a rigid order but rather are invoked as needed in response to the structure of a particular problem instance. Pattern matchers allow general rules to apply over a range of instances. These systems have an extreme flexibility that enables rel- atively small programs to exhibit a vast range of possible behaviors in response to differ- ing problems and situations.

Whether these systems can ultimately be made to exhibit the flexibility shown by a living organism is still the subject of much debate. Nobel laureate Herbert Simon has argued that much of the originality and variability of behavior shown by living creatures is due to the richness of their environment rather than the complexity of their own internal programs. In *The Sciences of the Artificial*, Simon (1981) describes an ant progressing circuitously along an uneven and cluttered stretch of ground. Although the ant’s path seems quite complex, Simon argues that the ant’s goal is very simple: to return to its

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 15

colony as quickly as possible. The twists and turns in its path are caused by the obstacles it encounters on its way. Simon concludes that

An ant, viewed as a behaving system, is quite simple. The apparent complexity of its behavior over time is largely a reflection of the complexity of the environment in which it finds itself.

This idea, if ultimately proved to apply to organisms of higher intelligence as well as to such simple creatures as insects, constitutes a powerful argument that such systems are relatively simple and, consequently, comprehensible. It is interesting to note that if one applies this idea to humans, it becomes a strong argument for the importance of culture in the forming of intelligence. Rather than growing in the dark like mushrooms, intelligence seems to depend on an interaction with a suitably rich environment. Culture is just as important in creating humans as human beings are in creating culture. Rather than deni- grating our intellects, this idea emphasizes the miraculous richness and coherence of the cultures that have formed out of the lives of separate human beings. In fact, the idea that intelligence emerges from the interactions of individual elements of a society is one of the insights supporting the approach to AI technology presented in the next section.

**1.1.5** Biological and Social Models of Intelligence: Agents Theories

So far, we have approached the problem of building intelligent machines from the view- point of mathematics, with the implicit belief of logical reasoning as paradigmatic of intel- ligence itself, as well as with a commitment to “objective” foundations for logical reasoning. This way of looking at knowledge, language, and thought reflects the rational- ist tradition of western philosophy, as it evolved through Plato, Galileo, Descartes, Leib- niz, and many of the other philosophers discussed earlier in this chapter. It also reflects the underlying assumptions of the Turing test, particularly its emphasis on symbolic reasoning as a test of intelligence, and the belief that a straightforward comparison with human behavior was adequate to confirming machine intelligence.

The reliance on logic as a way of representing knowledge and on logical inference as the primary mechanism for intelligent reasoning are so dominant in Western philosophy that their “truth” often seems obvious and unassailable. It is no surprise, then, that approaches based on these assumptions have dominated the science of artificial intelligence from its inception almost through to the present day.

The latter half of the twentieth century has, however, seen numerous challenges to rationalist philosophy. Various forms of philosophical relativism question the objective basis of language, science, society, and thought itself. Ludwig Wittgenstein's later philosophy (Wittgenstein 1953), has forced us to reconsider the basis on meaning in both natural and formal languages. The work of Godel (Nagel and Newman 1958) and Turing has cast doubt on the very foundations of mathematics itself. Post-modern thought has changed our understanding of meaning and value in the arts and society. Artificial intelli- gence has not been immune to these criticisms; indeed, the difficulties that AI has encoun- tered in achieving its goals are often taken as evidence of the failure of the rationalist viewpoint (Winograd and Flores 1986, Lakoff and Johnson 1999, Dennett 2005).

16 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

Two philosophical traditions, that of Wittgenstein (1953) as well as that of Husserl (1970, 1972) and Heidegger (1962) are central to this reappraisal of the Western philo- sophical tradition. In his later work, Wittgenstein questioned many of the assumptions of the rationalist tradition, including the foundations of language, science, and knowledge. Human language was a major focus of Wittgenstein’s analysis: he challenged the notion that language derived its meaning from any sort of objective foundation.

For Wittgenstein, as well as the speech act theory developed by Austin (1962) and his followers (Grice 1975, Searle 1969), the meaning of any utterance depends on its being situated in a human, cultural context. Our understanding of the meaning of the word “chair”, for example, is dependent on having a physical body that conforms to a sitting posture and the cultural conventions for using chairs. When, for example, is a large, flat rock a chair? Why is it odd to refer to the throne of England as a chair? What is the difference between a human being's understanding of a chair and that of a dog or cat, inca- pable of sitting in the human sense? Based on his attacks on the foundations of meaning, Wittgenstein argued that we should view the use of language in terms of choices made and actions taken in a shifting cultural context. Wittgenstein even extended his criticisms to science and mathematics, arguing that they are just as much social constructs as is language use.

Husserl (1970, 1972), the father of phenomenology, was committed to abstractions as rooted in the concrete *Lebenswelt* or *life-world*: a rationalist model was very much sec- ondary to the concrete world that supported it. For Husserl, as well as for his student Heidegger (1962), and their proponent Merleau-Ponty (1962), intelligence was not know- ing what was true, but rather knowing how to cope in a world that was constantly chang- ing and evolving. Gadamer (1976) also contributed to this tradition. For the existentialist/ phenomenologist, intelligence is seen as survival in the world, rather than as a set of logi- cal propositions about the world (combined with some inferencing scheme).

Many authors, for example Dreyfus and Dreyfus (1985) and Winograd and Flores (1986), have drawn on Wittgenstein’s and the Husserl/Heidegger work in their criticisms of AI. Although many AI practitioners continue developing the rational/logical agenda, also known as *GOFAI*, or *Good Old Fashioned AI*, a growing number of researchers in the field have incorporated these criticisms into new and exciting models of intelligence. In keeping with Wittgenstein’s emphasis on the anthropological and cultural roots of knowledge, they have turned to social, sometimes referred to as *agent-based* or *situated*, models of intelligent behavior for their inspiration.

As an example of an alternative to a logic-based approach, research in connectionist learning (Section 1.2.9 and Chapter 11) de-emphasizes logic and the functioning of the rational mind in an effort to achieve intelligence by modeling the architecture of the physical brain. Neural models of intelligence emphasize the brain’s ability to adapt to the world in which it is situated by modifying the relationships between individual neurons. Rather than representing knowledge in explicit logical sentences, they capture it implicitly, as a property of patterns of relationships.

Another biologically based model of intelligence takes its inspiration from the processes by which entire species adapt to their surroundings. Work in artificial life and genetic algorithms (Chapter 12) applies the principles of biological evolution to the prob- lems of finding solutions to difficult problems. These programs do not solve problems by

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 17

reasoning logically about them; rather, they spawn populations of competing candidate solutions and drive them to evolve ever better solutions through a process patterned after biological evolution: poor candidate solutions tend to die out, while those that show the promise for solving a problem survive and reproduce by constructing new solutions out of components of their successful parents.

Social systems provide another metaphor for intelligence in that they exhibit global behaviors that enable them to solve problems that would confound any of their individual members. For example, although no individual could accurately predict the number of loaves of bread to be consumed in New York City on a given day, the entire system of New York bakeries does an excellent job of keeping the city stocked with bread, and doing so with minimal waste. The stock market does an excellent job of setting the relative val- ues of hundreds of companies, even though each individual investor has only limited knowledge of a few companies. A final example comes from modern science. Individuals located in universities, industry, or government environments focus on common problems. With conferences and journals as the main communication media, problems important to society at large are attacked and solved by individual agents working semi-independently, although progress in many instances is also driven by funding agencies.

These examples share two themes: first, the view of intelligence as rooted in culture and society and, as a consequence, emergent. The second theme is that intelligence is reflected by the collective behaviors of large numbers of very simple interacting, semi- autonomous individuals, or *agents*. Whether these agents are neural cells, individual mem- bers of a species, or a single person in a society, their interactions produce intelligence.

What are the main themes supporting an agent-oriented and emergent view of intelligence? They include:

1. Agents are autonomous or semi-autonomous. That is, each agent has certain responsibilities in problem solving with little or no knowledge of either what other agents do or how they do it. Each agent does its own independent piece of the problem solving and either produces a result itself (does something) or reports results back to others in the community (communicating agent).

2. Agents are “situated.” Each agent is sensitive to its own surrounding environ- ment and (usually) has no knowledge of the full domain of all agents. Thus, an agent's knowledge is limited to the tasks to hand: “the-file-I’m-processing” or “the-wall-next-to-me” with no knowledge of the total range of files or physical constraints in the problem solving task.

3. Agents are interactional. That is, they form a collection of individuals that cooperate on a particular task. In this sense they may be seen as a “society” and, as with human society, knowledge, skills, and responsibilities, even when seen as collective, are distributed across the population of individuals.

4. The society of agents is structured. In most views of agent-oriented problem solving, each individual, although having its own unique environment and skill set, will coordinate with other agents in the overall problem solving. Thus, a final solution will not only be seen as collective, but also as cooperative.

18 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

5. Finally, the phenomenon of intelligence in this environment is “emergent.” Although individual agents are seen as possessing sets of skills and responsibili- ties, the overall cooperative result can be viewed as greater than the sum of its individual contributors. Intelligence is seen as a phenomenon resident in and emerging from a society and not just a property of an individual agent.

Based on these observations, we define an agent as an element of a society that can perceive (often limited) aspects of its environment and affect that environment either directly or through cooperation with other agents. Most intelligent solutions require a vari- ety of agents. These include rote agents, that simply capture and communicate pieces of information, coordination agents that can support the interactions between other agents, search agents that can examine multiple pieces of information and return some chosen bit of it, learning agents that can examine collections of information and form concepts or generalizations, and decision agents that can both dispatch tasks and come to conclusions in the light of limited information and processing. Going back to an older definition of intelligence, agents can be seen as the mechanisms supporting decision making in the con- text of limited processing resources.

The main requisites for designing and building such a society are:

1. structures for the representation of information,

2. strategies for the search through alternative solutions, and

3. the creation of architectures that can support the interaction of agents.

The remaining chapters of our book, especially Section 7.4, include prescriptions for the construction of support tools for this society of agents, as well as many examples of agent- based problem solving.

Our preliminary discussion of the possibility of a theory of automated intelligence is in no way intended to overstate the progress made to date or minimize the work that lies ahead. As we emphasize throughout this book, it is important to be aware of our limita- tions and to be honest about our successes. For example, there have been only limited results with programs that in any interesting sense can be said to “learn”. Our accomplish- ments in modeling the semantic complexities of a natural language such as English have also been very modest. Even fundamental issues such as organizing knowledge or fully managing the complexity and correctness of very large computer programs (such as large knowledge bases) require considerable further research. Knowledge-based systems, though they have achieved marketable engineering successes, still have many limitations in the quality and generality of their reasoning. These include their inability to perform *commonsense reasoning* or to exhibit knowledge of rudimentary physical reality, such as how things change over time.

But we must maintain a reasonable perspective. It is easy to overlook the accomplishments of artificial intelligence when honestly facing the work that remains. In the next section, we establish this perspective through an overview of several important areas of artificial intelligence research and development.

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 19

**1.2** Overview of AI Application Areas

*The Analytical Engine has no pretensions whatever to originate anything. It can do whatever we know how to order it to perform.*

—ADA BYRON, *Countess of Lovelace*

*I’m sorry Dave; I can’t let you do that.*

—HAL 9000 in *2001: A Space Odyssey* by Arthur C. Clarke

We now return to our goal of defining artificial intelligence through an examination of the ambitions and accomplishments of workers in the field. The two most fundamental con- cerns of AI researchers are *knowledge representation* and *search*. The first of these addresses the problem of capturing in a language, i.e., one suitable for computer manipu- lation, the full range of knowledge required for intelligent behavior. Chapter 2 introduces predicate calculus as a language for describing the properties and relationships among objects in problem domains that require qualitative reasoning rather than arithmetic calcu- lations for their solutions. Later, Section III discusses the tools that artificial intelligence has developed for representing the ambiguities and complexities of areas such as com- monsense reasoning and natural language understanding.

Search is a problem-solving technique that systematically explores a space of *prob- lem states*, i.e., successive and alternative stages in the problem-solving process. Exam- ples of problem states might include the different board configurations in a game or intermediate steps in a reasoning process. This space of alternative solutions is then searched to find an answer. Newell and Simon (1976) have argued that this is the essential basis of human problem solving. Indeed, when a chess player examines the effects of dif- ferent moves or a doctor considers a number of alternative diagnoses, they are searching among alternatives. The implications of this model and techniques for its implementation are discussed in Chapters 3, 4, 6, and 16. The auxiliary material for this book (see Preface) offers Lisp, Prolog, and Java implementations of these algorithms.

Like most sciences, AI is decomposed into a number of subdisciplines that, while sharing an essential approach to problem solving, have concerned themselves with different applications. In this section we outline several of these major application areas and their contributions to artificial intelligence as a whole.

**1.2.1** Game Playing

Much of the early research in state space search was done using common board games such as checkers, chess, and the 15-puzzle. In addition to their inherent intellectual appeal, board games have certain properties that made them ideal subjects for research. Most games are played using a well-defined set of rules: this makes it easy to generate the search space and frees the researcher from many of the ambiguities and complexities inherent in less structured problems. The board configurations used in playing games are

20 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

easily represented on a computer, requiring none of the complex formalisms needed to capture the semantic subtleties of more complex problem domains. As games are easily played, testing a game-playing program presents no financial or ethical burden. State space search, the paradigm underlying most game-playing research, is presented in Chapters 3 and 4.

Games can generate extremely large search spaces. These are large and complex enough to require powerful techniques for determining what alternatives to explore in the problem space. These techniques are called *heuristics* and constitute a major area of AI research. A heuristic is a useful but potentially fallible problem-solving strategy, such as checking to make sure that an unresponsive appliance is plugged in before assuming that it is broken or to castle in order to try and protect your king from capture in a chess game. Much of what we commonly call intelligence seems to reside in the heuristics used by humans to solve problems.

Because most of us have some experience with these simple games, it is possible to devise and test the effectiveness of our own heuristics. We do not need to find and consult an expert in some esoteric problem area such as medicine or mathematics (chess is an obvious exception to this rule). For these reasons, games provide a rich domain for the study of heuristic search. Chapter 4 introduces heuristics using these simple games. Game-playing programs, in spite of their simplicity, offer their own challenges, including an opponent whose moves may not be deterministically anticipated, Chapters 5 and 9, and the need to consider psychological as well as tactical factors in game strategy.

Recent successes in computer-based game playing include world championships in backgammon and chess. It is also interesting to note that in 2007 the full state space for the game of checkers was mapped out, allowing it to be from the first move, deterministic!

**1.2.2** Automated Reasoning and Theorem Proving

We could argue that automatic theorem proving is the oldest branch of artificial intelligence, tracing its roots back through Newell and Simon’s Logic Theorist (Newell and Simon 1963*a*) and General Problem Solver (Newell and Simon 1963*b*), through Rus- sell and Whitehead’s efforts to treat all of mathematics as the purely formal derivation of theorems from basic axioms, to its origins in the writings of Babbage and Leibniz. In any case, it has certainly been one of the most fruitful branches of the field. Theorem-proving research was responsible for much of the early work in formalizing search algorithms and developing formal representation languages such as the predicate calculus (Chapter 2) and the logic programming language Prolog.

Most of the appeal of automated theorem proving lies in the rigor and generality of logic. Because it is a formal system, logic lends itself to automation. A wide variety of problems can be attacked by representing the problem description and relevant background information as logical axioms and treating problem instances as theorems to be proved. This insight is the basis of work in automatic theorem proving and mathematical reasoning systems (Chapter 14).

Unfortunately, early efforts at writing theorem provers failed to develop a system that could consistently solve complicated problems. This was due to the ability of any

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 21

reasonably complex logical system to generate an infinite number of provable theorems: without powerful techniques (heuristics) to guide their search, automated theorem provers proved large numbers of irrelevant theorems before stumbling onto the correct one. In response to this inefficiency, many argue that purely formal, syntactic methods of guiding search are inherently incapable of handling such a huge space and that the only alternative is to rely on the informal, *ad hoc* strategies that humans seem to use in solving problems. This is the approach underlying the development of expert systems (Chapter 8), and it has proved to be a fruitful one.

Still, the appeal of reasoning based in formal mathematical logic is too strong to ignore. Many important problems such as the design and verification of logic circuits, verification of the correctness of computer programs, and control of complex systems seem to respond to such an approach. In addition, the theorem-proving community has enjoyed success in devising powerful solution heuristics that rely solely on an evaluation of the syntactic form of a logical expression, and as a result, reducing the complexity of the search space without resorting to the *ad hoc* techniques used by most human problem solvers.

Another reason for the continued interest in automatic theorem provers is the realization that such a system does not have to be capable of independently solving extremely complex problems without human assistance. Many modern theorem provers function as intelligent assistants, letting humans perform the more demanding tasks of decomposing a large problem into subproblems and devising heuristics for searching the space of possible proofs. The theorem prover then performs the simpler but still demand- ing task of proving lemmas, verifying smaller conjectures, and completing the formal aspects of a proof outlined by its human associate (Boyer and Moore 1979, Bundy 1988, Veroff 1997, Veroff and Spinks 2006).

**1.2.3** Expert Systems

One major insight gained from early work in problem solving was the importance of domain-specific knowledge. A doctor, for example, is not effective at diagnosing illness solely because she possesses some innate general problem-solving skill; she is effective because she knows a lot about medicine. Similarly, a geologist is effective at discovering mineral deposits because he is able to apply a good deal of theoretical and empirical knowledge about geology to the problem at hand. Expert knowledge is a combination of a theoretical understanding of the problem and a collection of heuristic problem-solving rules that experience has shown to be effective in the domain. Expert systems are constructed by obtaining this knowledge from a human expert and coding it into a form that a computer may apply to similar problems.

This reliance on the knowledge of a human domain expert for the system’s problem solving strategies is a major feature of expert systems. Although some programs are writ- ten in which the designer is also the source of the domain knowledge, it is far more typical to see such programs growing out of a collaboration between a domain expert such as a doctor, chemist, geologist, or engineer and a separate artificial intelligence specialist. The domain expert provides the necessary knowledge of the problem domain through a

22 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

general discussion of her problem-solving methods and by demonstrating those skills on a carefully chosen set of sample problems. The AI specialist, or *knowledge engineer*, as expert systems designers are often known, is responsible for implementing this knowledge in a program that is both effective and seemingly intelligent in its behavior. Once such a program has been written, it is necessary to refine its expertise through a process of giving it example problems to solve, letting the domain expert criticize its behavior, and making any required changes or modifications to the program’s knowledge. This process is repeated until the program has achieved the desired level of performance.

One of the earliest systems to exploit domain-specific knowledge in problem solving was DENDRAL, developed at Stanford in the late 1960s (Lindsay et al. 1980). DEN- DRAL was designed to infer the structure of organic molecules from their chemical for- mulas and mass spectrographic information about the chemical bonds present in the molecules. Because organic molecules tend to be very large, the number of possible structures for these molecules tends to be huge. DENDRAL addresses the problem of this large search space by applying the heuristic knowledge of expert chemists to the structure elucidation problem. DENDRAL’s methods proved remarkably effective, routinely find- ing the correct structure out of millions of possibilities after only a few trials. The approach has proved so successful that descendants extensions of DENDRAL are cur- rently used in chemical and pharmaceutical laboratories throughout the world.

Whereas DENDRAL was one of the first programs to effectively use domain-specific knowledge to achieve expert performance, MYCIN established the methodology of con- temporary expert systems (Buchanan and Shortliffe 1984). MYCIN uses expert medical knowledge to diagnose and prescribe treatment for spinal meningitis and bacterial infec- tions of the blood. MYCIN, developed at Stanford in the mid-1970s, was one of the first programs to address the problems of reasoning with uncertain or incomplete information. MYCIN provided clear and logical explanations of its reasoning, used a control structure appropriate to the specific problem domain, and identified criteria to reliably evaluate its performance. Many of the expert system development techniques currently in use were first developed in the MYCIN project (Chapter 8).

Other early expert systems include the PROSPECTOR program for determining the probable location and type of ore deposits based on geological information about a site (Duda et al. 1979*a*, 1979*b*), the INTERNIST program for performing diagnosis in the area of internal medicine, the Dipmeter Advisor for interpreting the results of oil well drilling logs (Smith and Baker 1983), and XCON for configuring VAX computers. XCON was developed in 1981, and at one time every VAX sold by Digital Equipment Corporation was configured by that software. Numerous other expert systems are currently solving problems in areas such as medicine, education, business, design, and science (Waterman 1986, Durkin 1994). See also current proceedings of the Inovative Applications of Artifi- cial Intelligence (IAAI) Confersnces.

It is interesting to note that most expert systems have been written for relatively spe- cialized, expert level domains. These domains are generally well studied and have clearly defined problem-solving strategies. Problems that depend on a more loosely defined notion of “common sense” are much more difficult to solve by these means. In spite of the promise of expert systems, it would be a mistake to overestimate the ability of this technology. Current deficiencies include:

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 23

1. Difficulty in capturing “deep” knowledge of the problem domain. MYCIN, for example, lacks any real knowledge of human physiology. It does not know what blood does or the function of the spinal cord. Folklore has it that once, when selecting a drug for treatment of meningitis, MYCIN asked whether the patient was pregnant, even though it had been told that the patient was male. Whether this actually occurred or not, it does illustrate the potential narrowness of knowl- edge in expert systems.

2. Lack of robustness and flexibility. If humans are presented with a problem instance that they cannot solve immediately, they can generally return to an examination of first principles and come up with some strategy for attacking the problem. Expert systems generally lack this ability.

3. Inability to provide deep explanations. Because expert systems lack deep knowledge of their problem domains, their explanations are generally restricted to a description of the steps they took in finding a solution. For example, they often cannot tell “why” a certain approach was taken.

4. Difficulties in verification. Though the correctness of any large computer system is difficult to prove, expert systems are particularly difficult to verify. This is a serious problem, as expert systems technology is being applied to critical applications such as air traffic control, nuclear reactor operations, and weapons systems.

5. Little learning from experience. Current expert systems are handcrafted; once the system is completed, its performance will not improve without further attention from its programmers, leading to doubts about the intelligence of such systems.

In spite of these limitations, expert systems have proved their value in a number of important applications. Expert systems are a major topic in this text and are discussed in Chapters 7 and 8. Current applications can often be found in the proceedings of the Inova- tive Applications of Artificial Intelligence (IAAI) conferences.

**1.2.4** Natural Language Understanding and Semantics

One of the long-standing goals of artificial intelligence is the creation of programs that are capable of understanding and generating human language. Not only does the ability to use and understand natural language seem to be a fundamental aspect of human intelligence, but also its successful automation would have an incredible impact on the usability and effectiveness of computers themselves. Much effort has been put into writing programs that understand natural language. Although these programs have achieved success within restricted contexts, systems that can use natural language with the flexibility and general- ity that characterize human speech are beyond current methodologies.

Understanding natural language involves much more than parsing sentences into their individual parts of speech and looking those words up in a dictionary. Real understanding depends on extensive background knowledge about the domain of discourse and the

24 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

idioms used in that domain as well as an ability to apply general contextual knowledge to resolve the omissions and ambiguities that are a normal part of human speech.

Consider, for example, the difficulties in carrying on a conversation about baseball with an individual who understands English but knows nothing about the rules, players, or history of the game. Could this person possibly understand the meaning of the sentence: “With none down in the top of the ninth and the go-ahead run at second, the manager called his relief from the bull pen”? Even though all of the words in the sentence may be individually understood, this sentence would be gibberish to even the most intelligent non-baseball fan.

The task of collecting and organizing this background knowledge in such a way that it may be applied to language comprehension forms the major problem in automating natural language understanding. Responding to this need, researchers have developed many of the techniques for structuring semantic meaning used throughout artificial intelligence (Chapters 7 and 15).

Because of the tremendous amounts of knowledge required for understanding natural language, most work is done in well-understood, specialized problem areas. One of the earliest programs to exploit this “micro world” methodology was Winograd’s SHRDLU, a natural language system that could “converse” about a simple configuration of blocks of different shapes and colors (Winograd 1973). SHRDLU could answer queries such as “what color block is on the blue cube?” as well as plan actions such as “move the red pyr- amid onto the green brick”. Problems of this sort, involving the description and manipula- tion of simple arrangements of blocks, appeared with surprising frequency in early AI research and are known as “blocks world” problems.

In spite of SHRDLU’s success in conversing about arrangements of blocks, its methods did not generalize from that domain. The representational techniques used in the program were too simple to capture the semantic organization of richer and more complex domains in a useful way. Much of the current work in natural language under- standing is devoted to finding representational formalisms that are general enough to be used in a wide range of applications yet adapt themselves well to the specific structure of a given domain. A number of different techniques (many of which are extensions or modifi- cations of *semantic networks*) are explored for this purpose and used in the development of programs that can understand natural language in constrained but interesting knowl- edge domains. Finally, in current research (Marcus 1980, Manning and Schutze 1999, Jurafsky and Martin 2009) stochastic models, describing how words and language struc- tures “occur” in use, are employed to characterize both syntax and semantics. Full compu- tational understanding of language, however, remains beyond the current state of the art.

**1.2.5** Modeling Human Performance

Although much of the above discussion uses human intelligence as a reference point in considering artificial intelligence, it does not follow that programs should pattern them- selves after the organization of the human mind. Indeed, many AI programs are engi- neered to solve some useful problem without regard for their similarities to human mental architecture. Even expert systems, while deriving much of their knowledge from human

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 25

experts, do not really attempt to simulate human internal mental problem solving pro- cesses. If performance is the only criterion by which a system will be judged, there may be little reason to attempt to simulate human problem-solving methods; in fact, programs that take nonhuman approaches to solving problems (chess) are often more successful than their human counterparts. Still, the design of systems that explicitly model aspects of human performance is a fertile area of research in both AI and psychology.

Human performance modeling, in addition to providing AI with much of its basic methodology, has proved to be a powerful tool for formulating and testing theories of human cognition. The problem-solving methodologies developed by computer scientists have given psychologists a new metaphor for exploring the human mind. Rather than casting theories of cognition in the vague language used in early research or abandoning the problem of describing the inner workings of the human mind entirely (as suggested by the behaviorists), many psychologists have adopted the language and theory of computer science to formulate models of human intelligence. Not only do these techniques provide a new vocabulary for describing human intelligence, but also computer implementations of these theories offer psychologists an opportunity to empirically test, critique, and refine their ideas (Luger 1994; See the Cognitive Science Society’s journal and conferences). The relationship between artificial and human intelligence is summarized in Chapter 16.

**1.2.6** Planning and Robotics

Research in planning began as an effort to design robots that could perform their tasks with some degree of flexibility and responsiveness to the outside world. Briefly, planning assumes a robot that is capable of performing certain atomic actions. It attempts to find a sequence of those actions that will accomplish some higher-level task, such as moving across an obstacle-filled room.

Planning is a difficult problem for a number of reasons, not the least of which is the size of the space of possible sequences of moves. Even an extremely simple robot is capable of generating a vast number of potential move sequences. Imagine, for example, a robot that can move forward, backward, right, or left, and consider how many different ways that robot can possibly move around a room. Assume also that there are obstacles in the room and that the robot must select a path that moves around them in some efficient fashion. Writing a program that can discover the best path under these circumstances, without being overwhelmed by the huge number of possibilities, requires sophisticated techniques for representing spatial knowledge and controlling search through possible environments.

One method that human beings use in planning is *hierarchical problem decomposi- tion*. If you are planning a trip from Albuquerque to London, you will generally treat the problems of arranging a flight, getting to the airport, making airline connections, and find- ing ground transportation in London separately, even though they are all part of a bigger overall plan. Each of these may be further decomposed into smaller subproblems such as finding a map of the city, negotiating the subway system, and finding a decent pub. Not only does this approach effectively restrict the size of the space that must be searched, but also supports the saving of frequently used subplans for future use.

26 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

While humans plan effortlessly, creating a computer program that can do the same is a difficult challenge. A seemingly simple task such as breaking a problem into independent subproblems actually requires sophisticated heuristics and extensive knowledge about the planning domain. Determining what subplans should be saved and how they may be generalized for future use is an equally difficult problem.

A robot that blindly performs a sequence of actions without responding to changes in its environment or being able to detect and correct errors in its own plan could hardly be considered intelligent. A robot may not have adequate sensors to locate all obstacles in the way of a projected path. Such a robot must begin moving through the room based on what it has “perceived” and correct its path as other obstacles are detected. Organizing plans in a fashion that allows response to changing environmental conditions is a major problem for planning (Lewis and Luger 2000, Thrun et al. 2007).

Finally, robotics was one of the research areas in AI that produced many of the insights supporting agent-oriented problem solving (Section 1.1.4). Frustrated by both the complexities of maintaining the large representational space as well as the design of search algorithms for traditional planning, researchers, including Agre and Chapman (1987) , Brooks (1991a), Thrun et al. (2007), restated the problem in terms of the interac- tion of multiple semi-autonomous agents. Each agent was responsible for its own portion of the problem task and through their coordination the larger solution would emerge.

Planning research now extends well beyond the domains of robotics, to include the coordination of any complex set of tasks and goals. Modern planners are applied to agents (Nilsson 1994) as well as to control of particle beam accelerators (Klein et al. 1999, 2000).

**1.2.7** Languages and Environments for AI

Some of the most important by-products of artificial intelligence research have been advances in programming languages and software development environments. For a num- ber of reasons, including the size of many AI application programs, the importance of a prototyping methodology, the tendency of search algorithms to generate huge spaces, and the difficulty of predicting the behavior of heuristically driven programs, AI programmers have been forced to develop a powerful set of programming methodologies.

Programming environments include knowledge-structuring techniques such as object-oriented programming. High-level languages, such as Lisp and Prolog, which sup- port modular development, help manage program size and complexity. Trace packages allow a programmer to reconstruct the execution of a complex algorithm and make it pos- sible to unravel the complexities of heuristic search. Without such tools and techniques, it is doubtful that many significant AI systems could have been built.

Many of these techniques are now standard tools for software engineering and have little relationship to the core of AI theory. Others, such as object-oriented programming, are of significant theoretical and practical interest. Finally, many AI algorithms are also now built in more traditional computing languages, such as C++ and Java.

The languages developed for artificial intelligence programming are intimately bound to the theoretical structure of the field. We have built many of the representational struc- tures presneted in this book in Prolog, Lisp and Java and make them available in Luger

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 27

and Stubblefield (2009) and on the internet. In this book we remain apart from religious debates over their relative merits different languages. Rather, We adhere to the adage “the good professional knows all her tools.”

**1.2.8** Machine Learning

Learning has remained a challenging area for AI. The importance of learning, however, is beyond question, particularly as this ability is one of the most important components of intelligent behavior. An expert system may perform extensive and costly computations to solve a problem. Unlike a human being, however, if it is given the same or a similar prob- lem a second time, it usually does not remember the solution. It performs the same sequence of computations again. This is true the second, third, fourth, and every time it solves that problem—hardly the behavior of an intelligent problem solver. The obvious solution to this problem is for programs to learn on their own, either from experience, analogy, examples, being “told” what to do, or rewarded or punished depending on results.

Although learning is a difficult area, there are several programs that suggest that it is not impossible. One early program is AM, the *Automated Mathematician*, designed to dis- cover mathematical laws (Lenat 1977, 1982). Initially given the concepts and axioms of set theory, AM was able to induce such important mathematical concepts as cardinality, integer arithmetic, and many of the results of number theory. AM conjectured new theo- rems by modifying its current knowledge base and used heuristics to pursue the “best” of a number of possible alternative theorems. More recently, Cotton et al. (2000) designed a program that automatically invents “interesting” integer sequences.

Early influential work also includes Winston’s research on the induction of structural concepts such as “arch” from a set of examples in the blocks world (Winston 1975*a*). The ID3 algorithm has proved successful in learning general patterns from examples (Quinlan 1986*a*). *Meta-DENDRAL* learns rules for interpreting mass spectrographic data in organic chemistry from examples of data on compounds of known structure. *Teiresias*, an intelli- gent “front end” for expert systems, converts high-level advice into new rules for its knowledge base (Davis 1982). *Hacker* devises plans for performing blocks world manipu- lations through an iterative process of devising a plan, testing it, and correcting any flaws discovered in the candidate plan (Sussman 1975). Work in explanation-based learning has shown the effectiveness of prior knowledge in learning (Mitchell et al. 1986, DeJong and Mooney 1986). There are also now many important biological and sociological models of learning; we review these in the connectionist learning and emergent learning chapters.

The success of machine learning programs suggests the existence of a set of general learning principles that will allow the construction of programs with the ability to learn in realistic domains. We present several approaches to learning in Section IV.

**1.2.9** Alternative Representations: Neural Nets and Genetic Algorithms

Most of the techniques presented in this AI book use explicitly represented knowledge and carefully designed search algorithms to implement intelligence. A very different approach

28 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

seeks to build intelligent programs using models that parallel the structure of neurons in the human brain or the evolving patterns found in genetic algorithms and artificial life.

A simple schematic of a neuron (Figure 1.2) consists of a cell body that has a number of branched protrusions, called *dendrites*, and a single branch called the *axon*. Dendrites receive signals from other neurons. When these combined impulses exceed a certain threshold, the neuron fires and an impulse, or *spike*, passes down the axon. Branches at the end of the axon form *synapses* with the dendrites of other neurons. The synapse is the point of contact between neurons; synapses may be either *excitatory* or *inhibitory*, either adding to the total of signals reaching the neuron or subtracting from that total.

This description of a neuron is excessively simple, but it captures those features that are relevant to neural models of computation. In particular, each computational unit com- putes some function of its inputs and passes the result along to connected units in the net- work: the final results are produced by the parallel and distributed processing of this network of neural connections and threshold weights.

Neural architectures are appealing mechanisms for implementing intelligence for a number of reasons. Traditional AI programs can be brittle and overly sensitive to noise. Human intelligence is much more flexible and good at interpreting noisy input, such as a face in a darkened room or a conversation at a noisy party. Neural architectures, because they capture knowledge in a large number of fine-grained units distributed about a net- work, seem to have more potential for partially matching noisy and incomplete data.

With genetic algorithms and artificial life we evolve new problem solutions from components of previous solutions. The genetic operators, such as *crossover* and *mutation*, much like their genetic equivalents in the natural world, work to produce, for each new generation, ever better potential problem solutions. Artificial life produces its new genera- tion as a function of the “quality” of its neighbors in previous generations.

Both neural architectures and genetic algorithms provide a natural model for parallel- ism, because each neuron or segment of a solution is an independent unit. Hillis (1985) has commented on the fact that humans get faster at a task as they acquire more knowl- edge, while computers tend to slow down. This slowdown is due to the cost of sequen- tially searching a knowledge base; a massively parallel architecture like the human brain would not suffer from this problem. Finally, something is intrinsically appealing about approaching the problems of intelligence from a neural or genetic point of view. After all, the evolved brain achieves intelligence and it does so using a neural architecture. We present neural networks, genetic algorithms, and artificial life, in Chapters 10 and 11.

Synapse

Cell body

Dendrite

Axon

Figure 1.2 A simplified diagram of a neuron, from

Crick and Asanuma (1986).

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 29

**1.2.10** AI and Philosophy

In Section 1.1 we presented the philosophical, mathematical, and sociological roots of artificial intelligence. It is important to realize that modern AI is not just a product of this rich intellectual tradition but also contributes to it.

For example, the questions that Turing posed about intelligent programs reflect back on our understanding of intelligence itself. What is intelligence, and how is it described? What is the nature of knowledge? Can knowledge be represented? How does knowledge in an application area relate to problem-solving skill in that domain? How does *knowing what* is true, Aristotle’s *theoria*, relate to *knowing how* to perform, his *praxis*?

Answers proposed to these questions make up an important part of what AI research- ers and designers do. In the scientific sense, AI programs can be viewed as experiments. A design is made concrete in a program and the program is run as an experiment. The pro- gram designers observe the results and then redesign and rerun the experiment. In this manner we can determine whether our representations and algorithms are sufficient mod- els of intelligent behavior. Newell and Simon (1976) proposed this approach to scientific understanding in their 1976 Turing Award lecture (Section VI). Newell and Simon (1976) also propose a stronger model for intelligence with their physical symbol system hypothe- sis: *the necessary and sufficient condition for a physical system to exhibit intelligence is that it be a physical symbol system*. We take up in Section VI what this hypothesis means in practice as well as how it has been criticized by many modern thinkers.

A number of AI application areas also open up deep philosophical issues. In what sense can we say that a computer can understand natural language expressions? To produce or understand a language requires interpretation of symbols. It is not sufficient to be able to say that a string of symbols is well formed. A mechanism for understanding must be able to impute meaning or interpret symbols in context. What is meaning? What is interpretation? In what sense does interpretation require responsibility?

Similar philosophical issues emerge from many AI application areas, whether they be building expert systems to cooperate with human problem solvers, designing computer vision systems, or designing algorithms for machine learning. We look at many of these issues as they come up in the chapters of this book and address the general issue of relevance to philosophy again in Section VI.

**1.3** Artificial Intelligence—A Summary

We have attempted to define artificial intelligence through discussion of its major areas of research and application. This survey reveals a young and promising field of study whose primary concern is finding an effective way to understand and apply intelligent problem solving, planning, and communication skills to a wide range of practical problems. In spite of the variety of problems addressed in artificial intelligence research, a number of important features emerge that seem common to all divisions of the field; these include:

1. The use of computers to do reasoning, pattern recognition, learning, or some

other form of inference.

30 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

2. A focus on problems that do not respond to algorithmic solutions. This underlies

the reliance on heuristic search as an AI problem-solving technique.

3. A concern with problem solving using inexact, missing, or poorly defined infor- mation and the use of representational formalisms that enable the programmer to compensate for these problems.

4. Reasoning about the significant qualitative features of a situation.

5. An attempt to deal with issues of semantic meaning as well as syntactic form.

6. Answers that are neither exact nor optimal, but are in some sense “sufficient”. This is a result of the essential reliance on heuristic problem-solving methods in situations where optimal or exact results are either too expensive or not possible.

7. The use of large amounts of domain-specific knowledge in solving problems.

This is the basis of expert systems.

8. The use of meta-level knowledge to effect more sophisticated control of problem solving strategies. Although this is a very difficult problem, addressed in rela- tively few current systems, it is emerging as an essential area of research.

We hope that this introduction provides some feel for the overall structure and significance of the field of artificial intelligence. We also hope that the brief discussions of such technical issues as search and representation were not excessively cryptic and obscure; they are developed in proper detail throughout the remainder of the book, but included here to demonstrate their significance in the general organization of the field.

As we mentioned in the discussion of agent-oriented problem solving, objects take on meaning through their relationships with other objects. This is equally true of the facts, theories, and techniques that constitute a field of scientific study. We have intended to give a sense of those interrelationships, so that when the separate technical themes of artificial intelligence are presented, they will find their place in a developing understanding of the overall substance and directions of the field. We are guided in this process by an observa- tion made by Gregory Bateson (1979), the psychologist and systems theorist:

Break the pattern which connects the items of learning and you necessarily destroy all quality.

**1.4** Epilogue and References

The field of AI reflects some of the oldest concerns of Western civilization in the light of the modern computational model. The notions of rationality, representation, and reason are now under scrutiny as perhaps never before, because we workers in AI demand to understand them algorithmically! At the same time, the political, economic, and ethical situation of our species forces us to confront our responsibility for the effects of our artifices.

Many excellent sources are available on the topics raised in this chapter: *Mind Design* (Haugeland 1997), *Artificial Intelligence: The Very Idea* (Haugeland 1985), *Brainstorms*

CHAPTER 1 / AI: HISTORY AND APPLICATIONS 31

(Dennett 1978), *Mental Models* (Johnson-Laird 1983), *Elbow Room* (Dennett 1984), *The Body in the Mind* (Johnson 1987), *Consciousness Explained* (Dennett 1991), and *Darwin’s Dangerous Idea* (Dennett 1995), *Prehistory of Android Epistemology* (Glymour, Ford, and Hayes 1995a), and *Sweet Dreems* (Dennett 2006).

Several of the primary sources are also readily available, including Aristotle’s *Phys- ics*, *Metaphysics*, and *Logic*; papers by Frege; and the writings of Babbage, Boole, and Russell and Whitehead. Turing’s papers are also very interesting, especially his discus- sions of the nature of intelligence and the possibility of designing intelligent programs (Turing 1950). Turing's famous 1937 paper *On Computable Numbers, with an Application to the Entscheidungsproblem* worked out the theory of Turing machines and the definition of computability. Turing’s biography, *Alan Turing: The Enigma* (Hodges 1983), makes excellent reading. Selfridge’s *Pandemonium* (1959) is an early example of learning. An important collection of early papers in AI may be found in Webber and Nilsson (1981).

*Computer Power and Human Reason* (Weizenbaum 1976) and *Understanding Computers and Cognition* (Winograd and Flores 1986) offer sobering comments on the limitations of and ethical issues in AI. *The Sciences of the Artificial* (Simon 1981) is a pos- itive statement on the possibility of artificial intelligence and its role in society.

The AI applications mentioned in Section 1.2 are intended to introduce the reader to the broad interests of AI researchers and outline many of the important questions under investigation. Each of these subsections referenced the primary areas in this book where these topics are presented. *The Handbook of Artificial Intelligence* (Barr and Feigenbaum 1989) also offers an introduction to many of these areas. The *Encyclopedia of Artificial Intelligence* (Shapiro 1992) offers a clear and comprehensive treatment of the field of arti- ficial intelligence.

Natural language understanding is a dynamic field of study; some important points of view are expressed in *Natural Language Understanding* (Allen 1995), *Language as a Cognitive Process* (Winograd 1983), *Computer Models of Thought and Language* (Schank and Colby 1973), *Grammar, Meaning and the Machine Analysis of Language* (Wilks 1972), *The Language Instinct* (Pinker 1994), *Philosophy in the Flesh* (Lakoff and Johnson 1999), and *Speech and Language Processing* (Jurafsky and Martin 2009); an introduction to the field is presented in our Chapters 7 and 15.

Using computers to model human performance, which we address briefly in Chapter 17, is discussed in some depth in *Human Problem Solving* (Newell and Simon 1972), *Computation and Cognition* (Pylyshyn 1984), *Arguments Concerning Representations for Mental Imagery* (Anderson 1978), *Cognitive Science: the Science of Intelligent Systems* (Luger 1994), *Problem Solving as Model Refinement: Towards a Constructivist Epistemology* (Luger et al. 2002), and *Bayesian Brain*, (Doya et al. 2007).

Machine learning is discussed in Section IV; the multi-volume set, *Machine Learning* (Michalski et al. 1983, 1986; Kodratoff and Michalski 1990), the *Journal of Artificial Intelligence* and the *Journal of Machine Learning* are important resources. Further refer- ences may be found in the four chapters of Section IV.

Finally, Chapter 12 presents a view of intelligence that emphasizes its modular struc- ture and adaptation within a social and natural context. Minsky’s *Society of Mind* (1985) is one of the earliest and most thought provoking articulations of this point of view. Also see *Android Epistemology* (Ford et al. 1995b) and *Artificial Life* (Langton 1995).

32 PART I / ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

**1.5** Exercises

1. Create and justify your own definition of artificial intelligence.

2. Give several other examples of Aristotle’s distinction between *matter* and *form*. Can you

show how your examples might fit into a theory of abstraction?

3. Much traditional Western thought has dwelt on the mind–body relationship. Are the mind

and body:

a. distinct entities somehow interacting, or

b. is mind an expression of “physical processes”, or

c. is body just an illusion of the rational mind?

Discuss your thoughts on the mind–body problem and its importance for a theory of artificial intelligence.

4. Criticize Turing’s criteria for computer software being “intelligent”.

5. Describe your own criteria for computer software to be considered “intelligent”.

6. Although computing is a relatively new discipline, philosophers and mathematicians have been thinking about the issues involved in automating problem solving for thousands of years. What is your opinion of the relevance of these philosophical issues to the design of a device for intelligent problem solving? Justify your answer.

7. Given the differences between the architectures of modern computers and that of the human brain, what relevance does research into the physiological structure and function of biological systems have for the engineering of AI programs? Justify your answer.

8. Pick one problem area that you feel would justify the energy required to design an expert system solution. Spell the problem out in some detail. Based on your own intuition, which aspects of this solution would be most difficult to automate?

9. Add two more benefits for expert systems to those already listed in the text. Discuss these in

terms of intellectual, social, or financial results.

10. Discuss why you think the problem of machines “learning” is so difficult.

11. Discuss whether or not you think it is possible for a computer to understand and use a natural

(human) language.

12. List and discuss two potentially negative effects on society of the development of artificial

intelligence technologies.

33

MOSBKUIN 11

**3.ITNA**

or bouw

-64

S*3*

Alina

HIZTEILE

panda te

**T Jordatlan**

mas

P A R T II

**ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH**

A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE (url IIa)

*We propose that a 2 month, 10 man (sic) study of artificial intelligence be carried out dur- ing the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these prob- lems if a carefully selected group of scientists work on it together for a summer.*

J. MCCARTHY, *Dartmouth College* M. L. MINSKY, *Harvard University* N. ROCHESTER, *I.B.M. Corporation* C.E. SHANNON, *Bell Telephone Laboratories*

August 31, 1955

Introduction to Representation and Search

From an engineering perspective, the description of artificial intelligence presented in Section 1.3 may be summarized as *the study of representation and search through which intelligent activity can be enacted on a mechanical device*. This perspective has dominated the origins and growth of AI.

The first modern workshop/conference for AI practitioners was held at Dartmouth College in the summer of 1956. The proposal for this workshop is presented as the intro- ductory quotation for Part II. This workshop, where the name *artificial intelligence* itself

35

was chosen, brought together many of the then current researchers focused on the integra- tion of computation and intelligence. There were also a few computer programs written by that time reflecting these early ideas. The main topics for discussion at this conference, abridged here from the original workshop proposal (url IIa), were:

**1. Automatic Computers**

If a machine can do a job, then an automatic calculator can be programmed to simu- late the machine.

**2. How Can a Computer be Programmed to Use a Language**

It may be speculated that a large part of human thought consists of manipulating words according to rules of reasoning and rules of conjecture.

**3. Neuron Nets**

How can a set of (hypothetical) neurons be arranged so as to form concepts?

**4. Theory of the Size of a Calculation**

If we are given a well-defined problem (one for which it is possible to test mechani- cally whether or not a proposed answer is a valid answer) one way of solving it is to try all possible answers in order. This method is inefficient, and to exclude it one must have some criterion for efficiency of calculation.

**5. Self-improvement** (Machine Learning)

Probably a truly intelligent machine will carry out activities which may best be described as self-improvement.

**6. Abstractions**

A number of types of “abstraction” can be distinctly defined and several others less distinctly. A direct attempt to classify these and to describe machine methods of form- ing abstractions from sensory and other data would seem worthwhile.

**7. Randomness and Creativity**

A fairly attractive and yet clearly incomplete conjecture is that the difference between creative thinking and unimaginative competent thinking lies in the injection of a some randomness.

It is interesting to note that the topics proposed for this first conference on artificial intelligence capture many of the issues, such as complexity theory, methodologies for abstraction, language design, and machine learning that make up the focus of modern

36 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

computer science. In fact, many of the defining characteristics of computer science as we know it today have their roots in AI. AI has also had its own historical and political strug- gles, with several of these early topics proposed for research, such as “neuron nets” and “randomness and creativity” were put into background mode for decades.

A powerful new computational tool, the Lisp language, emerged at about this time, built under the direction of John McCarthy, one of the original proposers of the Dartmouth Workshop. Lisp addressed several of the topics of the Workshop, supporting the ability to create relationships that could themselves be manipulated by other structures of the lan- guage. Lisp gave artificial intelligence both a highly expressive language, rich in abstrac- tion, as well as a medium for interpretation of these expressions.

The availability of the Lisp programming language did shape much of the early devel- opment of AI, in particular, the use of the predicate calculus as a representational medium as well as search to explore the efficacy of different logical alternatives, what we now call *graph search*. Prolog, created in the late 1970s, would offer AI a similar powerful compu- tational tool.

An introduction to the fundamental representation and search techniques supporting work in artificial intelligence make up the five chapters of Part II. The predicate calculus, graph search, heuristic and stochastic methods, and architectures (control systems) for intelligent problem solving make up the material of Part II. These technologies reflect the dominant techniques explored by the AI community during its first two decades.

**Representational Systems**

The function of any representation scheme is to capture - often called *abstract out* - the critical features of a problem domain and make that information accessible to a problem- solving procedure. *Abstraction* is an essential tool for managing complexity as well as an important factor in assuring that the resulting programs are computationally efficient. *Expressiveness* (the result of the features abstracted) and *efficiency* (the computationall complexity of the algorithms used on the abstracted features) are major dimensions for evaluating knowledge representation languages. Sometimes, expressiveness must be sacrificed to improve an algorithm’s efficiency. This must be done without limiting the representation’s ability to capture essential problem-solving knowledge. Optimizing the trade-off between efficiency and expressiveness is a major task for designers of intelligent programs.

Knowledge representation languages can also be tools for helping humans solve problems. As such, a representation should provide a *natural* framework for expressing problem-solving knowledge; it should make that knowledge available to the computer and assist the programmer in its organization.

The computer representation of floating-point numbers illustrates these trade-offs (see Fig. II.1). To be precise, real numbers require an infinite string of digits to be fully described; this cannot be accomplished on a finite device such as a computer. One answer to this dilemma is to represent the number in two pieces: its *significant* digits and the loca- tion within those digits of the decimal point. Although it is not possible to actually store a real number in a computer, it is possible to create a representation that functions ade- quately in most practical applications.

PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 37

The real number:

The decimal equivalent:

The floating point representation:

The representation in computer memory:

π

3.1415927 . . .

31416 1

Exponent Mantissa

11100010

Figure II.1 Different representations of the real number π.

Floating-point representation thus sacrifices full expressive power to make the repre- sentation efficient, in this case to make it possible. This representation also supports algo- rithms for multiple-precision arithmetic, giving effectively infinite precision by limiting round-off error to any pre-specified tolerance. It also guarantees well-behaved round-off errors. Like all representations, it is only an abstraction, a symbol pattern that designates a desired entity and not the entity itself.

The array is another representation common in computer science. For many prob- lems, it is more natural and efficient than the memory architecture implemented in com- puter hardware. This gain in naturalness and efficiency involves compromises in expressiveness, as illustrated by the following example from image processing. Figure II.2 is a digitized image of human chromosomes in a stage called *metaphase*. The image is processed to determine the number and structure of the chromosomes, looking for breaks, missing pieces, and other abnormalities.

The visual scene is made up of a number of picture points. Each picture point, or *pixel*, has both a location and a number value representing its intensity or *gray level*. It is natural, then, to collect the entire scene into a two-dimensional array where the row and column address gives the location of a pixel (X and Y coordinates) and the content of the array element is the gray level at that point. Algorithms are designed to perform operations like looking for isolated points to remove noise from the image, finding threshold levels for discerning objects and edges, summing contiguous elements to determine size or den- sity, and in various other ways transforming the picture point data. Implementing these algorithms is straightforward, given the array representation and the FORTRAN language, for example. This task would be quite cumbersome using other representations such as the predicate calculus, records, or assembly code, because these do not have a natural fit with the material being represented.

When we represent the picture as an array of pixel points, we sacrifice fineness of resolution (compare a photo in a newspaper to the original print of the same picture). In addition, pixel arrays cannot express the deeper semantic organization of the image. For example, a pixel array cannot represent the organization of chromosomes in a single cell nucleus, their genetic function, or the role of metaphase in cell division. This knowledge is more easily captured using a representation such as predicate calculus (Chapter 2) or semantic networks (Chapter 7). In summary, a representational scheme should be ade- quate to express all of the necessary information, support efficient execution of the result- ing code, and provide a natural scheme for expressing the required knowledge.

38 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

In general, the problems AI attempts to solve do not lend themselves to the represen- tations offered by more traditional formalisms such as arrays. Artificial intelligence is con- cerned with qualitative rather than quantitative problem solving, with reasoning rather than numeric calculation, and with organizing large and varied amounts of knowledge rather than implementing a single, well-defined algorithm.

For example, consider Figure II.3, the arrangement of blocks on a table. Suppose we wish to capture the properties and relations required to control a robot arm. We must deter- mining which blocks are stacked on other blocks and which blocks have clear tops so that they can be picked up.The *predicate calculus* offers a medium to capture this descriptive information. The first word of each expression (on, ontable, etc.) is a *predicate* denoting some property or relationship among its *arguments* (appearing in the parentheses). The arguments are symbols denoting objects (blocks) in the domain. The collection of logical clauses describes the important properties and relationships of this *blocks world*:

clear(c) clear(a) ontable(a) ontable(b) on(c, b) cube(b) cube(a) pyramid(c)

Figure II.2 Digitized image of chromosomes in metaphase.

PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 39

a bc

Figure II.3 A blocks world.

Predicate calculus provides artificial intelligence programmers with a well-defined language for describing and reasoning about qualitative aspects of a system. Suppose, in the blocks world example, we want to define a test to determine whether a block is clear, that is, has nothing stacked on top of it. This is important if the robot hand is to pick it up or stack another block on top of it. We can define a general rule:

∀ X ¬ ∃ Y on(Y,X) ⇒ clear(X)

This is read “for all X, X is clear if there does not exist a Y such that Y is on X.” This gen- eral rule can be applied to a variety of situations by substituting different block names, a, b, c, etc., for X and Y. By supporting general inference rules, predicate calculus allows economy of representation, as well as the possibility of designing systems that are flexible and general enough to respond intelligently to a range of situations.

The predicate calculus may also be used to represent the properties of individuals and groups. It is often not sufficient, for example, to describe a car by simply listing its component parts; we may want to describe the ways in which those parts are combined and the interactions between them. This view of structure is essential to a range of situations including taxonomic information, such as the classification of plants by genus and species, or a description of complex objects such as a diesel engine or a human body in terms of their component parts. For example, a simple description of a bluebird might be “a bluebird is a small blue-colored bird and a bird is a feathered flying verte- brate”, which may be represented as the set of logical predicates:

hassize(bluebird,small) hascovering(bird,feathers) hascolor(bluebird,blue) hasproperty(bird,flies) isa(bluebird,bird) isa(bird,vertebrate)

This predicate description can be represented graphically by using the *arcs*, or *links*, in a graph instead of predicates to indicate relationships (Fig. II.4). This *semantic network*, is a technique for representing semantic meaning. Because relationships are explicitly denoted in the graph, an algorithm for reasoning about a problem domain could make rel- evant associations by following the links. In the bluebird illustration, for example, the pro- gram need only follow one link to se that a blubird flies and two links to determine that a bluebird is a vertebrate.

40 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

vertebrate

isa

hascovering

hasproperty feathers

bird

isa

hassize

hascolor small

bluebird

Figure II.4 Semantic network description of a bluebird.

flies

blue

Perhaps the most important application for semantic networks is to represent mean- ings for language understanding programs. When it is necessary to understand a child’s story, the details of a journal article, or the contents of a web page, semantic networks may be used to encode the information and relationships that reflect the knowledge in that application. Semantic networks are discussed in Chapter 6, and their application to lan- guage understanding in Chapter 15.

**Search**

Given a representation, the second component of intelligent problem solving is *search*. Humans generally consider a number of alternative strategies on their way to solving a problem. A chess player typically reviews alternative moves, selecting the “best” accord- ing to criteria such as the opponent’s possible responses or the degree to which various moves support some global game strategy. A player also considers short-term gain (such as taking the opponent’s queen), opportunities to sacrifice a piece for positional advan- tage, or conjectures concerning the opponent’s psychological makeup and level of skill. This aspect of intelligent behavior underlies the problem-solving technique of *state space search*.Consider, for example, the game of tic-tac-toe. Given any board situation, there is only a finite number of moves that a player can make. Starting with an empty board, the first player may place an X in any one of nine places. Each of these moves yields a differ- ent board that will allow the opponent eight possible responses, and so on. We can repre- sent this collection of possible moves and responses by regarding each board configuration as a *node* or *state* in a graph. The *links* of the graph represent legal moves from one board configuration to another. The resulting structure is a *state space graph*.

The state space representation thus enables us to treat all possible games of tic-tac-toe as different paths through the state space graph. Given this representation, an effective game strategy will search through the graph for the paths that lead to the most wins and fewest losses and play in a way that always tries to force the game along one of these opti- mal paths, as in Figure II.5.

PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 41

X X X X X X X X XX X X X X X X X X X X X 0 0

0 0 0 0 0 0 0 0

0 0

X X

X X X 0 X 0 X 0 X X 0 X 0 X 0 X X 0 X 0 X 0 X X 0 X 0

XX X X

X X 0

Figure II.5 Portion of the state space for tic-tac-toe.

As an example of how search is used to solve a more complicated problem, consider the task of diagnosing a mechanical fault in an automobile. Although this problem does not initially seem to lend itself to state space search as easily as tic-tac-toe or chess, it actu- ally fits this strategy quite well. Instead of letting each node of the graph represent a “board state,” we let it represent a state of partial knowledge about the automobile’s mechanical problems. The process of examining the symptoms of the fault and inducing its cause may be thought of as searching through states of increasing knowledge. The starting node of the graph is empty, indicating that nothing is known about the cause of the problem. The first thing a mechanic might do is ask the customer which major system (engine, transmission, steering, brakes, etc.) seems to be causing the trouble. This is repre- sented by a collection of arcs from the start state to states that indicate a focus on a single subsystem of the automobile, as in Figure II.6.

Each of the states in the graph has arcs (corresponding to basic diagnostic checks) that lead to states representing further accumulation of knowledge in the diagnostic process.

42 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

start ask: where is the problem?

engine trouble transmission brakes **. . .**

Figure II.6 State space description of the first step in

diagnosing an automotive problem.

engine starts

engine won't start ask: . . . ask:

Will engine turn over?

turns over ask: . . .

start ask: where is the problem?

engine trouble ask: does the car start?

transmission

**. . .** ask: . . .

brakes ask: . . .

yes no

yes no

won't turn over ask: Do lights come on?

no yes battery dead

Figure II.7 State space description of the automotive

diagnosis problem.

battery ok

For example, the engine trouble node has arcs to nodes labeled engine starts and engine won t start. From the won t startnode we may move to nodes labeled turns over and won t turn over. The won t turn over node has arcs to nodes labeled battery dead and battery ok, see Figure II.7. A problem solver can diagnose car trouble by searching for a path through this graph that is consistent with the symptoms of a particular defective

PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH 43

car. Although this problem is very different from that of finding an optimal way to play tic-tac-toe or chess, it is equally amenable to solution by state space search.

In spite of this apparent universality, state space search is not, by itself, sufficient for automating intelligent problem-solving behavior; rather it is an important tool for the design of intelligent programs. If state space search were sufficient, it would be fairly sim- ple to write a program that plays chess by searching through the entire space for the sequence of moves that brought a victory, a method known as *exhaustive search*. Though exhaustive search can be applied to any state space, the overwhelming size of the space for interesting problems makes this approach a practical impossibility. Chess, for example, has approximately 10120 different board states. This is a number larger than the number of molecules in the universe or the number of nanoseconds that have passed since the *big bang*. Search of this space is beyond the capabilities of any computing device, whose dimensions must be confined to the known universe and whose execution must be com- pleted before the universe succumbs to the ravages of entropy.

Humans use intelligent search: a chess player considers a number of possible moves, a doctor examines several possible diagnoses, a computer scientist entertains different designs before beginning to write code. Humans do not use exhaustive search: the chess player examines only moves that experience has shown to be effective, the doctor does not require tests that are not somehow indicated by the symptoms at hand. Human problem solving seems to be based on judgmental rules that guide search to those portions of the state space that seem most “promising”.

These rules are known as *heuristics*, and they constitute one of the central topics of AI research. A heuristic (the name is taken from the Greek word “to discover”) is a strategy for selectively searching a problem space. It guides search along lines that have a high probability of success while avoiding wasted or apparently stupid efforts. Human beings use a large number of heuristics in problem solving. If you ask a mechanic why your car is overheating, she may say something like, “Usually that means the thermostat is bad.” If you ask a doctor what could cause nausea and stomach pains, he might say it is “probably either stomach flu or food poisoning.”

State space search gives us a means of formalizing the problem-solving process, and heuristics allow us to infuse that formalism with intelligence. These techniques are dis- cussed in detail in the early chapters of this book and remain at the heart of most modern work in AI. In summary, state space search is a formalism, independent of any particular search strategies, and used as a launch point for many problem solving approaches.

Throughout the text we continue to explore the theoretical aspects of knowledge rep- resentation and search and the use of this theory in building effective programs. The treat- ment of knowledge representation begins with Chapter 2 and the predicate calculus. Chapter 3 introduces search in the context of game graphs and other applications. In Chap- ter 4, heuristics are introduced and applied to graph search, including games. In Chapter 5 we present stochastic (probabilistic) techniques for building and organizing search spaces; these will be used later in areas including machine learning and natural language process- ing. Finally, Chapter 6 introduces the production system, blackboards and other software architectures that integrate representation and serach, thus supporting the building of intel- ligent problem solvers.

44 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

2 THE PREDICATE CALCULUS

*We come to the full possession of our power of drawing inferences, the last of our faculties; for it is not so much a natural gift as a long and difficult art.*

—C. S. PIERCE

*The essential quality of a proof is to compel belief*. —FERMAT

**2.0** Introduction

In this chapter we introduce the predicate calculus as a representation language for artificial intelligence. The importance of the predicate calculus was discussed in the introduction to Part II; its advantages include a well-defined *formal semantics* and *sound* and *complete* inference rules. This chapter begins with a brief (optional) review of the propositional calculus (Section 2.1). Section 2.2 defines the syntax and semantics of the predicate calculus. In Section 2.3 we discuss predicate calculus inference rules and their use in problem solving. Finally, the chapter demonstrates the use of the predicate calculus to implement a knowledge base for financial investment advice.

**2.1** The Propositional Calculus (optional)

**2.1.1** Symbols and Sentences

The propositional calculus and, in the next subsection, the predicate calculus are first of all languages. Using their words, phrases, and sentences, we can represent and reason about properties and relationships in the world. The first step in describing a language is to introduce the pieces that make it up: its set of symbols.

45

D E F I N I T I O N

PROPOSITIONAL CALCULUS SYMBOLS

The *symbols* of propositional calculus are the propositional symbols:

P, Q, R, S, ...

truth symbols:

true, false

and connectives:

∧, ∨, ¬, →, ≡

Propositional symbols denote *propositions*, or statements about the world that may be either true or false, such as “the car is red” or “water is wet.” Propositions are denoted by uppercase letters near the end of the English alphabet. Sentences in the propositional cal- culus are formed from these atomic symbols according to the following rules:

D E F I N I T I O N

PROPOSITIONAL CALCULUS SENTENCES

Every propositional symbol and truth symbol is a sentence.

For example: true, P, Q, and R are sentences.

The *negation* of a sentence is a sentence.

For example: ¬ P and ¬ false are sentences.

The *conjunction*, or *and*, of two sentences is a sentence.

For example: P ∧ ¬ P is a sentence.

The *disjunction*, or *or*, of two sentences is a sentence.

For example: P ∨ ¬ P is a sentence.

The *implication* of one sentence from another is a sentence.

For example: P → Q is a sentence.

The *equivalence* of two sentences is a sentence.

For example: P ∨ Q ≡ R is a sentence.

Legal sentences are also called *well-formed formulas* or *WFFs*.

In expressions of the form P ∧ Q, P and Q are called the *conjuncts*. In P ∨ Q, P and Q are referred to as *disjuncts*. In an implication, P → Q, P is the *premise* or *antecedent* and Q, the *conclusion* or *consequent*.

46 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

In propositional calculus sentences, the symbols ( ) and [ ] are used to group symbols into subexpressions and so to control their order of evaluation and meaning. For example, (P ∨ Q) ≡ R is quite different from P ∨ (Q ≡ R), as can be demonstrated using truth tables as we see Section 2.1.2.

An expression is a sentence, or well-formed formula, of the propositional calculus if and only if it can be formed of legal symbols through some sequence of these rules. For example,

((P ∧ Q) → R) ≡ ¬ P ∨ ¬ Q ∨ R

is a well-formed sentence in the propositional calculus because:

P, Q, and R are propositions and thus sentences.

P ∧ Q, the conjunction of two sentences, is a sentence.

(P ∧ Q) → R, the implication of a sentence for another, is a sentence.

¬ P and ¬ Q, the negations of sentences, are sentences.

¬ P ∨ ¬ Q, the disjunction of two sentences, is a sentence.

¬ P ∨ ¬ Q ∨ R, the disjunction of two sentences, is a sentence.

((P ∧ Q) → R) ≡ ¬ P ∨ ¬ Q ∨ R, the equivalence of two sentences, is a sentence.

This is our original sentence, which has been constructed through a series of applications of legal rules and is therefore “well formed”.

**2.1.2** The Semantics of the Propositional Calculus

Section 2.1.1 presented the syntax of the propositional calculus by defining a set of rules for producing legal sentences. In this section we formally define the *semantics* or “meaning” of these sentences. Because AI programs must reason with their representational structures, it is important to demonstrate that the truth of their conclusions depends only on the truth of their initial knowledge or premises, i.e., that log- ical errors are not introduced by the inference procedures. A precise treatment of seman- tics is essential to this goal.

A proposition symbol corresponds to a statement about the world. For example, P may denote the statement “it is raining” or Q, the statement “I live in a brown house.” A proposition must be either true or false, given some state of the world. The truth value assignment to propositional sentences is called an *interpretation*, an assertion about their truth in some *possible world*.

Formally, an interpretation is a mapping from the propositional symbols into the set {T, F}. As mentioned in the previous section, the symbols true and false are part of the set of well-formed sentences of the propositional calculus; i.e., they are distinct from the truth value assigned to a sentence. To enforce this distinction, the symbols T and F are used for truth value assignment.

CHAPTER 2 / THE PREDICATE CALCULUS 47

Each possible mapping of truth values onto propositions corresponds to a possible world of interpretation. For example, if P denotes the proposition “it is raining” and Q denotes “I am at work,” then the set of propositions {P, Q} has four different functional mappings into the truth values {T, F}. These mappings correspond to four different inter- pretations. The semantics of propositional calculus, like its syntax, is defined inductively:

D E F I N I T I O N

PROPOSITIONAL CALCULUS SEMANTICS

An *interpretation* of a set of propositions is the assignment of a truth value, either T or F, to each propositional symbol.

The symbol true is always assigned T, and the symbol false is assigned F.

The interpretation or truth value for sentences is determined by:

The truth assignment of *negation*, ¬ P, where P is any propositional symbol, is F if the assignment to P is T, and T if the assignment to P is F.

The truth assignment of *conjunction*, ∧, is T only when both conjuncts have truth value T; otherwise it is F.

The truth assignment of *disjunction*, ∨, is F only when both disjuncts have truth value F; otherwise it is T.

The truth assignment of *implication*, →, is F only when the premise or symbol before the implication is T and the truth value of the consequent or symbol after the implication is F; otherwise it is T.

The truth assignment of *equivalence*, ≡, is T only when both expressions have the same truth assignment for all possible interpretations; otherwise it is F.

The truth assignments of compound propositions are often described by *truth tables*. A truth table lists all possible truth value assignments to the atomic propositions of an expression and gives the truth value of the expression for each assignment. Thus, a truth table enumerates all possible worlds of interpretation that may be given to an expression. For example, the truth table for P ∧ Q, Figure 2.1, lists truth values for each possible truth assignment of the operands. P ∧ Q is true only when P and Q are both T. Or (∨), not (¬), implies (→), and equivalence (≡) are defined in a similar fashion. The construction of these truth tables is left as an exercise.

Two expressions in the propositional calculus are equivalent if they have the same value under all truth value assignments. This equivalence may be demonstrated using truth tables. For example, a proof of the equivalence of P → Q and ¬ P ∨ Q is given by the truth table of Figure 2.2.

By demonstrating that two different sentences in the propositional calculus have iden- tical truth tables, we can prove the following equivalences. For propositional expressions P, Q, and R:

48 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

¬ (¬ P) ≡ P

(P ∨ Q) ≡ (¬ P → Q)

the contrapositive law: (P → Q) ≡ (¬ Q → ¬ P)

de Morgan’s law: ¬ (P ∨ Q) ≡ (¬ P ∧ ¬ Q) and ¬ (P ∧ Q) ≡ (¬ P ∨¬ Q)

the commutative laws: (P ∧ Q) ≡ (Q ∧ P) and (P ∨ Q) ≡ (Q ∨ P)

the associative law: ((P ∧ Q) ∧ R) ≡ (P ∧ (Q ∧ R))

the associative law: ((P ∨ Q) ∨ R) ≡ (P ∨ (Q ∨ R))

the distributive law: P ∨ (Q ∧ R) ≡ (P ∨ Q) ∧ (P ∨ R)

the distributive law: P ∧ (Q ∨ R) ≡ (P ∧ Q) ∨ (P ∧ R)

Identities such as these can be used to change propositional calculus expressions into a syntactically different but logically equivalent form. These identities may be used instead of truth tables to prove that two expressions are equivalent: find a series of identi- ties that transform one expression into the other. An early AI program, the *Logic Theorist* (Newell and Simon 1956), designed by Newell, Simon, and Shaw, used transformations between equivalent forms of expressions to prove many of the theorems in Whitehead and Russell’s *Principia Mathematica* (1950). The ability to change a logical expression into a different form with equivalent truth values is also important when using inference rules (modus ponens, Section 2.3, and resolution, Chapter 14) that require expressions to be in a specific form.

P Q P Q>

T T T

T F F

F T F

F F F

Figure 2.1 Truth table for the operator ∧.

P Q ¬P ¬P >

Q P ⇒ Q (¬P > Q)=(P ⇒ Q) T T F T T T

T F F F F T

F T T T T T

F F T T T T

Figure 2.2 Truth table demonstrating the equivalence of

P → Q and ¬ P ∨ Q.

CHAPTER 2 / THE PREDICATE CALCULUS 49

**2.2** The Predicate Calculus

In propositional calculus, each atomic symbol (P, Q, etc.) denotes a single proposition. There is no way to access the components of an individual assertion. Predicate calculus provides this ability. For example, instead of letting a single propositional symbol, P, denote the entire sentence “it rained on Tuesday,” we can create a predicate weather that describes a relationship between a date and the weather: weather(tuesday, rain). Through inference rules we can manipulate predicate calculus expressions, accessing their individual components and inferring new sentences.

Predicate calculus also allows expressions to contain variables. Variables let us create general assertions about classes of entities. For example, we could state that for all values of X, where X is a day of the week, the statement weather(X, rain) is true; i.e., it rains every day. As we did with the propositional calculus, we will first define the syntax of the language and then discuss its semantics.

**2.2.1** The Syntax of Predicates and Sentences

Before defining the syntax of correct expressions in the predicate calculus, we define an alphabet and grammar for creating the *symbols* of the language. This corresponds to the lexical aspect of a programming language definition. Predicate calculus symbols, like the *tokens* in a programming language, are irreducible syntactic elements: they cannot be broken into their component parts by the operations of the language.

In our presentation we represent predicate calculus symbols as strings of letters and digits beginning with a letter. Blanks and nonalphanumeric characters cannot appear within the string, although the underscore, \_, may be used to improve readability.

D E F I N I T I O N

PREDICATE CALCULUS SYMBOLS

The alphabet that makes up the symbols of the predicate calculus consists of:

1. The set of letters, both upper- and lowercase, of the English alphabet.

2. The set of digits, 0, 1, ..., 9.

3. The underscore, \_.

*Symbols* in the predicate calculus begin with a letter and are followed by any sequence of these legal characters.

Legitimate characters in the alphabet of predicate calculus symbols include

a R 6 9 p \_ z

Examples of characters not in the alphabet include

50 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

# % @ / &

Legitimate predicate calculus symbols include

George fire3 tom\_and\_jerry bill XXXX friends\_of

Examples of strings that are not legal symbols are

3jack no blanks allowed ab%cd \*\*\*71 duck!!!

Symbols, as we see in Section 2.2.2, are used to denote objects, properties, or relations in a world of discourse. As with most programming languages, the use of “words” that suggest the symbol’s intended meaning assists us in understanding program code. Thus, even though l(g,k) and likes(george, kate) are formally equivalent (i.e., they have the same structure), the second can be of great help (for human readers) in indicating what relationship the expression represents. It must be stressed that these descriptive names are intended solely to improve the readability of expressions. The only “meaning” that predicate calculus expressions have is given through their formal semantics.

Parentheses “( )”, commas “,”, and periods “.” are used solely to construct well- formed expressions and do not denote objects or relations in the world. These are called *improper symbols*.

Predicate calculus symbols may represent either *variables, constants, functions*, or *predicates*. Constants name specific objects or properties in the world. Constant symbols must begin with a lowercase letter. Thus george, tree, tall, and blue are examples of well-formed constant symbols. The constants true and false are reserved as *truth symbols*. Variable symbols are used to designate general classes of objects or properties in the world. Variables are represented by symbols beginning with an uppercase letter. Thus George, BILL, and KAte are legal variables, whereas geORGE and bill are not.

Predicate calculus also allows functions on objects in the world of discourse. Func- tion symbols (like constants) begin with a lowercase letter. Functions denote a mapping of one or more elements in a set (called the *domain* of the function) into a unique element of a second set (the *range* of the function). Elements of the domain and range are objects in the world of discourse. In addition to common arithmetic functions such as addition and multiplication, functions may define mappings between nonnumeric domains.

Note that our definition of predicate calculus symbols does not include numbers or arithmetic operators. The number system is not included in the predicate calculus primi- tives; instead it is defined axiomatically using “pure” predicate calculus as a basis (Manna and Waldinger 1985). While the particulars of this derivation are of theoretical interest, they are less important to the use of predicate calculus as an AI representation language. For convenience, we assume this derivation and include arithmetic in the language.

Every function symbol has an associated *arity*, indicating the number of elements in the domain mapped onto each element of the range. Thus father could denote a function of arity 1 that maps people onto their (unique) male parent. plus could be a function of arity 2 that maps two numbers onto their arithmetic sum.

A *function expression* is a function symbol followed by its arguments. The arguments are elements from the domain of the function; the number of arguments is equal to the

CHAPTER 2 / THE PREDICATE CALCULUS 51

arity of the function. The arguments are enclosed in parentheses and separated by com- mas. For example,

f(X,Y) father(david) price(bananas)

are all well-formed function expressions.

Each function expression denotes the mapping of the arguments onto a single object in the range, called the *value* of the function. For example, if father is a unary function, then

father(david)

is a function expression whose value (in the author’s world of discourse) is george. If plus is a function of arity 2, with domain the integers, then

plus(2,3)

is a function expression whose value is the integer 5. The act of replacing a function with its value is called *evaluation*.

The concept of a predicate calculus symbol or term is formalized in the following def- inition:

D E F I N I T I O N

SYMBOLS and TERMS

Predicate calculus symbols include:

1. *Truth symbols* true and false (these are reserved symbols).

2. *Constant symbols* are symbol expressions having the first character lowercase.

3. *Variable symbols* are symbol expressions beginning with an uppercase character.

4. *Function symbols* are symbol expressions having the first character lowercase. Functions have an attached arity indicating the number of elements of the domain mapped onto each element of the range.

A t1 , *function* t2 , ... ,

tn *expression* , enclosed consists of in parentheses a function constant of arity n, followed by n terms,

and separated by commas.

A predicate calculus *term* is either a constant, variable, or function expression.

Thus, a predicate calculus *term* may be used to denote objects and properties in a problem domain. Examples of terms are:

52 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

cat times(2,3) Xblue mother(sarah) kate

Symbols in predicate calculus may also represent predicates. Predicate symbols, like constants and function names, begin with a lowercase letter. A predicate names a relationship between zero or more objects in the world. The number of objects so related is the arity of the predicate. Examples of predicates are

likes equals on near part\_of

An *atomic sentence*, the most primitive unit of the predicate calculus language, is a predicate of arity n followed by n terms enclosed in parentheses and separated by com- mas. Examples of atomic sentences are

likes(george,kate) likes(X,george) likes(george,susie) likes(X,X) likes(george,sarah,tuesday) friends(bill,richard) friends(bill,george) friends(father\_of(david),father\_of(andrew)) helps(bill,george) helps(richard,bill)

The predicate symbols in these expressions are likes, friends, and helps. A predicate symbol may be used with different numbers of arguments. In this example there are two different likes, one with two and the other with three arguments. When a predicate symbol is used in sentences with different arities, it is considered to represent two different rela- tions. Thus, a predicate relation is defined by its name and its arity. There is no reason that the two different likes cannot make up part of the same description of the world; however, this is usually avoided because it can often cause confusion.

In the predicates above, bill, george, kate, etc., are constant symbols and represent objects in the problem domain. The arguments to a predicate are terms and may also include variables or function expressions. For example,

friends(father\_of(david),father\_of(andrew))

is a predicate describing a relationship between two objects in a domain of discourse. These arguments are represented as function expressions whose mappings (given that the father\_of david is george and the father\_of andrew is allen) form the parameters of the predicate. If the function expressions are evaluated, the expression becomes

friends(george,allen)

These ideas are formalized in the following definition.

CHAPTER 2 / THE PREDICATE CALCULUS 53

D E F I N I T I O N

PREDICATES and ATOMIC SENTENCES

Predicate symbols are symbols beginning with a lowercase letter.

Predicates have an associated positive integer referred to as the *arity* or “argument number” for the predicate. Predicates with the same name but different arities are considered distinct.

An t1 , t2 atomic , ... ,

tn , sentence enclosed is in parentheses a predicate constant of arity n, and separated by commas.

followed by n terms,

The truth values, true and false, are also atomic sentences.

Atomic sentences are also called *atomic expressions*, *atoms*, or *propositions*.

We may combine atomic sentences using logical operators to form *sentences* in the predicate calculus. These are the same logical connectives used in propositional calculus: ∧, ∨, ¬, →, and ≡.

When a variable appears as an argument in a sentence, it refers to unspecified objects in the domain. First order (Section 2.2.2) predicate calculus includes two symbols, the *variable quantifiers* ∀ and ∃, that constrain the meaning of a sentence containing a vari- able. A quantifier is followed by a variable and a sentence, such as

∃ Y friends(Y, peter) ∀ X likes(X, ice\_cream)

The *universal quantifier*, ∀, indicates that the sentence is true for all values of the variable. In the example, ∀ X likes(X, ice\_cream) is true for all values in the domain of the defini- tion of X. The *existential quantifier*, ∃, indicates that the sentence is true for at least one value in the domain. ∃ Y friends(Y, peter) is true if there is at least one object, indicated by Y that is a friend of peter. Quantifiers are discussed in more detail in Section 2.2.2.

Sentences in the predicate calculus are defined inductively.

D E F I N I T I O N

PREDICATE CALCULUS SENTENCES

Every atomic sentence is a sentence.

1. If s is a sentence, then so is its negation, ¬ s.

2. If s1 and s2 are sentences, then so is their conjunction, s1 ∧ s2. 3. If s1 and s2 are sentences, then so is their disjunction, s1 ∨ s2. 4. If s1 and s2 are sentences, then so is their implication, s1 → s2. 5. If s1 and s2 are sentences, then so is their equivalence, s1 ≡ s2.

54 PART II / ARTIFICIAL INTELLIGENCE AS REPRESENTATION AND SEARCH

6. If X is a variable and s a sentence, then ∀ X s is a sentence.

7. If X is a variable and s a sentence, then ∃ X s is a sentence.

Examples of well-formed sentences follow. Let times and plus be function symbols of arity 2 and let equal and foo be predicate symbols with arity 2 and 3, respectively.

plus(two,three) is a function and thus not an atomic sentence.

equal(plus(two,three), five) is an atomic sentence.

equal(plus(2, 3), seven) is an atomic sentence. Note that this sentence, given the standard interpretation of plus and equal, is false. Well-formedness and truth value are independent issues.

∃ X foo(X,two,plus(two,three)) ∧ equal(plus(two,three),five) is a sentence since both conjuncts are sentences.

(foo(two,two,plus(two,three))) → (equal(plus(three,two),five) ≡ true) is a sen- tence because all its components are sentences, appropriately connected by logical operators.

The definition of predicate calculus sentences and the examples just presented suggest a method for verifying that an expression is a sentence. This is written as a recursive algorithm, verify\_sentence. verify\_sentence takes as argument a candidate expression and returns success if the expression is a sentence.

function verify\_sentence(expression); begin

caseexpression is an atomic sentence: return SUCCESS;

expression is of the form Q X s, where Q is either ∀ or ∃, X is a variable,

if verify\_sentence(s) returns SUCCESS then return SUCCESS else return FAIL; expression is of the form ¬ s:

if verify\_sentence(s) returns SUCCESS then return SUCCESS else return FAIL; expression is of the form s1 op s2, where op is a binary logical operator:

if verify\_sentence(s1) returns SUCCESS and

verify\_sentence(s2) returns SUCCESS then return SUCCESS else return FAIL; otherwise: return FAIL end end.

CHAPTER 2 / THE PREDICATE CALCULUS 55