国外计算机科学教材系列

人工智能

——理论与实践

Artificial Intelligence:

Theory and Practice

英文版

Thomas Dean
[美] James Allen 著
Yiannis Aloimonos

PEARSON Addison Wesley
PHEI
電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

国外计算机科学教材系列

人工智能——理论与实践

（英文版）

Artificial Intelligence:
Theory and Practice

Thomas Dean
[美] James Allen 著
Yiannis Aloimonos

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

这是一本阐述人工智能基本理论及其实际应用的教材，由三位资深的人工智能专家精心编著而成。针对机器智能系统开发中涌现出的表达与计算问题，本书介绍了最新的研究成果，并讨论了系统实现中涉及到的实际问题。作者深入探讨了用于解决学习、规划和不确定性问题的传统符号推理技术，例如演绎推理、对策树等，并介绍了神经网络、概率推理等新技术。书中出现的重要算法在每章后面都附有其Lisp实现的源代码，以供读者在实验时进行参考。另外，本书还给出了丰富的人工智能应用系统的实例。

本书可作为高等院校计算机、控制、机电、数学等专业人工智能课程的教材，也可供从事人工智能研究及应用的科学工作者和工程技术人员学习参考。

版权贸易合同登记号：图字：01-2003-0356

图书在版编目（CIP）数据

人工智能——理论与实践 = Artificial Intelligence：Theory and Practice/（美）迪安（Dean, T.）等著.
-北京：电子工业出版社，2003.6
（国外计算机科学教材系列）
ISBN 7-5053-8780-4

Ⅰ.人... Ⅱ.迪... Ⅲ.人工智能-教材-英文 Ⅳ. TP18

中国版本图书馆CIP数据核字（2003）第043200号

责任编辑：冯小贝
印 刷 者：北京东光印刷厂
出版发行：电子工业出版社 http://www.phei.com.cn
北京市海淀区万寿路173信箱 邮编：100036
经 销：各地新华书店
开 本：787 × 980 1/16 印张：37 字数：829千字
版 次：2003年6月第1版 2003年6月第1次印刷
定 价：49.00元

About the Authors

Thomas Dean is a Professor in the Computer Science Department at Brown University. His general research interests include temporal and spatial reasoning, planning, robotics, learning, and probabilistic inference. Professor Dean's recent work has led to the design and implementation of a temporal database system for applications involving mobile robots and factory automation. He is on the executive council and is a Fellow of the American Association for Artificial Intelligence (AAAI). He served as the program co-chair for the 1991 National Conference on Artificial Intelligence. Professor Dean was also a recipient of the NSF Presidential Young Investigator Award (1989–1994).

James Allen is the John H. Dessaurer Professor of Computer Science at the University of Rochester. He is a fellow of the AAAI and was a recipient of the NSF Presidential Young Investigator Award (1985–1989). In addition, Professor Allen was the Editor-in-Chief of *Computational Linguistics* from 1983 to 1993.

Yiannis Aloimonos is an Associate Professor at the Computer Science Department and the Institute for Advanced Computer Studies of the University of Maryland. He also heads the Computer Vision Laboratory of the Center for Automation Research. His research interests include computer vision and the integration of perception, reasoning, and action. He is a recipient of the NSF Presidential Young Investigator Award (1990–1995).

Preface

This book is designed to introduce students to a set of theoretical and computational techniques that serve as a foundation for the study of artificial intelligence (AI). The presentation is aimed at students with a background in computer science at about the sophomore or junior level in college. The emphasis is on algorithms and theoretical machinery for building and analyzing AI systems. Traditional symbolic AI techniques such as deductive inference, game-tree search, and natural language parsing are covered, as are hybrid approaches such as those employed in neural networks, probabilistic inference, and machine vision. The coverage is broad, with selected topics explored in greater depth but with no attempt to exhaustively survey the entire field.

Representation

The book focuses on the importance of representation in the core chapters dealing with logic, search, and learning. It incorporates a more formal treatment of AI than is found in most introductory textbooks. This formal treatment is reflected in the attention given to syntax and semantics in logic and in the material concerning the computational complexity of AI algorithms.

The material on learning draws on recent unifying work in computational learning theory to explain a variety of techniques from decision trees to neural networks.

The book provides a consistent pedagogic example of AI in the real world through examples focusing on AI systems corresponding to robots and software automation "softbots." A wide range of other examples are also introduced to characterize both the potential and the variety of AI applications. The chapters on natural language processing, planning, uncertainty, and vision supply a state-of-the-art perspective unifying existing approaches and summarizing challenging areas for future research.

This book is not meant as an exhaustive survey of AI techniques. Subjects such as qualitative reasoning about physical systems and analogical reasoning are only briefly touched on in this text. Other subjects are given much more attention in this book than in traditional texts. Learning, planning, and probabilistic reasoning are treated in some depth, reflecting their increased importance in the field. The chapter on vision (Chapter 9, Image Understanding) is substantial in its coverage of topics to reflect the importance of perception in understanding intelligence and building artifacts that interact with the world in useful and interesting ways.

Theory and Practice

Although the text emphasizes theoretical foundations, practical problems involved with the implementation of AI algorithms are addressed in every chapter. A self-contained introduction to symbolic programming in Common Lisp is provided to encourage students to perform computational experiments. Lisp code is given for many of the important algorithms described in the text; however, the text is designed so that the student can ignore Lisp and implementation issues altogether if he or she chooses. The code uses a carefully chosen subset of Common Lisp to teach algorithmic issues in AI. In contrast, other texts use AI algorithms to teach Lisp programming techniques.

All the algorithms in the text are described in English prose and pseudo code. In the case of algorithms that are also given in Lisp, most of the time the code appears in a Lisp Implementation appendix at the end of the chapter, but on some occasions it appears in the main body of the chapter. Code appears in the main body when it is considered particularly important that the student explore the underlying issues empirically. With most of the Lisp code relegated to appendices, instructors are free to choose the areas they want to emphasize empirically. The tight coupling between the descriptions of algorithms in the text and the accompanying Lisp code makes it easy for students to experiment, without the bother of using two texts with different perspectives and algorithmic approaches.

We use Lisp instead of Prolog because Lisp is closest in structure to languages such as Pascal and C that students are likely to be familiar with. We use Lisp instead of Pascal or C because the list processing and symbolic manipulation routines available in Lisp allow for elegant implementations of important algorithms that can be compactly listed. Note, however, that a library of C++ code is available (see the section on "Supplements") that mirrors the Common Lisp treatment in the text function for function and algorithm for algorithm.

To the Student

Preface material is usually aimed at instructors who are thinking of adopting a text for a course. Generally, students cut straight to the first chapter or the table of contents to get some idea of what the book is about. This book is designed to teach students about the theory and practice of building computer programs that perform interesting and useful tasks. With the exception of some diversions in the introductory chapter, we leave the philosophical conundrums to the philosophers and focus on techniques, algorithms, and analytical tools that we believe students will find useful in building sophisticated (even *intelligent*) computer programs.

The book describes precise problems, analyzes them from a computational perspective, and specifies efficient algorithms for their solution where possible. Along the way, we provide the necessary logic, computer science, and mathematics for you to understand the important issues and ultimately develop your own solutions and propose your own problems. Our hope is that you will find the techniques and ideas in this book useful, whether you pursue a career in engineering, computer science, business management, or any other area that requires you to think in terms of computational processes that have to interact with a complex and changing world.

To the Instructor

The core material in the book is in the first five chapters covering basic introductory and motivational material, symbolic programming for courses interested in implementation, representation and logic, search, and learning. Within these core chapters, instructors have considerable flexibility regarding what to include and how much time to spend on particular topics.

The choice of topics and allocation of lecture time will depend on the background of the students taking the course. Often students have a reasonable background in boolean logic from a previous course in computer science, engineering, or mathematics, in which case the chapter on representation and logic can move along rather quickly. Search issues are generally familiar to computer science students, and the basic blind-search methods, including depth-first and breadth-first search, should take very little time.

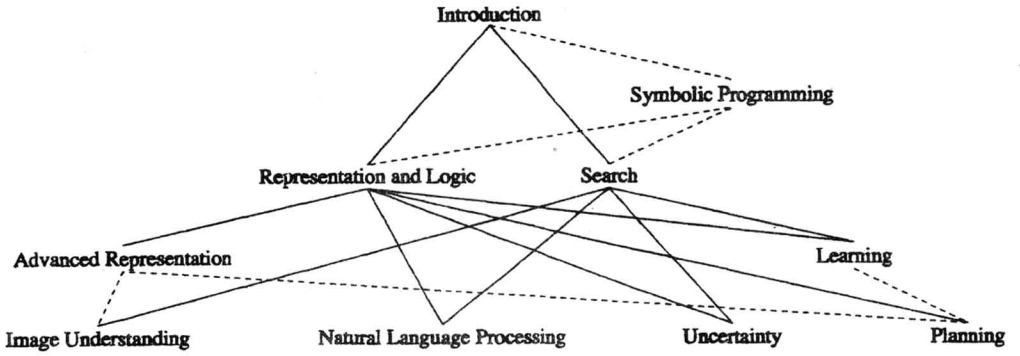

Figure 1 This graph illustrates some of the dependencies and connections among the chapters in this text. A solid line indicates a strong dependency between two chapters, and a dashed line indicates a connection or conditional dependency between two chapters that an instructor may wish to consider.

We recommend spending a significant amount of time on learning since the area is reasonably mature, the issues dramatically illustrate the role of representation and search, and students are generally fascinated with the prospect of building systems that learn.

Representation appears before search in the order of chapters because representation is the more fundamental idea as far as AI is concerned. We emphasize logic because it enables students to think precisely about representation. Pedagogically, there are situations that merit covering search before representation; if you are teaching Lisp and this is the students' first exposure to Lisp and symbolic programming, consider covering the search chapter first because the examples of search procedures provide a somewhat easier introduction to Lisp programming issues. With the exception of the section on discrimination networks at the end of the search chapter, the representation and search chapters can be covered in either order.

Figure 1 illustrates some of the dependencies and connections among the chapters in this text. A solid line indicates that one chapter should be covered before another chapter in order to fully understand all the material. A dashed line indicates a connection between two chapters that an instructor may wish to emphasize or a conditional dependency that an instructor may wish to account for. For example, the section on spatial representation and robot navigation in Chapter 6 (Advanced Representation) can be used to motivate and set the stage for topics covered in Chapter 9 (Image Understanding). All the chapters are conditionally dependent on Chapter 2 (Symbolic Programming) if implementation issues are to be covered and the students require instruction in symbolic programming methods. Additional information regarding chapter dependencies and synergies as well as suggestions for course syllabi are available in the Instructors Resource Guide (see the section on "Supplements").

Supplements

Source supplemental materials for this book are available via anonymous FTP from `aw.com` in the subdirectory `aw/dean`. The supplemental materials for the book include the following items:

- Instructor's Guide and Solutions Manual—contain notes on each chapter, solutions to selected exercises, additional exercises for the Lisp (Chapter 2) and vision (Chapter 9) chapters, and sample exams with answers. This guide is available only on a $3\frac{1}{2}$" disk from the publisher (ISBN 32548-4).
- Selected figures—selected figures in encapsulated PostScript format are available for overhead transparencies.
- Source code—the sample source code contained in the text is available in both Lisp and C++ implementations. Other implementations may be available (for example, the Scheme dialect of Lisp); check the `README` file in `bc/dean` for current status and recent developments.

To obtain the supplemental materials, FTP to bc.aw.com as follows:

```
% ftp aw.com
```

and log on as `anonymous`. Use your electronic mail address as your password and connect to the directory for this book by typing

```
% cd aw/dean
```

Before retrieving supplements, it is a good idea to look at the `README` file to see if changes have been made since this book went to press. You can retrieve this file by typing

```
% get README
```

Type `quit` to exit FTP and read the `README` file. (Although you could read the file online, it is courteous not to load the FTP server while you are just reading.) Then log back on when you are ready to download the files that you want. Using FTP to retrieve archived files can get complicated. The `README` file will give you some additional advice, but you may find it helpful to consult your favorite UNIX guide or local artwork wizard.

Thanks

We benefited from the efforts of many previous authors in organizing the material covered in this text and figuring out how to present the material to students. In particular, we would like to acknowledge the texts by Charniak and McDermott [1985], Nilsson [1980], and Winston [1979] that provided our first introductions to AI. We are also indebted to Davis [1990], Genesereth and Nilsson [1987], Ginsberg [1993], and Winston [1992], whose books we consulted while writing this text.

A lot of friends and colleagues contributed to this book by providing feedback to the authors during crucial times during its writing. In particular, we would like to thank the following people:

Mark Boddy	Robert Goldman	Bart Selman
Chris Brown	Steve Hanks	Jude Shavlik
Jack Breese	Eric Horvitz	Yoav Shoham
Eugene Charniak	Leslie Kaelbling	Mike Shwe
Ernie Davis	Paris Kanellakis	Austin Tate
Mark Drummond	Richard Korf	Prasad Tadepalli
Charles Dyer	Tom Mitchell	Dan Weld
Nort Fowler	Ann Nicholson	Mike Wellman

The production and editorial help from Addison-Wesley Publishing Company was exceptional. We would like to thank Carter Shanklin, the Acquisitions Editor for Computer Science at Addison-Wesley, for shepherding us through the entire process. Judith Hibbard, the Senior Production Editor, Melissa Standen, the Editorial Assistant, and Ari Davidow, the Technical Production Assistant, were amazingly helpful at all the right times. We would also like to thank Sara Larson from the University of Maryland at College Park, who helped with the vision chapter, and Mary Andrade and Dawn Nichols from Brown University, who provided invaluable assistance in keeping track of the many drafts and handling the extensive correspondence that was required.

A large number of students also provided feedback while we were writing the text, far too many to list, but the following deserve special mention:

Scott Baetz	Michael Littman
Ted Camus	Mike Perkowitz
Lloyd Greenwald	Kostadis Roussos
Shieu-Hong Lin	Smith Surasmith

A special thanks goes to Jon Monsarrat for his unflagging enthusiasm for the project and for the many hours he spent hacking UNIX, PostScript, and LaTeX. Finally, each of us would like to thank our respective spouses and loved ones for being supportive when it was needed most.

Thomas Dean
James Allen
Yiannis Aloimonos

目录概览

Contents