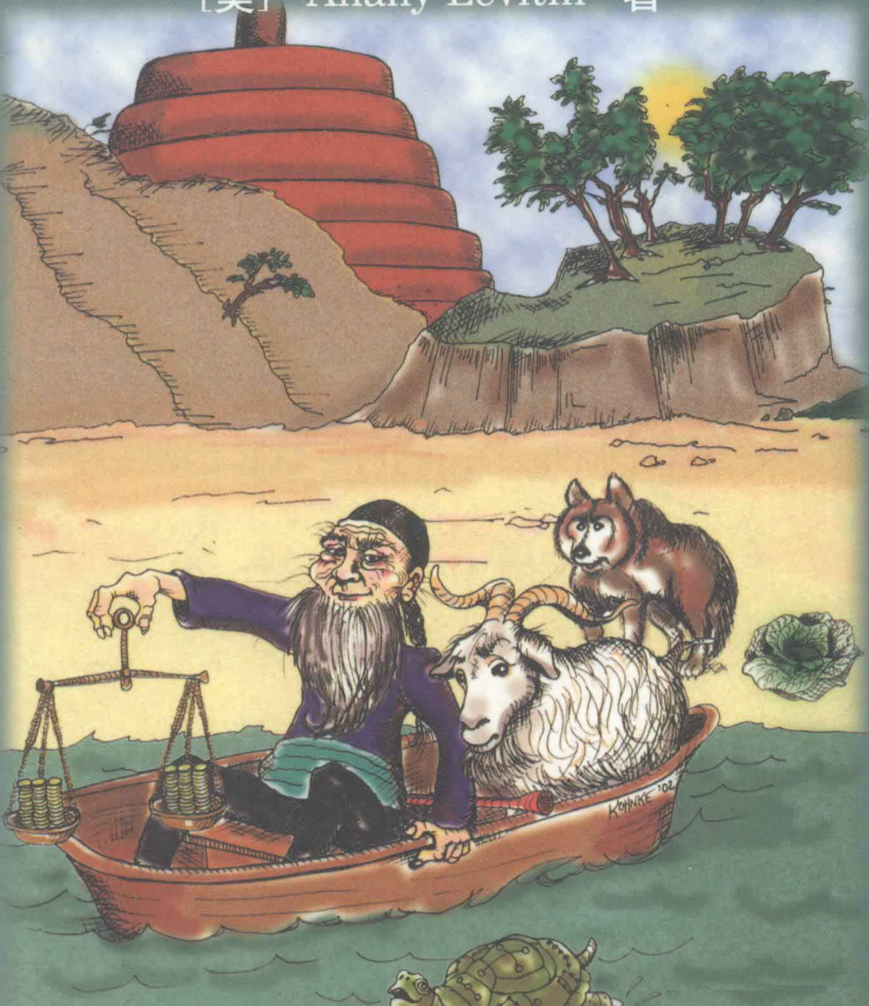
算法经典丛书

PEARSON
Addison
Wesley

算法设计与分析基础（影印版）

Introduction to The Design & Analysis of Algorithms

［美］ Anany Levitin 著



清华大学出版社

算法经典丛书

算法设计与分析基础

（影印版）

［美］ Anany Levitin　著

清华大学出版社

北　京

图书在版编目（CIP）数据

算法设计与分析基础 = Introduction to The Design & Analysis of Algorithms / [美]乐威汀著.—影印版. 北京　清华大学出版社，2003
（算法经典丛书）
ISBN 7-302-06796-1
Ⅰ. 算… Ⅱ.乐… Ⅲ.①算法设计—英文②算法分析—英文　Ⅳ. TP301.6
中国版本图书馆 CIP 数据核字（2003）第 054511 号

出 版 者：清华大学出版社　　**地　　址：**北京清华大学学研大厦
http://www.tup.com cn　　**邮　　编：**100084
社 总 机：010-62770175　　**客户服务：**010-62776969

文稿编辑：葛昊晗
封面设计：立日新设计公司
印 刷 者：世界知识印刷厂
发 行 者：新华书店总店北京发行所
开　　本：148×210　**印张：**16.5
版　　次：2003 年 8 月第 1 版　2003 年 8 月第 1 次印刷
书　　号：ISBN 7-302-06796-1/TP・5057
印　　数：1～3000
定　　价：39.00 元

推荐书目

欢迎选购清华大学出版社出版的外版计算机图书！如需购买以下图书，请与清华大学出版社发行部联系，联系电话为 010-62786544 或 62776969，通信地址为:

北京清华大学学研大厦 (邮编 100084)

以下图书的信息与服务网站，请访问：**http://www.ePress.cn** 和 **http://www.34.cn**，亦可通过电子信箱 **book@21bj.com** 咨询。

1. SEI软件工程丛书

《软件过程管理》(中文/影印版)	Watts S. Humphrey 著，定价: 59 元/59 元
《技术人员管理》(中文/影印版)	Watts S. Humphrey 著，定价: 49 元/49 元
《软件制胜之道》(中文/影印版)	Watts S. Humphrey 著，定价: 29 元/39 元
《软件构架评估》(中文/影印版)	Paul Clements 等著，定价: 39 元/49 元
《软件采办管理》(中文/影印版)	B. Craig Meyers 等著，定价: 49 元/59 元
《风险管理》(中文/影印版)	Elaine M. Hall 著，定价: 49 元/49 元
《用商业组件构建系统》(中文/影印版)	Kurt Wallnau 等著，定价: 49 元/59 元
《系统工程》(中文版)	George A. Hazelrigg 著，定价: 59 元
《项目管理原理》(中文版)	Adedeji B. Badiru 等著，定价: 59 元
《软件构架实践》(中文版)	Len Bass 等著，定价: 69 元
	本书荣获美国《软件开发》杂志第 9 届图书大奖
《CERT 安全指南》(中文版)	Julia H. Allen 著，定价: 49 元
《团队软件过程》(影印版)	Watts S. Humphrey 著，定价: 49 元
《CMMI 集成过程改进》(影印版)	Dennis M. Ahern 等著，定价: 39 元
《基于构架的软件项目管理》(影印版)	Daniel J. Paulish 著，定价: 39 元

2. 软件工程实践丛书

《面向对象 Java 编程思想》	Timothy Budd 著，定价: 45 元
《用户界面设计与开发精解》	R. J. Torres 著，定价: 49 元
《需求分析》(影印版)	David C. Hay 等著，定价: 39 元
《自动化软件测试》(影印版)	Elfriede Dustin 等著，定价: 39 元
《软件项目管理实践》	Pankaj Jalote 著，定价: 36 元

3. 其他软件工程类图书

《有效用例模式》（影印版）	Paul Bramble 等著，定价: 28 元
《有效用例模式》（中文版）	Paul Bramble 等著，即将出版
《用例建模》（影印版）	Kurt Bittner 等著，定价: 28 元
《用例建模》(中文版)	Kurt Bittner 等著，即将出版

4. Java技术丛书

《分布式 Java 2 数据库系统开发指南》	Stewart Birnam 著，定价: 36 元
《JAVA 开发人员年鉴》（影印版，第 1 卷）	Patrick Chan 等，定价: 68 元
《JAVA 开发人员年鉴》（影印版，第 2 卷）	Patrick Chan 等，定价: 68 元

5. 其他系统与编程类图书

《MySQL 核心编程》	Leon Atkinson 著，定价: 69 元
《Windows .NET Server 安全指南》	Cyrus Peikari 著，定价: 39 元

6. Deitel编程金典

《C++编程金典》（第 3 版）	Deitel 等著，定价: 118 元
《Perl 编程金典》	Deitel 等著，定价: 128 元
《C#高级程序员指南》	Deitel 等著，定价: 118 元
《VB .NET 高级程序员指南》	Deitel 等著，即将出版
《Python 编程金典》	Deitel 等著，即将出版

7. 算法经典丛书

《Java 算法》（第 3 版，第 1 卷，影印版）	Robert Sedgewick 著，定价: 68 元
《Java 算法》（第 3 版，第 1 卷）	Robert Sedgewick 著，即将出版
《C++算法》（第 3 版，第 2 卷）	Robert Sedgewick 著，即将出版

8. 国外经典教材

《电子商务网站开发指南》	Vivek Sharma 等著，定价: 59 元
《数据库系统原理》	Greg Riccardi 著，定价: 55 元
《电气工程学概论》（第 2 版）	John R. Cogdell 著，定价: 88 元
《网络管理》	Mani Subramanian 著，定价: 69 元
《微波晶体管放大器分析与设计》(第 2 版)	Guillermo Gonzalez 著，定价: 59 元
《C#解析教程》	Ira Pohl 著，定价: 39 元

Introduction to The Design & Analysis of Algorithms

Introduction to The Design & Analysis of Algorithms

ANANY LEVITIN
Villanova University

Addison Wesley

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

Senior Acquisitions Editor	*Maite Suarez-Rivas*
Executive Editor	*Susan Hartman Sullivan*
Executive Marketing Manager	*Michael Hirsch*
Production Supervisor	*Marilyn Lloyd*
Project Manager	*Sandra Rigney*
Composition	*Windfall Software, using ZzTEX*
Technical Art	*George Nichols*
Copyeditor	*Penelope D. Hull*
Proofreader	*Holly McLean-Aldis*
Text Design	*Sandra Rigney*
Cover Design	*Gina Hagen Kolenda*
Cover Illustration	*Jennifer M. Kohnke*
Senior Manufacturing Buyer	*Evelyn Beaton*

Access the latest information about Addison-Wesley titles from our World Wide Web site:
www.aw.com/cs

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.

The programs and applications presented in this book have been included for their instructional value. They have been tested with care but are not guaranteed for any purpose. The publisher does not offer any warranties or representations, nor does it accept any liabilities with respect to the programs or applications.

Library of Congress Cataloging-in-Publication Data

Levitin, Anany.
Introduction to the design and analysis of algorithms / Anany Levitin.
p. cm.
Includes bibliographic references and index.
ISBN 0-201-74395-7
1. Computer algorithms. I. Title.
QA76.9.A43 L48 2003
005.1—dc21 2002027987

ISBN 0-201-74395-7
1 2 3 4 5 6 7 8 9 10—HAM—06 05 04 03 02

To Maria and Miriam,
with my deepest gratitude

Preface

The most valuable acquisitions in a scientific or technical education are the general-purpose mental tools which remain serviceable for a life-time.

—George Forsythe, "What to do till the computer scientist comes" (1968)

Algorithms play the central role in both the science and the practice of computing. Recognition of this fact has led to the appearance of a considerable number of textbooks on the subject. By and large, they follow one of two alternatives in presenting algorithms. One classifies algorithms according to a problem type. Such a book would have separate chapters on algorithms for sorting, searching, graphs, and so on. The advantage of this approach is that it allows an immediate comparison of, say, the efficiency of different algorithms for the same problem. The drawback of this approach is that it emphasizes problem types at the expense of algorithm design techniques.

The second alternative organizes the presentation around algorithm design techniques. In this organization, algorithms from different areas of computing are grouped together if they have the same design approach. I share the belief of many (e.g., [BaY95]) that this organization is more appropriate for the basic course on the design and analysis of algorithms. There are three principal reasons for emphasis on algorithm design techniques. First, these techniques provide a student with tools for designing algorithms for new problems. This makes learning algorithm design technique a very valuable endeavor from the practical standpoint. Second, they seek to classify multitudes of known algorithms according to an underlying design idea. Learning to see such commonality among algorithms from different application areas should be a major goal of computer science education. After all, every science considers classification of its principal subject as a major if not the central point of its discipline. Third, in my opinion, algorithm design techniques have considerable utility as general problem-solving strategies, applicable to problems beyond computing.

Several textbooks exist that are organized around algorithm design techniques (see [BB96], [HSR98], [NN98]). The problem with these books is that they uncritically follow the same classification of design techniques. This classification has several serious shortcomings from both theoretical and educational points of view. The most significant of the shortcomings is failure to classify many important algorithms. This limitation has forced the authors of existing textbooks to depart from the design technique organization and to include chapters dealing with specific problem types. Unfortunately, such a switch leads to a loss of course coherence and almost unavoidably creates confusion in students' minds.

New Taxonomy of Algorithm Design Techniques

My frustration with the shortcomings of the existing classification of algorithm design techniques has motivated me to develop a new taxonomy of techniques [Lev99], which is the basis of this book. Here are the principal advantages of the new taxonomy:

- The new taxonomy is more comprehensive than the traditional one. It includes several strategies—brute force, decrease-and-conquer, transform-and-conquer, and time and space tradeoffs—that are rarely if ever recognized as important design paradigms.
- The new taxonomy covers naturally many classic algorithms (Euclid's algorithm, heapsort, search trees, hashing, topological sorting, Gaussian elimination, Horner's rule, to name a few) that the traditional taxonomy cannot classify. As a result, it makes it possible to present the standard body of classic algorithms in a unified and coherent fashion.
- It naturally accommodates the existence of important varieties of several design techniques. (For example, it recognizes three variations of decrease-and-conquer and three variations of transform-and-conquer.)
- It is better aligned with analytical methods for the efficiency analysis (see Appendix B).

Design Techniques as General Problem-Solving Strategies

Most applications of the design techniques in the book are to classic problems of computer science. The only innovation here is the inclusion of some material on numerical algorithms, which are covered within the same general framework. (An inclusion of numerical algorithms is encouraged by *Computing Curricula 2001* [CC01]—a new model curriculum for computer science programs.) But the design techniques can be considered general problem-solving tools, whose applications are not limited to traditional computing and mathematical problems. Two factors make this point particularly important. First, more and more computing applications go beyond the traditional domain, and there are reasons to believe that this trend will strengthen in the future. Second, developing students' problem-solving skills has come to be recognized as a major goal of college education. Among all the courses in a computer science curriculum, a course on the design and analysis of algorithms is uniquely suitable for this task because it can offer a student specific strategies for solving problems. I am not proposing that a course on the design and analysis of algorithms should become a course on general problem solving. But I do believe that the unique opportunity provided by studying the design and analysis of algorithms should not be missed. Toward this goal, the book includes applications to puzzles and puzzle-like games. Although using puzzles in teaching algorithms is certainly not a new idea, the book tries to do so systematically by going beyond a few standard examples.

Textbook Pedagogy

My goal was to write a text that would not trivialize the subject but still would be readable by most students on their own. Here are some of the things done in the book toward this objective.

- Sharing the opinion of George Forsythe (see the epigraph), I have sought to stress major ideas underlying the design and analysis of algorithms. In choosing specific algorithms to illustrate these ideas, I limited the number of covered algorithms to those that most clearly demonstrate an underlying design technique or analysis method. Fortunately, most classic algorithms satisfy this criterion.
- In Chapter 2, which is devoted to the efficiency analysis, the methods used for analyzing nonrecursive algorithms are separated from those typically used for analyzing recursive algorithms. The chapter also includes sections devoted to empirical analysis and algorithm visualization.
- The narrative is systematically interrupted by questions to the reader. Some of them are asked rhetorically, in anticipation of a concern or doubt, and are answered immediately. The goal of the others is to prevent the reader from drifting through the text without a satisfactory level of comprehension.
- Each chapter ends with a summary recapping the most important concepts and results discussed in the chapter.
- The book contains about 600 exercises. Some of them are drills; others make important points about the material covered in the body of the text or introduce algorithms not covered there at all. A few exercises take advantage of Internet resources. Several exercises are designed to prepare the reader for material covered later in the book. More difficult problems—there are not many of them—are marked with a special symbol in the Instructor's Manual. (Because designating problems as difficult may discourage some students from trying to tackle them, problems are not marked in the book itself.)

Puzzles, games, and puzzle-like questions are marked in the exercises in the textbook with a special icon.

- The book provides hints to all the exercises. Detailed solutions, except for programming projects, are provided in the Instructor's Manual available to qualified adopters from the publisher. (Contact your Addison-Wesley representative, or email aw.cse@aw.com.) The supplements available to all readers of this book are at www.aw.com/cssupport.

Prerequisites

The book assumes that a reader has gone through an introductory programming course and a standard course on discrete structures. With such a background, he or she should be able to handle the book's material without undue difficulty. Still, fundamental data structures, necessary summation formulas, and recurrence relations are reviewed in Section 1.4, Appendix A, and Appendix B, respectively. Calculus is used in only three sections (Sections 2.2, 10.4, and 11.4) and to a very limited degree; if students lack calculus as an assured part of their background, the portions of these three sections that involve calculus can be omitted without hindering their understanding of the rest of the material.

Use in the Curriculum

The book can serve as a textbook for the basic course on design and analysis of algorithms that is organized around algorithm design techniques. It might contain too much material for a typical one-semester course. By and large, portions of Chapters 3 through 11 can be skipped without danger of making later parts of the book incomprehensible to the reader. Any portion of the book can be assigned for self-study. In particular, Sections 2.6 and 2.7 on empirical analysis and algorithm visualization, respectively, can be assigned in conjunction with projects.

Here is a possible plan for a one-semester course; it assumes a 40-class meeting format.

Lecture	Topic	Sections
1, 2	Introduction	1.1–1.3
3, 4	Analysis framework; O, Θ, Ω notations	2.1, 2.2
5	Mathematical analysis of nonrecursive algorithms	2.3
6, 7	Mathematical analysis of recursive algorithms	2.4, 2.5 (+ App. B)
8	Brute-force algorithms	3.1, 3.2 (+ 3.3)
9	Exhaustive search	3.4
10–12	Divide-and-conquer: mergesort, quicksort, binary search	4.1–4.3
13	Other divide-and-conquer examples	4.4 or 4.5 or 4.6

14–16	Decrease-by-one: insertion sort, DFS & BFS, topological sorting	5.1–5.3
17	Decrease-by-a-constant-factor algorithms	5.5
18	Variable-size-decrease algorithms	5.6
19–21	Instance simplification, presorting, Gaussian elimination, balanced search trees	6.1–6.3
22	Representation change: heaps and heapsort	6.4
23	Representation change: Horner's rule and binary exponentiation	6.5
24	Problem reduction	6.6
25–27	Space-time tradeoffs: string matching, hashing, B-trees	7.2–7.4
28–30	Dynamic programming algorithms	3 from 8.1–8.4
31–33	Greedy algorithms: Prim's, Kruskal's, Dijkstra's, Huffman's	9.1–9.4
34	Lower-bound arguments	10.1
35	Decision trees	10.2
36	*P*, *NP*, and *NP*-complete problems	10.3
37	Numerical algorithms	10.4 (+ 11.4)
38	Backtracking	11.1
39	Branch-and-bound	11.2
40	Approximation algorithms for *NP*-hard problems	11.3

Acknowledgments

I would like to start by acknowledging the authors of other algorithm textbooks from whose insights and presentation ideas I have benefited both directly and indirectly. The advice and criticism of the book's reviewers have made the book better than it would have been otherwise. I am thankful to Simon Berkovich (George Washington University), Richard Borie (University of Alabama), Douglas M. Campbell (Brigham Young University), Bin Cong (California State University, Fullerton), Steve Homer (Boston University), Roland Hübscher (Auburn University), Sukhamay Kundu (Louisiana State University), Sheau-Dong Lang (University of Central Florida), John C. Lusth (University of Arkansas), John F. Meyer (University of Michigan), Steven R. Seidel (Michigan Technological University), Ali Shokoufandeh (Drexel University), and George H. Williams (Union College).

I am grateful to my colleague Mary-Angela Papalaskari, who used the manuscript in teaching a course on algorithms at Villanova and suggested several improvements to the text and the exercises. She also enthusiastically supported the idea of systematic utilization of puzzles in the book. Another colleague, John Matulis, used the manuscript in his sections, too, and provided me with useful feedback. A former student, Andiswa Heinegg, helped to prepare the manuscript and, through her critique, made it clearer in both content and style.

Students at Villanova have suffered the inconvenience of using the manuscript as their textbook over the past few semesters. I acknowledge their patience, useful feedback, and corrections to the errors and typos they found. The remaining errors are not, of course, their fault: I introduced them after they had taken the course.

I thank all the people at Addison-Wesley and its associates who worked on my book. I am especially grateful to my editor, Maite Suarez-Rivas, and her former assistant Lisa Hogue for sharing and supporting my enthusiasm for this project.

Finally, I am indebted to two members of my family. Living with a spouse who is writing a book is probably more trying than doing the actual writing. My wife, Maria, lived through two years of this, helping me any way she could. And help she did: all 250 or so figures in the book are her creations. My daughter Miriam has been my English prose guru over many years. Not only did she read large portions of the book; she was instrumental in finding the chapter epigraphs.

Anany Levitin
anany.levitin@villanova.edu
August 2002

Brief Contents

Contents