

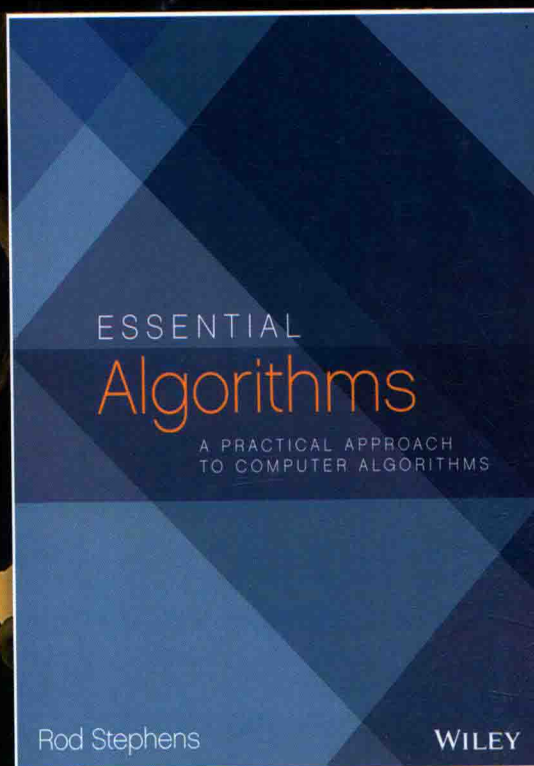
WILEY

算法基础

[美] 罗德·斯蒂芬斯 (Rod Stephens) 著

王宏志 黎玲利 译

Essential Algorithms
A Practical Approach to Computer Algorithms



机械工业出版社
China Machine Press

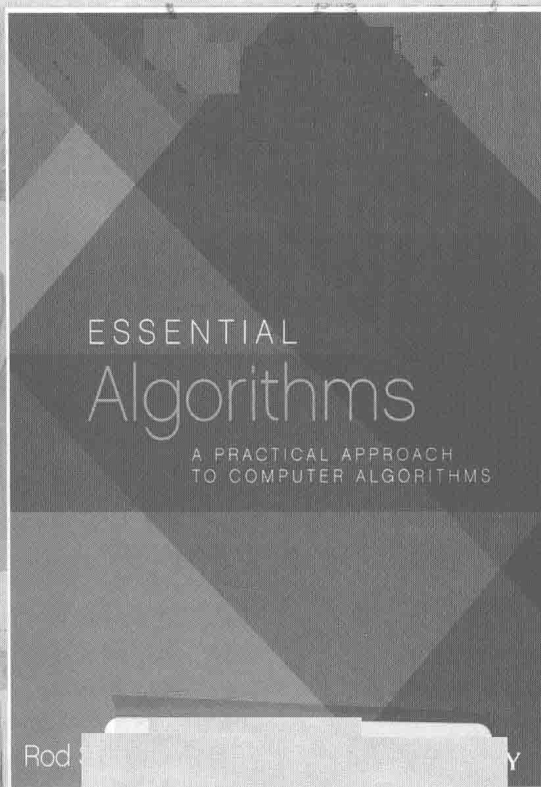
计 算 机 科 学 丛 书

算法基础

[美] 罗德·斯蒂芬斯 (Rod Stephens) 著

王宏志 黎玲利 译

thms



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

算法基础 / (美) 罗德·斯蒂芬斯 (Rod Stephens) 著; 王宏志, 黎玲利译. —北京: 机械工业出版社, 2017.4

(计算机科学丛书)

书名原文: Essential Algorithms: A Practical Approach to Computer Algorithms

ISBN 978-7-111-56092-0

I. 算… II. ① 罗… ② 王… ③ 黎… III. 电子计算机—算法理论 IV. TP301.6

中国版本图书馆 CIP 数据核字 (2017) 第 032015 号

本书版权登记号: 图字: 01-2013-8307

Copyright © 2013 by John Wiley & Sons, Inc., Indianapolis, Indiana

All Rights Reserved. This translation published under license. Authorized translation from the English language edition, entitled Essential Algorithms: A Practical Approach to Computer Algorithms, ISBN 978-1-118-61210-1, by Rod Stephens, Published by John Wiley & Sons. No part of this book may be reproduced in any form without the written permission of the original copyrights holder.

本书中文简体字版由约翰·威利父子公司授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

本书封底贴有 Wiley 防伪标签, 无标签者不得销售。

本书的撰写有机结合了理论与实现, 在讲授算法理论的同时, 也通过 C# 实例讲授了算法的实现。通过描述并分析一些重要的传统算法, 让读者了解每一个算法在什么时候使用较为适合, 并学会创造自己的算法。书中介绍的技巧让读者能从不同的角度看问题, 建立有用的方法工具, 从而解决实际问题, 抑或从容面对面试难题。

本书适合当作“算法设计与分析”和“数据结构与算法”两门课程的教材或参考书使用。特别是本书还融入和面试相关的内容, 对读者参加算法相关工作的面试很有帮助。

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 唐晓琳

责任校对: 殷虹

印刷: 北京建宏印刷有限公司

版次: 2017 年 5 月第 1 版第 1 次印刷

开本: 185mm×260mm 1/16

印张: 26

书号: ISBN 978-7-111-56092-0

定价: 79.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

文艺复兴以来，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域中取得了垄断性的优势；也正是这样的优势，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自1998年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson, McGraw-Hill, Elsevier, MIT, John Wiley & Sons, Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum, Bjarne Stroustrup, Brian W. Kernighan, Dennis Ritchie, Jim Gray, Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, Abraham Silberschatz, William Stallings, Donald E. Knuth, John L. Hennessy, Larry L. Peterson等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力相助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专门为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方式如下：

华章网站：www.hzbook.com

电子邮件：hzsj@hzbook.com

联系电话：(010) 88379604

联系地址：北京市西城区百万庄南街1号

邮政编码：100037



华章科技图书出版中心

算法的修养对计算机从业者来说至关重要，这种重要性不仅仅体现在计算机科学的研究中，也体现在软件开发的过程中，可以说开发一款优秀软件离不开高效的算法。因此，算法知识成为面试过程中的必考知识。正是由于其重要性，各出版社出版了林林总总的算法设计与分析教材和参考书。

和目前的算法教材相比，本书有着如下鲜明的特征：

第一，本书的撰写有机结合了理论与实现。算法是一门理论性很强的课程，这也使得传统的算法教材中较少有算法实现。本书则不同，在讲授算法理论的同时，也通过 C# 实例讲授了算法的实现，并且既有理论方面的习题，也有程序设计方面的习题。

第二，本书在内容上采取“广度优先”的组织思想，不仅讲授了基础的算法和数据结构知识，如数值算法、排序、搜索、链表、树算法、图算法，也介绍了算法设计与分析中的高级部分，如密码算法、计算复杂性理论、分布式算法等，这让读者阅读此书就能掌握多种不同的算法与数据结构。

第三，本书有机地融合了算法与数据结构的知识，可以同时当作“算法设计与分析”和“数据结构与算法”两门课程的教材或参考书使用。特别地，本书还融入和面试相关的内容，使得本书特别适合作为相关工作面试的参考资料。

译者应机械工业出版社的要求对本书进行了翻译。由于水平有限，而且翻译要求的时间紧张，译文中一定存在许多不足，在此敬请各位同行专家学者和广大读者批评指正，欢迎大家将发现的错误或提出的意见与建议发送到邮箱 wangzh@hit.edu.cn，以便将来改进本书的译本。

我要感谢哈尔滨工业大学的李沅泽、刘奇、樊昱才、张晓凯、魏思达、魏龔、董乐天、马靖昆、王必聪、季文骢、唐帮盈、司傲然、徐时雨、高桥、杨树佳、时春香、米尔根·紫帕、周舒妍、任雪健、韩雍博、陆鑫、罗嘉瑞、道发发、马常青、李宁、赵世琦、妥明翔、万晓珑、孙彤等同学参加了翻译工作。在完成译稿之后，哈尔滨工业大学的巢泽敏同学阅读全文并且对译稿提出了很多有益的意见，在此也表示感谢。同时感谢机械工业出版社的姚蕾编辑和朱劼编辑，由于她们的信任和支持，本书的翻译工作才得以顺利进行。

王宏志 黎玲利

2016年8月10日于哈尔滨

算法是使高效的程序成为可能的方法。它们解释了如何排列记录、搜索项、计算数值（比如质因子分解）、查找一个街道网络中的最短路径、确定可能通过通信网络的最大流。算法好坏的差别可能意味着是在一秒、一个小时内解决问题，还是永远也不能解决问题。

学习算法使你能建立有用的方法工具来解决具体的问题。它能帮助你理解在不同的情况下，哪个算法是最有效的，所以对于一个特定的问题，你就能选择最适合的算法。对某些数据而言性能优异的算法可能对其他的数据而言表现糟糕。所以知道如何选择一个最适合当前情况的算法是很重要的。

更重要的是，通过学习算法，你可以学习常规的问题解决技巧。即使你已知的任何算法都不能很好地适合当前的状况，你也可以把这些技巧应用到其他问题上。这些技巧让你从不同的角度看问题，使你能创造和分析自己的算法，从而解决你的问题，并且满足没有预料到的需求。

除了帮助你解决工作上的问题，这些技巧甚至会帮助你获得需要使用这些技巧的工作。许多大的科技公司，比如微软、谷歌、雅虎、IBM 等公司都想要让他们的程序员理解算法和相关问题的解决技巧。其中，有些公司因为让面试者在面试中解决算法编程和逻辑难题而“臭名昭著”。

好的面试官不一定期待你解决每一个难题。事实上，当你没有解决某个难题时，他们可能会了解更多。最好的面试官想看到你如何处理一个不熟悉的问题，而不是想要看到答案。他们想看到你是举起双手说这个问题在面试中是不合理的，还是你分析了这个问题，并提出了一条有希望的推理来用算法解决这个问题。“天哪，我不知道。或许我要上网搜一下”将会是一个坏的答案。“看起来一个递归的分治法可能有用”将会是一个好得多的答案。

这是一本易读的计算机算法导论书。它描述了一些重要的传统算法，并且说明了每一个算法在什么时候是适合的。它解释了如何分析算法从而理解它们的行为。最重要的，它教会了你创造自己新算法的技巧。

这里是本书中描述的一些有用的算法：

- 数值算法，比如随机化、分解因式、处理质数、数值积分
- 熟练操作常见数据结构的方法，比如堆、树、平衡树、B 树
- 排序和搜索
- 网络算法，比如最短路径、生成树、拓扑排序和流计算

这里是本书中解释的一些常规的问题解决技巧：

- 暴力或者穷举搜索
- 分治法
- 回溯法
- 递归
- 分支界限
- 贪心算法和爬山法

- 最小花费算法
- 缩小范围
- 启发式算法

为了帮助读者掌握这些算法，本书提供了一些练习，读者可以利用它们来探索自己的方法，以便修改书中的算法并把它们应用到新的情况中。这些练习也会帮助你巩固算法中的主要技巧。

最后，本书包含了解决面试中可能遇到的算法问题的技巧。算法技巧会让你解决许多面试中的难题。即使你不能用算法技巧解决每一个难题，至少表明你熟悉这些方法并且能用它们解决其他的问题。

算法选择

本书中的每一个算法都是因为一个或多个下列原因而被选择的：

- 这个算法是有用的，并且有经验的程序员应该能理解它如何工作以及如何程序中使它。
- 这个算法展示了你可以应用到其他问题中的重要算法编程技巧。
- 计算机科学专业的学生普遍学习这个算法，所以这个算法或者它使用的技巧可能出现在一个技术性面试中。

当读者读完本书并做完练习后，将会有一个好的算法基础并掌握解决自己编程问题的技巧。

本书的目标人群

本书主要针对三种读者：专业程序员、准备面试的程序员和学习编程的学生。

专业程序员将会发现本书中所描述的算法和技巧对于解决他们工作中遇到的难题是很有用的。即使遇到无法直接用书中的算法解决的问题，阅读本书中的算法也会让你从新的角度观察问题，从而发现新的解决办法。

准备工作面试的程序员可以使用本书来磨炼他们的算法技巧。你的面试可能不会包括书中描述的任何问题，但是可能包含一些足够相似的问题。你可以用从本书中学到的技巧解决它们。

学习编程的学生应该学习这些算法。本书中描述的许多方法是简单、富有魅力而且有效的。但是它们并不都是十分显而易见的，所以你不一定在偶然间自己发现它们。递归、分治、分支界限等技巧，还有使用众所周知的数据结构，这些对任何有兴趣编程的人都是十分需要掌握的。

注 就我个人而言，算法只是一种乐趣，它们就像填字游戏或者数独一样。我喜欢组成一个复杂的算法，倒一些数据进去，然后看到一个美丽的三维图形、一条匹配一系列点的曲线，或者一些其他的优雅的答案出现。我喜欢这种感觉。

充分运用本书

仅仅是阅读本书，就能学到一些新的算法和技巧。但是要真正掌握这些算法体现出的方法，就需要用它们工作，需要用某种编程语言实现它们。你还需要实验、修改这些算法，尝试旧问题的新变型。本书中的练习和面试问题能让你想到使用这些算法中技巧的新方法。

为了从本书中得到最大的收获，我强烈推荐用一种你最喜欢的语言实现尽可能多的算法。甚至用多种语言来实现，看看不同的语言是如何影响问题实现的。你应该研究这些练习并至少写下解决它们的提纲。理想状况下，你也应该实现它们。通常一个练习被编入本书是有原因的，你可能需要认真审视这个问题，才能发现这个原因。

最后，看看网上的一些面试问题，然后弄明白你将如何解决它们。在许多面试中，可能不会要求你实现一个解决方案，但是你应该能勾勒出大体的解决方案。如果你有时间实现解决方案，你将会学到更多。

理解算法是一个实践问题。不要害怕放下书，打开一个编译器，写一些真正的代码吧。

本书的网站

其实，本书有两个网站：Wiley 的版本和我的版本。每一个网站都包含了本书的源代码。

Wiley 的网址是 <http://www.wiley.com/go/essentialalgorithms>。你也可以去 <http://www.wiley.com> 通过书名[⊖]或 ISBN 来搜索这本书。一旦找到了这本书，可以点击“下载”（Downloads）标签来获取本书的所有源代码。一旦下载了这些代码，只需要用你喜欢的压缩工具解压它们即可。

注 在 Wiley 的网站，你可能会发现用 ISBN 搜索更容易。本书的 ISBN 是 978-1-118-61210-1。

要想在我的网站上找到这本书，请登录 <http://www.CSharpHelper.com/algorithms.html>。

本书的结构

这一部分介绍了本书的详细内容。

第 1 章 解释了分析算法中必须理解的概念。其中讨论了算法与数据结构之间的不同，引入了大 O 符号，并介绍了当实际因素比理论运行时间计算更重要时的时间问题。

第 2 章 解释了几个处理数值的算法。这些算法用于随机化数字和数组，计算最大公约数和最小公倍数，快速求幂，判断一个数字是否为质数。一些算法还引入了自适应求积和蒙特卡罗模拟这些重要的技巧。

第 3 章 解释了链表数据结构。这种灵活的结构可以用来存储随时间增长的列表。这些基础概念对于建立其他的链接数据结构，比如树和网络，也是很重要的。

第 4 章 解释了特殊的数组算法和数据结构，比如节省程序时间和存储空间的三角形数组和稀疏数组。

第 5 章 解释了让一个程序以先进先出（FIFO）和后进先出（LIFO）的顺序存储和取出元素的算法及数据结构。这些数据结构对于其他算法很有用。它们也可以用来模拟现实世界中的场景，比如在商店中排队结账。

第 6 章 解释了排序算法。这些算法展示了有用的算法技巧。不同的排序算法适应于不同种类的数据，并且有不同的理论运行时间，所以理解这些算法的特点是很有好处的。这些也是为数不多理论性能界限已知的算法，所以研究它们特别有趣。

第 7 章 解释了程序可以用来搜索有序列表的方法。这些算法展示了重要的技巧，比如说二分法和插值法。

⊖ 这里指原书名。——编辑注

第 8 章 解释了散列表——使用额外的存储空间来使一个程序快速查找特定元素的数据结构。它们有力地展示了时间与空间的权衡在许多程序中的重要性。

第 9 章 解释了递归算法——调用其自身的算法。递归技巧使某些算法更容易理解和实现，虽然它们有时会导致问题。所以这一章也介绍了当必要时如何从一个算法中移除递归。

第 10 章 解释了高度递归的树这一数据结构。它对于存储、操作和学习分层数据很有用，并且可以应用在意想不到的地方，比如求算术表达式的值。

第 11 章 解释了在自身随着时间增长时保持平衡的树。一般来说，树结构会变得很高很瘦，并且那将破坏树算法的性能。平衡树通过确保树不变得太高太瘦而解决这个问题。

第 12 章 解释了尝试解决可以被建模为一系列决策的问题的算法。这些算法通常在解决非常困难的问题中使用，所以它们一般只找到近似解而不是可能的最优解。然而，它们是非常灵活的，并且可以被应用到范围很广的问题中。

第 13 章 解释了基础网络算法，比如访问一个网络中的所有结点、检测循环、创建生成树、查找网络中的路径。

第 14 章 解释了更多的网络算法，比如用拓扑排序来安排相关的任务、地图着色、网络克隆、向雇员分配任务。

第 15 章 解释了操作字符串的算法。这些算法中的一部分，比如搜索子串，被内置为工具。大多数编程语言都可以直接使用它们，而不用自己编程。其他的算法，比如括号匹配和查找字符串差异，要求额外做一些工作，而且也展示了一些有用的技巧。

第 16 章 解释了如何加密和解密信息。其中涵盖了加密基础和几个有趣的加密技巧，比如 Vigenère 密码、分组密码和公钥加密。这一章没有叙述所有的特定加密算法的细节，比如 DES（数据加密标准）和 AES（高级加密标准），因为它们更适合在一本加密的专著中介绍。

第 17 章 解释了计算机科学中最重要的一类问题：P（可以在多项式时间内解决的问题）和 NP（不能在多项式时间内解决的问题）。这一章介绍了这些类别，证明一个问题属于 P 还是 NP，还有计算机科学中最深刻的问题：P 等于 NP 吗？

第 18 章 解释了在多处处理器上运行的算法。几乎所有的现代计算机都包含着多处处理器，并且未来的计算机将会包含更多，所以这些算法对于最有效地利用一台计算机的潜在能力至关重要。

第 19 章 介绍了一些窍门和技巧，读者可以用它们来攻克编程面试中可能遇到的难题和挑战。这章还包括一个网站列表，这些网站上有大量读者可以用作练习的难题。

附录 A 总结了本书中算法使用的观点和策略，使读者可以解决这里的算法没有明确涵盖的其他问题。

附录 B 包含每章末练习的解答。

使用本书需要的知识

读者不需要任何特殊的知识就能阅读本书和理解书中的算法。然而，如果读者真的想掌握这些知识，应该用某种实际的编程语言尽可能多地实现算法。用任何语言完成算法的具体实现将会帮助读者更好地理解算法的细节和语言要求的特殊处理方法。

当然，如果读者打算使用一种编程语言实现算法，需要配备合适的开发环境。

本书的网站上提供了在 Visual Studio 2012 中用 C# 编写的实现实例，读者可以下载并考查它们。如果想运行那些程序，需要在一台能很好运行 Visual Studio 的计算机上安装 C# 2012。

运行任何版本的 Visual Studio，都要求你有一台相当快的、有大的硬盘和内存的现代计算机。例如，我很高兴我的计算机有 2GB 内存、500GB 硬盘、1.83GHz 的 Intel Core 2 系统。硬盘空间比我需要的大许多，但是硬盘相对便宜，为什么不买大的呢？

读者可以在一台功能不是很强大的计算机上运行 Visual Studio，但是使用功能不强的计算机会非常缓慢且令人沮丧。Visual Studio 会占用很大的内存，所以，如果计算机遇到了性能问题，安装更多的内存可能会有帮助。

这些程序将会用 C# Express Edition 载入和执行，所以没有必要安装更贵的 C# 版本。你可以在 <http://www.microsoft.com/visualstudio/eng/downloads#d-express-windows-desktop> 上获得 C# Express Edition 的更多信息并下载 C#。

约定

为了帮读者充分利用本书，并了解正在发生的事情，我在本书中使用了几个约定。

漂亮的边框

像这样的边框包含着更多的信息和副主题。

注 像这样的格式有注释、指导、提示、技巧和当前讨论的旁白。

至于文本中的样式：

- 新的术语和重要的词语在被引入时表示为楷体。
- 击键操作，比如 Ctrl + A，意味着先按住 Ctrl 键再按 A 键。
- 文本中的 URL 和电子邮件地址用斜体表示，比如 *<http://www.CSharpHelper.com>* 和 *RodStephens@CSharpHelper.com*。

给我发电子邮件

如果你有任何疑问、意见或建议，请随时给我发电子邮件至 *RodStephens@CSharpHelper.com*。我不能保证解决你所有的算法问题，但我一定尽力为你指明正确的方向。

致谢

感谢 Bob Elliott、Tom Dinse、Gayle Johnson 和 Daniel Scribner，是他们的努力使本书得以出版。同时感谢技术编辑 George Kocur、Dave Colman 和 Jack Jianxiu Hao，他们的帮助确保了本书中的信息尽可能准确。

目 录

Essential Algorithms: A Practical Approach to Computer Algorithms

出版者的话

译者序

前言

第 1 章 算法基础知识 1

1.1 方法 1

1.2 算法和数据结构 2

1.3 伪代码 2

1.4 算法的特点 4

1.4.1 大 O 符号 5

1.4.2 常见的运行时间函数 7

1.4.3 可视化函数 12

1.5 实际因素 12

1.6 总结 13

练习 13

第 2 章 数值算法 16

2.1 随机化数据 16

2.1.1 随机数生成 16

2.1.2 随机化数组 20

2.1.3 生成不均匀分布 21

2.2 寻找最大公约数 21

2.3 求幂运算 23

2.4 有关素数的运算 24

2.4.1 寻找素数因子 24

2.4.2 寻找素数 26

2.4.3 素性测试 27

2.5 进行数值积分 28

2.5.1 矩形规则 28

2.5.2 梯形规则 29

2.5.3 自适应求积 30

2.5.4 蒙特卡罗积分 32

2.6 查找零 32

2.7 总结 34

练习 34

第 3 章 链表 36

3.1 基本概念 36

3.2 单链表 37

3.2.1 遍历链表 37

3.2.2 查找单元格 37

3.2.3 使用哨兵 38

3.2.4 在开头添加单元格 39

3.2.5 在结尾添加单元格 40

3.2.6 在某个单元格后插入单元格 40

3.2.7 删除单元格 41

3.3 双向链表 42

3.4 有序链表 43

3.5 链表算法 44

3.5.1 复制链表 44

3.5.2 链表的插入排序 45

3.6 链表的选择排序 46

3.7 多线程链表 47

3.8 循环链表 48

3.8.1 标记单元格 49

3.8.2 使用散列表 50

3.8.3 链表回溯 51

3.8.4 反转链表 51

3.8.5 乌龟和兔子 53

3.8.6 双向链表中的循环问题 55

3.9 总结 55

练习 55

第 4 章 数组 57

4.1 基本概念 57

4.2 一维数组 58

4.2.1 查找元素 58

4.2.2 查找最大值、最小值、平均值 59

4.2.3 插入元素 60

4.2.4 移除元素 61

4.3 非零下界 61

4.3.1 二维数组	61	第 7 章 搜索	110
4.3.2 多维数组	62	7.1 线性搜索	110
4.4 三角形数组	64	7.2 二分搜索	111
4.5 稀疏数组	66	7.3 插值搜索	112
4.5.1 找到行或列	67	7.4 总结	113
4.5.2 获取值	68	练习	113
4.5.3 设置值	69	第 8 章 散列表	114
4.5.4 删除值	71	8.1 散列表的基础知识	114
4.6 矩阵	72	8.2 链	115
4.7 总结	74	8.3 开放寻址	116
练习	74	8.3.1 删除记录	117
第 5 章 栈和队列	76	8.3.2 线性探测	118
5.1 栈	76	8.3.3 二次探测	119
5.1.1 栈的链表实现	76	8.3.4 伪随机探测	120
5.1.2 栈的数组实现	77	8.3.5 双散列	120
5.1.3 双向栈	78	8.3.6 有序散列	121
5.1.4 栈的算法	79	8.4 总结	122
5.2 队列	84	练习	123
5.2.1 队列的链表实现	84	第 9 章 递归	125
5.2.2 队列的数组实现	85	9.1 基础算法	125
5.2.3 专用队列	86	9.1.1 阶乘	125
5.3 总结	87	9.1.2 斐波那契数	127
练习	87	9.1.3 汉诺塔	128
第 6 章 排序	89	9.2 图算法	130
6.1 时间复杂度为 $O(N^2)$ 的算法	89	9.2.1 科赫曲线	130
6.1.1 数组中的插入排序	89	9.2.2 希尔伯特曲线	131
6.1.2 数组中的选择排序	90	9.2.3 谢尔宾斯基曲线	132
6.1.3 冒泡排序	91	9.2.4 垫片	134
6.2 时间复杂度为 $O(N \log N)$ 的算法	93	9.3 回溯算法	134
6.2.1 堆排序	93	9.3.1 八皇后问题	136
6.2.2 快速排序	98	9.3.2 骑士巡游	138
6.2.3 归并排序	103	9.4 选择与排列	140
6.3 时间复杂度为亚 $O(N \log N)$ 的算法	105	9.4.1 循环选择	140
6.3.1 计数排序	106	9.4.2 重复选择	141
6.3.2 桶排序	107	9.4.3 不重复选择	142
6.4 总结	108	9.4.4 元素可重复的排列	143
练习	108	9.4.5 元素不重复的排列	144

9.5 消去递归.....	145	11.2.2 删除值.....	189
9.5.1 尾递归的消除.....	145	11.3 B 树.....	191
9.5.2 存储中间值.....	146	11.3.1 添加值.....	191
9.5.3 一般递归的消除.....	148	11.3.2 删除值.....	192
9.6 总结.....	150	11.4 平衡树变体.....	193
练习.....	151	11.4.1 自上而下的 B 树.....	193
第 10 章 树.....	153	11.4.2 B+树.....	193
10.1 树的术语.....	153	11.5 总结.....	194
10.2 二叉树属性.....	155	练习.....	195
10.3 树的表示.....	157	第 12 章 决策树.....	196
10.3.1 建立树的通用方法.....	157	12.1 游戏搜索树.....	196
10.3.2 构造完全树.....	159	12.1.1 极小化极大值算法.....	197
10.4 树的遍历.....	160	12.1.2 初始步骤和反应.....	199
10.4.1 前序遍历.....	160	12.1.3 启发式游戏树.....	200
10.4.2 中序遍历.....	162	12.2 搜索通用决策树.....	201
10.4.3 后序遍历.....	163	12.2.1 优化问题.....	202
10.4.4 深度优先遍历.....	164	12.2.2 穷举搜索.....	202
10.4.5 遍历的运行时间.....	164	12.2.3 分支界限.....	203
10.5 排序树.....	165	12.2.4 决策树的启发式搜索.....	205
10.5.1 添加结点.....	165	12.2.5 其他决策树问题.....	209
10.5.2 查找结点.....	166	12.3 总结.....	212
10.5.3 删除结点.....	167	练习.....	195
10.6 线索树.....	168	第 13 章 基本网络算法.....	214
10.6.1 建立线索树.....	169	13.1 网络术语.....	214
10.6.2 使用线索树.....	171	13.2 网络的表示方法.....	216
10.7 特化树算法.....	172	13.3 网络的遍历.....	218
10.7.1 动物游戏.....	172	13.3.1 深度优先遍历.....	218
10.7.2 表达式求值.....	173	13.3.2 广度优先遍历.....	220
10.7.3 四叉树.....	175	13.3.3 连通性测试.....	221
10.7.4 Trie 树.....	179	13.3.4 生成树.....	223
10.8 总结.....	182	13.3.5 最小生成树.....	224
练习.....	182	13.4 寻找路径.....	225
第 11 章 平衡树.....	185	13.4.1 寻找任一路径.....	225
11.1 AVL 树.....	185	13.4.2 标签设置最短路径.....	225
11.1.1 添加值.....	185	13.4.3 标签校正最短路径.....	227
11.1.2 删除值.....	187	13.4.4 任意两点间最短路径.....	228
11.2 2-3 树.....	187	13.5 总结.....	232
11.2.1 添加值.....	188	练习.....	232

第 14 章 更多的网络算法	234	16.4 分组密码	267
14.1 拓扑排序	234	16.4.1 代换 - 置换网络	267
14.2 回路检测	236	16.4.2 Feistel 密码	268
14.3 地图着色	237	16.5 公钥加密和 RSA	269
14.3.1 两色着色	237	16.5.1 欧拉函数	270
14.3.2 三色着色	238	16.5.2 在取模运算下的乘法 逆元素	270
14.3.3 四色着色	239	16.5.3 一个 RSA 的例子	270
14.3.4 五色着色	239	16.5.4 现实思考	271
14.3.5 其他地图着色算法	241	16.6 加密技术的其他用途	271
14.4 最大流	242	16.7 总结	272
14.4.1 工作分配	243	练习	273
14.4.2 最小割	244		
14.5 总结	245	第 17 章 复杂性理论	274
练习	246	17.1 符号	274
第 15 章 字符串算法	247	17.2 复杂性分类	275
15.1 括号匹配	247	17.3 归约	277
15.1.1 求算术表达式	248	17.3.1 3SAT	278
15.1.2 构建解析树	248	17.3.2 二分图匹配	278
15.2 模式匹配	249	17.4 NP 难问题	279
15.2.1 DFA	249	17.5 检测、报告和优化问题	279
15.2.2 为正则表达式建立 DFA	251	17.5.1 检测 $\leq p$ 报告	279
15.2.3 NFA	252	17.5.2 报告 $\leq p$ 优化	280
15.3 字符串搜索	253	17.5.3 报告 $\leq p$ 检测	280
15.4 计算编辑距离	256	17.5.4 优化 $\leq p$ 报告	280
15.5 总结	258	17.6 NP 完全问题	281
练习	258	17.7 总结	282
		练习	283
第 16 章 密码学	260	第 18 章 分布式程序设计	284
16.1 术语	260	18.1 并行的种类	284
16.2 换位密码	261	18.1.1 脉动阵列	284
16.2.1 行 / 列换位	261	18.1.2 分布式计算	286
16.2.2 列换位	262	18.1.3 多 CPU 的处理	287
16.2.3 路由加密算法	263	18.1.4 竞争条件	287
16.3 替换密码	264	18.1.5 死锁	290
16.3.1 凯撒替换	264	18.1.6 量子计算	291
16.3.2 维吉尼亚密码	265	18.2 分布式算法	291
16.3.3 简单替换密码	266	18.2.1 调试分布式算法	292
16.3.4 一次性密码本	266	18.2.2 高度并行算法	292

18.2.3 归并排序 293

18.2.4 哲学家进餐 294

18.2.5 两个将军问题 296

18.2.6 拜占庭将军 297

18.2.7 一致性 298

18.2.8 领导人选举 301

18.2.9 快照 301

18.2.10 时钟同步 302

18.3 总结 303

练习 303

第 19 章 面试难题 305

19.1 面试难题提问 306

19.2 面试难题回答 307

19.3 总结 309

练习 311

附录 A 算法概念综述 312

附录 B 练习解答 320

索引 371

算法基础知识

在开始算法学习之前，你需要一点背景知识。简单地说，首先需要知道的是算法是完成某些事情的方法。它定义了用某个方法执行一个任务的步骤。

这个定义看起来足够简单，但是没有人为了做一件十分简单的工作而写算法。没有人写指令来获取数组中的第四个元素。这只是假设这是一个数组定义的一部分，并且你知道怎么做（如果你知道如何在这个问题中使用编程语言）。

一般来说，人们只是为了完成复杂的任务而写算法。算法解释了如何找到一个复杂代数问题的答案，如何在一个包含数千条街道的网络中找到最短路径，抑或是如何在数百个投资中找到最好的投资组合来使利益最大化。

本章介绍一些基本的算法概念。如果想从算法学习中获得最大的收获，就应该理解它们。

跳过这一章然后去学某个特定的算法似乎很有诱惑力，但是你至少应该浏览一下这一章。请着重注意 1.4.1 节，因为对运行时间是否有良好理解意味着算法能否在数秒、数小时内完成任务，还是永远不能完成任务。

1

1.1 方法

为了尽可能地了解一个算法，你需要做出更多努力而非简单地遵循它的步骤。你需要理解以下内容：

算法的行为：它寻找出了最好的方法还是仅仅找出了一个可行的解决办法？可能有多个最佳的解决方案吗？是否有一个方法可以从其他解决方案中找到一个“最佳”解决方案？

算法的速度：这个算法是快还是慢？还是它通常是快的，但有时对于某些输入慢？

算法的存储要求：这个算法需要多大的存储空间？这个大小合理吗？这个算法是否需要数十亿 TB 的存储空间，比一台计算机可能有的存储空间还多（至少对现在的计算机来说如此）？

算法设计主要使用的技巧：是否可以利用这些技术解决类似的问题？

这本书涵盖了所有这些主题。然而，本书没有试图用数学的精确理论去涵盖每一个算法的每一个细节。本书使用一种直观的方法来解释算法及其性能，但是本书没有用严谨的细节去分析算法的性能。尽管这种细节分析的证明可能是有趣的，但它也可能让人混乱，并且会占用大量的篇幅，提供了对大多数程序员来说不必要的细节。毕竟，这本书主要适用于需要完成工作的专业程序员。

本书把具有相关主题的算法归在一章。有时这个主题是它们所执行的任务（排序、搜索、网络算法），有时是它们所使用的数据结构（链表、数组、散列表、树），有时是它们使用的技巧（递归、决策树、分布式算法）。粗看起来，这些分组似乎是任意的，但当你读到这些算法的时候，可以看到它们很好地组合在一起。

除了这些类别，许多算法的基本主题跨越章节界限。例如，树算法（第 10、11 和 12 章）往往是高度递归的（第 9 章）。链表（第 3 章）可用于建立数组（第 4 章）、散列表（第 8 章）、

栈（第5章）和队列（第5章）。引用和指针的概念可以用来建立链表（第3章）、树（第10、11和12章）和网络（第13和14章）。在阅读时，请关注这些共同的线索。附录A中总结了常见的策略，这些策略能使这些想法更容易被理解。

1.2 算法和数据结构

算法是完成某个任务的方法。数据结构是一种安排数据的方法。这个方法使解决某个特定的问题更简单。数据结构可以是一种在数组中放置数值的方法，一个以某种特定结构链接物体的链表、一棵树、一个图、一个网络，甚至更奇异的东西。

通常算法是与数据结构紧密联系在一起。比如，第15章中描述的编辑距离算法使用了一个网络来确定两个字符串的相似程度。这个算法与网络紧紧地联系在一起，没有网络它就不能工作。

通常一个算法意味着：“建立一个特定的数据结构，然后用一个特定的方法使用它。”没有数据结构就没有算法。如果不打算在算法中运用数据结构，那么设计这个算法也就没有意义。

1.3 伪代码

为了使本书中描述的算法尽可能有用，首先我们用直观的术语来描述它们。有了这个高层次的解释，可以能够用大多数的编程语言来实现这些算法。

然而，一个算法的实现经常包含很多难以实现的琐碎细节。为了使这些细节易于处理，算法也用伪代码来描述。伪代码是很像编程语言但又不是真正的编程语言的一种文本。伪代码提供了代码实现算法过程中会用到的结构和细节，同时又不与某种特定的编程语言联系在一起。希望你能把这些伪代码翻译成真正的代码，然后在你的计算机上执行。

下面的代码片段展示了计算两个整数的最大公约数（GCD）算法的伪代码示例：

```
// 寻找 a、b 的最大公约数
//  $GCD(a, b) = GCD(b, a \bmod b)$ .
Integer: Gcd(Integer: a, Integer: b)
    While (b != 0)
        // 计算余数
        Integer: remainder = a Mod b

        // 计算  $GCD(b, remainder)$ 
        a = b
        b = remainder
    End While

    //  $GCD(a, 0)$  为 a
    Return a
End Gcd
```

取模操作

取模操作，在伪代码中写作 Mod。它意味着整除之后的余数。比如说， $13 \bmod 4 = 1$ 。因为 13 被 4 除，商是 3，余数是 1。

伪代码以一个注释开头。注释以字符“//”开始然后延伸到这一行的结束。