Broadview
www.broadview.com.cn

PEARSON

算法分析导论（第2版）

AN INTRODUCTION TO THE ANALYSIS OF ALGORITHMS

SECOND EDITION

[美] Robert Sedgewick
Philippe Flajolet 著

英文版

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

算法分析导论（第2版）

AN INTRODUCTION TO THE

ANALYSIS OF ALGORITHMS

SECOND EDITION

[美] Robert Sedgewick
Philippe Flajolet 著

英文版

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

内 容 简 介

本书全面介绍了算法的数学分析中所涉及的主要技术。涵盖的内容来自经典的数学课题（包括离散数学、初等实分析、组合数学），以及经典的计算机科学课题（包括算法和数据结构）。本书的重点是“平均情况”或“概率性”分析，书中也论述了“最差情况”或“复杂性”分析所需的基本数学工具。

本书第1版为行业内的经典著作，本版不仅对书中图片和代码进行了更新，还补充了新章节。全书共9章，第1章是导论；第2~5章介绍数学方法；第6~9章介绍组合结构及其在算法分析中的应用。除每章包含的大量习题以及参考文献外，本书特设配套免费学习网站，为读者提供了很多关于算法分析的补充材料，包括课件和相关网站的链接，帮助读者提高学习兴趣，完成更深入的学习。

本书适合作为高等院校数学、计算机科学以及相关专业的本科生和研究生的教材，也可供相关技术人员和爱好者学习参考。

版权贸易合同登记号 图字：01-2013-4144

图书在版编目（CIP）数据

算法分析导论：第2版：英文/（美）塞奇威克（Sedgewick,R.），（美）弗拉若莱（Flajolet,P.）著. —北京：电子工业出版社，2015.6
书名原文：An Introduction to the Analysis of Algorithms,2E
ISBN 978-7-121-26070-4

Ⅰ.①算… Ⅱ.①塞… ②弗… Ⅲ.①算法分析－英文 Ⅳ.① TP311

中国版本图书馆 CIP 数据核字 (2015) 第 099853 号

策划编辑：张春雨
责任编辑：徐津平
印　　刷：北京丰源印刷厂
装　　订：河北省三河市路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编：100036
开　　本：787×980　1/16　印张：36.75　字数：602千字
版　　次：2015年6月第1版
印　　次：2015年6月第1次印刷
定　　价：128.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

序

分析算法可以给人带来两方面的快乐。其一，人们可以尽享优雅计算过程中所蕴含的让人沉醉的数学模式；其二，他们所学得的理论知识可以让自己更好更快地完成工作，这无疑是最实际的好处。

尽管数学模型只是对真实世界的一种理想化近似，但它对所有的科学活动而言都可谓是一剂灵丹妙药。在计算机中，数学模型往往可以精确地描述计算机程序所创造的世界，我想数学模型的重要性也因此大大增加了。我想，这也是为什么我在读研究生时会沉迷于算法分析，以至于这成为了我迄今为止的主要工作。

但直到今天，算法分析在很大程度上还是局限在相关专业的研究生和科研人员的圈子里。算法分析的概念既不晦涩也不复杂，但确实比较新，所以相关概念的学习和使用都还需要一些时间才能成熟。

现在，经过 40 余年的发展，算法分析已经非常成熟，足以成为计算机专业标准课程中的一部分。Sedgewick 和 Flajolet 写的这本众人翘首以盼的教科书也因此备受欢迎。本书的作者不仅仅是这个领域的世界级专家，同时也是算法分析的布道大师。我坚信，这本书会让每一位细细品读的计算机研究人员从中获益。

D. E. Knuth

前 言

本书的主要目的是介绍算法数学分析中涉及的主要技术。涵盖的内容都来自经典的数学和计算科学领域，包括来自数学领域的离散数学、实分析基础和组合数学，来自计算机领域的算法和数据结构等。本书的重点是“平均情况”或是“概率性”的分析，对“最差情况”或“复杂性”分析所需的基本数学工具也会涉及。

我们假设读者对计算机科学和实分析的基本概念是比较熟悉的。简单来说，读者要能够编写程序和证明定理。除此之外，本书对读者没有其他要求。

本书主要是为了用作高阶算法分析课程的教科书。书中涵盖了计算机专业学生所需离散数据的基本技术、组合数学和重要离散结构的基本属性，因此也可以用在计算机专业的离散数学课程中。通常此类课程所涵盖的内容都比较广泛，但很多教师都用自己的方法来选取其中的一部分内容教给学生。本书还可以用于给数学和应用数学的学生介绍计算机科学中与算法和数据结构有关的基本原则。

关于算法的数学分析的论文非常多，但这个领域的学生和研究人员却很难学习到该领域中广泛使用的方法和模型。本书的目标就是解决这个问题，一方面要让读者知道这个领域所面临的挑战，另一方面要让读者有足够的背景支持来学习为应对这些挑战而正在开发的工具。我们列出了相关的参考文献，因此本书可以作为研究生算法分析入门课程的基础教材，也可以用作想学习该领域中相关文献的数学或计算机领域研究人员的参考资料。

准备工作。假设读者有大学一二年级或与之等同的数学基础。组合数学和离散数学方面的基本课程，以及实分析、数值方法和基本数论方面的课程，有助于理解本书（有些内容跟本书是交叉的）。本书将涉及这些领域，但我们在这里只会对必要的内容做介绍。我们也会列出参考文献供想进一步学习的

人参考。

读者需要一到两个学期大学水平的编程经验，包括了解基本数据结构。我们不会涉及编程和具体实现的问题，但算法和数据结构是我们的核心。同数学方面的基础知识一样，本书中我们会简要介绍基本的信息，同时列出标准教材和知识来源供读者参考。

相关书籍。与本书有关的书籍包括 D. E. Knuth 写的 *Art of Computer Programming*（《计算机程序设计艺术》），Sedgewick 和 Wayne 写的 *Algorithms, Fourth Edition*（《算法》第 4 版），Cormen、Leiserson、Rivest 和 Stein 写的 *Introduction to Algorithms*（《算法导论》），以及我们自己写的 *Analytic Combinatorics*（《分析组合论》）。本书可以作为这些书的补充材料。

从基本思路上来说，本书与 D. E. Knuth 的书是最相似的。但这本书的重点是算法分析中的数学技术，而 D. E. Knuth 的那些书是内容更丰富的百科全书，其中算法的各种性质是主角，算法分析方法只是配角。本书可以作为学习 D. E. Knuth 书中进阶内容的基础。这本书还涵盖了 D. E. Knuth 的书诞生之后在算法分析领域出现的新方法和新结果。

本书尽可能地涵盖各种重要、有趣的基础算法，例如，Sedgewick 的《算法》（现在是第 4 版了，跟 K. Wayne 合著的）中所介绍的那些算法。《算法》一书涵盖了经典的排序、搜索和用于处理图和字符串的算法。我们这本书的重点是介绍用于预测算法性能和比较算法性能优劣的数学知识。

Cormen、Leiserson、Rivest 和 Stein 合著的《算法导论》是算法方面的标准教材，其中提供了关于算法设计的各种文献。《算法导论》一书（及相关的文献）关注的是算法设计和理论，大部分是基于最差性能边界的。而本书关注的则是算法分析，尤其是各种科学研究（而不是理论研究）所需的技术。第 1 章就是为此做铺垫的。

本书还会介绍分析组合学，它可以让读者开阔视野，也有助于开发用于新研究的高级方法和模型，而且不局限于算法分析，它还可以用于组合学和更广泛的科学应用。那一卷对读者的数学水平要求要高一些，差不多需要本科高年级学生或研究生一年级的水平。当然，仔细阅读本书也有助于读者学

习相关的数学知识。我的目标是尽可能让读者感兴趣，有动力更深入地学习。

如何使用这本书。本书读者在数学和计算机科学方面的知识储备肯定是不一样的。所以，读者要注意本书的结构：全书共 9 章，第 1 章是导论，第 2~5 章介绍数学方法，最后 4 章是组合结构及其在算法分析中的应用，具体如下：

导论

第 1 章 算法分析

离散数学方法

第 2 章 递归关系

第 3 章 生成函数

第 4 章 渐近逼近

第 5 章 分析组合学

算法与组合结构

第 6 章 树

第 7 章 排列

第 8 章 串和 trie 树

第 9 章 字与映射

第 1 章会介绍全书的主要内容，帮助读者理解本书的基本目标以及其余各章的目的。第 2~4 章涵盖了经典离散数学中的方法，重点是基本概念和技术。第 5 章是一个转折点，其中涵盖了分析组合学。计算机和计算模型的出现带来了很多新问题，这些问题往往涉及大型的离散结构。分析组合学的内容就是研究这些离散结构以解决新出现的问题。第 6~9 章则回到了计算机科学，涵盖各种组合结构的属性，它们跟基本算法的关系，以及分析结果。

我们试图让本书是自包含的，但本书的组织结构可以让老师很方便地根据学生和老师自己的背景知识及经验选取要重点讲解的部分。一种比较偏数学的教学方案是重点讲解本书第一部分的理论和证明，然后过渡到第 6~9 章的相关应用。另一个比较偏计算机科学的方案是简要介绍第 2~5 章的数学工具，重点放在本书第二部分跟算法有关的内容。但我们的根本目标还是要让

绝大部分的学生能通过本书学习到数学和计算机科学的新知识。

本书还罗列了很多参考资料以及数百个练习，鼓励读者去研究原始材料，更深入地研究本书中的内容。根据我们的教学经验，讲师可以通过计算实验室和家庭作业，灵活地组织授课和阅读材料。本书的内容为学生深入学习诸如Mathematica、MAPLE或SAGE之类的符号计算系统提供了一个很好的框架。更重要的是，对学生而言，通过比较数学研究结果和实际测试结果得出的结论是有重要价值的，不可忽略。

配套网站。本书的一个重要特点就是拥有配套的网站 aofa.cs.princeton.edu。这个网站是免费的，提供了很多关于算法分析的补充材料，包括课件和相关网站的链接，其中有一些关于算法和分析组合学的网站。这些材料对讲授这些内容的教师和自学者都很有帮助。

致谢。我们非常感谢普林斯顿大学的 INRIA 和国家科学基金，它们是我们这本书的主要资助者。其他的资助来自布朗大学、欧共体（Alcom 项目）、国防分析研究院、法国研究与技术部、斯坦福大学、布鲁塞尔自由大学和施乐帕克研究中心。本书历时多年才得以付梓，在此不可能一一列出诸多为本书提供支持和帮助的人和机构，我们对此深表歉意。

正如书中所说，D. E. Knuth 对本书有重要的影响。

过去几年，在普林斯顿、巴黎和普罗维登斯听过我课的学生们为本书提供了宝贵的反馈意见。全世界各地的学生和教师也为本书的第 1 版提出了很多好建议。我们在此特别感谢 Philippe Dumas、Mordecai Golin、Helmut Prodinger、Michele Soria、Mark Daniel Ward 和 Mark Wilson 的帮助。

Ph. F. 和 R. S. 1995 年 9 月于科孚岛

R. S. 2012 年 12 月于巴黎

第 2 版声明

2011 年 3 月，我和我的妻子 Linda 去了一个美丽而遥远的地方旅行，因此有几天没上网。当我找到机会处理邮件时，却得到一个让人震惊的消息：我的挚友和同事 Philippe 突然去世了。由于无法及时赶到巴黎参加葬礼，Linda 和我为我们的好朋友写了悼词，在此分享给本书的读者，让我们一起缅怀我们的这位朋友。

悲痛万分，我在一个遥远的地方缅怀我长期以来的好友和同事 Philippe Flajolet。很遗憾不能亲自参加葬礼，但我坚信，将来一定有很多机会来缅怀 Philippe，希望到时我能参与。

Philippe 才华横溢、匠心独运、求知若渴、不知疲倦，同时慷慨大方、乐善好施，他的生活方式深深地感染了我。他改变了很多人的生活，包括我自己的。随着我们的诸多研究论文变成综述，进而发展成若干篇章，第一本书，第二本书，最终写书成为了我们一生的工作——在跟 Philippe 携手同行时，我与全世界很多学生和合作者都感受到了 Philippe 的真诚和友好。我们曾在全世界各个地方的咖啡馆、酒吧、午餐室或客厅里一起工作。无论何处，Philippe 的工作方式始终不变。我们会先聊聊某个朋友的趣事，然后开始工作。一个眼神的交流，真诚的笑声，抽两根烟，喝两口啤酒，咬两口牛排，吃点薯条，然后听到一声“好……”，我们就这样解决一个又一个问题，证明一条又一条定理。对很多与他合作过的人而言，这些时刻都铭记心中。

这个世界又少了一个聪明高效的数学家。Philippe 的离世意味着很多秘密再也无法揭开。但他给世人留下了很多追随他脚步的继承者，继续追寻他的数学梦想。我们会在会议上为他举杯，我们会以

他的研究成果为基础继续前进，我们的论文会加上一句铭文“谨以此纪念 Philippe Flajolet”，我们会将此一代代传承下去。亲爱的朋友，我们非常想念你，我坚信，你的精神会在我们的工作中得到永生。

我们合著的《算法分析导论》一书的第 2 版就是这样完成的。谨以此书纪念 Philippe Flajolet，也希望更多的人通过这本书继续追寻 Philippe 的数学梦想。

R. S. 2012 年 10 月于罗得岛州詹姆斯敦

TABLE OF CONTENTS

NOTATION

$\lfloor x \rfloor$ *floor function*
largest integer less than or equal to x

$\lceil x \rceil$ *ceiling function*
smallest integer greater than or equal to x

$\{x\}$ *fractional part*
$x - \lfloor x \rfloor$

$\lg N$ *binary logarithm*
$\log_2 N$

$\ln N$ *natural logarithm*
$\log_e N$

$\binom{n}{k}$ *binomial coefficient*
number of ways to choose k out of n items

$\begin{bmatrix} n \\ k \end{bmatrix}$ *Stirling number of the first kind*
number of permutations of n elements that have k cycles

$\left\{ {n \atop k} \right\}$ *Stirling number of the second kind*
number of ways to partition n elements into k nonempty subsets

ϕ *golden ratio*
$(1 + \sqrt{5})/2 = 1.61803\cdots$

γ *Euler's constant*
$.57721\cdots$

σ *Stirling's constant*
$\sqrt{2\pi} = 2.50662\cdots$

CHAPTER ONE

ANALYSIS OF ALGORITHMS

MATHEMATICAL studies of the properties of computer algorithms have spanned a broad spectrum, from general complexity studies to specific analytic results. In this chapter, our intent is to provide perspective on various approaches to studying algorithms, to place our field of study into context among related fields and to set the stage for the rest of the book. To this end, we illustrate concepts within a fundamental and representative problem domain: the study of sorting algorithms.

First, we will consider the general motivations for algorithmic analysis. Why analyze an algorithm? What are the benefits of doing so? How can we simplify the process? Next, we discuss the theory of algorithms and consider as an example mergesort, an "optimal" algorithm for sorting. Following that, we examine the major components of a full analysis for a sorting algorithm of fundamental practical importance, quicksort. This includes the study of various improvements to the basic quicksort algorithm, as well as some examples illustrating how the analysis can help one adjust parameters to improve performance.

These examples illustrate a clear need for a background in certain areas of discrete mathematics. In Chapters 2 through 4, we introduce recurrences, generating functions, and asymptotics—basic mathematical concepts needed for the analysis of algorithms. In Chapter 5, we introduce the *symbolic method*, a formal treatment that ties together much of this book's content. In Chapters 6 through 9, we consider basic combinatorial properties of fundamental algorithms and data structures. Since there is a close relationship between fundamental methods used in computer science and classical mathematical analysis, we simultaneously consider some introductory material from both areas in this book.

1.1 Why Analyze an Algorithm? There are several answers to this basic question, depending on one's frame of reference: the intended use of the algorithm, the importance of the algorithm in relationship to others from both practical and theoretical standpoints, the difficulty of analysis, and the accuracy and precision of the required answer.

The most straightforward reason for analyzing an algorithm is to discover its characteristics in order to evaluate its suitability for various applications or compare it with other algorithms for the same application. The characteristics of interest are most often the primary resources of time and space, particularly time. Put simply, we want to know how long an implementation of a particular algorithm will run on a particular computer, and how much space it will require. We generally strive to keep the analysis independent of particular implementations—we concentrate instead on obtaining results for essential characteristics of the algorithm that can be used to derive precise estimates of true resource requirements on various actual machines.

In practice, achieving independence between an algorithm and characteristics of its implementation can be difficult to arrange. The quality of the implementation and properties of compilers, machine architecture, and other major facets of the programming environment have dramatic effects on performance. We must be cognizant of such effects to be sure the results of analysis are useful. On the other hand, in some cases, analysis of an algorithm can help identify ways for it to take full advantage of the programming environment.

Occasionally, some property other than time or space is of interest, and the focus of the analysis changes accordingly. For example, an algorithm on a mobile device might be studied to determine the effect upon battery life, or an algorithm for a numerical problem might be studied to determine how accurate an answer it can provide. Also, it is sometimes appropriate to address multiple resources in the analysis. For example, an algorithm that uses a large amount of memory may use much less time than an algorithm that gets by with very little memory. Indeed, one prime motivation for doing a careful analysis is to provide accurate information to help in making proper tradeoff decisions in such situations.

The term *analysis of algorithms* has been used to describe two quite different general approaches to putting the study of the performance of computer programs on a scientific basis. We consider these two in turn.

The first, popularized by Aho, Hopcroft, and Ullman [2] and Cormen, Leiserson, Rivest, and Stein [6], concentrates on determining the growth of the worst-case performance of the algorithm (an "upper bound"). A prime goal in such analyses is to determine which algorithms are optimal in the sense that a matching "lower bound" can be proved on the worst-case performance of any algorithm for the same problem. We use the term *theory of algorithms*