TURING 图灵计算机科学丛书

PEARSON Addison Wesley

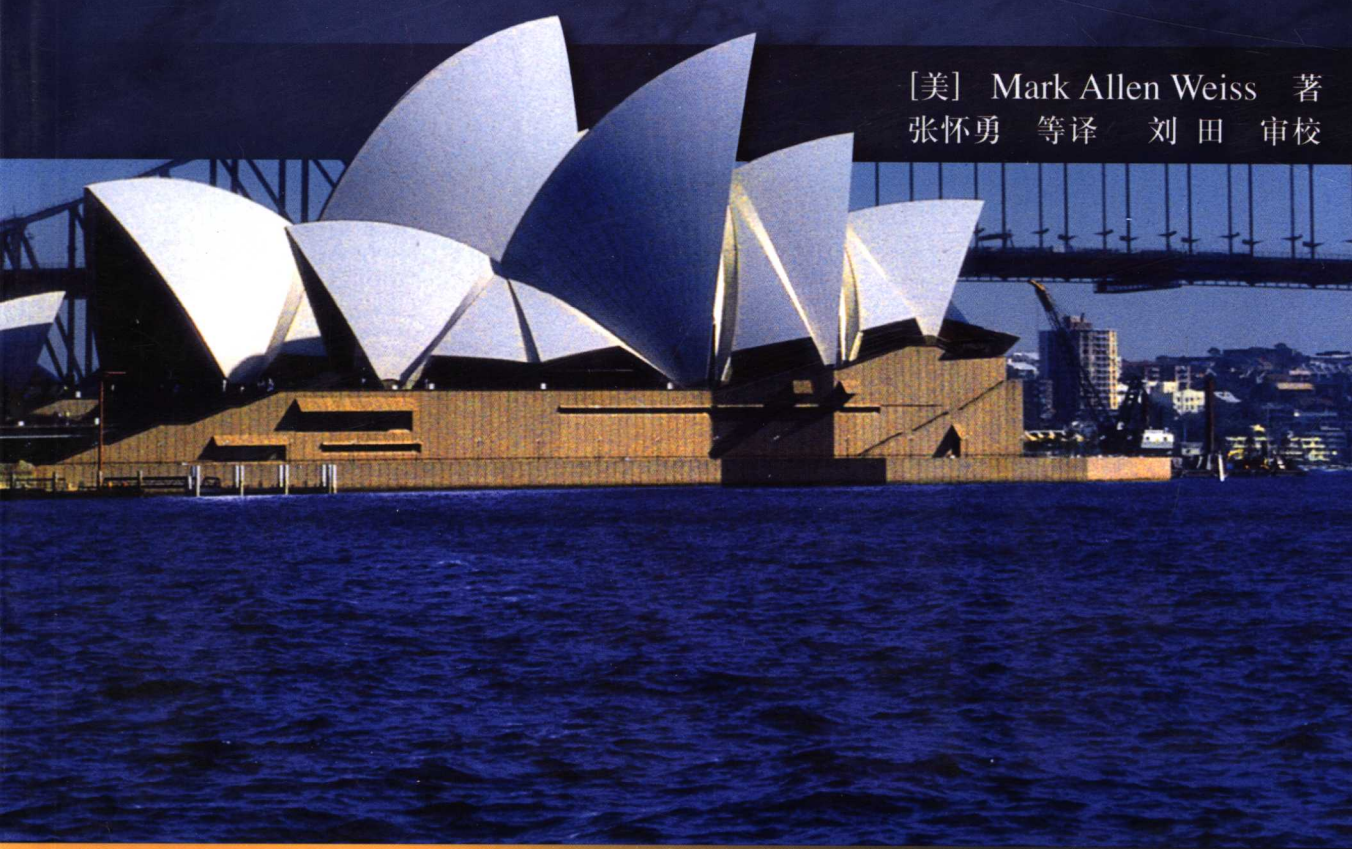
数据结构与算法分析

C++描述

（第3版）

Data Structures and Algorithm Analysis in C++

Third Edition



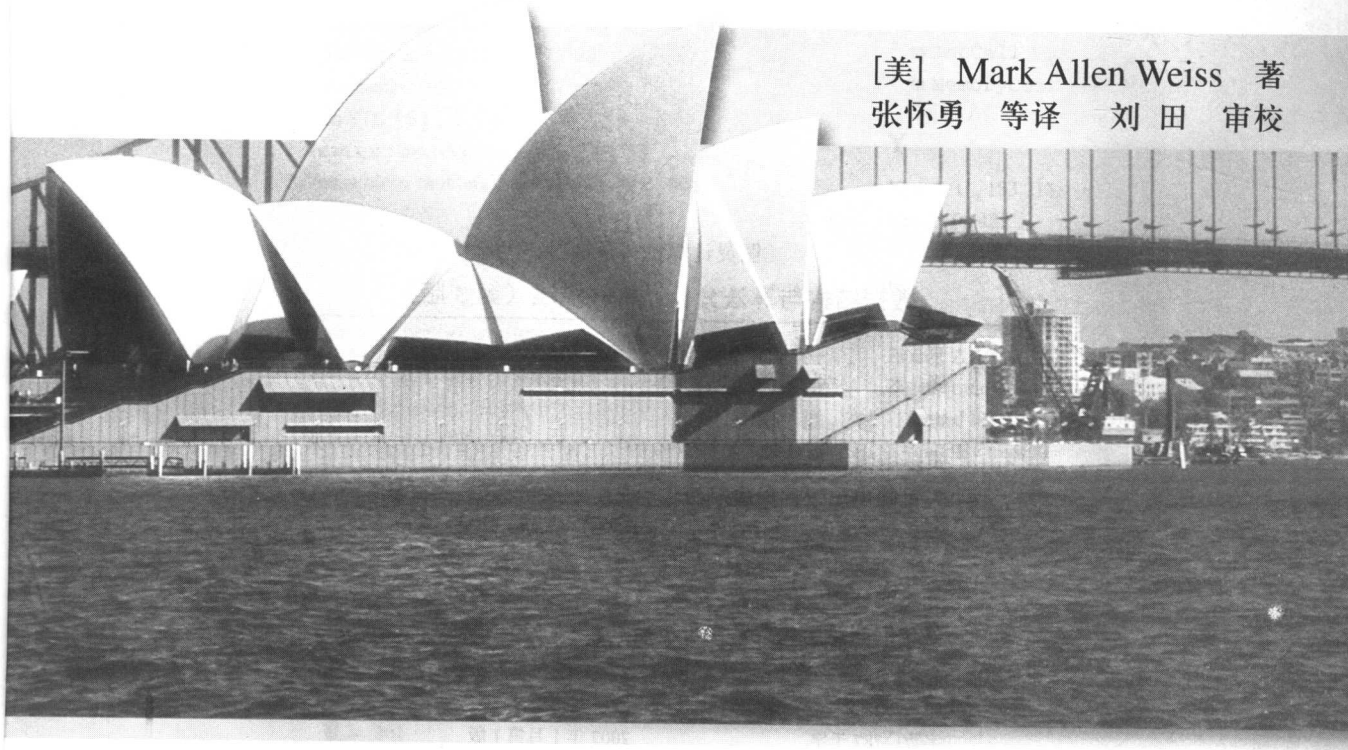
[美] Mark Allen Weiss 著
张怀勇 等译 刘 田 审校

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵计算机科学丛书

数据结构与算法分析
C++描述
（第3版）

Data Structures and Algorithm Analysis in C++
Third Edition



[美] Mark Allen Weiss 著
张怀勇 等译 刘田 审校

人民邮电出版社
北京

图书在版编目（CIP）数据

数据结构与算法分析：C++描述：第3版 /（美）维斯著；张怀勇等译.
—北京：人民邮电出版社，2007.1
（图灵计算机科学丛书）

ISBN 978-7-115-13923-8

Ⅰ.数… Ⅱ.①维…②张… Ⅲ.①数据结构②算法分析③C语言－程序设计 Ⅳ.TP311.12

中国版本图书馆CIP数据核字（2006）第133750号

内 容 提 要

本书是数据结构和算法分析的经典教材，书中使用主流的程序设计语言C++作为具体的实现语言。书的内容包括表、栈、队列、树、散列表、优先队列、排序、不相交集算法、图论算法、算法分析、算法设计、摊还分析、查找树算法、*k*-d树和配对堆等。本书适合作为计算机相关专业本科生的数据结构课程和研究生算法分析课程的教材。本科生的数据结构课程可以使用本书第1章～第9章，多学时课程还可以讲解第10章；研究生算法分析课程可以使用第6章～第12章。

图灵计算机科学丛书

数据结构与算法分析：C++描述（第3版）

◆ 著 [美]Mark Allen Weiss
译 张怀勇 等
审 校 刘 田
责任编辑 杨海玲

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092 1/16
印张：28
字数：787千字 2007年1月第1版
印数：1-4 000册 2007年1月北京第1次印刷

著作权合同登记号 图字：01-2006-3686号

ISBN 978-7-115-13923-8/TP・4910

定价：49.00元

读者服务热线：(010)88593802 印装质量热线：(010)67129223

版 权 声 明

译 者 序

数据结构与算法分析是计算机专业最重要的基础课之一，在计算机科学及相关领域有广泛应用。本书的原文是国外数据结构与算法分析课程的经典教材，有诸多版本在国内引进出版，包括C语言版、Java语言版等，对国内的数据结构与算法分析领域的教学工作有很大的积极影响。

书中论述了数据结构（大量数据的组织方法）以及算法分析（算法运行时间的估算）。书中采用当前流行的面向对象语言C++来描述数据结构和算法，并提供了大部分算法的C++程序和伪代码例程。本书不但详细介绍了当前流行的论题和新的变化，讨论了算法设计技巧，而且在研究算法的性能、效率以及对运行时间分析的基础上考查了一些高级数据结构，从历史的角度和近年的发展对数据结构的活跃领域进行了简要的概括。

本书可作为高级数据结构课程或研究生一年级算法分析课程的教材。本书在教会学生良好的程序设计技巧的同时让学生具备算法分析能力，使得他们能够开发高效的程序。

译者在翻译过程中，参考了天津师范大学冯舜玺老师翻译的本书Java版的译文，而且得到冯老师的诸多帮助，使翻译工作得以尽快完成，在此深表谢意。特别感谢北京大学刘田老师，译稿成文过程中得到他的悉心审阅，提高了本书的翻译质量。我还要借此机会感谢本书的合译者吴怡，感谢她出色的翻译工作。我还想感谢人民邮电出版社图灵公司的编辑们，感谢他们为使最后的散稿成书所做的出色工作。

张怀勇

2006年7月

前　言

目的/目标

本书全面论述了数据结构和算法分析，即组织大量数据的方法和对算法运行时间的估计。随着计算机的速度越来越快，对于能够处理大量输入数据的程序的需求变得日益迫切。具有讽刺意味的是，由于在输入量很大时程序的效率明显降低，因此这又要求更加关注效率问题。通过在实际编程之前对算法进行分析，学生可以确定一个特定的解法是否可行。例如，在本书中学生可看到一些特定的问题，并了解精心的实现如何能够把处理大量数据的时间从16年减至不到1秒。因此，本书中论述的算法和数据结构均进行了运行时间方面的分析。在某些情况下，还研究了影响实现运行时间的一些微小细节。

一旦确定了解法，接着就要编写程序。随着计算机功能的日益强大，它们必须解决的问题也越来越大，越来越复杂，这就要求开发更加复杂的程序。本书的目的是在教会学生使用良好的程序设计技巧的同时让学生具备算法分析能力，使得他们开发的这类程序具有最高效率。

本书适用于本科生的高级数据结构课程或是研究生的算法分析课程。使用本书的学生应该具有中等程度的程序设计方面的知识，包括指针、递归和面向对象程序设计，还要具有离散数学的某些知识。

方法

虽然本书中的内容大部分都是与语言无关的，但是，程序设计还是需要使用某种特定的语言。正如书名指出的，我们为本书选择了C++。

C++已经成为系统编程的主流语言。除了修正了许多C语言的语法方面的缺陷之外，C++还提供了直接结构（类和模板）来实现抽象数据类型的通用数据结构。

撰写本书最困难的部分是确定C++在书中所占的比例。使用太多的C++的特性将使教材变得很晦涩，使用得太少又会和使用支持类的C语言撰写的教材区别不大。

我们采用的方法是以基于对象的方法来阐述文中的内容。这样，本书中几乎没有用到继承性。我们采用类模板来描述通用数据结构。通常情况下尽量避免使用深奥的C++特性，而是使用标准C++中的vector和string类。通过采用C++的现代特性来取代在本书第1版中所用的次级特性简化了很多代码。本书前几版已经实现了类模板，方法是将类模板的接口与其实现分离开。毫无疑问，这是一种好方法，但是这种方法暴露出了编译的问题，读者事实上很难利用这些代码。本版中，在线的源代码将类模板作为一个单元来表示，而不再将接口和实现分离开。本书第1章对书中所用到的C++特性进行了介绍，并阐述了对类模板的处理方法。附录描述了如何重写类模板，用于分离编译。

以Java和C++两种语言描述的数据结构的完全版在因特网上可以得到。我们使用类似的编码习惯使得这两种语言间的相似性表现得更加明显。

第3版中的主要变化

第3版中包含了大量的错误修正，并且书中的大部分章节都经过了修订以提高其可读性。另外，本版还有以下几方面的变化：

- 书中的所有代码都做了更新以适应现代的C++特性。
- 第3章进行了大量的修订，并且讨论了标准vector和list类的使用及其实现。标准vector类在其他数据结构的实现中被广泛使用。
- 第4章修订后包含了关于set和map类的讨论，并通过一个扩展的例子介绍了它们在有效算法设计中的应用。在第9章中也包含了一个使用标准map类来实现最短路径算法的例子。
- 第7章讨论了标准sort算法，其中包括一个关于实现重载的标准sort算法所涉及的技巧。
- 书中提供的源代码都已经进行了简化，从而避免了与类模板的分离编译有关的复杂语法。修订后的代码可以使读者更专心于算法本身，而不是过多地关注C++。

内容概述

第1章包含离散数学和递归的一些内容。我相信熟悉递归的唯一办法是反复不断地看一些好的用法。因此，除第5章外，递归遍布本书每一章的例子中。第1章还介绍了一些C++的内容，包括C++基础知识的回顾以及C++类设计中模板和重要结构的讨论。

第2章讨论算法分析。这一章阐述了渐近分析和它的主要弱点。这里提供了许多例子，包括对对数运行时间的深入解释。通过直观地把一些简单递归程序转变成迭代程序，对它们进行分析。这一章还介绍了更复杂的分治程序，不过有些分析（求解递归关系）要到第7章再详细进行。

第3章包括表、栈和队列。这一章较之以前的版本进行了大量的修订。现在包含了关于STL vector和list类的讨论，包括有关迭代的内容，并且提供了STL vector和list类的重要子集的实现。

第4章讨论树，重点在查找树，包括外部查找树（B树）。UNIX文件系统和表达式树是作为例子介绍的。本章还介绍了AVL树和伸展树。查找树实现细节的更详细讨论可在第12章找到。树的另外一些内容，如文件压缩和博弈树，延迟到第10章讨论。外部介质上的数据结构作为几章中的最后论题来考虑。本版中新增的部分是对STL set和map类的讨论，包括一个讲解了如何使用三个分离的图来高效率地解决问题的例子。

第5章相对较短，主要讨论散列表。这里进行了某些分析，本章末尾讨论了可扩展散列。

第6章是关于优先队列的。这一章还讲解了二叉堆，还有一些附加内容，论述优先队列某些理论上很有趣的实现方法。斐波那契堆在第11章讨论，配对堆在第12章讨论。

第7章是排序。它是关于编程细节和分析的非常特殊的一章，讨论并比较了所有重要的通用排序算法。对插入排序、希尔排序、堆排序以及快速排序这四种算法进行了详细的分析。这一章末尾还讨论了外部排序。

第8章讨论不相交集算法并证明其运行时间。这一章短且特殊，如果不讨论Kruskal算法则这一章可跳过。

第9章讲授图论算法。图论算法之所以重要，不仅因为它们在实践中频繁出现，而且因为它

们的运行时间特别依赖于数据结构的恰当使用。实际上，所有标准算法都是和相应的数据结构、伪代码以及运行时间的分析一起介绍的。为把这些问题放在一个适当的环境下讨论，书中提供了对复杂性理论（包括NP完全性和不可判定性）的简短讨论。

第10章通过考查一般的问题求解技巧来介绍算法设计。这一章含有大量的实例。这里及后面各章使用了伪代码，使学生对一个示例算法的理解不受具体实现细节的困扰。

第11章处理摊还分析。对第4章和第6章的三种数据结构以及本章介绍的斐波那契堆进行了分析。

第12章讨论查找树算法、*k*-d树和配对堆。不同于其他各章，这一章给出了查找树和配对堆完全的仔细的实现。材料的安排使得教师可以把一些内容纳入到其他各章的讨论之中。例如，第12章中的自顶向下红黑树可以和（第4章的）AVL树一起讨论。

第1章～第9章为大多数一学期的数据结构课程提供了足够的材料。如果时间允许，第10章也可以包括进来。研究生的算法分析课程可以使用第7章到第11章的内容。在第11章分析的高级数据结构可以很容易地在前面各章中查到。第9章中对NP完全性的讨论太过简短，以至于不能用于算法分析课程。可以使用论述NP完全性的其他书籍来补充本书的这部分不足。

练习

每章末尾提供的练习与正文中讲授内容的顺序相匹配。最后的练习是把一章作为一个整体来处理而不是针对特定的一节来考虑的。较困难的练习标有一个星号，更难的练习标有两个星号。

参考文献

参考文献位于每章的最后。一般说来，这些参考文献或者是历史性的，代表着书中材料的原始来源，或者阐述对书中给出的结果的扩展和改进。有些文献提供了一些习题的解法。

补充材料

所有的读者都可以在图灵网站www.turingbook.com或www.aw.com/cssupport网站得到下面的补充材料：

- 书中例子的程序代码。

此外，以下材料可从Addison-Wesley的教师资源中心（www.aw.com/irc）获得，但是这部分内容仅针对有资质的教师。教师可以访问Addison-Wesley的教师资源中心或向当地的Addison-Wesley代表申请下面这些资料。

- 书中部分练习的答案。
- 书中的图。

致谢

在本书各个版本的准备过程中，我得到许多人的帮助。有些人在本书的其他版本中提到过，谢谢他们每一位。

同往常一样，Addison-Wesley的专家们使得本书的写作过程更加轻松。我愿意借此机会感谢本书的编辑Michael Hirsch和文字编辑Marilyn Lloyd。我还想感谢Paul Anagnostopoulos和他在Windfall Software的同事，感谢他们的出色工作使最后的书稿成书。特别感谢我的爱妻Jill，感谢她所做的一切。

最后，我还想感谢广大的读者，他们发来电子邮件并指出前面各版中的一些错误和矛盾之处。我的网页http://www.cis.fiu.edu/~weiss将包含更新的（C++的、C的以及Java的）源代码、勘误表以及指向提交问题报告的一个链接。

Mark Allen Weiss
于佛罗里达州迈阿密

目　录

第1章 引　论

本章阐述本书的目的和目标，并且简要复习离散数学和程序设计的一些概念。我们将要了解程序在合理范围内的大规模输入情况下运行性能与其在中等规模输入下的运行性能同等重要。

- 概括本书其余部分所需要的基本的数学基础。
- 简要复习一下递归。
- 概括用于本书的C++语言的一些重要特点。

1.1 本书讨论的内容

设有一组N个数，要确定其中第k个最大者。这称为**选择问题**（selection problem）。对于大多数学习过一两门程序设计课程的学生来说，编写一个解决这种问题的程序不会有什么困难。“显而易见的”解决方法是相当多的。

该问题的一种解法就是将这N个数读进一个数组中，再通过某种简单的算法，比如冒泡排序法，以递减顺序将数组排序，然后返回位置k上的元素。

更好一点的算法可以先把前k个元素读入数组并（以递减的顺序）对其排序。接着，将剩下的元素逐个读入。当读到一个新元素时，如果该元素小于数组中的第k个元素则忽略之，否则就将其放到数组中的正确位置上，同时将数组中的一个元素挤出数组。当算法终止时，返回第k个位置上的元素作为答案。

这两种算法的编码都很简单，建议读者试一试。此时读者自然要问：哪个算法更好？更重要的是，是否两个算法都足够好？使用含有1000万个元素的随机文件，在k=5 000 000的条件下进行模拟发现，两个算法都不能在合理的时间内结束；每种算法都需要计算机处理若干天才能终止（尽管最终给出了正确答案）。在第7章讨论的另一种算法在1s左右就给出答案。因此，虽然所提出的两个算法都是正确的，但是都不能看作是好算法，因为相对于第三种算法能够在合理的时间内处理的输入规模而言，这两种算法都是完全不实用的。

第二个问题是求解一个流行的字谜游戏。输入由一个二维字母数组和一个单词表组成。目标是要找出字谜中的单词。这些单词可能是水平、垂直或沿任何对角线方向放置的。例如，图1-1所示的字谜由单词this、two、fat和that组成。单词this从第一行第一列的位置（即(1, 1)）处开始，并延伸至(1, 4)，单词two从(1, 1)到(3, 1)，fat从(4,1)到(2, 3)，而that则从(4, 4)到(1, 1)。

同样，现在至少有两种显而易见的算法可以求解这个问题。对列出的每个单词，检查每一个有序三元组（行，列，方向）来看看这个单词是否存在。这需要大量嵌套的`for`循环，但基本上是显而易见的算法。

	1	2	3	4
1	t	h	i	s
2	w	a	t	s
3	o	a	h	g
4	f	g	d	t

图1-1　字谜示例

或者这样，对于每一个在字谜界限以内的有序四元组（行，列，方向，字符数），检查所指的单词是否在单词表中。同样，这导致大量嵌套的for循环。如果任意单词中的最大字符数已知，就有可能节省一些时间。

上述两种解法都相当容易编码，并且都能求解通常在杂志上发表的许多实际的字谜游戏。这些字谜通常有16行、16列以及40个左右的单词。然而，假设我们把字谜变成只给出谜板而单词表基本上是一本英语词典，则使用上面提出的两种解法来解决这个问题都需要相当可观的时间，因此这两种方法都是不可接受的。不过，即使单词表很大，这样的问题还是有可能在几秒钟内解决的。

在许多问题中，写出一个可以工作的程序是不够的。如果这个程序是在大规模的数据集上运行，那么运行时间就成了问题。在本书中我们将看到，对于大规模的输入，如何估计程序的运行时间，更重要的是，弄清如何在尚未具体编码的情况下比较两个程序的运行时间。我们还将看到提高程序的运行速度以及确定程序瓶颈的技巧。这些技巧将使我们能够找到需要着重优化的那些代码段。

1.2　数学知识复习

2　本节列出一些需要记住或是能够推导出的基本公式，并且复习基本的证明方法。

1.2.1　指数

$$X^A X^B = X^{A+B}$$

$$\frac{X^A}{X^B} = X^{A-B}$$

$$(X^A)^B = X^{AB}$$

$$X^N + X^N = 2X^N \neq X^{2N}$$

$$2^N + 2^N = 2^{N+1}$$

1.2.2　对数

在计算机科学中，所有的对数都是以2为底的，除非另有声明。

定义1.1　$X^A = B$当且仅当 $\log_X B = A$。

由该定义可以得到下面几个等式。

定理1.1　$\log_A B = \dfrac{\log_C B}{\log_C A}$；$A, B, C > 0$，$A \neq 1$。

证明　令$X = \log_C B$，$Y = \log_C A$，以及$Z = \log_A B$。于是，由对数的定义得$C^X = B$，$C^Y = A$以及$A^Z = B$。联合这三个等式则有$B = C^X = (C^Y)^Z$。因此，$X = YZ$，这意味着$Z = X/Y$，定理得证。　■

定理1.2 $\log AB = \log A + \log B$；$A, B > 0$。

证明 令$X = \log A$，$Y = \log B$，$Z = \log AB$。此时由于假设默认的底为2，所以$2^X = A$，$2^Y = B$，$2^Z = AB$。联合最后的三个等式则有$2^X 2^Y = AB = 2^Z$。因此$X + Y = Z$，这就证明了该定理。 ■

其他一些有用的公式如下，它们都能够用类似的方法推导出来。

$$\log A/B = \log A - \log B$$

$$\log(A^B) = B \log A$$

$\log X < X$ 对所有的$X > 0$成立

$$\log 1 = 0, \log 2 = 1, \log 1024 = 10, \log 1\,048\,576 = 20$$

3

1.2.3 级数

最容易记忆的公式是

$$\sum_{i=0}^{N} 2^i = 2^{N+1} - 1$$

和

$$\sum_{i=0}^{N} A^i = \frac{A^{N+1} - 1}{A - 1}$$

在第二个公式中，如果$0 < A < 1$，则

$$\sum_{i=0}^{N} A^i \leqslant \frac{1}{1 - A}$$

当N趋于∞时，该和趋向于$1/(1-A)$。这些公式是“几何级数”公式。

可以用下面的方法推导关于$\sum_{i=0}^{\infty} A^i$ $(0 < A < 1)$的公式。令S是其和，于是

$$S = 1 + A + A^2 + A^3 + A^4 + A^5 + \cdots$$

于是

$$AS = A + A^2 + A^3 + A^4 + A^5 + \cdots$$

如果将这两个方程相减（这种运算只允许对收敛级数进行），则等号右边1以外的所有项相消，结果得到：

$$S - AS = 1$$

这就是说

$$S = \frac{1}{1 - A}$$

可以用相同的方法计算$\sum_{i=1}^{\infty} i/2^i$，这是一个经常出现的和。把它写成

$$S = \frac{1}{2} + \frac{2}{2^2} + \frac{3}{2^3} + \frac{4}{2^4} + \frac{5}{2^5} + \cdots$$

两边乘以2得到

$$2S = 1 + \frac{2}{2} + \frac{3}{2^2} + \frac{4}{2^3} + \frac{5}{2^4} + \frac{6}{2^5} + \cdots$$

将这两个方程相减得到

$$S = 1 + \frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^5} + \cdots$$

因此，$S = 2$。

4

经常用来分析的另一种级数是算术级数。任何这样的级数都可以用基本公式求其值。

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2} \approx \frac{N^2}{2}$$

例如，为求和 $2+5+8+\cdots+(3k-1)$，将其改写为 $3(1+2+3+\cdots+k)-(1+1+1+\cdots+1)$，显然，这就是$3k(k+1)/2-k$。另一种记忆方法则是将第一项与最后一项相加（和为$3k+1$），第二项与倒数第二项相加（和也是$3k+1$），依次类推。由于有$k/2$个这样的数对，因此总和就是$k(3k+1)/2$，这与前面的答案相同。

下面介绍的两个公式就没有那么常见了。

$$\sum_{i=1}^{N} i^2 = \frac{N(N+1)(2N+1)}{6} \approx \frac{N^3}{3}$$

$$\sum_{i=1}^{N} i^k \approx \frac{N^{k+1}}{|k+1|} \qquad k \neq -1$$

当$k=-1$时，后一个公式不成立。此时需要下面的公式，这个公式在计算机科学中远比在其他数学科目中用得多。数H_N叫作调和数，其和叫作调和和。下面近似式中的误差趋向于$\gamma \approx 0.577\,215\,66$，称为**欧拉常数**（Euler's constant）。

$$H_N = \sum_{i=1}^{N} \frac{1}{i} \approx \log_e N$$

下面两个公式只不过是一般的代数运算：

$$\sum_{i=1}^{N} f(N) = Nf(N)$$

$$\sum_{i=n_0}^{N} f(i) = \sum_{i=1}^{N} f(i) - \sum_{i=1}^{n_0-1} f(i)$$

1.2.4　模运算

如果N整除$A-B$，那么就说A与B模N同余（congruent），记为$A \equiv B \pmod{N}$。直观地看，这意味着无论A或B被N除，所得余数都是相同的。因此，$81 \equiv 61 \equiv 1 \pmod{10}$。与等号一样，若$A \equiv B \pmod{N}$，则$A+C \equiv B+C \pmod{N}$ 以及$AD \equiv BD \pmod{N}$。

有许多定理适用于模运算，其中有些需要用数论来特别证明。本书将尽量少用模运算，前面那些定理就足够了。

5

1.2.5　证明方法

证明数据结构分析中的结论的两种最常见的方法是归纳法证明和反证法证明（偶尔也不得已用唬人法证明[1]，这只有教授们才用）。证明定理不成立的最好方法是举出一个反例。

1. 归纳法证明

由归纳法进行的证明有两个标准的部分。第一步是证明基准情形（base case），就是确定定理对于某个（某些）小的（通常是退化的）值的正确性；这一步总是很简单的。接着，进行**归纳假设**（inductive hypothesis）。一般说来，这意味着假设定理对直到某个有限数k的所有情况都是成立的。然后使用这个假设证明定理对下一个值（通常是$k+1$）也是成立的。至此定理得证（在k是有限的情形下）。

1. 就是用一句高深莫测的“显然”来代替证明。——译者注