“十二五”普通高等教育本科国家级规划教材

数据结构

（第3版）

刘大有 杨博 黄晶 朱允刚 谷方明 姜丽 编著

高等教育出版社

"十二五"普通高等教育本科国家级规划教材

数据结构

（第3版）

刘大有 杨博 黄晶 朱允刚 谷方明 姜丽 编著

高等教育出版社·北京

内容提要

本书系统介绍了数据结构的概念、原理、技术和应用实例，由纸介质部分和在线数字化资源部分所组成，是一部“纸介质教材”和“数字化资源”相辅相成、紧密结合的“新形态教材”。

本书的纸介质部分主要包括数学准备、绪论、基本数据结构、排序与查找等内容，共8章。其中，第1章“数学准备”，系统地介绍与算法分析紧密相关的数学分支(生成函数与渐近表示除外，渐近表示在第2章简要介绍)的基本知识；第2章“绪论”，对算法描述语言ADL和算法书写规范、数据结构与算法的基本概念、算法分析基础等进行阐述；第3、4章介绍线性结构，系统地描述线性表、堆栈、队列、数组和字符串等结构的存储、操作和应用；第5章“树与二叉树”，在详细刻画树和二叉树结构的基础上，从应用和数据结构扩展的视角渐进地讨论线索二叉树、哈夫曼树、并查集和决策树等内容；第6章“图”，系统地阐述图的基本概念、基本存储结构和基本算法，新增了带约束的最短路径算法和功能同Warshall算法但更高效的传递闭包求解算法，从应用的视角讨论复杂网络概念和基于图的典型信息搜索算法；第7、8章“排序”与“查找”，深入讨论排序和查找的重要内容，并给出典型算法的描述、时间复杂性分析和相关算法的比较等。

本书的数字化资源部分主要包括以下几部分：算法的C++程序代码，与ADL算法描述相呼应，为读者上机实践提供方便；习题答案或解题思路；重要内容的讲解视频；较难算法的动画演示程序。这些内容均以数字化形式存于网站，读者使用移动终端扫描纸介质教材上的二维码便可随时随地访问与之对应的数字化资源。

本书可作为高等院校计算机科学与技术、软件工程及相关专业的教材和教学参考书，也可供相关专业的工程技术人员参考使用。

图书在版编目(CIP)数据

数据结构/刘大有等编著.--3版.--北京:高等教育出版社,2017.3

ISBN 978-7-04-046787-1

Ⅰ.①数… Ⅱ.①刘… Ⅲ.①数据结构-高等学校-教材 Ⅳ.①TP311.12

中国版本图书馆CIP数据核字(2016)第280997号

策划编辑 倪文慧　责任编辑 倪文慧　封面设计 于文燕　版式设计 王艳红
插图绘制 杜晓丹　责任校对 刘春萍　责任印制 韩　刚

出版发行	高等教育出版社	网　址	http://www.hep.edu.cn
社　址	北京市西城区德外大街4号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印　刷	河北新华第一印刷有限责任公司		http://www.hepmall.com
开　本	787mm×1092mm　1/16		http://www.hepmall.cn
印　张	22.25	版　次	1999年6月第1版
字　数	500千字		2017年3月第3版
购书热线	010-58581118	印　次	2017年3月第1次印刷
咨询电话	400-810-0598	定　价	43.00元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换

物 料 号　46787-00

数字资源使用说明页

与本书配套的数字课程资源发布在高等教育出版社易课程网站，请登录网站后开始课程学习。

一、注册/登录

访问 http://abook.hep.com.cn/18567916，点击“注册”，在注册页面输入用户名、密码及常用的邮箱进行注册。已注册的用户直接输入用户名和密码登录即可进入“我的课程”页面。

二、课程绑定

点击“我的课程”页面右上方“绑定课程”，正确输入教材封底防伪标签上的20位密码，点击“确定”完成课程绑定。

三、访问课程

在“正在学习”列表中选择已绑定的课程，点击“进入课程”即可浏览或下载与本书配套的课程资源。刚绑定的课程请在“申请学习”列表中选择相应课程并点击“进入课程”。

四、与本书配套的易课程数字课程资源包括案例素材，以便读者学习使用。

账号自登录之日起一年内有效，过期作废。

如有账号问题，请发邮件至：abook@hep.com.cn。

前　言

数据结构与算法紧密相关且两者均非常重要

两个思想改变了世界。一些人认为是约翰内斯·古腾堡(Johannes Gutenberg)在15世纪发明的金属字活板印刷术"typography"改变了世界,据说这一发明受了中国印刷术的影响(早在11世纪的北宋庆历年间,中国的毕昇就发明了胶泥活板印刷术);然而,另外一些人则坚信改变世界的关键原因是算法(algorithms),而非印刷术。

算法就是求解问题的步骤,它必须是确定的(无歧义)、可行的、有限的;此外,它有零个或多个输入,但必须有输出。注意不要将算法和计算机程序等同起来,后者可作为描述前者的手段之一。除了使用算法描述语言描述算法之外,还可用流程图、形式语言、程序设计语言,甚至是自然语言等对算法进行描述。"算法是贯穿在所有计算机程序设计中的一个基本概念"(1997年,著名计算机科学家、图灵奖获得者D.E.Knuth)和"算法是计算机科学的灵魂"(2008年,著名计算机理论科学家Umesh Vazirani)等观点被计算机界的众多学者所认同。

算法学是系统研究算法的一门科学。通常,它主要包括算法设计、算法正确性证明及算法分析三部分。算法设计是创建算法的过程,其主要研究算法结构及良好的创建方法、算法与数据结构间的关系和作用等。算法时空复杂性分析和算法(或其关键部分)正确性证明的重要性自不待言,在算法的时间复杂性分析方面,有时只得到复杂性的阶是不够的,还要给出更精确的分析结果,譬如时间复杂性阶前面的系数。

自20世纪60年代初开始,计算机已更频繁地用于数值只是偶尔出现的一些问题上,对于这些问题主要使用计算机的逻辑判断而不是它的算术运算能力。当然,"非数值分析"实际上是关于这一研究领域的一个极其负面的称谓,最好应有一个正面的名称。"信息处理"对于这一领域好像太宽了,而"程序设计技术"又显得过窄了,称其为"计算机算法分析"似乎恰当一些。

本书的前言本应从数据结构说起,但这里却从算法道来,其原因是数据结构与算法是一对密不可分的"孪生兄弟"。数据结构是计算机算法设计的基础,它在计算机科学中占有十分重要的地位。正如图灵奖获得者霍尔(C.A.R.Hoare)于1972年所阐明的:不了解施加于数据上的算法就不知道怎样去构造数据结构,反之,若不深入研究作为其基础的数据结构,则算法的优美结构与设计将无从谈起。

实际上,不仅数据结构与算法紧密相关,而且两者与数学均有着重要的关系。下面用一个问题求解过程来概要说明数据结构与相关知识的结合应用,以及对数学进行严格训练的必要性。同时编者还期待通过数据结构的学习,使读者对科学研究方法有一个初步的了解。

一个问题求解的大致过程如下。

第1步:运用数学知识(如数学分析、代数、离散数学、组合数学、运筹学、概率论等),对待求

解问题进行数学建模。

第 2 步:运用数据结构、算法设计等知识设计一个好的计算机算法。

第 3 步:运用数学知识(本书第 1 章所涉及的知识等),展开如下子步骤。

子步骤 1　对算法(或算法的关键部分)进行正确性证明。

子步骤 2　对算法进行时空复杂性分析。

子步骤 3　若算法的时空复杂性超出相关要求,则对算法进行优化并返回子步骤 1。

第 4 步:运用程序设计与程序测试知识完成算法的编程与测试,对测试结果给出分析,给出改进算法(包括重选或构造新的数据结构)和/或改进编程等反馈信息。

注:① 对于复杂算法(就整个算法而言)的正确性证明,是迄今仍未解决的难题;② 问题求解过程可能会在上述 4 个步骤和三个子步骤中进行多次重复。

本书撰写思路和主要特点

1. 与《数据结构(第 2 版)》相比在内容上的调整

(1) 精简部分内容

从 2010 年《数据结构(第 2 版)》(以下简称"第 2 版")中,删去了"递归""内存管理""文件"和"随机数"4 章和附录等内容,对"绪论""线性表堆栈和队列""数组和字符串""树""图""排序"和"查找"等各章进行了深度改写。新版紧紧围绕数据结构的核心,着力构建启发式内容,力求阐述严谨,深入浅出,讲解透彻。

(2) 增加部分重要的新内容

新版增加了一些重要的新内容,下面只介绍其中的一部分。

① 算法分析或其关键部分的正确性证明是读者学习数据结构需要掌握的重要技巧,为此增加了新的一章"数学准备"。从 D.E.Knuth 的《The Art of Computer Programming:Vol. 1 Fundamental Algorithms》(Third Edition)中,围绕与算法分析等紧密相关的"数学归纳法""数、幂与对数""和与积""整数函数和初等数论""排列和阶乘""二项式系数""调和数"和"斐波那契数"等数学分支,选择了更为基本的数学知识,并对其中"非显然"的结果(或结论)给出了具体推导过程。特别要指出的一点是,这些对算法学十分重要的数学知识在传统的数学课程中却只进行了较为浅显的讨论。

通过对这些经过选择、补充的数学准备知识的介绍,将为读者阅读、理解书中的算法分析、算法关键部分的证明以及从事算法学研究打下一定的基础。

② 在第 6 章"图"中新增了 6.6.4 节"满足约束的最短路径",给出了其 ADL 描述和实例分析。增加该算法的原因是其应用越来越广泛,虽难度较大,但建立在第 2 版最短路径算法的基础上,可满足读者对学习新知识的渴望,且较易于被他们所掌握。在 6.7.2 节中增加了 Kruskal 算法的 ADL 描述、时间复杂性分析和关键部分的正确性证明,从而使这一内容更加系统与完整。针对第 2 版 5.8.1 节只给出了时间复杂性为 $O(n^3)$ 的 Warshall 算法,新版增加了一个时间复杂性为 $O(n^{2.5})$ 的更高效的传递闭包求解算法,该内容不仅展示了提高算法效率的一种途径,而且

其与 Warshall 算法两者又能作为一个启发式的教学案例。

③ 增加了背景历史介绍，以及推荐读物与参考文献。为使读者能全面了解、掌握相关的内容，激发其学习兴趣，并了解相关的前沿研究成果，本书在大多数章中都增加了与重要知识点相关的背景与历史介绍，以及推荐读物与参考文献，描述了相关知识点的由来及重要应用，和/或给出了相关的应用实例。

此外，将第 2 版附录（算法的 C++代码、习题答案或解题思路）与教师讲课视频、算法动画等以数字化资源存于网站。将纸质教材、网上数字教学资源和移动终端相结合，对重点和难点教学内容提供教学讲解视频和动画演示，用移动终端扫描纸介质教材中的二维码，读者便可进行在线学习。总之，读者可方便、灵活、随时随地选用“纸介质教材”和/或“数字化教材”，从而为其构建一个理想的学习环境，充分满足其个性化需求。

2. 用算法描述语言 ADL 和 C++程序设计语言描述算法

用面向对象的程序设计语言描述施加于数据结构之上的算法，不仅有利于面向对象技术与数据结构知识的结合，而且也为上机实验和课程设计提供了方便。但若换一个视角，这一做法也有不足之处，用高级程序设计语言描述算法不仅要占用较多篇幅，而且还会涉及较多的程序设计细节，算法的要点有可能湮没于过多的细节之中，不利于读者将注意力聚焦于数据结构与算法本身。基于这些考虑，在纸介质教材中采用了一种简洁的算法语言 ADL 描述算法，而将算法的 C++程序代码存于网站，通过移动设备可随时随地进行浏览。

3. 突出启发式教学

针对一些重要知识点，着力设计启发式教学内容。

① 在选择与组织材料时，着力构造由多个求解同一问题的不同算法组成的“算法链条”。假定其由 N 个算法 $A, A^1, A^2, \cdots, A^N$ 组成，其中，A^{i+1} 是对 A^i 做出改进（或重要改进）的算法，或者相对 A^i 来说，A^{i+1} 是一个创新算法（$1 \leqslant i \leqslant N-1$）。在具体介绍算法 A^{i+1} 之前，恰当地进行提示，使多数读者能给出 A^{i+1} 的描述（或关键思想）。

② 在对某算法进行具体描述之前，阐明该算法的关键思想，使部分读者能给出该算法的描绘（或较粗略的描绘）。

③ 对算法涉及的关键难点给出严格证明，这不仅能帮助读者透彻理解算法，而且能培养其严谨的科学精神。

④ 在每章的结尾处给出相关算法的比较，点明对其进一步改进或提升的思路。

⑤ 针对一些较难的算法，设计能帮助读者理解的动画程序。

⑥ 针对一些典型算法，选择与改写恰当的例子，进一步加深读者对算法机理和重要性的理解。

4. 重视算法时空复杂性分析

对经典算法，或给出严格的时空复杂性分析，或给出更具体的分析结果。特别是对具有相同时间复杂性阶的一批相关算法（系指求解同一个问题的不同算法），不但给出算法时间复杂性的阶，而且还给出“阶”前面的系数。只有在考虑这些算法时间复杂性的阶的基础上，再给出“阶”

前面的系数,才能辨识它们的优劣。在不降低算法分析的深度和严谨性的前提下,深入浅出地阐明推导过程。

5. 将科研成果转化为“数据结构”内容

例如,客观世界的很多系统或待解难题都可以抽象成一个图结构。由于这些系统和难题都有很高的复杂性,因此称与之对应的“图结构”为“复杂网络”,诸如社会网络、生物网络、万维网,以及大型电网、交通网和电信网等。从应用的角度,第6章简要介绍了复杂网络的概念,并以万维网为例,深入浅出地介绍了基于图的信息搜索算法 PageRank,该算法是谷歌(Google)搜索引擎的核心算法,被普遍认为是近年来“图结构”最成功的应用实例之一。

上述内容有助于读者深入了解“图结构”的重要性,激发其学习“图结构”的兴趣,启发其面向应用设计出更有效的图算法,并将“图结构”与相关新兴研究领域联系起来。

6. 强调数学上的严格

本书重视内容的严谨性,并试图使读者在这方面受到一定的训练。具体做法主要包括:对书中与某些算法之正确性有关的一些问题,以及与算法复杂性分析或数据结构概念相关的重要定理、引理等都给出了严格的数学证明;对主要概念都试图给出严格的形式化定义。

7. 习题与因材施教

新版每章之后都附加了精选的习题,并且对每道习题都列出了其重要属性及属性值,以使读者更好地了解和选择习题,具体包括:按难度将习题分为5个等级,5级是最高的难度级别;标注解答习题所需的大致平均时间;对用到高等数学知识的习题进行标记;对难度级别≥3的习题,一般给出了提示或分级提示(这些习题通常是有相当难度的题目,建议在读了第一级提示后仍然没有解题思路,才去读第二级提示,依此类推)。

本书编写分工为:刘大有对全书进行了架构设计和统稿,并撰写了第1章、第2章、前言和内容提要;杨博撰写了第6章;黄晶撰写了第3章及第4章;姜丽、朱允刚撰写了第8章;谷方明撰写了第5章;朱允刚撰写了第7章。吉林大学数据结构教学团队的部分教师指导学生对一些较难的算法设计实现了动画程序。朱允刚指导学生进行了本书涉及的数字化资源的网上试验。团队成员虞强源、刘亚波、贾海洋、高滢和赖永等在撰写人二校的基础上又对书稿进行了仔细校对,提出了许多修改建议。对此,我代表撰写组对他们表示由衷的谢意。

鉴于计算机科学技术的飞速发展,因时间和水平所限,书中不足之处在所难免,敬请专家和读者批评指正。

刘大有
2017年2月

郑重声明

目　录

第1章 数学准备

本章将深入介绍算法分析中常用的多种数学公式,以使读者能了解这些公式的含义,并能在算法分析中运用它们。此外,这些数学知识中的一部分对于算法的正确性证明也是十分有用的。

本书关于算法分析的大多数计算都可通过大学的数学知识来完成,具有初等微积分知识的读者将能理解其中几乎所有的数学问题。

在算法分析中涉及的数学技巧通常有独特的风味。譬如,人们经常会对有限个有理数进行求和,或者寻找递推关系。然而这些在算法分析中非常重要的技巧,在传统的数学课程中却只给出较为浅显的讨论。本章将对需要定义的符号和最有用的计算类型与技术作深入的讨论和介绍。

本章从参考文献[1]所阐述的数学归纳法,数、幂与对数,和与积,整数函数和初等数论,排列和阶乘,二项式系数,调和数,斐波那契数等数学分支中选择了更为基本、读者更容易理解的内容,同时对一些“非显然的推导结果”给出了必要的推导步骤。

1.1 数学归纳法

设 $P(n)$ 为关于整数 n 的某个命题;例如,$P(n)$ 可以是“n 乘以 $n+3$ 是一个偶数”,再如,$P(n)$ 是“如果 $n\geqslant 10$,则 $2^n>n^3$”。若想证明对于所有的正整数 n,$P(n)$ 为真,则有一个包括以下两个步骤的重要方法:

① 证明 $P(1)$ 为真。

② 证明“如果 $P(1)$,$P(2)$,$\cdots$,$P(n)$ 都为真,则 $P(n+1)$ 也为真”。

注意:②之证明系指任何正整数 n。

有一个例子,自古代就有许多人独立地发现过下列等式序列:

$$\begin{aligned}1&=1^2\\1+3&=2^2\\1+3+5&=3^2\\1+3+5+7&=4^2\\1+3+5+7+9&=5^2\\&\cdots\end{aligned}\tag{1-1}$$

可将式(1-1)归纳成一般形式:

$$1+3+\cdots+(2n-1)=n^2\tag{1-2}$$

暂把式(1-2)称为 $P(n)$,并证明对所有的正整数 n,$P(n)$为真。遵照上述步骤,有

① 因为 $1=1^2$,故 $P(1)$为真。

② 如果 $P(1)$, $P(2)$,…, $P(n)$都为真,特别是 $P(n)$为真,则式(1-2)成立;在式(1-2)的两边均加上 $2n+1$,得到 $1+3+\cdots+(2n-1)+(2n+1)=n^2+2n+1=(n+1)^2$,这就证明了 $P(n+1)$也为真。

可将这个方法看作一个算法的证明步骤。假如上面的步骤①、②已被证明,那么下面的算法I,对于任意的正整数 n,将给出 $P(n)$的一个证明。

[**算法 I**][1](构造一个证明)给定一个正整数 n,该算法将输出 P(n)为真的一个证明。

I1. [证明 P(1)] 置 k ← 1,并按照①输出 P(1)的一个证明.

I2. [k=n?] 如果 k=n,算法终止. //要求的证明已被输出

I3. [证明 P(k+1)] /* 按照② */输出"如果 P(1), P(2),…, P(k)都为真,则 P(k+1)为真"的证明,并输出"我们已经证明了 P(1),P(2),…,P(k),因此 P(k+1)为真".

//至此,已经验证了当 n=k+1 时,命题为真

I4. [k 加 1] k ← k+1 并转到步骤 I2. ▌

数学归纳法不但包括猜测,而且还包含一个命题的结论性证明。其实,它是无穷多个命题(对于每个 n 都有一个)的一个证明。它之所以被称为归纳法,仅仅是因为在可以应用数学归纳法技术之前,人们必须首先判断出需要证明的究竟是什么。虽然数学归纳法的名字中有"归纳"二字,但是它并非是不严谨的归纳推理法,它属于完全严谨的演绎推理法。事实上,所有数学证明都是演绎法。

这里介绍的归纳法,实际上被称作强归纳法,或第二数学归纳法。

最常见的数学归纳法(亦称第一数学归纳法)是,证明当 n 为任意一个正整数时命题 $P(n)$成立(或曰为真)。其证明过程分成如下两个步骤。

① 证明当 $n=1$ 时命题成立。

② 假设当 $n=m$ 时命题成立,那么可推导出在 $n=m+1$ 时命题也成立(m 代表任意正整数)。

第一数学归纳法的原理在于:首先证明在正整数参数 n 取某个起点值 n_0(正整数)时命题 $P(n_0)$成立,然后证明当 n 从一个值到下一个值的过程有效。当这两点都已经证明,那么对于 n 的任意取值都可通过反复使用这个方法推导出来。

1.2　数、幂与对数

① 整数是包括负的、零和正的全体数:

$$\cdots -3,-2,-1,0,1,2,3\cdots \tag{1-3}$$

② 有理数是两个整数 p 与 q 之比,其中 q 为正整数。实数是一个由十进制数展开的量 x:

---

① 说明:为与程序中变量一致,本书算法中变量统一排正体。

$$x=n+0.d_1d_2d_3\cdots \tag{1-4}$$

其中,n 是一个整数;每个 d_i 是 0~9 之间的一个数字,$d_i\in[0,9]$,且这个数字序列不以无穷多个 9 结尾。式(1-4)的表示意味着对于所有正整数 k,有下式:

$$n+\frac{d_1}{10}+\frac{d_2}{100}+\cdots+\frac{d_k}{10^k}\leqslant x<n+\frac{d_1}{10}+\frac{d_2}{100}+\cdots+\frac{d_k}{10^k}+\frac{1}{10^k} \tag{1-5}$$

③ 数的分类(这里仅考虑实数,不涉及虚数)如图 1.1 所示。

$$\text{复数}\begin{cases}\text{实数}\begin{cases}\text{有理数}\begin{cases}\text{整数}\begin{cases}\text{正整数}\\0\\\text{负整数}\end{cases}\\\text{分数}\end{cases}\\\text{无理数}\end{cases}\\\text{虚数}\end{cases}$$

图 1.1 数的分类

④ 无理数的例子:

$\pi=3.141\ 592\ 653\ 589\ 79\cdots$,系指一个圆中圆周与直径的比值。

$\phi=1.618\ 033\ 988\ 749\ 89\cdots$,黄金比率,即$((1+\sqrt{5})/2)$。

⑤ 对所有实数值 x,定义 b^x。

首先假设 $b>1$。如果 x 由式(1-4)给出,那么有

$$b^{n+\frac{d_1}{10}+\cdots+\frac{d_k}{10^k}}\leqslant b^x<b^{n+\frac{d_1}{10}+\cdots+\frac{d_k}{10^k}+\frac{1}{10^k}} \tag{1-6}$$

这样就把 b^x 定义成唯一的正实数。式(1-6)中右端与左端的差是

$$b^{n+\frac{d_1}{10}+\cdots+\frac{d_k}{10^k}}\times(b^{\frac{1}{10^k}}-1)$$

并且这个差小于 $b^{n+1}(b-1)/10^k$(推导略)。如果将 k 取得充分大,那么可以得到任意希望精度的 b^x。譬如,我们发现

$$10^{0.301\ 029\ 99}=1.999\ 999\ 973\ 9\cdots,10^{0.301\ 030\ 00}=2.000\ 000\ 019\ 9 \tag{1-7}$$

当 $b<1$ 时,定义 $b^x=(1/b)^{-x}$;并且当 $b=1$ 时,有 $b^x=1$。

⑥ 对数。假定给定一个正实数 y,能否找到一个实数 x,使得 $y=b^x$?回答是肯定的,只要 $b\neq1$。因为当给定 $b^x=y$ 时,可简单地反过来使用式(1-6)来确定 n 和 $d_1,d_2,\cdots$。得到的数 x 被称作 y 的关于底 b 的对数,将其写成 $x=\log_b y$,由此又有

$$x=b^{\log_b x}=\log_b(b^x) \tag{1-8}$$

作为一个例子,等式(1-7)表明

$$\log_{10}2=0.301\ 029\ 99\cdots \tag{1-9}$$

由指数律得出：

$$\text{如果 } x>0 \text{ 且 } y>0\text{，则} \log_b(xy)=\log_b x+\log_b y \tag{1-10}$$

$$\text{如果 } c>0\text{，则} \log_b(c^y)=y\log_b c \tag{1-11}$$

等式(1-9)是所谓的常用对数。将二进制对数简记为

$$\log_2 x=\lg x \tag{1-12}$$

式(1-10)、式(1-11)加之下面的换底规则就构成了对数运算的基本规则：

$$\log_c x=\frac{\log_b x}{\log_b c} \tag{1-13}$$

在大多数情况下，底10或底2都不是用于计算的最方便的底。实数 e=2.718 281 828 459 045… 的对数有较简单的性质。以 e 为底的对数被称作自然对数，记为

$$\ln x=\log_e x \tag{1-14}$$

使用得越多，就会越觉得 $\ln x$ 自然。

1.3 和与积

1. 求和

设 $a_1, a_2\cdots$ 是数的任意序列。人们经常对诸如 $a_1+a_2+\cdots+a_n$ 这样的和感兴趣，并用下面两个等价符号中的一个将这个和写得更紧凑：

$$\sum_{j=1}^{n} a_j \text{ 或 } \sum_{1\leqslant j\leqslant n} a_j \tag{1-15}$$

如果 n 为零或负数，则定义这个和数值为0(零)。一般来说，如果 $R(j)$ 表示 j 需要满足的任意关系，记为

$$\sum_{R(j)} a_j \tag{1-16}$$

式(1-16)表示那些 a_j 之和，其中 j 是满足条件 $R(j)$ 的整数。如果不存在满足 $R(j)$ 的 j，则式(1-16)表示0(零)。

一般说来，仅当和数为有限时，即满足 $R(j)$ 的有限个 j 值，并且 $a_j\neq 0$，式(1-16)才被采用。如果要求一个包括无穷多个非零项的序列之和，例如，

$$\sum_{j=1}^{\infty} a_j=\sum_{j\geqslant 1} a_j=a_1+a_2+a_3+\cdots$$

则必须使用微积分技术，式(1-16)就变成式(1-17)：

$$\sum_{R(j)} a_j=\left(\lim_{n\to\infty}\sum_{R(j):-n\leqslant j<0} a_j\right)+\left(\lim_{n\to\infty}\sum_{R(j):0\leqslant j<n} a_j\right) \tag{1-17}$$

假定式(1-17)中的两个极限都存在。如果有一个极限不存在或两个极限都不存在，则无限

和是发散的,就是说式(1-17)的值是不存在的;否则它是收敛的。当在符号∑下放置两个或多个条件时,正像式(1-17)那样,表示所有的条件都必须被满足。

2. 关于求和的4个代数运算

有4个简单的代数运算对求和来说非常重要,熟悉它们才能求解许多问题,亦称之为4条规则。

(1) 关于和数之积的分配率

$$\Big(\sum_{R(i)} a_i\Big)\Big(\sum_{S(j)} b_j\Big) = \sum_{R(i)}\Big(\sum_{S(j)} a_i b_j\Big) \tag{1-18}$$

下面借助一个简单例子让读者得到直观认识,以更好地理解该定律:

$$\begin{aligned}\Big(\sum_{i=1}^{2} a_i\Big)\Big(\sum_{j=1}^{3} b_j\Big) &= (a_1 + a_2)(b_1 + b_2 + b_3)\\ &= (a_1b_1 + a_1b_2 + a_1b_3) + (a_2b_1 + a_2b_2 + a_2b_3)\\ &= \sum_{i=1}^{2}\Big(\sum_{j=1}^{3} a_i b_j\Big)\end{aligned}$$

习惯上,将式(1-18)中的括号去掉:

$$\sum_{R(i)}\Big(\sum_{S(j)} a_i b_j\Big) = \sum_{R(i)}\sum_{S(j)} a_i b_j$$

(2) 改变变量

$$\sum_{R(i)} a_i = \sum_{R(j)} a_j = \sum_{R(p(j))} a_{p(j)} \tag{1-19}$$

式(1-19)表达了两种变换。第一种仅仅是将下标变量的名称从 i 变成 j。在第二种情况中,$p(j)$ 是 j 的函数,它表示相关值的一个排列,确切地说,是对每个满足关系 $R(i)$ 的整数 i,恰好存在满足关系 $p(j)=i$ 的一个整数 j。特别是,对在应用中经常被使用的 $p(j)=c\pm j$ 的重要情况(其中 c 是不依赖于 j 的一个整数),这个条件总是满足的。例如,

$$\sum_{1\leqslant j\leqslant n} a_j = \sum_{1\leqslant j-1\leqslant n} a_{j-1} = \sum_{2\leqslant j\leqslant n+1} a_{j-1} \tag{1-20}$$

(3) 交换求和的次序

$$\sum_{R(i)}\sum_{S(j)} a_{ij} = \sum_{S(j)}\sum_{R(i)} a_{ij} \tag{1-21}$$

$$\sum_{R(i)} (b_i + c_i) = \sum_{R(i)} b_i + \sum_{R(i)} c_i \tag{1-22}$$

在更为一般的情况下,即关系 $S(j)$ 既依赖于 j 又依赖于 i,也经常需要交换求和的次序。对于这种情况,可以用 $S(i, j)$ 表示这样的关系。求和之变换,总可用如下等式表达:

$$\sum_{R(i)}\sum_{S(i,j)} a_{ij} = \sum_{S'(j)}\sum_{R'(i,j)} a_{ij} \tag{1-23}$$

其中,$S'(j)$ 是关系"有一个整数 i 使得 $R(i)$ 和 $S(i, j)$ 两者都为真";$R'(i,j)$ 是关系"$R(i)$ 和 $S(i, j)$ 两者都为真"。例如,假定有一求和是 $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n} a_{ij}$,则 $S'(j)$ 是关系"存在一个整数 i 使得 $1\leqslant i\leqslant n$ 和 $1\leqslant j\leqslant i$",即 $1\leqslant j\leqslant n$;$R'(i,j)$ 是关系"$1\leqslant i\leqslant n$ 和 $1\leqslant j\leqslant i$",即 $j\leqslant i\leqslant n$。因此