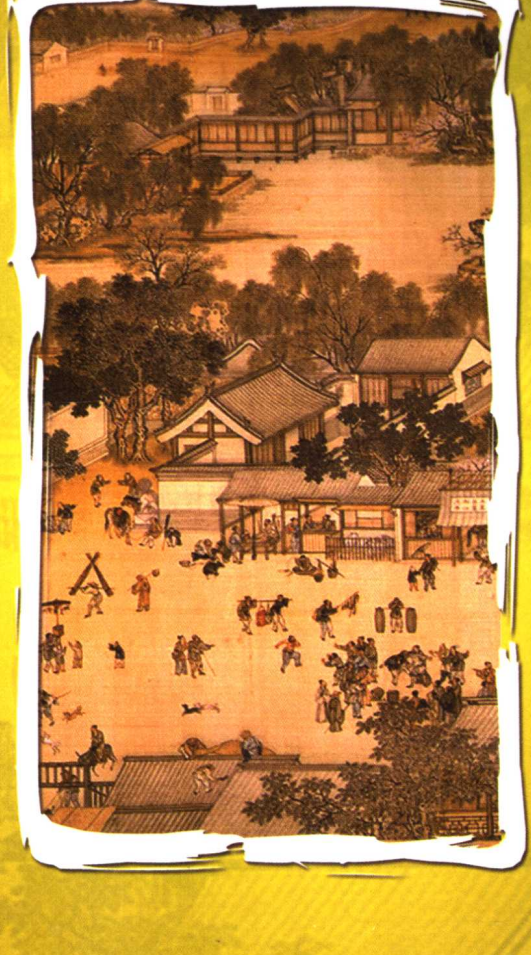
TURING 高等院校计算机教材系列

计算机算法

设计与分析导论

朱清新　杨凡　钟黔川　等编著



人民邮电出版社
POSTS & TELECOM PRESS

TURING 高等院校计算机教材系列

计算机算法
设计与分析导论

朱清新　杨凡　钟黔川　等编著

人民邮电出版社
北　京

图书在版编目（CIP）数据

计算机算法：设计与分析导论 / 朱清新等编著 . —北京：人民邮电出版社，2008.1
（高等院校计算机教材系列）
ISBN 978-7-115-16833-7

Ⅰ. 计… Ⅱ. 朱… Ⅲ. ① 电子计算机－算法设计－高等学校－教材 ② 电子计算机－算法分析－高等学校－教材 Ⅳ. TP301.6

中国版本图书馆 CIP 数据核字（2007）第 143474 号

内 容 提 要

本书为高等学校计算机专业基础课程算法设计与分析教材。全书从算法设计和算法分析的基本概念和方法入手，系统介绍了算法设计方法与分析技巧。全书分为3个部分：第一部分介绍算法的基本概念、算法的数学基础以及算法复杂度分析；第二部分针对排序问题和图的问题，讨论各种已有的算法，并介绍常用的算法设计方法包括分治法、贪心法、动态规划法、回溯法和分支限界法，并介绍了计算的复杂性以及NP完全问题；第三部分讲述并行计算模型和并行算法设计技术。书中每章后面都附有一定数量的习题，帮助读者理解和掌握书中的内容。

本书适合作为计算机以及相关学科高年级本科生及研究生算法设计与分析课程的教材和参考书，同时也可作为算法研究者的参考书。

高等院校计算机教材系列

计算机算法：设计与分析导论

◆ 编　　著　朱清新　杨　凡　钟黔川　等
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
三河市海波印务有限公司印刷
新华书店总店北京发行所经销

◆ 开本：800×1000　1/16
印张：18
字数：424 千字　　2008 年 1 月第 1 版
印数：1–4 000 册　　2008 年 1 月河北第 1 次印刷

ISBN 978-7-115-16833-7/TP

定价：35.00 元

读者服务热线：(010)88593802　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

算法设计与分析不仅是计算机应用领域必不可少的基础知识，也是计算机科学与技术专业学生的必修课程，对于从事计算机系统结构、系统软件和应用软件研究与开发人员来说同样是非常重要的一门学科。本书通过对计算机算法设计与分析理论的系统介绍与讨论，有助于读者理解和掌握算法设计的主要方法，培养分析算法的计算复杂性的能力，为独立地设计算法和对已有算法进行理论分析打下基础。

算法设计与分析学科所涉及的领域非常广泛，通常包括下面几方面的内容：一是迄今为止人们所设计的各种基本的和经典的算法，如排序、搜索、图的算法、组合算法、序列比对算法和大量的数值计算算法；二是关于算法分析和算法设计策略、可计算性理论和问题复杂性等方面的理论研究，如计算模型、NP 完全问题和问题复杂度下界等理论；三是近年来在随机算法、近似算法、加密算法、分布式并行算法、智能优化算法以及其他算法新领域方面的最新研究成果。

作为“算法设计与分析”课程的本科生及研究生教材，本书在内容组织安排上体现理论与实际应用并重的原则，兼顾串行算法与并行算法两大部分。

全书共分为 11 章，内容安排如下：第 1 章介绍算法的基本概念，包括算法的定义和特征、问题的形式化定义、算法的效率和复杂度的定义以及算法的数学基础；第 2 章讨论常用的算法设计与分析技术，包括算法复杂性的度量和算法的渐近增长率分析，算法的平均复杂度及其最差情况复杂度，算法的优化与最优算法的特征，以及常用的算法设计策略，如分治法、回溯法、贪心法、动态规划等方法的原理和应用实例；第 3 章讨论排序问题，针对各种经典的排序算法分类，重点分析 4 类具有代表性的排序算法的特点，即基于相邻元素比较的排序算法；基于分治法策略的排序算法，基于分配（映射）的排序算法和基于最大堆结构的排序算法；第 4 章讨论与图有关的算法，主要对图的搜索（遍历）问题、最短路径和最小生成树问题、欧拉回路问题和网络流理论以及在各个领域中的应用进行讨论；第 5 章介绍 NP 完全性理论，包括图灵机模型、Cook 定理、NP 完全性证明、近似算法和 DNA 计算等基本概念和主要研究成果。

从第 6 章开始至第 11 章是关于并行计算的内容。第 6 章介绍并行计算的基本概念，包括并行计算机体系结构和通信机制、并行计算模型；第 7 章讨论并行算法设计技术，包括串行算法的并行化方法和并行算法设计的 PCAM 方法论；第 8 章讨论并行算法的效率分析，包括并行算法的性能指标和效率分析；第 9 章介绍各种计算模型上的并行求和与排序算法；第 10 章介绍几种并行数值算法，包括矩阵运算、线性方程组求解、快速傅里叶变换和离散小波变换并行算法；第 11 章简单介绍常见的并行程序设计工具和并行程序设计语言 HPF。

本书是在电子科技大学开设的“算法设计与分析”课程讲义的基础上，参考了近年来国内外

多种算法设计与分析的优秀教材和大量论文编写而成的。全书由朱清新（前言，第 1 章至第 3 章、第 7 章和第 8 章）、杨凡（第 6 章和第 9 章）、钟黔川（第 10 章和第 11 章）、王伟东（第 4 章）、曹涌（第 5 章）共同编写，并由朱清新统稿。本书的写作还得到了许多研究生的协助，彭博整理了第 2 章和第 3 章的部分资料，匡平和卿利搜集整理了第 9 章的部分资料，赵怀玉、白勇、杨懿和程勇新改写并重新绘制了第 4 章、第 7 章和第 8 章的部分算法和插图，此外，还有其他学生提供了帮助，作者在此向他们表示衷心的感谢。

本书编写工作得到了电子科技大学研究生教材建设项目的资助，作者在此表示诚挚的谢意。

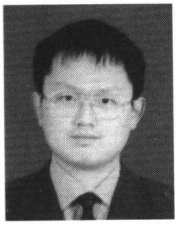
由于作者水平有限，书中定有许多不妥及谬误之处，敬请读者批评指正。

朱清新

2007 年 6 月于成都

作 者 简 介

朱清新 1982年1月毕业于四川师范大学数学系，获学士学位；1984年7月毕业于北京理工大学基础科学部应用数学专业，获硕士学位；1984年8月任西南技术物理研究所工程师、副研究员；1988年8月赴加拿大留学；1993年3月获渥太华大学数学系控制论专业博士学位；1993年5月至1996年3月在渥太华大学电子工程系和加拿大卡尔顿大学计算机学院从事博士后研究兼攻读计算机硕士（二学位）；在此期间曾任加拿大Nortel公司和OmniMark公司高级研究员；1998年3月回国应聘到电子科技大学（UESTC）工作，1999年聘为教授，2001年聘为博士生导师；2002年9月至2003年3月赴加拿大蒙特利尔Concordia大学计算机系任高级访问学者。现任电子科技大学计算机学院学术委员会主任，计算运筹学研究室主任。主要研究领域为计算机图形与视觉、生物信息技术、最优化与仿真技术。主要学术兼职有：美国数学学会（AMS）会员，中国计算机学会（CCF）高级会员、信息存储专业委员会委员，四川省计算机学会多媒体专业委员会主任。主持国家和省部级科研项目10余项。在国内外学术刊物上发表论文100多篇，出版专著3本，其中《离散和连续空间中的最优搜索理论》一书入选“华夏英才基金学术文库”。



杨凡 广东梅县人。2001年毕业于重庆大学，获得工学硕士学位，随后在迈普通讯股份有限公司工作，负责路由器、交换机上的路由协议软件的开发。现在电子科技大学计算机学院攻读计算机应用技术专业博士学位，主要研究领域为蛋白质序列分析和结构预测。

钟黔川 重庆市江津人。1993年毕业于四川师范大学物理系，获学士学位；2001年毕业于电子科技大学应用物理系光学专业，获硕士学位；2003年9月至今于电子科技大学计算机科学与工程学院攻读计算机应用技术专业博士学位。现任西昌学院信息技术系讲师，主要研究方向为信息安全、混沌密码、数字水印技术，以第一作者身份在国内外期刊上发表多篇论文。

目　录

第1章 引 论

1.1 算法的基本概念

计算机科学的主要研究内容就是如何通过计算机来把一个给定的输入转化为所需要的输出，其中算法分析与设计学科所研究的主要内容是怎样高效地进行这种转化，而如何使转化的过程更加规范化和易于管理则是软件工程学所关注的问题。

在20世纪40年代计算机问世以前，数学家们就在积极地从事算法研究。那时候的算法通常表示为一系列简单运算步骤的集合，这些运算步骤用于计算或求解一个数学问题。在早期的算法研究中，人们的研究重点集中在可计算性理论上。这个理论描述了在算法上可解的问题的特征，并且确定哪些是不可解的问题。1936年，阿兰·图灵（Alan Turing）在他的一篇划时代的文章中给出了一个重要的结果，他证明了所谓“停机问题”（halting problem）是不可解的[①]。停机问题是指对于任意给定的一个计算机算法（或程序）和所有可允许的输入，确定程序是否能够最终结束运行（即停机）。换句话说，就是确定该程序会不会陷入死循环。阿兰·图灵的结果表明了，停机问题是不可能用算法来求解的，即不存在用来解决这个问题的计算机程序。

我们说一个问题在算法上是可解的，意思是可以编写一个计算机程序，在运行时间足够长和存储空间足够大的条件下，它对任何可容许的输入都可得出正确的答案。虽然可计算性理论对于现代计算机科学的发展产生了深刻的影响，但是仅仅从理论上证明一个问题是可计算的并不能保证在实际中也一定是可解的。例如，我们可以很容易地编写出一个完美的下棋程序，因为所有棋子在棋盘上的全部走法是有限的；所有这些棋子走法的组合差不多有10^{50}种，用这样的程序下棋可能需要运行几千年才会得出结果，因此这个下棋程序在实际中是行不通的。

下面我们先给出算法的一个非正式的定义。

定义1.1 *算法是以一步接一步的方式来详细地描述计算机如何将输入转化为所要求的输出的过程，或者说，算法是对计算机上执行的计算过程的具体描述。*

① Alan Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, Series 2, 42 (1936), pp 230-265. 在这篇划时代的论文中，图灵提出了图灵机的概念，给出了停机问题的定义并且证明了它是不可解问题。

此外，一个算法还必须具备以下性质：

(1) 算法首先必须是正确的，即对于任意的一组输入，包括合理的输入与不合理的输入，总能得到预期的输出。如果一个算法只是对合理的输入才能得到预期的输出，而在异常情况下却无法预料输出的结果，那么它就不是正确的。

(2) 算法必须是由一系列具体步骤组成的，并且每一步都能够被计算机所理解和执行，而不是抽象和模糊的概念。

(3) 每个步骤都有确定的执行顺序，即上一步在哪里，下一步是什么，都必须明确，无二义性。

(4) 无论算法有多么复杂，都必须在有限步之后结束并终止运行，即算法的步骤必须是有限的。在任何情况下，算法都不能陷入无限循环中。

一个问题的解决方案可以有多种表达方式，但只有满足以上4个条件的解才能称之为算法。虽然算法和计算机程序紧密相关，但二者也存在区别：计算机程序是算法的一个实例，是将算法通过某种计算机语言表达出来的具体形式；同一个算法可以用任何一种计算机语言来表达。

如果把算法的定义归纳为用计算机来解决问题的一个详尽的有限步骤，那么问题的确切定义是什么呢？通常我们所说的“问题”是一个非常宽泛而模糊的概念，并且在不同的场合有着不同的解释。比如说：“这辆车出了点问题”，这里所说的“问题”指的是故障，而“我们正在研究这个问题”，则是指一个研究课题，或者是一种未知的对象。在计算机科学研究领域里，问题（计算问题）被定义为一个需要执行或实现的任务。如果从不同角度来看，问题的确切定义可以采用不同的形式化定义方法。例如，从数学角度来看，可以把问题定义为一个数学函数，函数的定义域是该问题所有可能的输入的集合，函数的值域是该问题的全部输出的集合；而问题就是从输入集合到输出集合的一个映射关系，即把定义域中的任意一个元素，也就是任意一个输入，映射到值域中的一个点，即对应的输出。

在用函数来定义问题时，我们把一个问题的输入集合中的元素，即输入变量所取的值，称为函数的**参数**或**实参**，而输入集合（输入变量）本身则称为函数的**形参**。这种说法也用在计算机程序设计中，我们把子程序称为函数，函数的输入变量所取的值称为参数或实参，而输入变量本身则称为函数的形参。

作为问题定义的函数应该是确定性的函数，而不是模糊的、随机的函数。也就是说，对于每一个确定的输入，无论该函数运行多少次，其输出值总是确定的、唯一的，不能有任何随机性和模糊性。这个特征对于我们定义的计算问题十分重要。

如果把数学函数的定义进一步抽象化，则可以把问题 Q 定义为两个集合 I 和 S 的乘积空间 $I \times S$ 上的一个二元关系，即 $Q \subset I \times S$。其中集合 I 叫作问题**实例**（instance）的集合，由所有的输入值组成；集合 S 叫作问题的**解集**，相当于输出的集合；而抽象化的问题 Q 就是集合 I 与集合 S 的乘积空间的一个子集。

以数学函数的概念来定义计算问题的优点是清晰严密，但是没有突出计算机科学领域的特点。计算机科学领域的问题的输入比较复杂，不像一般数学函数的输入通常是单个的点，如实数、复数或二维、三维空间的点等。举例来说，我们所面临的问题的输入可能是由许多信息组成的集

合，比如一张表格或其他复合类型的数据。所以从计算机科学的角度来看，更倾向于把问题理解成一组输入和输出，以及存在于它们之间的一个变换关系，即输入和输出是按照一定的方式联系起来的。除此之外，问题的定义中还应该包括相关的资源配置约束条件，即这个问题的任何可接受的解决方案所必需消耗的计算资源。如果没有特别说明，一般是指解决这个问题所需要的时间，有时也包括其他计算资源，如存储空间或通信资源等。

计算机科学领域中的问题大致可分为以下3种类型。

(1) 判定性问题：这类问题的输出是给出一个是与否的判断。例如，连通性问题、回路问题、查找与排序问题以及字符串匹配等。

(2) 最优值或最优化问题：这类问题是在所有可能的解中求出最优解。例如，求函数的最大值、最短路径问题以及最小生成树问题等。

(3) 数值计算问题：这类问题是在一定的约束条件（如精度范围）下求近似解。例如，解方程组和矩阵运算等。

在本书后面的几章中，我们将详细讨论这几类问题的例子。

解决一个问题的算法不是唯一的，同一个问题可能有许多算法。算法分析的任务就是估计算法的时间和空间复杂度，分析和比较各种算法的优劣。而算法设计的任务有两个：第一是设计容易理解、容易编程实现且容易调试的算法；第二是使算法能够有效地使用计算机资源，减少计算机的工作量，即节省时间、空间和计算机硬件资源。这两个目标通常是相互冲突的。在针对具体问题设计算法时，必须适当地折中考虑，在高效率和易理解性之间取得平衡。

下面简单介绍一些常用的算法设计方法。

(1) 分治法（divide and conquer）。分治法的思想是把问题分解成同类型的一些子问题，每个子问题又分成更多子问题，最后通过子问题的解求出原始问题的解，所以分治法的技术通常是导致一个递归算法。使用分治法设计的算法有二分查找算法、快速排序算法以及归并排序算法等。

(2) 贪心法（greedy method）。贪心法是根据一个事先确定的判定准则，每次都选取最优的部分解添加到当前的局部解集中，最后由一系列的局部最优解组成全局解。用贪心法设计的算法有选择排序算法以及图的最小生成树算法等。贪心法的实质是在面临多种选择时，永远选择最优的一个。

(3) 回溯法（back tracking）。回溯法的思想与老鼠走迷宫有点类似。迷宫中有很多岔路，老鼠不知道哪条路可以走出去，那它应该采取什么策略呢？它可以一直往前走，直到没路可走时为止，然后退回到上一个岔路口，重新选择一条没走过的路继续往前走。如此下去，老鼠总能找到一条走出迷宫的路，除非迷宫里没有路可以走出去。这种方式就是应用回溯法的一个典型例子。使用回溯法设计的算法有关于图的遍历问题的深度优先搜索算法等。

(4) 动态规划（dynamic programming）。动态规划技术是20世纪70年代中由美国数学家Bellman为解决最优控制问题而提出的，其主要思想是最优化原则，即用一系列子问题的最优解去构造全局的最优解。使用动态规划技术设计的算法有最短路径算法以及矩阵链乘积算法等。

1.2 算法的数学基础

本节介绍算法设计和分析中常用的一些数学基础知识，包括集合论、逻辑学、概率论以及代数领域的知识。

1.2.1 集合论

集合论是算法分析中用得最多的数学工具。后面在讨论算法的增长率时就定义了3个集合O、Θ和Ω。例如，对于某个给定的函数$f(n)$，一个算法属于$O(f(n))$，就表示在n趋于无穷大时这个算法的时间复杂度小于或等于$f(n)$，也就是它的增长速度比$f(n)$慢。集合Θ和Ω也有类似的意义。集合就是将互不相同的元素集中起来，通常这些元素都是同一种类型的，具有某些共同的性质。例如，整数集合的元素都是整数，实数集合的元素都是实数，等等。一个元素e是集合S的成员，用符号$e \in S$表示，读作e在S中或e属于S。

一个特定的集合可以用列举或描述的方法给出。列举就是将集合里的元素排列在花括号中，比如，$S_1 = \{a,b,c\}$。对于那些有很多个或无限多个元素的集合，就不能采用列举的方法来给出，描述这类集合可以采取把集合中的元素所满足的条件写出来的方式，例如，$S_2 = \{x \mid x$是整数的平方$\}$，读作“所有元素x都是整数的平方的集合”。有一个特殊的集合叫空集，用Φ表示。空集中没有任何元素，但是它的意义非常重要。后面我们将会看到，如果没有空集，那么许多有关集合的运算都会无法进行。

集合中的元素是没有先后顺序的。因此前面例子中S_1也可以表示为$\{b,c,a\}$或者$\{c,a,b\}$等。如果集合S_1的所有元素都在集合S_2中，那么S_1称为S_2的**子集**，用$S_1 \subseteq S_2$表示。而S_2称为S_1的**超集**，用$S_2 \supseteq S_1$表示。一个集合也是自身的子集和超集，空集是任何集合的子集。

下面定义集合的运算。设S和T是任意两个集合，S和T的交集定义为

$$S \cap T = \{x \mid x \in S \text{ 且 } x \in T\}$$

即交集$S \cap T$是由S和T中相同元素组成的集合。S和T的并集定义为

$$S \cup T = \{x \mid x \in S \text{ 或 } x \in T\}$$

即并集是所有S的元素和T的元素组成的集合。S和T的差集表示为

$$S \setminus T = \{x \mid x \in S \text{ 但 } x \notin T\}$$

即差集是所有属于S但不属于T的元素组成的集合。

如果存在一个由有限多个整数构成的集合$N = \{n_1, n_2, \cdots, n_k\}$，$N$中的元素和$S$中的元素有着一一对应关系，那么称集合$S$是有限集。$S$的**基数**或**势**（cardinality）等于k，表示为$|S| = k$。任意一个有n个元素的有限集的全部子集（包括空集）的数目为2^n，其中基数为k的子集的个数为

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

一组具有确定顺序的元素称为一个**序列**。除了元素之间具有确定的顺序之外，序列与集合的

不同之处还在于序列中的元素是可以重复的。与集合的描述方式类似，也可以用列举或给出通项公式的方法表示一个序列，列举的方法是将序列中的元素用一对圆括号括起来。例如，$S_1=(a,b,c)$，$S_2=(b,c,a)$，$S_3=(a,a,b,c)$。

如果一个序列与前n个正整数的序列（1, 2, 3,…, n）中的元素有一一对应关系，则称此序列是有限的。如果有限序列中的元素互不相同，那么称这个序列是一个有限集合的**置换**或**排列**（permutation）。一个有n个元素的集合有$n!$个互不相同的排列。

一个具有k个元素的有序序列有时也称作k元组。例如，二元组（有序数对）(x, y)，三元组(x, y, z)等。一个k元组也表示由k个集合的叉乘得到的（k维）乘积空间中的点。这里定义两个集合S和T的叉乘为

$$S\times T=\{(x,y)\mid x\in S, y\in T\}$$

乘积空间$S\times T$中的点是有序数对。显然，乘积空间$S\times T$的基数为$|S\times T|=|S|\cdot|T|$。常见的一种乘积空间是由相同的集合叉乘得到的，例如$R\times R$，或写为R^2，表示二维欧氏空间（实平面）。

一个n元关系定义为n维乘积空间中的一个子集。这个子集可以是有限的也可以是无限的，可能是空集也可能是整个乘积空间。n元关系中最重要的例子是二元关系$R\subseteq S\times T$。例如，定义在自然数集上的“小于”关系，可以表示为$R=\{(x,y)\mid x\in N, y\in N, x<y\}$。又如，设$P$是所有人的集合，我们可以定义一个“亲子关系”$R\subseteq P\times P$，$(x,y)\in R$表示$x$是$y$的双亲之一，因此$R$是$P\times P$的一个子集。虽然很多二元关系中的两个元素都是同一种类型的，但是这并不是必需的。特别地，集合$\{(x,y)\mid x\in S, y\in T\}$也是一个二元关系。

当关系$R\subseteq S\times S$，即R中的每个关系的两个元素都来自于同一个集合S时，这些二元关系可能具有以下一些重要的性质。

- 自反性，对所有$x\in S$，有$(x,x)\in R$。
- 对称性，若$(x,y)\in R$，则有$(y,x)\in R$。
- 反对称性（不满足对称性），若$(x,y)\in R$，则有$(y,x)\notin R$。
- 传递性，若$(x,y)\in R$且$(y,z)\in R$，则必有$(x,z)\in R$。

例如，上面讨论过的“小于”关系具有反对称性和传递性。又如，“等于”关系满足自反性、对称性和传递性，但不满足反对称性。

如果一个二元关系同时满足自反性、对称性和传递性，那么这个关系称为等价关系，记作“$\cong$”。等价关系是一类十分重要的关系，因为通过它可以将下层集合S划分为若干个互不相交的子集，每一个子集叫作一个等价类。例如，$[x]=\{y\in S\mid x\cong y\}$表示$S$中所有与$x$等价的元素的集合。根据等价关系的传递性，等价类中的所有元素都相互“等价”。

例1.1 （同余关系）令N是全体正整数的集合，对任意的$n\in N$，定义$R=\{(x,y)\mid x\in N, y\in N$，且$(x-y)$能够被$n$整除$\}$，容易验证$R$是$N$上的一个等价关系，称之为同余关系。全体正整数按$n$的同余关系划分为$n$个同余类，每个类中的数除以$n$得到的余数都相同，所以具有相同余数的数都相互等价。

我们还可以用二元关系的概念给出数学函数的另一个定义：函数$f(x)$是一个二元关系$R = \{(x, y) \mid x \in S,\ y \in T,\ 且y = f(x)\}$。值得注意的是，函数关系定义中的$S$中的元素不能重复出现，因为$S$相当于函数的定义域，而$T$是函数的值域，定义域中的点不能有重复，而值域中的点可以重复。

许多二元关系问题都可以归结为图论的问题。图论问题包含了很大一类具有挑战性的算法问题。比如，在一个包含许多相互依赖的工作的大项目中，我们会遇到许多诸如“工作x的开展依赖于工作y的完成”的情况；在工作人数固定的条件下，怎样安排工作时间表，使完成工作的时间最短等问题。第4章会研究很多这样的问题。

1.2.2 逻辑学

逻辑学是用形式语言来规范自然语言叙述的系统方法，是使我们可以更精确地进行推理和推导的工具。在逻辑学中最简单的命题叫作原子公式，复杂的命题可以通过原子公式和逻辑连接符来表示，称为逻辑表达式。逻辑表达式可以通过真值表来加以验证。**真值表**是一种由逻辑变量的所有可能取值组合及其对应的逻辑函数值所构成的表格。例如，函数$F = \neg B \wedge A + \neg A \wedge C$的真值表如表1-1所示。

表1-1 真值表

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

常用的逻辑连接符包括∧（与）、∨（或）和¬（非），这3个符号也叫作**布尔运算符**。还有一个常用的符号是⇒，$A \Rightarrow B$表示从事件A可以推出事件B，即如果A为真，就一定有B为真。不过这个运算符并不是一个新的布尔运算符，因为$A \Rightarrow B$等价于$\neg A \vee B$，这个逻辑恒等式可以通过真值表加以验证。

另外两个非常有用的恒等式称为德·摩根（DeMorgan）定律：

$$\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$$
$$\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$$

这里用符号⇔来表示“逻辑上等价于”这个概念。德·摩根定律的证明可以用类似于集合论中证明集合关系的方法，以第二个公式的证明为例，如图1-1所示。用文氏图的表示方法：阴影部分表示属于A或属于B的点组成的集合（$A \vee B$）。阴影部分以外的区域表示集合$A \vee B$的取非$\neg(A \vee B)$。$\neg A$是整个空间中除去A剩下的部分，$\neg B$是整个空间除去B剩下的部分。$\neg A \wedge \neg B$是这两部分的交集，它正好等于阴影部分以外的区域。

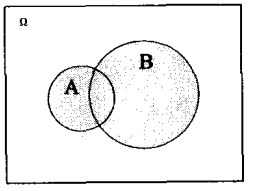


图1-1 证明德·摩根定律的文氏图

限制词all和some也是两种重要的逻辑连接符。all叫作整体性限定符，记作$\forall$；$\forall x$读作“对所有x”。some叫作存在性限定符，记作$\exists$；$\exists x$读作“存在x”。这些连接符可以应用到含有x的叙述中。例如：

- 整体性限定（包含全体的点）——$\forall x\, P(x)$为真，当且仅当$P(x)$对所有x都成立。
- 存在性限定——$\exists x\, P(x)$为真，当且仅当对某个x有$P(x)$成立。
- 普遍的隐含关系——$\forall x(A(x) \Rightarrow B(x))$为真，当且仅当对所有$x$，如果$A(x)$成立则$B(x)$成立。

整体性限定和存在性限定的逻辑关系可以相互转化。$\forall x\,P(x)$表示对所有x，$P(x)$都成立，逻辑上等价于不存在x，使得$P(x)$不成立，即$\neg\exists x(\neg P(x))$。$\exists x\,P(x)$表示存在x使$P(x)$成立，逻辑上等价于不是所有x都使$P(x)$不成立，即$\neg\forall x(\neg P(x))$。

证明一个逻辑论断除了前面提到的真值表的方法以外，还可以通过已经证明了的逻辑等价关系或逻辑恒等式来完成。下面再给出几个重要的逻辑恒等式：

(1)（反证法）$A\Rightarrow B \Leftrightarrow (A\wedge\neg B)\Rightarrow B$

(2)（逆反法）$A\Rightarrow B \Leftrightarrow (\neg B)\Rightarrow\neg A$

(3)（反例法）$\neg(\forall x\,P(x)) \Leftrightarrow \exists x\,\neg P(x)$

我们用反证法来证明$[B\wedge(B\Rightarrow C)]\Rightarrow C$成立。假设$C$为假，即$\neg C$成立，则有

$$\begin{aligned}\neg C\wedge[B\wedge(B\Rightarrow C)] &\Leftrightarrow \neg C\wedge[B\wedge(\neg B\vee C)]\\ &\Leftrightarrow \neg C\wedge[(B\wedge\neg B)\vee(B\wedge C)]\\ &\Leftrightarrow \neg C\wedge[(B\wedge C)]\\ &\Leftrightarrow \neg C\wedge B\wedge C\text{（恒为假）}\end{aligned}$$

这样我们就推出了一个矛盾：假设$\neg C$为真且已知$[B\wedge(B\Rightarrow C)]$为真，但$\neg C\wedge[B\wedge(B\Rightarrow C)]$不为真！因此$\neg C$为真的假设是不成立的，从而$C$为真成立，$[B\wedge(B\Rightarrow C)]\Rightarrow C$得证。

1.2.3 概率论

概率论是研究非确定性现象的一个数学分支。更确切地说，概率论与数理统计是研究随机现象的统计规律的一个数学分支。要理解什么是随机现象，还得从随机试验说起。下面来看两类简单的试验。

- 试验I：一个盒子中有10个完全相同的白球，搅匀后从中任意摸取一球。
- 试验II：一个盒子中有10个大小相同的球，其中5个是白色的，另外5个是黑色的，搅匀后从中任意摸取一球。

这两类试验的区别在于：对于试验I，在没有取出球之前，就能确定取出的必定是白球，即根据试验开始时的条件就可以确定试验的结果，这种类型的试验所产生的结果称为**确定性事件**；而对于试验II，在球没有取出以前不能确定将要取出的是白球还是黑球，即从试验开始时的条件不能确定试验的结果，这种类型的试验所产生的结果称为**随机事件**。对于试验II，乍一看似乎没有什么规律可言，但是如果从盒子中反复取球许多次，每次取出球并记下它的颜色后仍放回盒子中，那么当试验的次数n充分大时，出现白球的次数和出现黑球的次数是很接近的，即出现黑球的次数与出现白球的次数的比值趋于1。这个事实是因为在盒子中的白球数等于黑球数，从中任意摸取一个球，取得白球或黑球的机会是相等的；或者说这两个事件（取出白球或取出黑球）发生的概率都是1/2。

有人认为随机事件的产生是因为我们对一个现象出现的原因还缺乏全面的认识，随着科学的发展和人类认识的深化，总有一天会不再存在不可预言的随机现象。但是更多的人认为这种看法是不正确的。因为虽然可以通过增加条件组的条件数目来减少随机性，但随机因素的影响总是不