高等院校信息技术规划教材

算法与数据结构

(第2版)

宁正元 赖贤伟 编著

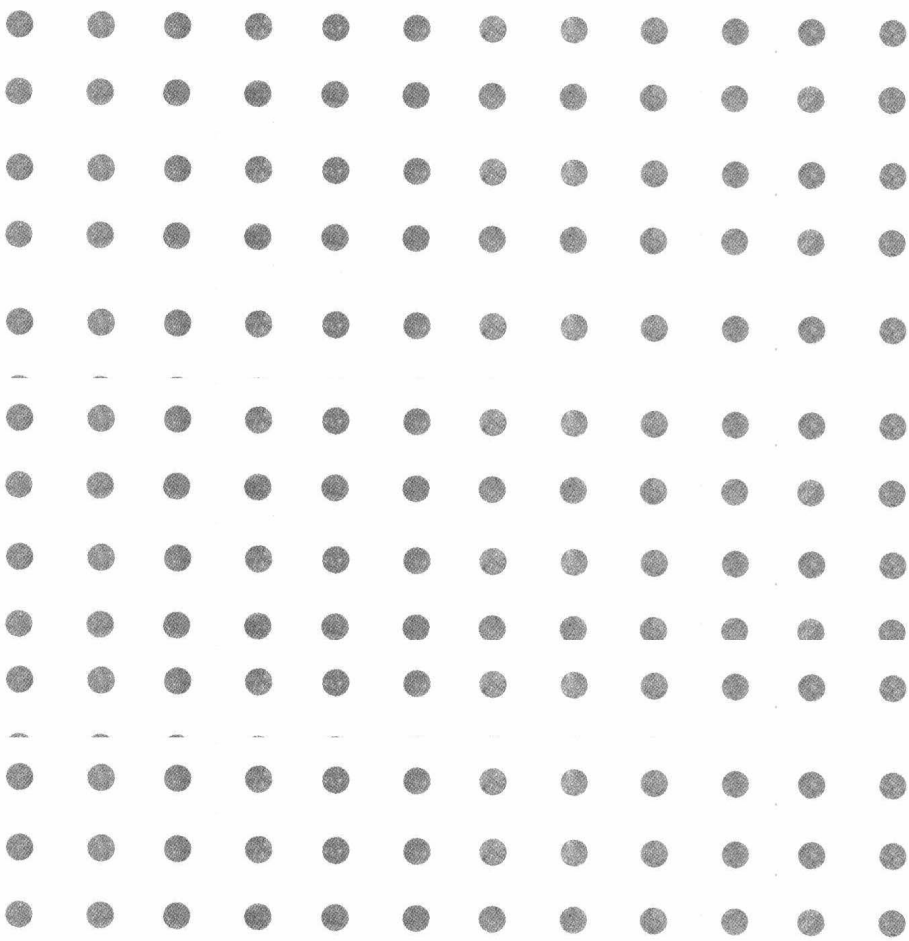
清华大学出版社

高等院校信息技术规划教材

算法与数据结构

（第2版）

宁正元 赖贤伟 编著



清华大学出版社
北京

内容简介

本书是针对应用型本科教学特点和需求编写的课程教材之一，它覆盖了《中国计算机科学与技术学科教程2002》中关于核心课程“算法与数据结构”的所有知识单元和课程提纲，系统介绍了各种常用数据结构的有关知识和各种基本的检索排序算法。每章配有足量的例题、习题和上机实验题，并有由本社出版与之配套使用的《算法与数据结构习题精解与实验指导》和电子教案，便于教师组织教学和学生自学。

全书以知识单元为基本构件，便于分解和重组，可以满足不同院校计算机科学与技术学科各专业的教学需求，也可作为从事计算机科学与技术工作的科技人员的参考用书。

图书在版编目（CIP）数据

算法与数据结构 / 宁正元，赖贤伟编著．—2版．--北京：清华大学出版社，2012.3
（高等院校信息技术规划教材）
ISBN 978-7-302-27492-6

Ⅰ．①算…　Ⅱ．①宁…②赖…　Ⅲ．①算法分析－高等学校－教材②数据结构－高等学校－教材　Ⅳ．①TP301.6②TP311.12

中国版本图书馆CIP数据核字(2011)第253601号

责任编辑：袁勤勇　顾　冰
封面设计：傅瑞学
责任校对：李建庄
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　**邮　　编：**100084
社 总 机：010-62770175　　**邮　　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**17.25　　**字　　数：**431千字
版　　次：2005年11月第1版　2012年3月第2版　　**印　　次：**2012年3月第1次印刷
印　　数：1～3000
定　　价：24.00元

产品编号：044578-01

本科教学的特点和需要,结合现阶段大学生的素质特点与学习现状,按照《中国计算机科学与技术学科教程2002》中设定的"算法与数据结构"课程教学大纲要求编写了这本教材。书中强调算法与数据结构之间密不可分的联系,体现程序设计方法中抽象、枚举、归纳三个原则的运用,反复再现算法的复杂性、效率、折中、重用等重要概念,并增加了"数据结构的程序实现"一章内容,有利于读者在较深层次上掌握好"算法与数据结构"这一核心课程。

在内容的选取上体现了突出应用的原则——针对应用型本科课程教与学的特点,在实例列举上力求做到典型和恰当,并在各章都附有相应的练习题和上机实验题;以实例介绍各种数据结构的应用,为后续课程的衔接——应用各种数据结构奠定了良好的基础。

在内容的组织上遵循循序渐进的原则——由程序与算法导入,从数据类型、数组和串开讲数据结构,与前导课程"高级语言程序设计"(C语言程序设计)内容前后贯通,无缝隙衔接浑然一体;内容由浅入深、按常用数据结构、简单数据结构、树与二叉树、图与网、检索和排序的次序安排主要教学内容。

在内容的叙述上符合通俗易懂的原则——在算法的描述上力求结构清晰、描述正确、易读易理解,并对每一个算法都做了大量的注释。在文字叙述上力求做到由浅入深和深入浅出,用语大众化,增强可读性。

围绕基本内容,除了增加"数据结构的程序实现"一章内容外,还介绍了一些同类其他书籍中没有或很少讨论的内容,如由层次序列和中序序列恢复二叉树、黄金点检索、精算点检索、共享栈插入排序等。

全书内容共8章。第1章介绍算法与程序的基本概念、表示、设计和分析方法,是在前导课程"高级语言程序设计"基础上的进一步引申和总结;第2章由前导课程中的数据类型概念引出数据结构的有关基本概念,并介绍读者已在前导课程中熟知的数组结构和串结构的进一步内容;第3章深入浅出地介绍一些简单的线性结构(如顺序表、链表、栈、队列、广义表等结构)的定义、基本算法和典型应用;第4章和第5章介绍非线性结构中的树、二叉树、图与网的有关内容;第6章介绍各类数据结构在程序设计语言中的实现策略和大批量数据的组织策略,并结合实际问题讨论了数据结构在问题建模中的应用方法;第7章和第8章介绍了在实际应用中广泛使用的检索和排序的基本算法,并进行了必要的算法分析。全书以知识单元为基本构件,各知识单元内容相对独立,具有可分解性、可重组性,便于不同院校根据各自的需要组织教学。

全书侧重应用性,力求教学内容与应用实际紧密结合;强调实践性,建议读者多动手,通过实际编程实现各种算法,并在深入分析的基础上改进算法或提出新的算法,强调多做习题和上机实验题的重要性。《中国计算机科学与技术学科教程2002》中建议本课程的讲授课时为72学时,实验课时为32学时;各校可根据自己的实际情况作适当调整。

本书由宁正元教授和赖贤伟博士共同编著,其中第1～8章由宁正元教授执笔,赖贤伟博士针对本书制作了电子教案。王秀丽教授、陈国龙教授、康宝生教授、林大辉副教授、易金聪副教授、黄思先副教授、刘雄恩副教授、黄建实验师、林敏讲师等在本书的成稿前期工作或精品课程建设中做了部分工作或提供了有益帮助,高等教育出版社计算机分

社社长张龙和清华大学出版社计算机分社高级策划编辑袁勤勇对本书的出版自始至终都给予了极大的支持和鼓励，作者在此一并表示最诚挚的感谢。

鉴于时间仓促和作者水平所限，书中难免还有错、谬、疏、漏之处，敬请同行专家和广大读者不吝赐教，我们将不胜感激。

作 者

2011 年 12 月

目录 Contents

第1章 chapter 1

算法与程序

开篇要讲的是算法与程序，因为程序＝算法＋数据结构，数据结构是与算法、程序相伴而生的。算法与程序既是计算机科学与技术学科的基础，又是计算机科学与技术学科永恒的主题。程序设计是算法与数据结构课程的基础，也是算法与数据结构课程的归宿。算法与数据结构以高级语言程序设计和离散结构作为前导课程，又服务于程序设计与软件开发；它主要研究的是复杂程序（或算法）中的数据的结构、组织和管理技术，研究复杂程序（或算法）的设计问题。

本章主要介绍算法的基本概念、特性、表示、评价和设计方法，并简要介绍程序的概念、特性、与算法的关系以及对实际问题的求解方法。

1.1 算法的基本概念

算法是计算学科的核心概念。研究算法能使我们深刻地理解问题的本质并进而找到可能的求解技术，算法设计的优劣决定着程序以至软件系统的性能。无论用计算机解决哪一方面的问题，都必须设法用数学方法来描述或模拟这些实际问题，把对实际问题的可行解决方案归结为计算机能够执行的若干个步骤，然后再把这些步骤用一组计算机指令进行描述形成所谓的程序，最后交给计算机去执行。

1.1.1 什么是算法

早在公元前300年左右出现的著名的欧几里德算法，就描述了求解两个整数的最大公因子的解题步骤。要求解的问题描述为："给定两个正整数 m 和 n，求它们的最大公因子，即能同时整除 m 和 n 的最大整数"。欧几里德当时给出的算法如下：

（1）以 n 除 m，并令所得余数为 r（必有 $r<n$）；

（2）若 $r=0$，输出结果 n，算法结束；否则继续步骤（3）；

（3）令 $m=n$ 和 $n=r$，返回步骤（1）继续进行。

由此，可以得出这样的结论，算法就是求解问题的方法和步骤。这里的方法和步骤是一组严格定义了运算顺序的规则；每一个规则都是有效的，且是明确的；按此顺序将在有限次数下终止。

有关算法(Algorithm)一词的定义不少,但其内涵基本上是一致的。最为著名的定义是计算机科学家D. E. Kunth在其巨著《计算机程序的艺术》(Art of Computer Program)第一卷中所做的有关描述。其非形式化的定义是:

一个算法,就是一个有穷规则的集合,其中之规则定义了一个解决某一特定类型问题的运算序列。

算法的形式化定义如下所述:

算法是一个四元组,即(Q,I,Ω,F)。其中:

(1) Q是一个包含子集I和Ω的集合,它表示计算的状态;

(2) I表示计算的输入集合;

(3) Ω表示计算的输出集合;

(4) F表示计算的规则,它是由Q至它自身的函数,且具有自反性,即对任何一个元素$q\in Q$,有$F(q)=q$。

一个算法是对于任何的输入元素x,都在有穷步骤内终止的一个计算方法。在算法的形式化定义中,对任何一个元素$x\in I$,x均满足性质

$$x_0 = x, \quad x_{k+1} = F(x_k), \quad (k \geqslant 0)$$

该性质表示任何一个输入元素x均为一个计算序列,即$x_0,x_1,x_2,\cdots,x_k$;该序列在第k步结束计算。

1.1.2 算法的基本特性

为了读者能更准确地理解和掌握算法定义的内涵,下面给出算法的5个重要特性:

(1) 输入(Input):一个算法应该有0个或多个输入。对于数值计算类问题,算法一定会有相应的输入数据;而对于非数值计算类问题,算法则不一定有输入数据,如制表画线等。

(2) 输出(Output):一个算法应该有一个或多个输出。设计算法的目的是为了求解给定的问题,给定问题的解是算法最终的输出结果。没有输出结果的算法是没有任何意义的。

(3) 确定性(Definiteness):算法的每一个步骤必须要有确切的定义。也就是说,算法中所有有待执行的动作必须严格地规定和描述,不能模棱两可含混不清,不能存在歧义性(也称二义性,即有不同的解释)。如欧几里德算法中步骤(1)明确规定是"以n除m",而不能是"以n除m或以m除n"这类有两种可能做法的规定。

(4) 有穷性(Finiteness):一个算法在执行有穷步骤之后必然会结束。也就是说,算法是由有序的操作序列组成的,它所包含的计算步骤是有限的。如在欧几里德算法中,由于在步骤(1)之后必有$r<n$,步骤(3)中令$n=r$后下一次执行步骤(1)时已使n的值减少,这种正整数的递降序列使得计算过程最后必然会终止。因此,不论给定m和n的初始正整数值有多大,步骤(1)的执行都是有穷次的。

(5) 有效性(Effectiveness):组成算法的每一个操作都应该是在特定环境下允许使用的,可以执行的,并能在有限时间内完成,最后得出确定的结果。只要算法中有一个操作是不可执行的,整个算法就不具有有效性。

1.2 算法的表示

算法是问题求解方法及过程的精确描述，描述算法的工具直接影响着算法的质量。算法的表示方法很多，下面介绍几种常用的算法表示方法。

1.2.1 自然语言表示

自然语言即人们日常使用的语言，如汉语、英语、日语、法语、德语等。使用自然语言描述算法，人们比较容易接受和理解。如前面的欧几里德算法就是用自然语言描述的。然而，自然语言也具有许多缺点，在使用自然语言描述算法时一定要引起注意：

(1) 自然语言存在着歧义性，容易导致算法的不确定性；

(2) 自然语言较冗长，使得描述不够简捷；

(3) 自然语言的表示形式是顺序的，描述分支选择和转移时不够直观；

(4) 自然语言与计算机程序设计语言的差别较大，不易转换为程序。

1.2.2 流程图表示

流程图是描述算法的图形工具，它采用如下所示的一组图形符号来表示算法：

⬭起止框，表示算法的开始或结束；只有一个入口或一个出口。

▱输入输出框，表示算法中数据信息的输入和输出；有一个入口和一个出口。

◇判断框，表示条件判断；有一个入口和多个出口。

▭处理框，表示算法中的一个(或一组)运算或处理；有一个入口和一个出口。

→流程线，表示算法中各步骤之间的次序关系。

○连接点，表示算法中的连接位置，主要用于同一算法在不同页描述时的接续等情况。

---▭注释框，用于对算法中某些操作的注释说明。

欧几里德算法的流程图描述如图 1-1 所示。同自然语言相比，用流程图描述算法直观，可以一目了然；算法步骤间用流程线连接，次序关系清楚，容易理解；可以很方便地表示顺序、选择和循环结构，不依赖于任何计算机和计算机程序设计语言，有利于不同环境下的程序设计。但是，流程图也还存在一些缺点，诸如：

(1) 不易于表示算法的层次结构。

(2) 不易于表示数据结构和模块调用关

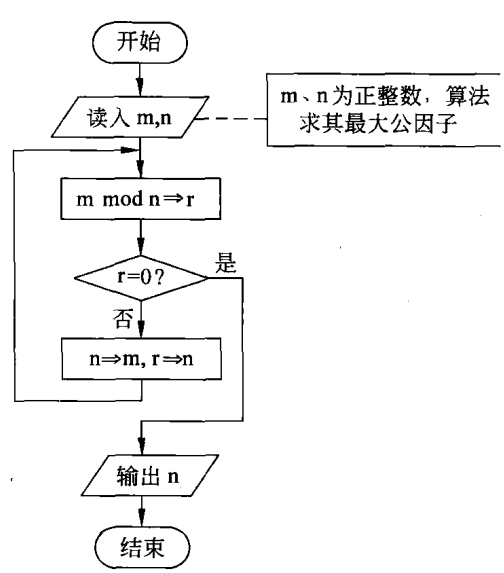


图 1-1 欧几里德算法的流程图表示

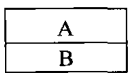
系等重要信息。

(3) 容易使人过早地考虑算法的控制流程,而忽视算法的全局结构。

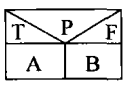
(4) 用流程线代表控制流,控制转移随意性较大。若对流程线的使用不加限制,随着求解问题规模和复杂度的增加,流程图会变得很复杂,使人难以阅读、理解和修改,从而使算法的可靠性难以保证。

1.2.3 N-S 图表示

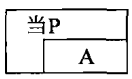
为了克服传统流程图的缺点,1973 年美国学者纳斯(I. Nassi)和施内德曼(B. Shneiderman)提出了一种表示算法的较好工具——N-S 图。它独立于任何计算机和计算机语言,又能很方便地转换为计算机语言程序;它去掉了流程线全部用矩形方框来描述,限制了算法流程转移控制的随意性,又节省了篇幅;它很容易表示算法中的嵌套关系和模块中的层次关系,功能域可以从框图中直接反映出来;它仍是图形工具,阅读起来直观、明确、容易理解。N-S 图的基本图形符号如下:



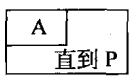
顺序结构。由两个或多个矩形框组成。其中 A 和 B 可以是基本操作,也可以是其他基本结构(如选择结构、循环结构)。



选择结构。当条件 P 成立时执行操作 A,否则执行操作 B。

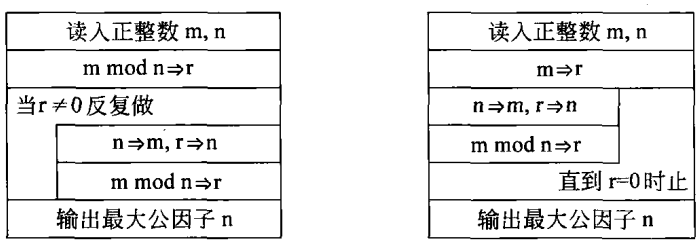


当型循环结构。当条件 P 成立时反复执行操作 A,直到条件 P 不成立时止。



直到型循环结构。反复执行操作 A,直到条件 P 成立时止。

由于 N-S 图像一个多层的盒子,因此它也被称作盒图。用 N-S 图表示欧几里德算法如图 1-2 所示。



(a) 当型循环结构实现　　(b) 直到型循环结构实现

图 1-2　欧几里德算法的 N-S 图表示

1.2.4 伪代码表示

伪代码是介于自然语言和计算机程序语言之间的一种表示算法的工具。它用文字和符号描述问题的求解方法和步骤而不使用图形符号。如同一篇文章自上而下书写,每

行写一个基本操作，而用若干行写出一个基本结构。因而书写方便，格式紧凑，清晰、易读、好理解，也更容易转化为某一计算机程序语言表示的程序。和图形工具相比较，便于修改，但直观性能较弱。下面给出欧几里德算法的一种伪代码描述：

```
Begin(算法开始)
  Read(m,n)
  m mod n⇒r
  while r≠0 do
   {n⇒m
    r⇒n
    m mod n⇒r
    }
  write(n)
End(算法结束)
```

1.2.5 程序语言表示

计算机程序设计语言，是计算机能够接受、理解和执行的算法描述工具。计算机不能直接识别自然语言、流程图、N-S 图和伪代码等工具描述的算法，而设计算法的目的就是要用计算机来解决问题，算法最终都要用某一具体的计算机程序设计语言来表示。从这个意义上讲，流程图、N-S 图和伪代码都仅仅是为了求解问题而设计算法时所使用的辅助工具。为了更好更准确地描述算法，人们使用图形或表格还创造了许多专用的算法设计辅助工具，如 PAD 图、判定表、数据流图、Warnier-Orr 图等。在此不再一一介绍这些工具，感兴趣的读者可阅读相关书籍。

和自然语言一样，计算机程序设计语言也是串行的描述，很不直观。对于较复杂的问题，人们很难用计算机程序设计语言直接写出程序。所以在算法设计阶段，一般是先采用某个专用的辅助工具来描述算法，在算法设计好之后，再把它转化为某一具体程序设计语言描述的程序。作为例子，下面给出欧几里德算法的 C 语言描述：

```
#include "stdio.h"
main()
 {int m,n,r;
  printf("请输入两个正整数：");
  scanf("%d%d",&m,&n);
  printf("\n%d和%d的最大公约数是：",m,n);
  r=m%n;
  while(r!=0)
    {m=n; n=r;
     r=m%n;
     }
  printf("%d\n",n);
  }
```