普通高等教育“十一五”国家级规划教材

高等学校规划教材

算法与数据结构（C++版）

漆涛 漆溢 蒋砚军 编著

计算机学科教学计划

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

普通高等教育"十一五"国家级规划教材
高等学校规划教材

算法与数据结构
(C++版)

漆 涛 漆 溢 蒋砚军 编著

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

内容简介

本书是普通高等教育“十一五”国家级规划教材，系统介绍各种数据结构、常用算法及算法分析技术。数据结构的内容包括线性结构、树形结构、哈希结构、索引结构；算法方面的内容包括选择算法、查找算法、排序算法。本书还较为详细地分析了各种算法的时间复杂度和空间复杂度，介绍了分摊复杂度分析技术。作为各种数据结构和算法的应用，本书给出了图的标准界面及其实现。利用这个标准界面，实现了图论中的一些经典算法。

本书以算法为主线组织内容，仿照 C++ 标准模板库的界面给出了许多算法和数据结构的实现。本书可作为高校计算机相关专业“数据结构”课程的教材，也可作为计算机工作者的参考书。

图书在版编目(CIP)数据

算法与数据结构：C++版/漆涛，漆溢，蒋砚军编著. —北京：电子工业出版社，2009.9
高等学校规划教材
ISBN 978-7-121-09451-4

Ⅰ.算… Ⅱ.①漆…②漆…③蒋… Ⅲ.①算法分析-高等学校：技术学校-教材②数据结构-高等学校：技术学校-教材③C语言-程序设计-高等学校：技术学校-教材 Ⅳ.TP301.6 TP311.12 TP312

中国版本图书馆 CIP 数据核字(2009)第 152354 号

责任编辑：凌 毅 特约编辑：梁卫红
印 刷：北京市李史山胶印厂
装 订：
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：787×1092 1/16 印张：20 字数：512 千字
印 次：2009 年 9 月第 1 次印刷
印 数：4000 册 定价：29.00 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。

前　言

“算法与数据结构”是计算机专业学生的一门基础课程。计算机应用软件、操作系统、编译器、数据库等大的计算机科学的分支会应用不同的数据结构和算法。它不仅是计算机专业学生的必修课，也是许多非计算机专业学生了解和学习计算机的选修课。“算法与数据结构”已经成为了解和学习计算机的学生的重要基础课。

本书主要内容包括线性结构、树形结构、选择算法、查找算法、排序算法和图，还介绍了字符串匹配算法、堆栈和函数调用，C++标准模板库及伪随机数产生算法。本书以算法为主线，并以此组织该书的结构。在实现算法的过程中，根据需要，选择不同的数据结构。算法的目的是完成任务。期望以此向读者展示在任何权衡各种利弊的实现过程中选择适当的数据结构。第7章讨论选择结构，介绍了数组和树实现的堆。第8章集中讨论了查找算法。第9章较为全面地介绍排序算法。第10章给出了图的标准界面及其实现，利用这个标准界面，实现了图论中的一些经典算法。

本书充实了算法分析内容，给出了算法复杂度严格的定义。第2、6章对该书算法分析中所涉及的数学知识给出了全面的介绍，对大部分算法，给出了准确且严格的复杂度分析。书中尽量选择适当的分析方法，使得分析的结果更加准确、简单。例如，分析快速排序与归并排序通常使用基本定理(参见2.2.4节)。但是基本定理仅是一种快速分析复杂度的方法，并不能够给出精确的结果。为此，书中将这两种排序算法对应于二叉树，将算法的性能对应于二叉树的某种指标。引入次满二叉树的概念，算法的最佳情形就是对应的二叉树为次满的。算法的平均复杂度就是二叉树的平均路径长度。将算法的性能对应于二叉树也为最佳归并排序给出了理论依据。又如，引入奇度概念，并给出堆排序建堆过程准确的键值比较次数分析。

鉴于分摊复杂度分析已经是算法分析和设计的主要工具，本书详细介绍了分摊复杂度的概念及其应用。

本书对数据结构方面的内容作了精简处理，详细介绍了线性结构、树形结构及哈希结构，没有将图作为数据结构对待。在介绍树形结构的同时更加注重树在算法分析中的作用。

“算法与数据结构”是一门实践性很强的课程。在计算机上实现并运行书中介绍的算法是学习这门课程的最好方法之一。本书选用带模板的C++作为描述算法语言，它可以不加修改，直接使用。作者相信这样会有利于读者理解课程的内容，激发读者自己实现算法，从而提高学习效率。受篇幅限制，书中只给出了部分算法的关键部分的实现。如果感兴趣，读者可以在电子工业出版社提供的网站(www.hxedu.com.cn)上免费下载算法的完整代码。

C++语言引入模板，将类型作为参数，和数据结构不谋而合。在数据结构中，不关注数据元素是什么，而是关注数据元素之间的关系。用带模板的C++描述算法最为合适。C++语言引入模板使得构建通用库函数成为可能。C++标准模板库(STL)就是这样的通用库函数，它集中体现了数据结构的研究成果，将其中的经典算法、经典结构实现为库函数供用户使用。本书对STL予以特别关注。第11章专门介绍STL中的基本概念：迭代子、泛函、算法、容器和适配器。本书第3章介绍线性结构时，考虑到线性结构比较简单，加入了许多STL的内容。例如，将线性结构看成容器、向量、线性表的界面都仿照STL中的界面，实现了链表的迭代子

类;在第10章中引入Weight_traits杂项模板类,这些都是STL中的概念。作者希望通过本书的介绍,使得更多的读者熟悉和使用STL。

本书每章都附有一定量的习题,它们是理解和扩展本书内容的必要组成部分,大部分习题都比较难,希望读者能够重视它们。

本书是在北京邮电大学多年讲授"算法与数据结构"的基础上,结合国外的一些先进教材及作者对这门课程理解的基础上编写而成的。感谢北京邮电大学计算机学院的领导和同事多年的支持和帮助;也要感谢使用过本书初稿的学生们,他们在使用过程中提出了许多宝贵意见。

本书被选作高等教育"十一五"国家级规划教材,感谢评审专家的信任,希望本书的内容不会令他们太失望。

感谢电子工业出版社的责任编辑凌毅同志,她专业和耐心的工作,使得本书避免了许多错误。

本书提供配套的电子课件和程序源代码,读者可登录华信教育资源 www.hxedu.com.cn,注册后免费下载。

尽管作者尽了最大的努力,但鉴于水平有限,书中不免有不妥之处,希望读者不吝赐教。作者在新浪网上为本书开了博客,地址是 http://blog.sina.com.cn/qtdatastructure。读者如有意见和建议,可以在博客上留言,也可以将意见和建议直接发送到 qit2009@yeah.net,作者将非常感谢。

漆 涛 北京邮电大学计算机学院
漆 溢 中国科学院国家授时中心
蒋砚军 北京邮电大学计算机学院

2009年8月

目　录

第1章 绪 论

算法与数据结构不仅是计算机科学与技术学科一门重要的基础课程，也是许多后继课程（如操作系统、数据库和编译原理等）的先修课程。它不仅是计算机专业学生的必修课程，也是许多非计算机专业学生了解和学习计算机的选修课。在高等院校中，不仅计算机专业，许多其他非计算机专业也开设这门课程。事实上，“算法与数据结构”已经成为学生了解和学习计算机的重要基础课。

算法与数据结构是伴随着计算机应用的普及与深入而产生的一门课程。早期的电子计算机是为数值计算而发明和设计的。应用在诸如弹道计算、天气预报等领域。现在计算机的应用已经渗透到社会的各个领域，如信息处理、图像识别、人工智能和电子交易等。这些领域大部分都是非数值计算领域，例如图像，其本质是人的感觉器官对客观事物的反映。现在的计算机不仅可以表示图像，而且可以存储、传输甚至识别图像。

算法与数据结构是研究现实世界中非数值计算问题的程序设计、信息的计算机表示、计算机操作对象以及它们之间的关系的学科。现实世界是缤纷复杂的，而计算机所能表示的只有0与1，其存储器也是线性的，中央处理器本质上也只能做有限位的加法运算。算法与数据结构就像是横架在现实世界和计算机世界之间的一座桥梁，是利用计算机解决实际问题不可缺少的工具。

离散数学也是以非数值问题为研究对象。与之相比，算法与数据结构更注重于算法的实现，而离散数学偏重于算法的理论。

1.1 利用计算机解决问题的几个步骤

利用计算机解决问题可归纳为以下几个步骤：

① 分析实际的具体问题，从中抽象出一个适当的数学模型；

② 设计一个求解此数学模型的算法；

③ 编制计算机程序实现此算法，最终得到解答。

下面以数值计算作为一个例子。

例1.1（人口预测） 已知量：每个人的净生殖率 r，时间 t，$t=0$ 时的人口数 $N(0)$。净生殖率等于出生率减去死亡率。

① 所求量 $N(t)$

② 数学模型 $\frac{1}{N}\frac{\mathrm{d}N}{\mathrm{d}t}=r$

③ 求解模型的算法 $N(t)=N(0)\mathrm{e}^{rt}$

④ 编制计算机程序

```
#include <cmath>  //for std::exp()
int population(int ini,double birth_rate,double time_at)
{ return ini*std::exp(birth_rate*time_at); }
```

给出初始人口数及生殖率，调用上面的函数可以预测出任何时刻的人口数量。在本书中，

问题的最终解体现为一个函数。输入数据为函数参数。尽量避免使用全局数据。问题的解通常为函数的返回值，在少部分情况下，也可能是函数的输出参数。输出参数通常是指针，极少情况下还可能是引用。

现实世界中的非数值问题的例子很多，例如图书馆书目检索问题、人机对弈问题以及多叉路口交通灯管理问题等。解决非数值计算问题的步骤与解决数值问题的步骤是相同的，唯一的区别是描述非数值问题的数学模型不同。而这正是算法与数据结构所要研究的内容。

1.2 基本概念和术语

数据：客观事物的数字化表示。客观事物（如图像、棋局、交通路口信号灯以及图书馆图书编目等）本身并不能存放在计算机中，但是这些客观事物在经过数字化处理之后，它们的数字化表示就可以在计算机中表示、存储和运算。

数据元素：数据的基本单位。这也是本门课程考虑的基本单元。其在计算机程序设计中通常是作为一个整体来对待的。不同的问题有着不同的数据元素。计算机硬件工作者考虑的数据几乎都是 0 与 1，考虑用何种电路去实现 0 与 1 之间的运算。对于它们来说，0 与 1 就是数据元素。而这门课程不研究计算机硬件，它需要更高的抽象，所考虑的数据元素可以是 0 和 1，也可以是别的。例如在求解偏微分方程时，实数被认为是数据的基本单元；在计算机图书管理系统中，数据元素可能包括书名、作者名、ISBN、分类号以及索书号等；而在对弈问题中数据元素为对弈过程中的棋局。有些数据元素是计算机高级语言所固有的，例如整数和实数等，而有些数据元素需要一个结构体来描述。算法与数据结构就是研究数据元素之间关系的学科。

数据项：数据元素的组成部分。通常认为数据项是不可分割的最小单位。

数据对象：具有相同性质的数据元素的集合。例如：

$$N=\{0,1,2,3,\cdots\}$$

$$\text{英文字母表}=\{a,b,c,\cdots,z\}$$

$$\text{围棋棋局}=\{q_1,\cdots,q_{3^{19\times19}}\}$$

数据结构：由三部分组成，一是数据元素的集合，二是集合元素之间存在的某种关系，三是定义在这些元素上的一些运算。即：

$$\text{数据结构}=(D,S,O_p)$$

其中，D 是数据元素的有限集合；S 为 D 上的一个关系，即，$S\subseteq D\times D$；而 O_p 就是定义在 D 上的运算。

在逻辑上，数据元素之间的关系可被分为四大类基本关系。

1. **集合结构**

$$D=\{d_1,d_2,\cdots,d_n\},n\geqslant 0 \quad \text{而} \quad S=\varnothing$$

即元素与元素之间没有任何关系。

2. **线性结构**

$$D=\{d_1,d_2,\cdots,d_n\},n\geqslant 0$$

$$S=\{(d_1,d_2),(d_1,d_3),\cdots,(d_1,d_n),(d_2,d_3),\cdots,(d_2,d_n),\cdots,(d_{n-1},d_n)\}$$

即元素之间存在全序关系。

3. **树形结构**

现实世界中有很多的树形结构的例子，如国家行政机关、家族族谱等等级关系都是树形结构。

4. 图结构(网状结构)

S 可以是 $D\times D$ 的任意的子集。

本书主要介绍线性结构和树形结构。事实上本书并不将集合结构与图结构作为数据结构来处理。集合的关系集合为空集合。而图结构,其关系集合不受任何限制。对于这种不受任何限制的关系,要想得到高效的算法是很困难的。况且对于每一种数据结构,都有一组标准的运算。但是由于图集合的关系的任意性,还没有一组大家公认且满意的标准运算。而图结构关系复杂,定义在其上的运算集合过分庞大,以致没有一种物理结构能够使所有运算得以高效地实现。

将图结构不作为一种数据结构来处理还有另外一个原因。上面的关于图结构的定义并不能完全抽象和描述现实世界。虽然图结构中的关系集合可以是任意的,但是这个定义却排除了两个元素之间有多个关系的情况,例如考虑全国的交通系统,现在的图结构的定义就不能方便地描述北京和大连之间火车、飞机和汽车等多种交通形式并存的现实。在许多的教科书中也不允许自身到自身的关系存在。而现实是,大连的星海广场的确提供环大连的观光直升飞机服务,它的起点和终点都是大连。本书将图结构作为介绍算法的一种工具来看待,以算法为主线,根据不同的算法来设计图的物理结构。另外,本书给出图的一个新定义。

不同的数据结构会有不同的操作集合。这些操作可分为:①构造函数及析构函数;②询问类操作;③查找类操作;④添加和删除类操作。在不同的数据结构中,这些操作的具体实现肯定是不同的。但是维护操作界面的一致性也是非常诱人的。它可以减轻使用者的记忆负担,方便用户使用。下面是最为常用的数据结构栈的界面:

```
template<typename T>          //T: 栈中数据元素的类型
struct Stack {                //类的实现细节略去
  Stack(void);                //构造函数
  ~Stack(void);               //析构函数
  bool empty(void) const      //查询栈是否为空
  int size(void) const ;      //查询栈中元素的个数
  T& top(void);               //返回对栈顶元素的引用
  void  push(T const& );      //压栈
  void  pop(void);            //出栈
};
```

在这里,只给出了栈的操作界面。数据结构需要研究的是如何根据界面中的这些操作,选择适当的物理结构来高效快速地实现它们。

数据结构有两个方面的属性,即逻辑结构与物理结构。以上谈到的结构为数据结构的逻辑结构。而物理结构是指数据结构在计算机中存储和表示的结构,又称为存储结构。

计算机存储器的最小单元为字节,它通常由 8 个二进制位组成。而在实际问题中,数据元素的大小是未知的。通常都会用连续的若干字节来表示一个数据元素。不同的高级程序语言会提供不同的方式将若干个数据项聚合在一起,例如 C 语言的结构体。本书不考虑数据元素是怎样在计算机内部表示的,仅考虑数据结构(D,S)在计算机内部的表示和存储问题,即不仅要存储所有的数据元素 D,而且还能表示数据元素之间的关系集合 S。选择具体的物理结构需要考虑定义在数据结构上的操作集合 O_p,甚至还要考虑操作的使用概率。物理结构总体上被分成 4 类:①顺序存储结构;②链式存储结构;③哈希存储结构;④索引存储结构。

顺序存储结构是将数据元素 $D=\{d_1,d_2,\cdots,d_n\}$连续地存放在计算机的存储器中,而用数据元素存放的位置来体现数据元素之间的关系。链式存储结构不仅存储数据元素本身,而且

还存储指向另外一个数据元素的指针，通过这个指针来体现两个数据元素之间的关系。

数据结构的一个主要任务是研究不同逻辑结构用何种物理结构来存储，使得定义在这个数据结构上的操作能够高效地完成。哈希存储结构和索引存储结构是两种特别的存储结构，在后续的章节中会仔细讨论它们。

在不同的物理结构中，希望读者能特别重视顺序存储结构。它有利于计算机内存与外存之间的数据交换。在许多时候，需要将内存中的数据结构保存到外存（也就是文件）中。而下次程序启动时，只要从文件中将数据结构读入到内存中即可。而在外存中，所有的数据都是顺序存放的。链式存储结构需要存储数据元素在内存中的地址，将这些地址保存到文件中是没有意义的。所以顺序存储结构非常适合内/外存数据传输。如果确实需要链式存储结构，为了方便数据的导入/导出，也可以使用变通的方法，即在内存中申请一块连续的内存，而将内存地址修改为偏移量。所以本质上还是使用顺序结构来模拟链式结构。

1.3 算法及其复杂度分析

所谓算法，是解决某类特定问题的求解步骤的一种描述。它是指令的有限序列。每条指令可以表示一个或多个操作。算法需要具有以下几个方面的性质。

输入：即算法所要操作的对象。理论上，任何算法都要有一定的输入。但是有些算法要求非常明确，将输入数据都固化到算法中了。这样的算法表面上好像没有输入。

输出：即问题的答案。它是必须要有的。

有穷性：对于合法的输入，算法所需执行的指令是有限的。算法能够在有限时间内结束。

确定性：每条指令必须是有确切含义的，无二义的。

可行性：指令可以在现有的计算技术基础上实现。

算法的设计和描述也要遵循以下几个要素。

正确性：设计出的算法必须对所有合法的输入都能给出正确的结果。

可读性：算法的设计者通常不是算法的使用者。写出的算法需要由别人来阅读理解和使用，所以算法一定要使用易于理解的方式来描述。

健壮性：指对于非法的数据能够给出适当的处理。

时间效率：指算法执行要快，所需时间要短。

空间效率：指执行算法所需的存储量要小。

在上面的5个要素中，最为重要的是后面两个，它们构成了算法分析的两个方面：时间复杂度分析和空间复杂度分析。算法的时间复杂度本质上是对算法执行时间的某种估计。先来考察影响一个算法执行时间的部分因素：①算法的设计；②问题的规模；③计算工具的速度。

第一个因素是算法分析的本职工作。第二个因素也是不得不考虑的。两个数的排序算法与10000个数的排序算法所需的时间当然是不一样的。而第三个因素是复杂度分析必须设法剔除的。同样一个算法，在算盘和计算机上执行的速度当然是不同的。算法的时间复杂度应该和计算工具无关。计算工具可以改变，但算法的时间复杂度不变。算法的复杂度只和算法本身及问题的规模相关。通常用一个整数或者实数 n 来刻画问题的规模，这样算法在一个具体的计算工具上的执行时间可以表示为 $t(n)$。算法时间复杂度的准确定义将在第2章给出。

算法的空间复杂度是指执行算法所需要的额外计算机存储。所谓“额外”是指保存算法执行期间的中间结果所需的空间。通常不统计保存算法输入/输出数据所需的计算机存储空间。