

高等院校信息技术规划教材

# 数据结构与算法

赵玉兰 等编著

INFORMATION TECHNOLOGY  
INFORMATION TECHNOLOGY  
INFORMATION TECHNOLOGY

清华大学出版社



高等院校信息技术规划教材

# 数据结构与算法

赵玉兰 王俊义 周建涛 王景荣 刘咏梅 刘玉林 编著

清华大学出版社  
北京

## 内 容 简 介

数据结构与算法是计算机科学中一门综合性的专业基础课,它不仅是计算机学科的核心课程,而且已成为其他非计算机专业的热门选修课之一。

本书从抽象类型的角度描述了各种逻辑结构,即线性结构、树形结构、集合和图形结构。书中由简单到复杂,循序渐进,对各种数据结构从逻辑结构、存储结构和基本操作方面进行了详细的介绍;本书另外一个特点是对各种算法进行了算法分析,对典型算法还给出了算法正确性的证明。最后一章对一些常用的算法,如“分而治之法”、“动态规划法”、“贪心法”和“回溯法”等技术进行了详细的介绍,为设计高效的程序,即以最小的成本、最快的速度和最好的质量开发出适合各种应用需求的软件奠定了基础。

全书从面向对象的角度出发,利用 C++ 语言对书中的算法进行了描述,并配有注解,有利于读者的理解;本书概念严谨、语言通俗易懂、条理清楚、图文并茂,既便于教学,又便于自学。

本书可作为计算机类专业或信息类专业的本科或专科教材,也可作为有关科研人员的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

## 图书在版编目(CIP)数据

数据结构与算法/赵玉兰等编著. —北京:清华大学出版社,2008.10

(高等院校信息技术规划教材)

ISBN 978-7-302-18384-6

I. 数… II. 赵… III. ① 数据结构—高等学校—教材 ② 算法分析—高等学校—教材 IV. TP311.12

中国版本图书馆 CIP 数据核字(2008)第 123104 号

责任编辑:袁勤勇 薛 阳

责任校对:焦丽丽

责任印制:杨 艳

出版发行:清华大学出版社

<http://www.tup.com.cn>

社 总 机:010-62770175

投稿与读者服务:010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

质 量 反 馈:010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

地 址:北京清华大学学研大厦 A 座

邮 编:100084

邮 购:010-62786544

印 装 者:北京鑫海金澳胶印有限公司

经 销:全国新华书店

开 本:185×260 印 张:20

字 数:462 千字

版 次:2008 年 10 月第 1 版

印 次:2008 年 10 月第 1 次印刷

印 数:1~4000

定 价:28.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:010-62770177 转 3103 产品编号:030480-01

# 前言

## foreword

随着计算机的问世,数据作为计算机程序的处理对象也随之产生了。在计算机的发展初期,计算机主要用于数值性计算,处理的是数值性数据,其特点是数据量少,计算比较复杂。在这一阶段,“数据结构与算法”还未形成一门系统的学科,而是零星地分布在程序设计、图论、集合、代数、操作系统和编译原理等课程中。随着计算机的发展,计算机应用领域的不断扩大,计算机不仅要处理数值性数据,也要处理非数值性数据。据统计,现在非数值性计算占到了90%以上,而且由于数据量越来越大,数据之间的关系也越来越复杂,这么多的数据在计算机中并不是杂乱无章地存放,而是有其内在的联系,只有把它们之间的关系分析清楚,才能有效地对数据进行处理。因此,除了考虑数据本身的数学特性之外,还必须考虑数据的存储结构,在这种情况下,“数据结构与算法”这门课程也随之形成。

“数据结构与算法”的研究不仅涉及计算机硬件(特别是编码理论、存储装置和存取方法等)的研究范围,而且和计算机软件的研究有着密切的关系。计算机科学各领域及有关的应用软件都要用到数据结构。无论是编译原理还是操作系统,都涉及数据元素在存储器中的分配问题,在研究信息检索时也必须考虑如何组织数据,以便查找和存取数据元素更为方便。例如,语言编译要使用栈、散列表及语法树;操作系统中要用到队列、存储管理及目录树等;数据库系统运用线性表、多链表及索引树等进行数据管理;而在人工智能领域,依求解问题性质的差异将涉及到各种不同的数据结构,如广义表、集合、搜索树及各种有向图等。“数据结构与算法”这门课的目的就是介绍一些最常用的数据结构,阐明数据结构内在的逻辑关系,讨论它们在计算机中的存储表示,并结合各种典型应用说明它们在进行各种运算(操作)时的动态性质及实际的执行算法。这样,不仅为学习后续课程提供必要的知识准备,而且更重要的是进一步提高软件设计和编程水平。并且通过对不同存储结构和相应算法的对比以及上机编程练习,增强根据求解问题性质选择合理的数据

结构并控制求解算法的时间、空间复杂度的能力。

本课程要求学生具有一定的计算机基础知识,既要对诸如集合、表、树和图等知识有初步的了解,还要求学生具有利用程序设计语言熟练编程的能力。本课程的前导课是高等数学、高级语言程序设计、离散数学;它又是操作系统、编译原理、数据库原理、算法分析、人工智能、图像处理等专业课程的前导课,具有承上启下的作用。

这本书是作者十几年教学经验的总结,同时征得了部分高校同行的意见,书上的算法都经过上机验证,同时与本书配套的 PPT 课件也相继完成,以便教学参考。

由于作者水平有限,书中难免存在错误和不足,恳请读者给予指正。

作者

2008 年 5 月于呼和浩特

# 目录

## Contents

|                          |    |
|--------------------------|----|
| 第 1 章 概述 .....           | 1  |
| 1.1 数据结构的发展 .....        | 1  |
| 1.2 数据结构 .....           | 2  |
| 1.2.1 数据结构简介 .....       | 2  |
| 1.2.2 基本概念 .....         | 4  |
| 1.3 数据的逻辑结构 .....        | 4  |
| 1.3.1 预备知识 .....         | 5  |
| 1.3.2 数据结构的分类 .....      | 6  |
| 1.4 抽象数据类型 .....         | 7  |
| 1.5 数据的存储结构 .....        | 9  |
| 1.5.1 顺序存储结构 .....       | 9  |
| 1.5.2 链式存储结构 .....       | 10 |
| 1.6 算法与算法分析 .....        | 10 |
| 1.6.1 算法 .....           | 11 |
| 1.6.2 算法性能分析和度量 .....    | 11 |
| 1.6.3 算法的描述 .....        | 16 |
| 1.7 ADT 的表示与实现间的关系 ..... | 19 |
| 习题 1 .....               | 19 |
| 第 2 章 基本数据结构 .....       | 21 |
| 2.1 线性表 .....            | 21 |
| 2.1.1 ADT 线性表 .....      | 21 |
| 2.1.2 线性表的顺序存储 .....     | 23 |
| 2.1.3 线性表的链式存储 .....     | 27 |
| 2.2 数组 .....             | 34 |
| 2.2.1 数组的定义 .....        | 35 |
| 2.2.2 数组的存储 .....        | 35 |

|                             |           |
|-----------------------------|-----------|
| 2.2.3 特殊矩阵 .....            | 37        |
| 2.2.4 稀疏矩阵 .....            | 39        |
| 2.3 字符串 .....               | 44        |
| 2.3.1 串的实现 .....            | 44        |
| 2.3.2 串的模式匹配算法 .....        | 49        |
| 习题 2 .....                  | 55        |
| <b>第 3 章 栈、队列与广义表 .....</b> | <b>57</b> |
| 3.1 栈 .....                 | 57        |
| 3.1.1 ADT 栈 .....           | 57        |
| 3.1.2 栈的实现 .....            | 58        |
| 3.1.3 栈与递归 .....            | 61        |
| 3.2 队列 .....                | 65        |
| 3.2.1 ADT 队列 .....          | 65        |
| 3.2.2 队列的实现 .....           | 66        |
| 3.3 栈与队列的应用 .....           | 71        |
| 3.3.1 栈的应用 .....            | 71        |
| 3.3.2 队列的应用 .....           | 78        |
| 3.4 广义表 .....               | 79        |
| 3.4.1 广义表的定义和基本运算 .....     | 79        |
| 3.4.2 广义表的存储结构 .....        | 82        |
| 3.4.3 广义表基本操作的实现 .....      | 85        |
| 习题 3 .....                  | 87        |
| <b>第 4 章 树与二叉树 .....</b>    | <b>90</b> |
| 4.1 树的定义和相关术语 .....         | 90        |
| 4.2 二叉树 .....               | 92        |
| 4.2.1 ADT 二叉树 .....         | 92        |
| 4.2.2 二叉树的遍历 .....          | 93        |
| 4.2.3 二叉树的性质 .....          | 95        |
| 4.2.4 二叉树的实现 .....          | 96        |
| 4.2.5 二叉树遍历的非递归实现 .....     | 106       |
| 4.2.6 线索二叉树 .....           | 109       |
| 4.3 树与森林 .....              | 113       |
| 4.3.1 树与森林的遍历 .....         | 114       |
| 4.3.2 树的存储结构 .....          | 114       |
| 4.4 森林与二叉树的关系 .....         | 116       |

|                                  |            |
|----------------------------------|------------|
| 4.5 Huffman 树与编码 .....           | 117        |
| 4.5.1 Huffman 树 .....            | 118        |
| 4.5.2 Huffman 编码 .....           | 119        |
| 习题 4 .....                       | 125        |
| <b>第 5 章 集合与查找 .....</b>         | <b>127</b> |
| 5.1 集合及其实现 .....                 | 127        |
| 5.1.1 ADT 集合 .....               | 127        |
| 5.1.2 用位向量实现集合 .....             | 128        |
| 5.2 具有 Merge、Find 的 ADT 集合 ..... | 131        |
| 5.2.1 ADT MFSet 集合 .....         | 131        |
| 5.2.2 MFSet 的实现 .....            | 131        |
| 5.2.3 MFSet 实现算法的改进 .....        | 133        |
| 5.2.4 MFSet 的应用——求等价类 .....      | 135        |
| 5.3 静态查找 .....                   | 136        |
| 5.3.1 查找 .....                   | 136        |
| 5.3.2 顺序查找 .....                 | 138        |
| 5.3.3 二分查找 .....                 | 140        |
| 5.3.4 静态次优查找树 .....              | 143        |
| 5.3.5 索引顺序查找 .....               | 146        |
| 5.4 哈希表 .....                    | 147        |
| 5.4.1 哈希函数的构造 .....              | 148        |
| 5.4.2 闭哈希法 .....                 | 151        |
| 5.4.3 开哈希法 .....                 | 156        |
| 5.5 二叉查找树 .....                  | 158        |
| 5.5.1 ADT 二叉查找树 .....            | 158        |
| 5.5.2 二叉查找树的实现 .....             | 159        |
| 5.5.3 二叉查找树的应用 .....             | 164        |
| 5.6 平衡二叉查找树(或称 AVL 树) .....      | 165        |
| 5.6.1 AVL 树及其性质 .....            | 166        |
| 5.6.2 最小局部的不平衡类型及平衡方法 .....      | 167        |
| 5.6.3 AVL 树的实现 .....             | 172        |
| 5.7 B-树与 B+树 .....               | 176        |
| 5.7.1 B-树 .....                  | 177        |
| 5.7.2 B+树 .....                  | 183        |
| 5.8 键树 .....                     | 185        |
| 5.8.1 双链表 .....                  | 186        |
| 5.8.2 Trie 树 .....               | 187        |

|                                                   |            |
|---------------------------------------------------|------------|
| 习题 5 .....                                        | 188        |
| <b>第 6 章 图 .....</b>                              | <b>191</b> |
| 6.1 图的基本概念 .....                                  | 191        |
| 6.1.1 图的定义 .....                                  | 191        |
| 6.1.2 图的术语 .....                                  | 192        |
| 6.2 图的存储结构 .....                                  | 195        |
| 6.2.1 数组(邻接矩阵、相邻矩阵)表示法 .....                      | 196        |
| 6.2.2 邻接表表示法 .....                                | 197        |
| 6.2.3 邻接多重表表示法 .....                              | 200        |
| 6.2.4 十字链表 .....                                  | 201        |
| 6.3 图的遍历 .....                                    | 202        |
| 6.3.1 深度优先遍历 .....                                | 203        |
| 6.3.2 广度优先遍历 .....                                | 204        |
| 6.4 无向图的应用 .....                                  | 206        |
| 6.4.1 最小(代价)生成树 .....                             | 206        |
| 6.4.2 关节点和双连通图 .....                              | 210        |
| 6.5 有向图的应用 .....                                  | 213        |
| 6.5.1 拓扑排序 .....                                  | 213        |
| 6.5.2 关键路径 .....                                  | 216        |
| 6.6 最短路径 .....                                    | 220        |
| 6.6.1 从一个顶点(源点)到其他顶点的最短路径 .....                   | 221        |
| 6.6.2 每一对顶点间的最短路径(all-pairs shortest-paths) ..... | 223        |
| 6.6.3 传递闭包 .....                                  | 225        |
| 习题 6 .....                                        | 227        |
| <b>第 7 章 排序 .....</b>                             | <b>229</b> |
| 7.1 内部排序方式 .....                                  | 229        |
| 7.2 插入排序 .....                                    | 231        |
| 7.2.1 直接插入排序 .....                                | 231        |
| 7.2.2 折半插入排序 .....                                | 232        |
| 7.2.3 希尔排序 .....                                  | 233        |
| 7.3 选择排序 .....                                    | 234        |
| 7.3.1 直接选择排序 .....                                | 235        |
| 7.3.2 树形选择排序 .....                                | 236        |
| 7.3.3 堆排序 .....                                   | 236        |

|                             |            |
|-----------------------------|------------|
| 7.4 交换排序 .....              | 239        |
| 7.4.1 冒泡排序 .....            | 239        |
| 7.4.2 快速排序 .....            | 241        |
| 7.5 归并排序 .....              | 244        |
| 7.6 基数排序 .....              | 245        |
| 7.6.1 多排序码排序 .....          | 246        |
| 7.6.2 链式基数排序 .....          | 246        |
| 7.7 各种内部排序算法的比较 .....       | 249        |
| 7.7.1 各种排序方法的比较 .....       | 249        |
| 7.7.2 排序问题的下界 .....         | 250        |
| 习题 7 .....                  | 251        |
| <b>第 8 章 外部排序</b> .....     | <b>253</b> |
| 8.1 外部排序的方法 .....           | 253        |
| 8.1.1 外部排序的基本过程 .....       | 253        |
| 8.1.2 多路平衡归并 .....          | 255        |
| 8.1.3 置换-选择排序 .....         | 257        |
| 8.2 最佳归并树 .....             | 261        |
| 习题 8 .....                  | 263        |
| <b>第 9 章 动态存储管理</b> .....   | <b>264</b> |
| 9.1 概述 .....                | 264        |
| 9.2 可利用空间表 .....            | 266        |
| 9.2.1 可利用空间表结构 .....        | 266        |
| 9.2.2 分配 .....              | 267        |
| 9.2.3 回收 .....              | 268        |
| 9.3 伙伴系统 .....              | 268        |
| 9.3.1 指数伙伴系统的可利用空间表结构 ..... | 269        |
| 9.3.2 分配 .....              | 271        |
| 9.3.3 回收 .....              | 272        |
| 9.4 一个小型的动态存储管理系统 .....     | 273        |
| 9.4.1 系统内存结构 .....          | 273        |
| 9.4.2 内存分配 .....            | 275        |
| 9.4.3 内存释放与垃圾回收 .....       | 275        |
| 习题 9 .....                  | 276        |

|                              |     |
|------------------------------|-----|
| 第 10 章 算法分析与设计技术 .....       | 277 |
| 10.1 递归算法的分析 .....           | 277 |
| 10.2 递归式求解 .....             | 278 |
| 10.2.1 展开递归式 .....           | 278 |
| 10.2.2 猜解 .....              | 278 |
| 10.2.3 大递归式的一般解 .....        | 279 |
| 10.3 分而治之与递归 .....           | 281 |
| 10.4 动态规划法 .....             | 283 |
| 10.4.1 动态规划法的基本思想 .....      | 283 |
| 10.4.2 最优二叉查找树 .....         | 286 |
| 10.5 贪心法 .....               | 289 |
| 10.6 搜索法 .....               | 291 |
| 10.6.1 搜索法的基本思想 .....        | 291 |
| 10.6.2 启发式搜索 .....           | 293 |
| 10.7 NP 完全性问题 .....          | 295 |
| 10.7.1 问题的描述形式 .....         | 296 |
| 10.7.2 P 类问题与 NP 类问题 .....   | 297 |
| 10.7.3 NP-Complete 类问题 ..... | 299 |
| 10.7.4 近似算法 .....            | 300 |
| 习题 10 .....                  | 302 |
| 参考文献 .....                   | 303 |

## 概 述

## 1.1 数据结构的发展

作为一门系统的课程,数据结构理论的产生和发展是与计算机应用领域的延伸以及程序设计技术的发展密切相关的。

1946年2月,世界上诞生了第一台数字积分计算机 ENIAC(Electronic Numerical Integrator and Computer)。ENIAC 的诞生具有极其重要的历史意义,它与机械计算工具有着本质的区别,研发的本质动力是将人类劳动从繁杂的计算中解放出来。在计算机发展的初级阶段,计算机领域所讨论问题的重心是如何为高等数学、离散数学中所描述的数学模型重新确立算法、构造新的模型,以便于借用计算机最本质的运算来进行描述和完成计算,如弹道计算、矩阵运算、函数计算、方程组求解、定积分计算等。总的来说,这类科学计算数据量小,数据类型单一,核心问题仍集中于构造新的算法。而程序设计技术是以机器语言、汇编语言的机制为基础的,因此在当时出现了“计算机科学的核心问题是算法问题”的论点,从而形成“以算法为核心”的认识。

随着计算机应用范围向信息领域的延伸,计算机面对的更多的是用于控制、管理和信息处理等非数值性计算工作。加工对象由纯粹的数值发展到字符、表格等各种具有一定结构的数据。这类非数值性计算数据量巨大,数据类型相对复杂,计算却比较简单。人们开始意识到数据表示法在程序设计中的独特地位。程序中常用的一些数据表示,如表(线性表、链表、广义表等)、栈、队、树、图等被单独分离进行研究。面向结构的程序设计理念形成之后,人们就越来越重视分析待处理对象的特性和考虑数据间存在的关系、数据的组织形式和数据的表示方法。数据结构及抽象数据类型就是在这种情况下初步形成的,其主要标志是1968年美国唐·欧·克努特教授(D. E. Knuth)的巨著《计算机程序设计技巧》(*The art of computer programming*)的第一卷《基本算法》的发表,该书首次系统地阐述了数据的逻辑结构和存储结构以及定义在数据上的操作。

进入20世纪90年代以来,电子、计算机、声像、通信技术得到空前的融合,计算机的应用已经渗透到现实社会的方方面面。目前计算机所处理的问题,不仅信息量巨大,而且数据类型复杂,处理过程涉及多种技术。以多媒体数据为例,多媒体数据实际上由多种不同类型的数据组成,通常包括文字、图像、图形、声音、视频图像、动画等不同的数据类型,而且同一类型的数据可以有不同的表示方法。特别是多媒体数据中的声音和视频

图像数据都是与时间有关的信息。在处理多媒体数据的过程中,要涉及多种技术,如压缩编码技术、信息存储技术、数据库技术、通信技术。因此,人们再一次意识到要设计一个“好”的程序,不仅要注重求解方法而且要注重数据特性,两者均不可偏废。此时,面向对象的程序设计方法逐步地被人们认识、认可。面向对象技术中的核心元素——对象,是描述实体的属性与操作的封装体,内含了数据结构中的主要因素:数据成分与操作。面向对象技术实质上是数据结构概念的自然扩展与继续,因此也推动了数据结构理论的

进一步发展和成熟。

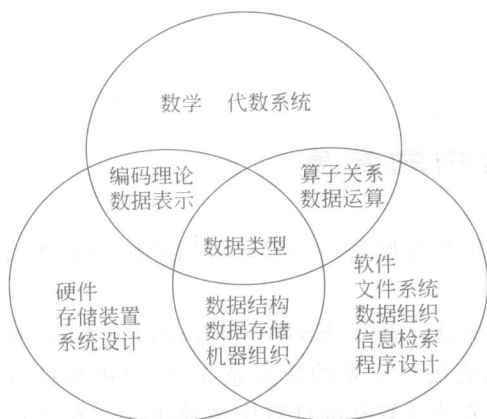


图 1.1 数据结构的地位

“数据结构与算法”作为计算机科学中一门综合性的专业主干基础课程之一,其研究不仅涉及计算机硬件(特别是编码理论、存储装置和存取方法等)的研究范围,而且和计算机软件的研究也有着极为密切的关系,无论是编译程序还是操作系统,都涉及数据元素在存储器中的空间分配问题。如图 1.1 所示在研究信息检索技术时必须考虑如何组织数据,以便查找和存取数据元素更为方便。因此,可以认为数据结构是介于数学、计算机硬件、计算机软件三者之间的一门核心课程。

## 1.2 数据结构

### 1.2.1 数据结构简介

数据结构理论虽然已经从萌芽开始走向了成熟,但是,站在不同的角度,数据结构的侧重点将有所不同。下面将给出实例来说明从非数值性计算的角度出发,数据结构所表示的内在含义。

#### 例 1.1 电话号码簿。

在日常生活中,电话号码簿可以说是随处可见的,但是它却涵盖了朴素的数据结构理论。假设让我们来设计一本某地区或更小范围内的电话号码簿,首先要做的是统计该地区的所有电话号码,接着要考虑的是如何表示这些号码和安排号码的位置,使得用户使用号码簿查询号码时更便捷、更有效。所以,在设计电话号码簿时涉及对数据(电话号码)的分析和数据间的位置关系的考虑。用户在使用号码簿查询号码的行为则是数据结构理论中研究的算法。

假设呼和浩特地区的电话号码簿的组织设计如 1.2 图所示。当用户有急事要联系内蒙古大学计算中心的相关人员时,查询电话号码的办法是根据号码的安排策略,从教育→大中专院校→内蒙古大学→计算机学院→计算中心找到所需的电话号码。在计算中心所有人员的电话号码中可以进行顺序查找或者仍按照号码的安排策略查找。从电

话号码簿的组织设计图可以看出该图像一棵倒长的“树”，号码的查询过程就是从树根沿着树杈到达某个叶子的过程。由此可以看出，“树”可以作为某些非数值性计算问题的数学模型。

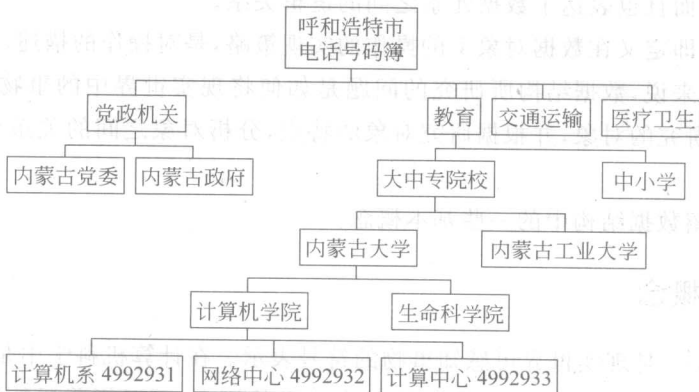


图 1.2 电话号码数据安排

### 例 1.2 图书馆书籍管理系统。

作为一名读者,当需要借阅书籍时,首先要到图书馆图书管理系统去查阅图书目录。通常情况下,图书的目录是按照一定的内容进行分类的,以便加快检索速度,如图 1.3 所示,有学科索引、书名索引、作者索引、出版社索引等。从该图以及我们对图书管理系统的了解可知:该系统涉及的数据有书名、学科、作者、出版社这些非数值性数据,同时也具有借阅书籍数目、赔偿金额等一些数值性数据。面向读者的操作有检索、借书、还书、赔偿等,面向管理者的操作有采编、购书、出库、上架、整理、清点等。如何组织和安排这些不同性质的数据,采用何种算法来实现上述操作使图书管理系统运行高效、使用便捷,这就是数据结构理论研究的问题。

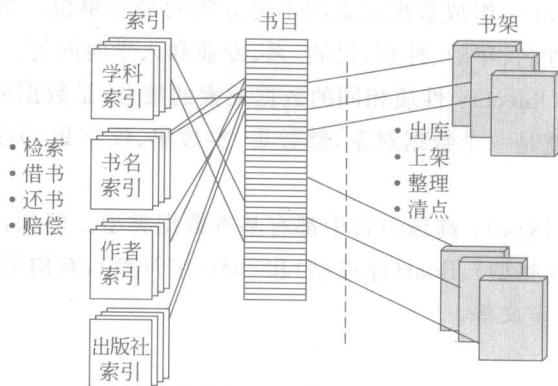


图 1.3 图书馆的体系结构

由上面两个例子可知,数据结构所研究内容的着重点主要体现在以下 3 个方面。

一是数据间的逻辑关系,即数据元素之间的关系。在解决同一问题时,采用不同的

数据组织策略将产生不同的数据逻辑关系。如图书管理系统中,不同的索引将导致同一数据处于不同的位置,与不同的元素建立不同的关系。

二是数据的存储关系,即数据在计算机中的存储结构。它不仅体现了数据元素在计算机中的表示,而且也表达了数据元素之间的逻辑关系。

三是算法,即定义在数据对象上的操作的实现策略,是对操作的描述。

因此,简单来说,数据结构所研究的问题是如何将现实世界中的事物合理描述为计算机世界中所研究的对象,并根据研究对象的特点,分析对象之间的关系、存储结构和操作的学科。

下面先介绍数据结构中的一些基本概念。

### 1.2.2 基本概念

数据(Data):是现实世界可感知事物的符号表示。在计算机科学中是指可被计算机表示、存储和加工处理的信息的总称。它是个广义的概念,可以指普通的数据(可参加算术运算),也可以指符号(源程序、产品名称等)或数字化声音、图形和图像等。

数据元素(Data Element):能独立、完整地描述现实世界中的实体的某些特性的数据单位称为数据元素,是数据的基本单位,相对独立,通常作为一个整体看待。例如,学生基本情况表(如表 1.1 所示)中的每条记录都是一个数据元素。

表 1.1 学生基本情况表

| 学号      | 姓名  | 年龄  | 性别  | 民族  | 系   | 专业    | 入学时间   |
|---------|-----|-----|-----|-----|-----|-------|--------|
| ...     | ... | ... | ... | ... | ... | ...   | ...    |
| 0130012 | 李名  | 19  | 男   | 汉   | 计算机 | 计算机应用 | 2003.9 |
| ...     | ... | ... | ... | ... | ... | ...   | ...    |

数据项(Data Item):组成数据元素的不可分割的最小单位。例如,学生基本情况表中的数据项有学号、姓名、年龄、性别、民族、系、专业和入学时间等。

数据对象(Data Object):性质相同的数据元素的集合,是数据的一个子集。例如,所有学生记录的集合,就是一个数据对象,整数集、实数集、汉字集、英语词汇等都是不同应用领域的数据对象。

数据类型(Data Type):高级语言中都有基本数据类型。例如,在 C、C++ 和 Java 语言中,有基本类型 int(整型)、float(浮点型)和 char(字符型),有构造类型如数组、结构、联合、指针、枚举型和自定义等。

## 1.3 数据的逻辑结构

现实世界中的任何一个对象都不是孤立存在的,而是与其他的对象存在着这样或那样的联系。在数据结构理论中,数据元素作为对现实世界对象的描述,必然保持对象之

间的关系,也就是说,数据元素之间存在着一定的关系。我们把数据之间的关系称为数据的逻辑结构,经常把数据的逻辑结构简称为数据结构。数据的逻辑关系完全可以借用数学中的关系理论进行表达。

### 1.3.1 预备知识

#### • 偶对

在现实生活中,当用计算机处理问题时,总会发现有许多事物是成对出现的,而且与其他事物之间存在固定关系,如父子关系、同学关系、邻里关系等。在数学中,表示两个事物之间所具有的固定关系的方法是偶对。如甲与乙具有邻里关系通常表示为偶对(甲,乙)。偶对分为无序偶对和有序偶对,如果偶对所表示的对象之间的关系没有次序之分,如邻里关系,则为无序偶对,用圆括号来表示。如果偶对表示的对象之间的关系是有次序的,如父子关系,则为有序偶对,用尖括号表示。例如,(a,b)表示a和b是邻居,这个关系是没有次序的,a和b是邻居,则b和a也是邻居,即 $(a,b)=(b,a)$ ;  $\langle a,b \rangle$ 表示a是b的父亲,这个关系是有次序的,a是b的父亲不能说b是a的父亲,即 $\langle a,b \rangle \neq \langle b,a \rangle$ 。

可以用图形表示偶对,如图1.4所示,用圆圈表示数据元素,无序偶对的数据元素间用一无向的线段连接,有序偶对的数据元素间用一有向的线段连接。

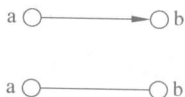


图 1.4 偶对图形表示法

#### • (直接)前驱与(直接)后继(Previous And Next)

如果有 $\langle a,b \rangle$ ,称a是b的(直接)前驱,b是a的(直接)后继。

#### • 笛卡儿积(Descartes Set)

给定两个集合A和B,如果有序偶对的第一个分量是A的一个元素,第二个分量是B的一个元素,则所有这种有序偶对的集合称为集合A和B的笛卡儿积。

记为 $A \times B = \{ \langle x,y \rangle \mid x \in A \wedge y \in B \}$ 。

例如,集合 $A = \{a,b\}$   $B = \{1,2,3\}$ ,则,A和B的笛卡儿积为

$$A \times B = \{ \langle a,1 \rangle, \langle a,2 \rangle, \langle a,3 \rangle, \langle b,1 \rangle, \langle b,2 \rangle, \langle b,3 \rangle \}$$

#### • 二元关系(Duality Relationship)

R是集合A上的二元关系:  $R \subseteq A \times A$

#### • 数据的逻辑结构

由数据元素集及其逻辑关系组成,可以形式地描述为一个二元素:  $DS = (D, S)$ ,其中,数据结构是一个二元素,D是数据元素的有限集合;S是D上的有限关系集合。

**例 1.3** 一周有7天,它们之间有先后的关系,其逻辑关系可表示为:

$$\text{WEEK} = (D, R)$$

$$D = \{\text{Sun, Mon, Tue, Wed, Thu, Fri, Sat}\}$$

$$R = \{ \langle \text{Sun, Mon} \rangle, \langle \text{Mon, Tue} \rangle, \langle \text{Tue, Wed} \rangle, \langle \text{Wed, Thu} \rangle, \langle \text{Thu, Fri} \rangle, \langle \text{Fri, Sat} \rangle, \langle \text{Sat, Sun} \rangle \}$$

关系图如图1.5所示。



图 1.5 一周的逻辑结构

例 1.4 行政机构及其隶属关系的逻辑结构可表示为：

$IMU = (Dep, R)$

$Dep = \{imu, fcs, fse, his, cs, csc, nc, phy, math\}$

$R = \{ \langle imu, fcs \rangle, \langle imu, fse \rangle, \langle fcs, cs \rangle, \langle fcs, csc \rangle, \langle fcs, nc \rangle, \langle fse, phy \rangle, \langle fse, math \rangle \}$

关系图如图 1.6 所示。

例 1.5 A、B、C、D 4 个球队进行循环比赛，一个队在一天中最多进行一场比赛，C 和 D 已进行过比赛，还有 5 场比赛：AB(AB 表示 A 队与 B 队比赛)，AC, AD, BC, BD。若以 (AB, AC) 表示 AB 和 AC 不能在同一天进行，则可以表示为：

$Arrange = (D, S)$

$D = \{AB, AC, AD, BC, BD\}$

$S = \{ (AB, AC), (AB, AD), (AB, BC), (AB, BD), (AC, AD), (AC, BC), (AD, BD), (BC, BD) \}$

关系图如图 1.7 所示。从图 1.7 可以看出，实线连接的顶点表示不可以同一天举行比赛，由虚线连接的顶点表示比赛可以在同一天举行，所以 4 个球队进行的循环赛至少需要 3 天，AB 一天，AC 与 BD 一天，AD 与 BC 一天。

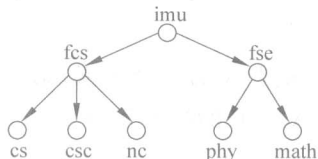


图 1.6 行政机构

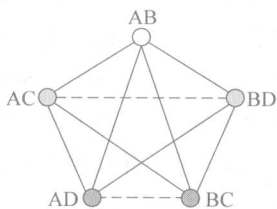


图 1.7 球队比赛图

- 开始结点和终端结点

无前驱的结点是开始结点，无后继的结点是终端结点。

### 1.3.2 数据结构的分类

根据数据元素间关系的不同特性，通常数据结构可以分为以下 4 种基本逻辑结构。

(1) 线性结构(Linear Structure)：数据元素之间存在“一对一”的关系。逻辑结构可以表示为  $LS = (D, R)$ 。

其中， $D = \{d_1, d_2, \dots, d_n\}$ ； $R = \{ \langle d_i, d_{i+1} \rangle \mid i = 1, 2, 3, \dots, n-1 \}$

(2) 集合：是数据结构中逻辑结构较为“松散”的一种结构，结构中的数据元素之间除了“同属于一个集合”的关系外，别无其他关系。