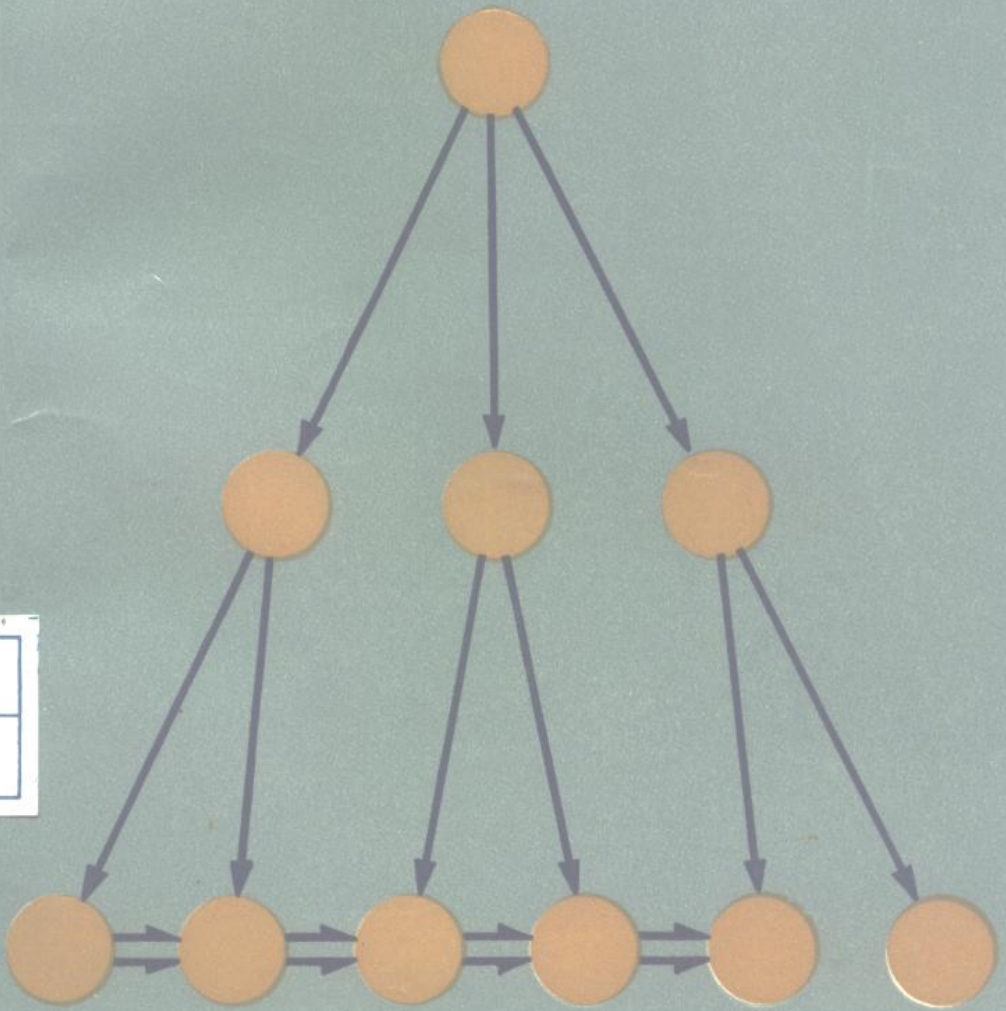
数据结构

魏晴宇 晋良颖 编著



数 据 结 构

魏晴宇　晋良颖 编著

中国人民大学出版社

数 据 结 构

魏晴宇　晋良颖 编著

*

中国人民大学出版社出版发行
（北京西郊海淀路39号）
中国人民大学出版社印刷厂印刷
（北京鼓楼西大石桥胡同61号）
新 华 书 店 经 销

*

开本：850×1168毫米32开　印张：10.25
1988年4月第1版　1988年4月第1次印刷
字数：245 000　册数：1—5 000

*

ISBN 7—300—00222—6
O·12　　定价：3.10元

前　言

这本教材是为信息管理系的同学编写的。

关于数据结构的讨论，是随着电子计算机日益广泛地应用于各种非数值计算问题而逐渐发展起来的。在60年代初，国内外都还没有专门的“数据结构”课程，它的一部分内容，散见于编译原理和操作系统课程中。从60年代中期开始，这门课程陆续在一些国家的计算机系出现。但当时名称很不统一，通常称为“表处理语言”，主要介绍 SLIP 系统、IPL-V 系统、LISP 系统、SNOBOL 系统等。后两者慢慢演变成了专门的程序语言。真正确定“数据结构”为一门课程，是在1968年以后。

随着人们在计算机方面的实践和认识的发展，人们逐渐认识到，算法和数据结构是计算机上所处理问题的两个基本组成部分，这两个部分紧密相关，但同时又各自具有不同的内容。在有关算法的论述中，人们所探讨的是：对于一个给定的问题，用什么方法来计算它；如果有不同的方法，那么哪一种方法更好。在有关数据结构的论述中，人们所探讨的是：对于一个给定的问题，它的数据在逻辑上有什么特征，应该如何来组织它的数据，才便于人们对数据的使用和处置。显然，这是一个问题的两个方面，它正如一本书的名称：《算法＋数据结构＝程序》。

因此，在数据结构的讨论中，虽然也要涉及到某些算法问题，但重点却不在此，而在于在不同的条件下应该如何来组织数据和操纵数据。关于算法的讨论，那是另一门课程——“算法基础”的任务。尤其这本教材是为信息管理系的同学编写的，自然

地，将把重点放到应用方面上，对于系统软件和硬件方面的某些特殊问题，不拟作太多论述，甚至舍弃。

这本教材已在中国人民大学经济信息管理系使用了七年，使用中陆续进行了一些增删和修改，应该说是有一定教学实践基础的。但计算机科学的飞速发展，仍然使我们感到这本教材还不够完备，很难概括各方面应用中所涉及到的各种数据结构问题。同时，由于我们的水平有限，不足之处在所难免。我们热忱地欢迎读者给予指正，以便将来修改。

编　者

1986.8.

目　录

第一章 概 论

§1. 数学准备

本节的内容是以后经常要用到的一部分数学知识，这里将它们简单地叙述一下，但不进行详细的论述。这有两个目的，一方面是为了浏览一下有关的材料，知道以后将用到的主要是哪些数学内容。另一方面则是为了给出一些必须的概念和名词，并且统一某些符号的使用。如果从另一个意义上来讲，这些内容则可以帮助我们从更一般、更抽象的观点上来概括和整理数据结构中的某些基本问题。

假定给了一个集合A，A中包含着n个元a_1，a_2，……，a_n($n\geqslant 0$)，则称集合A是一个**有限集**，记为

$$A=\{a_1,\ a_2,\ \cdots\cdots,\ a_n\}$$

如果集合A中的每个元都具有某种特定的性质p，而不在集合A中的任何元都不具备性质p，那么有时也将集合A记为

$$A=\{x\mid x\text{具有性质}p\}$$

当集合A中不包括任何元时，称A是一个**空集**，空集用符号ϕ表示。如果集合A中包含的元是无限多时，称A是一个**无限集**。但今后的讨论将主要是在有限集上进行，因此，如果未作说明，所说的集合都是指有限集。

如果A是一个集合，a是集合A中的一个元，则称元a**属于**集合A，记为$a\in A$。如果一个元b不是集合A中的元，则称元b**不属于**集合A，记为$b\notin A$。

如果A和B是两个集合，A中的每一个元都属于集合B，则称集合B**包含**集合A，或称A是B的一个**子集**，记为$A\subseteq B$。如果A是B的子集，并且B中至少有一个元a不属于A，则称A是B的**真子集**。

如果给定集合A和B，它们满足$A\subseteq B$和$B\subseteq A$，则称集合A和B**相等**，记为$A=B$。

对于集合A和B，定义它们的**并集**$A\cup B$，**交集**$A\cap B$和**差集**$A-B$如下：

$$A\cup B=\{x\mid x\in A\text{或}x\in B\}$$
$$A\cap B=\{x\mid x\in A\text{且}x\in B\}$$
$$A-B=\{x\mid x\in A\text{且}x\notin B\}$$

在$A\cup B$的定义中，“或”字没有相互排斥的意思。即当$x\in A$并且$x\in B$时，仍有$x\in A\cup B$。另外，也可以将$\cup$、$\cap$和$-$视为定义在集合上的三种运算，正如在数的集合上定义的运算加、减、乘等一样。

如果集合A和B满足$A\cap B=\phi$时，就称集合A和B是**不相交**的。按照交集的定义，这也就是说，集合A和B没有公共的元。

如果集合A_1，A_2，……，A_n都是集合A的子集，并且

$$A=A_1\cup A_2\cup\cdots\cdots\cup A_n\text{①}$$
$$A_i\cap A_j=\phi\qquad i,\ j=1,\ 2,\ \cdots\cdots,\ n$$
$$i\neq j$$

则称集合A_1，A_2，……，A_n构成了集合A的一个**划分**。

如果a_1，a_2是两个元，按先后顺序将它们排列在一起，并且作为一个整体来看待，则称它为一个**序偶**，记为$\langle a_1,\ a_2\rangle$。注意$\langle a_1,\ a_2\rangle$和$\langle a_2,\ a_1\rangle$是不同的，因为它们的排列顺序不同。更一

① 并的运算满足结合律，所以本式有意义。

般地讲，若a_1，a_2，……，a_n是n个元，将它们按先后顺序排列，并且记为$\langle a_1, a_2, \cdots\cdots, a_n\rangle$，则称$\langle a_1, a_2, \cdots\cdots, a_n\rangle$是一个**n元组**。对于任何两个$n$元组$\langle a_1, a_2, \cdots\cdots, a_n\rangle$和$\langle b_1, b_2, \cdots\cdots, b_n\rangle$，当且仅当$a_i = b_i$对于$i = 1, 2, \cdots\cdots, n$都成立时，才称它们是**相等的**，并且记为

$$\langle a_1, a_2, \cdots\cdots, a_n\rangle = \langle b_1, b_2, \cdots\cdots, b_n\rangle$$

在n元组$\langle a_1, a_2, \cdots\cdots, a_n\rangle$中，$a_i$称为它的第$i$个**分量**。这里的$i$可以等于1，2，……，$n$，它指明了$a_i$在$n$元组中的位置。

如果给定了两个集合A和B，则定义它们的**直接积** $A \times B$如下：

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

所以$A \times B$是由A中的元与B中的元组成的所有序偶的集合。这个概念也可以推广到n个集合A_1，A_2，……，A_n的情形，定义

$$\begin{aligned} & A_1 \times A_2 \times \cdots\cdots \times A_n \\ &= (\cdots\cdots((A_1 \times A_2) \times A_3)\cdots\cdots \times A_n) \\ &= \{\langle a_1, a_2, \cdots\cdots, a_n\rangle \mid a_i \in A_i, i \\ &\qquad = 1, 2, \cdots\cdots, n\} \end{aligned}$$

集合$A \times B$的每个子集R都称为从A到B的**一个关系**。或者称R是$A \times B$上的一个关系。当$A = B$时，就简称R是定义在A上的**一个关系**。

如果R是定义在集合A上的一个关系，则$\langle a, b\rangle \in R$时，有时也将它记为$aRb$，即从$a$到$b$关系$R$成立。此时，称元$a$是元$b$的**前缀**，元$b$是元$a$的**后继**。

若R是集合A上的一个关系，对于A中的任何一个元a，都有$\langle a, a\rangle \in R$，则称关系$R$是**自返的**。如果对于任何一个元$a \in A$，都有$\langle a, a\rangle \in R$，则称关系$R$是**反自返的**。如果对于$A$中任何元$a$和$b$，当$\langle a, b\rangle \in R$时，必有$\langle b, a\rangle \in R$，则称关系$R$是**对称的**。如果

$\langle a, b\rangle \in R$且$\langle b, a\rangle \in R$，则$a=b$时，称$R$是**反对称的**。如果对于$A$中的任何元$a$、$b$、$c$，当$\langle a, b\rangle \in R$，并且$\langle b, c\rangle \in R$，必有$\langle a, c\rangle \in R$时，则称关系$R$是**传递的**。

当集合A上定义的关系R，具有自返，对称和传递三个性质时，就称R是一个**等价关系**。此时，如果$\langle a, b\rangle \in R$，就称元$a$和$b$是**等价的**。

当集合A上定义了一个等价关系R时，可以根据这个关系，将A中的元分为若干部分，让它们分别属于A的若干个子集。这些子集互不相交，并且使同一个子集的任何两个元都具有关系R，而任何两个不同子集中的任何两个元，都不满足关系R。即，这些子集构成了集合A的一个划分，而这些子集都称为关系R的**等价类**。

反之，如果给了集合A的一个划分，则可以定义一个关系R：同一个子集中任何两个元都具有关系R，不同子集的任何两个元都不满足关系R。容易验证，这样定义的关系R是A上的一个等价关系。

如果R是定义在集合A上的一个关系，它是反自返的，反对称和传递的，则称R是集合A上的一个**偏序**。显然，对于一个定义在有限集上的偏序说来，至少有一个元没有前缀，至少有一个元没有后继。

如果R是定义在集合A上的一个关系，R是传递的，并且对于集合A中的任何元a、b都满足：

（1）$\langle a, b\rangle \in R$，或$\langle b, a\rangle \in R$，或两者都成立。

（2）若$\langle a, b\rangle \in R$并且$\langle b, a\rangle \in R$，则$a=b$。

这时就称R是集合A上的一个**完全序**或**线性序**。

若S是一个集合，在S上定义一个线性序R。$A \subseteq S$。根据关系R，可以将A中所有的元排成一个序列$B=a_1, a_2, \cdots\cdots, a_n$，使$\langle a_i, a_{i+1}\rangle \in R$，$i=1, 2, \cdots\cdots, n-1$。序列$B$称为集合$A$按照

线性序R所作的一个**排序**。①

如果f是定义在$A\times B$上的一个关系，对于每个$x\in A$，都有唯一的一个$y\in B$，使得$\langle x, y\rangle\in f$，则称f是定义在A上，并且在B中取值的一个**函数**，记为$f: A\to B$。当$\langle x, y\rangle\in f$时，常常又记为$f(x)=y$。如果给定了$f: A\to B$，对于任何的x_1，$x_2\in A$，当$x_1\neq x_2$时，必有$f(x_1)\neq f(x_2)$，则称函数f是一个**一对一**的。

一个**有向图**G是指一个序偶$\langle A, R\rangle$，其中的A是一个集合，A中的元称为图G的**结点**。R是定义在A上的一个关系，它的每一个元都称为图G的**弧**。如果a、$b\in A$，$\langle a, b\rangle\in R$，则称弧$\langle a, b\rangle$是从结点a出发，到达结点b的。如果a_1，a_2，……，$a_n\in A$，并且$\langle a_i, a_{i+1}\rangle\in R$，$i=1, 2, \cdots\cdots, n-1$，则称序列$a_1$，$a_2$，……，$a_n$构成图$G$的一条路，当各$a_i$都不相同时，这条路称为一条**简单路**，当$a_1=a_n$，而$a_1$，$a_2$，……，$a_{n-1}$都不相同时，这条路称为一条（**简单**）**回路**。如果$\langle a_i, a_i\rangle\in R$对结点$a_i$成立，则称图$G$在结点$a_i$处有一个**圈**。

一个有向图经常可以用一个几何图形来表示，如图1.1，它表示了一个有向图$G=\langle A, R\rangle$，其中

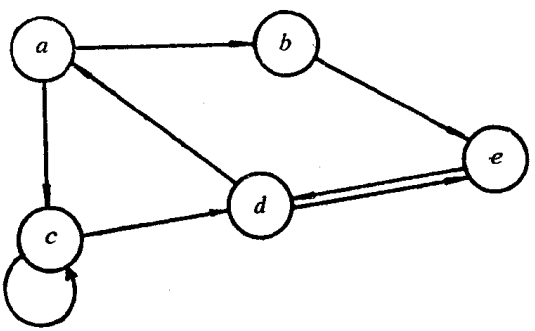


图 1.1

① 一般在集合A中，不考虑元的重复出现，但在实际工作中，有时允许A中元重复出现，今后按后一种情况理解。

$$A=\{a,\ b,\ c,\ d,\ e\}$$

$$R=\{\langle a,\ b\rangle,\ \langle a,\ c\rangle,\ \langle b,\ e\rangle,\ \langle c,\ c\rangle,$$
$$\langle c,\ d\rangle,\ \langle d,\ a\rangle,\ \langle d,\ e\rangle,\ \langle e,\ d\rangle\}$$

一个有向图总是和一个矩阵联系着，这个矩阵的每一行都表示图G中的一个结点a_i，$(i=1,\ 2,\ \cdots\cdots,\ n)$，每一列也都表示图中的一个结点$a_j(j=1,\ 2,\ \cdots\cdots,\ n)$。当结点$a_i$和$a_j$之间具有弧$\langle a_i,\ a_j\rangle$时，则矩阵$R$中的元$r_{i,j}=1$，否则$r_{i,j}=0$，即

$$r_{i,j}=\begin{cases}1 & \langle a_i,\ a_j\rangle\in R \quad (i,\ j=1,\ 2,\ \cdots\cdots,\ n)\\ 0 & \text{否则}\end{cases}$$

这个矩阵称为图G的**关联矩阵**，记为M_G。例如图1.1所表示的图G的关联矩阵如下：

$$M_G=\begin{pmatrix}0&1&1&0&0\\0&0&0&0&1\\0&0&1&1&0\\1&0&0&0&1\\0&0&0&1&0\end{pmatrix}$$

显然，在有向图和关联矩阵之间具有一一对应的关系，即对于任何一个有向图G，总存在一个它的关联矩阵M_G，反之，对于任何一个元素都是0和1的方阵，总有一个有向图是以它为关联矩阵的。因此，对于图G的一些研究，常常也可以通过矩阵M_G来进行。

如果在图G中，不考虑方向问题，即当$\langle a,\ b\rangle\in R$时，有$\langle b,\ a\rangle\in R$，那么就得到**无向图**。对于无向图，同样可以引进路、简单路、回路和圈的概念，也可以给出关联矩阵。由于无向图中不必考虑方向问题，所以常常可以使问题更简单一些。

§2. 数据类型

在电子计算机出现后的最初一段时间，它主要用于自然科学和工程技术部门，用来进行一些复杂的、计算量比较大的数值计算工作。那时，人们所利用的，主要是计算机的快速进行算术运算的能力。从人们当时的认识来讲，电子计算机也无非是一种新的，快速计算的工具而已。所以在计算机的使用中，尽管计算的问题是多方面的，但从加工的对象和所要求进行的加工工作来讲，主要的仍是数学中常见到的一些东西。即要求加工的对象仍然是实数、复数、某些特殊的函数、矩阵等等。而所进行的加工基本上都是数值计算，如计算函数值，解方程组，求一个微分方程的近似解，算一个逆矩阵等等。这一点在早期的一些高级语言中可以明显地看出来。如ALGOL-60，FORTRAN，都着眼于科技方面的计算。因而在这些语言中，数据的类型都比较简单，只有整型量、实型量、布尔型量。符号型的量没有给予充分的考虑，如果允许使用，那也只是为了打印一些表头，或是输出时在某些位置上印一点特殊的指定符号而已。就是说，数据本身没有什么结构方面的问题。

但是随着计算机的发展、计算机应用范围的扩大，情况开始起了变化。这种变化首先表现在计算机上所加工的对象和所进行的加工工作的变化中。

由于计算机的使用扩展到了工业、商业、政府机关部门，而且进入了这些部门的日常工作，而这些工作中，又遇到了各种各样的不同类型的数据。

例1 在学校的工作中，可以遇到下列形式的数据（表1.1），要求进行加工。

这是一张假想的表，实际生活中所常常见到的报表比它更复杂。现以此为例分析所发生的变化。

表 1.1

学生证号	姓名	性别	入学年龄	在校表现	政治面目	毕业时间	成绩						
							一年级				二年级		
							数学	物理	政治	外语	化学	语文	生物

首先可以看到，加工对象起了变化。原来计算机上要求加工的对象，不过是一些数和数组，而这里所要求加工的对象却是报表。此表是由若干行组成的，而每一行都反映了一个学生的完整情况。即这里作为加工的基本单位，应该是表中的行。

其次，现在所遇到的这种加工单位——行的本身也有了结构问题。它是由若干项组成的。这些项之间在逻辑上有一定的联系，一般地说不允许颠倒顺序或随意割裂。如果孤立地考虑其中某个项，就可能使之丧失原有含义。例如，从表中的某一行取出一个“男”或“90”来，那么就什么用处也没有了。加工单位的逻辑结构现在成了一个极端重要、完全不允许忽视的问题。可是这样一种加工单位能用数组来表示吗？即使不说不行，至少也应该说不方便。从表中可以看到，每一行中都可能既有整数（如学生证号），也有实数（如各门功课的分数），既有布尔型数（如用它来表示性别），也有符号型数（如姓名）。这样一个复杂情况用数组来表示会有一定困难，更何况这里还有“成绩”、“一年级”、“二年级”之类的问题。这使程序员们感到在处理这些问题时，原来的语言就不那么方便了。

第三，这里所说的计算机加工工作，一般地说都不是很复杂的数值计算，并不要求有很复杂的计算过程，相反，在计算上常常是很简单的。就是说，这里所使用的，主要是计算机的判断方面的能力，而不是计算方面的能力。所说的加工工作，如查看某个学生的情况，按某门功课的分数高低将学生们排序；搞一张某年毕业的学生成绩表；检查团员们的学习情况；加工一张新的报表等等。这些工作看来似乎简单，但却有困难的地方和特殊的要求。它们要求的数据存贮量常常是相当大的，而且这种工作必须经常地、频繁地进行（例如，一个商店每日的库存情况和销售情况的整理汇总，一个银行每日的业务工作，都要求每日必须加工完毕）。

第四，不能不看到，每个单位每个部门都有自己所要求的报表格式，很难象在科技计算时那样，要求所有计算课题中的数据都用数或数组形式来组织，很难给所有的用户提供一个他们都满意的数据类型。

情况的变化发展使人们不能不考虑以下的几个问题。

（1）对于所出现的各种类型的数据，如何从数学上给予概括。因为只有进行了这样的概括才便于给予统一的描述。

（2）对于各种类型的数据来说，最经常要求在计算机上所进行的加工工作是哪些？这些工作有什么特点？

（3）对于数据之间逻辑上的联系，应该如何来描述？

（4）这些概括、描述和常用的加工工作如何在计算机上实现？如何更好地实现？

这样一些问题提出的结果，自然地也就导致人们对于数据结构的讨论和研究。

本节将讨论第一个问题。

例2 考虑下列四种类型的数据（见表1.2）。

它是经过简化后的数据类型。在表1.2中（1）所表示的是一