用于计算机科学的组合学

S. G. Williamson 著

黄彩玉 译

西安电子科技大学情报资料室

一九八九年三月

译者说明

本书译自《Combinatorics for Computer Science》(S. Gill Williamson 著)中的"PART I: Linear order"，除第四章和第五章外，从第一章至第三章都多少不同地删去了部分练习题。虽然如此，译文本身仍然保留着连贯性和系统性。

由于本人水平有限，译文难免有错误，欢迎批评指教。

译者 董彩玉

1989.2

目　录

第一章　线性次序的基本概念

我们将从便于对基本有限集进行运算的观点出发，去学习排列、子集、图、树的结构、偏序集和全序集等主题，它们除了其本身很有趣以外，还是描述、设计种种算法的基础，这些算法的研究极大地提高了我们的水平。另外，在算法的设计和分析中，线性表的结构和使用是最基本的技术，而从某些“形式”观念出发去研究线性次序，可获得若干有用的基本概念。

1.1　定义

设S是集合，S上的关系ρ是从笛卡儿乘积S×S到任一集T（含有二个元素）的函数。

例如，T={0，1}，T={-1，+1}，T={假，真}都可用作关系ρ的值域T。如果集T是确定的且S是有限集，则易见有2^{p}（$p=|s|^{2}$）个从S×S到T的函数，（如果S是有限集，|S|表示集S的基数或元素数），虽然，从整体上看关系这个概念是不重要的，但是当研究算法时两种特殊的关系却起着重要的描述性作用，因此，下面定义ρ：S×S→T是一种关系，规定T={假，真}，用记号“xρy”表示ρ(x，y)=真，用“xφy”表示“ρ(x，y)=假”。

1.2　定义

定义在S上的一个关系ρ为：

（1）若对所有x∈s，有xρx，称ρ有自反性。

（2）若对一切x、y∈s，若xρy则yρx，称ρ具有“对称性”。

（2′）若对所有的x，y∈s，xρy且yρx，则x=y，称ρ具有“反对称性”。

（3）若对一切x，y、z∈S，xρy且yρz，则xρz，称ρ具有“传递性”。

如果关系ρ满足1，2，3，称ρ是“等价关系”，如果关系ρ满足1，2′及3，称ρ有“有序关系”。

因为等价关系与集合的划分之间存在着对应，故等价关系的一般结构是很容易理解的。

1.3　定义

设S是一个集合，S的一个划分是S的一个子集集合$\mathcal{C}$，$\mathcal{C}$满足$\bigcup_{A\in\mathcal{C}} A=S$，当A、B∈$\mathcal{C}$时，或者A=B，或者A与B不相交，即A∩B=φ，由此定义得到S的子集是$\mathcal{C}$的元素，它被称为$\mathcal{C}$的块，如果每一块中只有一个元素，则$\mathcal{C}$是离散的，空集φ∉$\mathcal{C}$。

如果N^{*}是正整数集合，则$\mathcal{C}$={E，O}是N^{*}的一个划分，其中E是N^{*}中的偶数集合，O是N^{*}中的奇数集合。集合{1，3，7}，{2，4，5，6}，{8}}是S={1，…，8}的一个划分。

1.4　定义

设ρ是定义在S上的一个等价关系，对每个s∈S，令E_{s}={x；x∈S，xρs}，E_{s}称为s关于ρ的等价类，简称为s的等价类。

1.5 定理(划分和等价关系之间的关系)

若ρ是集合S上的等价关系，则ρ的全部等价类的集合$\mathscr{C}=\{E_s: s\in S\}$是S的一个划分，反之，若$\mathscr{C}$是S的任一划分，且x、y∈S，当x、y属于$\mathscr{C}$中同一块时有xρy，于是，ρ是S上的一个等价关系，而$\mathscr{C}$是等价类的集合。

本科生的许多数学课程(离散数学、实分析、逻辑、代数、群论、线性代数)中经常要使用定理1.5，读者应尽力自证之，需要时可查阅有关参考书。从定理1.5知等价关系的一般结构是很清楚的，但是，某些关系在开始时并不明显表现为等价关系，特别对传递性有时难以证得成立，若对ρ能证得定理成立，则从定理1.5得到该划分为等价类集合。

1.6 注释

整数1，…，n的集合用记号$\underline{n}$表示，如果A和B是集合，用f：A→B表示函数f，(其定义域是A，值域是B)，用B^A表示这类函数集，(注意：如果A和B是有限集，|A|=a，|B|=b，则$|B^A|=b^a$) Image(f)是集合$\{f(a): a\in A\}$，对每个b∈B，$f^{-1}(b)$被称为“b的逆象”，它的集合是$\{a: a\in A, f(a)=b\}$，集合$\mathscr{C}=\{f^{-1}(b): b\in \text{Image}(f)\}$是A的一个划分，被称为Coimage(f)，若Image(f)=B，则f是满射的，如果Coimage(f)是离散集，则f是单射的，如果f既是满射又是单射的，则f是双射的，显然，若|A|=|B|(两者都是有限的)，则f是单射的当且仅当f是满射的，A^A的单射被称为A的排列，这里A是有限集。

举例说明上述概念，考虑$f\in \underline{6}^{\underline{6}}$，其中$f=\begin{pmatrix}1&2&3&4&5&6\\1&3&1&3&2&2\end{pmatrix}$，Image(f)={1, 2, 3}，Coimage(f)={{1, 3}, {5, 6}, {2, 4}}，该函数可用one-line记号法记为(1 3 1 3 2 2)，如果知道了定义域，并且指定若干次序，则此记号便指定了一个函数，用one－line记号，$\underline{6}$的有些排列可视为(5 6 3 2 1 4)或(4 2 3 5 6 1)。

我们在等价关系一节中将看到许多例子，在此先提几个例子。(研究等价关系时，用记号～比用记号ρ更方便)。

1.7 等价关系的例子

(1) $S=\underline{2}^{\underline{3}}$，其中f～g满足Image(f)=Image(g)，等价类是{(111)}，{(112)，(121)，(211)，(122)，(212)，(221)}和{(222)}，这里对所有函数都用one－line记号。

(2) 如果Coimage(f)=Coimage(g)，则f～g，于是$\underline{2}^{\underline{3}}$的等价类为{(111)，(222)}，{(112)，(221)}，{(121)，(212)}，和{(122)，(211)}。

(3) $S=\underline{2}^{\underline{3}}$，其中f～g满足Max(f)=Max(g)，有两个等价类。

(4) $S=\underline{2}^{\underline{3}}$，其中f～g满足f是g的循环移位：(112)，(211)，(121)满足任一个是其他各个的循环。

(5) $S=\underline{2}^{\underline{3}}$，其中f～g满足f(1)+f(2)+g(3)=g(1)+g(2)+g(3)。

下面为懂一些图论的读者举些例子：

(6) 设图G=(V, E)，若图中存在从x到y的一条路径，则x～y，由此定义了等价关系．等价类用于确定g的连通部分。

(7) 若图G=(V,E)，如果e和f位于G的同一个简单回路(无自回路)中，则e～f，由此定义了等价关系，等价类用于确定G的强连通部分，设e～e，请读者自验传递性。

(8) 设S={(a,b)：a和b都是整数，b≠0}，如果ab′=ba′，定义(a,b)～(a′,b′)，

这个等价关系用于有理数的形式定义中。

(9) 设$\{a_n\}$和$\{b_n\}$是两个有理数的无限集合，如果$\lim_{n\to\infty}(a_n-b_n)=0$，则确定$\{a_n\}\sim\{b_n\}$，这是等价关系，用于实数理论的形式研究中。

有序关系的结构比等价关系复杂，有序关系中常使用记号$x\leqslant y$或$x\preccurlyeq y$，集S与有序关系$\preccurlyeq$一起称为“偏序集”记作$(S,\preccurlyeq)$，又用$x\prec y$意味着$x\preccurlyeq y$，但$x\neq y$。

1.8 定义

设$(S,\preccurlyeq)$是有序集，若$x\prec y$且$x\preccurlyeq z\preccurlyeq y$能导出$x=z$或$y=z$(对一切$z\in s$)，称y复盖x，如果y复盖x，称y是x的后继，x是y的前导，记作$x\overset{c}{\prec}y$。

1.9 有序集的例子

(1) (R，$\leqslant$)表示通常次序的实数集。

(2) (N，$\leqslant$)表示有通常次序的正整数集。

(3) 给定集合A，令$S=\mathcal{P}(A)$表示A的幂集，则$(S,\subseteq)$是偏序集，此处记号“$\subset$”表示普通集合的包含关系。

(4) $(\pi(A),\preccurlyeq)$，这里$\pi(A)$=集合A的划分，如果π_2是π_1的加细，则$\pi_2\preccurlyeq\pi_1$。例如，设$A=\underline{8}$，$\pi_1=\{\{1,3,5\},\{2,4,6\},\{7,8\}\}$，$\pi_2\{\{1,3\},\{5\},\{2,6\},\{4\},\{7,8\}\}$，则$\pi_2\preccurlyeq\pi_1$。(进一步细分块$\pi_1$得到块$\pi_2$)。

(5) $(N,\preccurlyeq)$，这里$x\preccurlyeq y$当且仅当$x\mid y$(x整除y)。

(6) $\underline{r}^{\underline{d}}$(从$\underline{d}$到$\underline{r}$的函数集)，这里$f\preccurlyeq g$当且仅当$f(i)\leqslant g(i)$(对一切i成立)。

1.10 定义

给出偏序集$P=(S,\leqslant)$，再令$P_{c.c.}=(S,\overset{c}{\prec})$，我们构造$P_{c.c.}$的图满足：用一条线从a联到b当且仅当a复盖b，该图$P_{c.c.}$称为P的哈希图。

1.11 以x|y为序关系，集合$S=\underline{12}$的哈希图

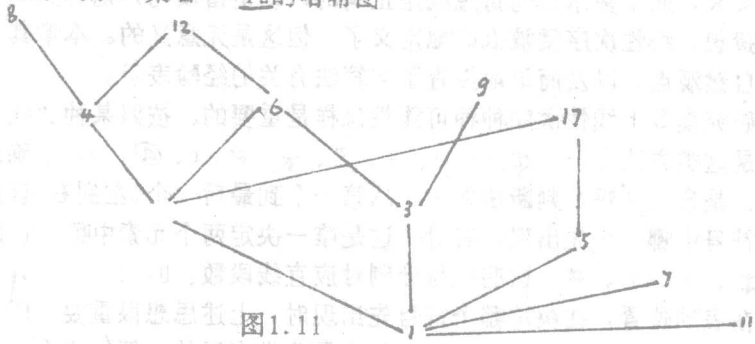


图1.11

1.12 带有包含关系的$\underline{2}$的全体子集$(\mathcal{P}(2),\subseteq)$的哈希图，($\underline{5}$，$\leqslant$)的哈希图。

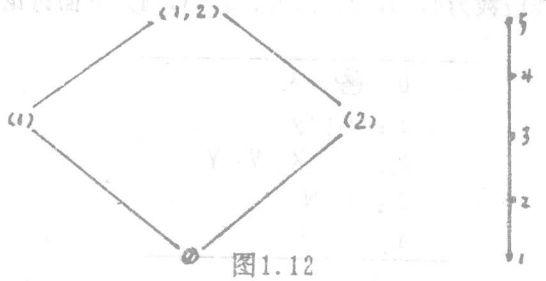


图1.12

1.13 练习

(1) 设μ_n是所有$n \times n$阶实值矩阵的集合，如果$A+D=B+C$通过$(A, B)\sim(C, D)$定义了$\mu_n \times \mu_n$上的一种关系$\sim$。证明“$\sim$”是$\mu_n \times \mu_n$上的等价关系。

(2) 令μ_n如同(1)中所设，如果存在非奇异矩阵P满足$PAP^{-1}=B$，则定义$A \sim B$，试证$\sim$是μ_n上的等价关系，这个等价关系，对已经学过线性代数的学生而言，能很好地说明等价类。

(3) 设ρ_1是S上的关系，ρ_2T上的关系，若$s \rho_1 s'$且$t \rho_2 t'$，用它定义$S \times T$上的关系ρ_3，记为$(s, t) \rho_3 (s', t')$，如果ρ_1和ρ_2都是等价关系，则ρ_3也是等价关系。如果ρ_1和ρ_2都是有序关系，则ρ_3也是有序关系。

(4) 构造$(\mathcal{T}(4), \subseteq)$和$(18, |)$的哈希图，可查阅例题1.9(3)和1.9(5)，再查阅图1.11(关于$(\underline{12}, 1)$)，

(5) 设ρ是S上的一个具有自反、对称性的关系，如果S中存在序列$u_1, \cdots, u_p$，满足$s \rho u_1, u_1 \rho u_2, \cdots, u_p \rho t$，由此定义了关系$\rho'$，记为$s \rho' t$，试证$\rho'$是S上的等价关系。

(6) 参照例题1.9(3)和1.9(6)，考虑$(\mathcal{T}(d), \leftarrow)$和$(\{0, 1\}^d$，对$A \in \mathcal{T}(d)$，若$x \notin A$，则$f_A(x)=0$，若$x \in A$，则$f_A(x)=1$，由此定义了$f_A \in \{0, 1\}^d$。证明映射$\Phi(A)=f_A$是从$\mathcal{T}(d)$到$\{0, 1\}^d$的双射。证明$A \sqsubseteq B$当且仅当$f_A \leqslant f_B$。这种映射$\Phi$称为在两个偏序集之间的保序双射，这一个函数$f_A$称为A的特征函数。

对偏序集有一个影响大且很吸引人的数学理论，然而，从算法的角度看，最基本的技术包含着研究线性次序的各种类型，如果对每个属于S的x，y，或者$x \leqslant y$，或者$y \leqslant x$，则偏序$(S, \leqslant)$是线性有序的。线性有序集的哈希图是一个链。(例如图1.12的$(\underline{5}, \leqslant)$)。线性有序的纯数学观念允许存在于许多别的分技中。费尔马猜想正整数a，b，c和$n>2$，$a^n+b^n \neq c^n$。设$S=\{x, y\}$，如果费尔马的猜想是正确的，表为$x<y$，若费尔马的猜想是错误的，表为$y<x$，如果费尔马的猜想或是正确的，或是错误的，则定义了线性次序。如果能解决费尔马猜想，线性次序便被很好地定义了，但这是无意义的。本章其余部分给出了线性次序的一个自然观点，以及简单地探查了与算法有关的经验表示。

实际上，研究集S上线性次序种种可能性怎样是重要的。按照某种“线性排列”列出集合中的元素就是这类方法之一。¢、%、&、$、*、#、(、@、)、十确定了课文符号集上的线性次序。是$<#吗？判断法之一：从第一个到最后一个(左到右)读这个排列，检查被比较的两个符号中哪一个先出现，有时，这是唯一决定两个元素中哪一个小的方法。考虑线性有序集@、¢、x、z、#。这些符号分别对应直线段数：0，1，2，3，4，这些数字自然地决定了符号在表的位置，在决定哪个符号先出现时，上述思想很重要，即使这种对应不是双射的，(对应于每个符号的数字不唯一)，却也是非常有用的。例如表@、&、+、x、v、y、z、N、#对应的直线段数为0，0，2，2，2，3，3，4，下面讨论表1.14。

1.14 检验表

```
0 : @  &
1 : empty-
2 : +  X  V  Y
3 : Z  N
4 : #
```

为了比较表中的元素首先要计算它们的直线段数。对＋号算得 2，对N算得 3。易知 2＜3，则＋＜N。若要比较＋和V，算得两者都为 2，这时需到表1.14中找出标号 2 的那行，就可决定是否成立＋＜V。上述方法可在大范围内应用(计算机科学家称为哈希法)。下面提出许多有趣的问题。表1.14给出了子集标记为整数或符号(例如 2)，首先要知道实地查找其中的子表有多难?

假如不想比较表中两个符号，而想知道表中给定元素前导或后继课文元素，这种数据的编制类型有用吗? 如何衡量这种方法所提供的有用程度? 一种最简单的计算方法（直接查找方法)：假定某些特殊符号类是直接查找的符号，粗略地说，如果某个符号标记着一个子集例如表1.14，那末，给出这个符号，在不依赖于“问题难度”(可用某种方式测量)的小的固定时间数内，我们可以直接到子表中去寻找。例如，假想表1.14为一个很大的译本，给出大数 n，时间数取为在较小表1.14中寻找标号为 2 的子表所用的时间，按照这种方法的规定，我们可以直接到标号为 n 的表中查找，当然实际上这是不可能的，因为光读一个很大的阿拉伯数字就需要1,000年，然而，实践表明用计算复杂度来衡量，直接查找法对某种基本运算是好方法，从这点上看，对被称为链表的线性有序集编制计算方法是有用的。图1.15表明方格或“位置”的排列，其中有些放着某类基本符号，每个方格有一个地址，例如A_4，B_6等等，给定地址就能找到并读出该方格的内容，在图1.15中指出初始地址D_3，然后按照方块中的指令行动，就读出了@ # $ ¢ &。这种“数据结构”就叫做链表

1.15 链接表

开始于D_3：

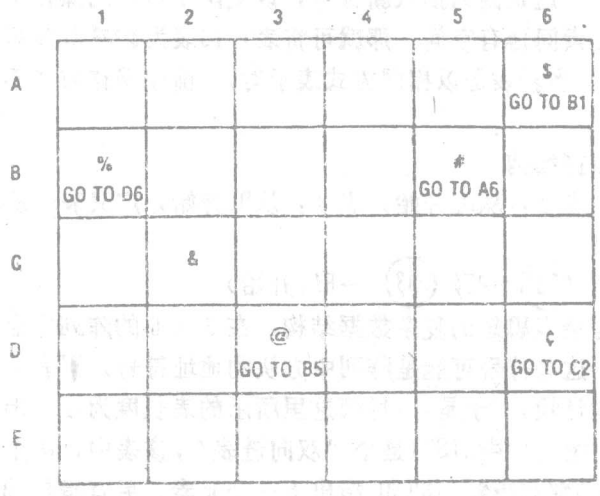


图1.15

“直接查找法”的基本假设给出了像“D5”或“E2”那样的符号，在一定时间内就可查到在1.15中所示结构的区域。计算复杂性的课程要用严格的术语去表示，这里仅用直观的观念就足够了。像1.15那样的结构允许在相当范围内扩充，并且对于所有结构都有同样固定的查找时间。粗糙的想法是需要有一个开头，以便考虑算法的有效性，研究计算复杂度可以解决上述问题，并可引出一个值得研究的实际程序设计问题。

1.16 图1.15的结构图表

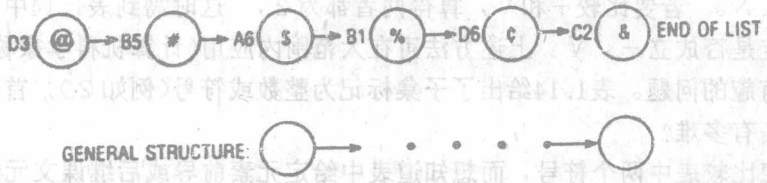


图1.16

图1.15所示的数据结构相关的可以提出许多基本问题，为了得到链表1.15的形像感觉，可用图1.16所示的结构图来表示，表中的元素显示在圈中，每个元素的地址写在圈的左边，1.16的箭头不光表示链表中某元素的地址，而且还表示了下一个元素的地址，下一个元素的地址(用箭头表示)名叫“指针”。

读者应改虑与图1.15的链表数据结构有关的某些基本问题：如何执行在$和%之间“插入新符号¢”的操作呢？显然，应先确定未用上的方块如E_4，且将新符号¢放入该方块中，然后找到放有$的方块$A_6$，用“GO TO E_4”取代“GO TO B1”，对块E_4加入指针“GO TO B1”，完成这些操作需多少时间呢？是先找空块，我们用扫描大量的块去发现一个空块呢还是跟踪空块，以致在一定时间内发现空块呢？下一步必须找到含有$的块，我们曾假设过在一定时间内能找到此块，这个块给出了地址(如“A_6”)但没有内容“$”。如果表的符号是地址符号，则完满地解决了这个问题。在最简单存储模型中所提出的一些问题，在与“现实世界”的计算情况打交道过程中被强调了许多次，我们乐观地估计到上述插入新符号的操作在一定时间内能够完成，这一点已从直接查找法中看到，那就有可能相信插入新符号所需时间不依赖于链表的长度，这是因为插入新符号，改变两个指针的操作不依赖于链表的长度。如果组成数据结构的元素间含有空间，那就可将表1.15表为@ # $ % ¢ &。现在，在$和%间插入一个新符号¢。当新表是以相同方式表示时，前述操作与“不依赖于表长的一定时间”无关。

1.17 更复杂的数据结构

表1：从E_3开始，表2：从D_4开始，表3：从D_3开始，广义表：从E_1开始。

广义表的结构图表：

E1 (D6) →D5 (E5) →E6 (D3) →E1(开始)

图1.17显示了含有链表思想的复杂数据结构。在5×6的阵列中存有三张表，表中出现“A2”“B3”等符号，这些符号可能是阵列中方块的地址符号，W表示用一定时间在一些大范围阵列中能够找到地址块，于是，尽管这里所示的表长度为4，但是尚可考虑长度为n(n>4)的类似的表。图1.17中LIST1是个“双向链表”，该表中，每个含有元素的方块（除第一个和最后一个以外)都有指针 AHEAD 指出下一个元素，另有指针 BACK 指出前面元素。因为存在表中的符号也是地址，所以我们可用这些地址去存储它在表中的地址，在A2中找到了E2，E2是表1中含有A2的块，同样，LIST1 中含有第一个元素的块还含有指针，它指向表1的最后一个元素，反之亦然。LIST2 是单链表，其中每一块含有一个指针，它指向表中第一块，LIST3 是链表，它带有从最后一个元素到第一个元素的指针，图1.17还显示一种重要思想：无论多么复杂的数据结构都可以使用指针。

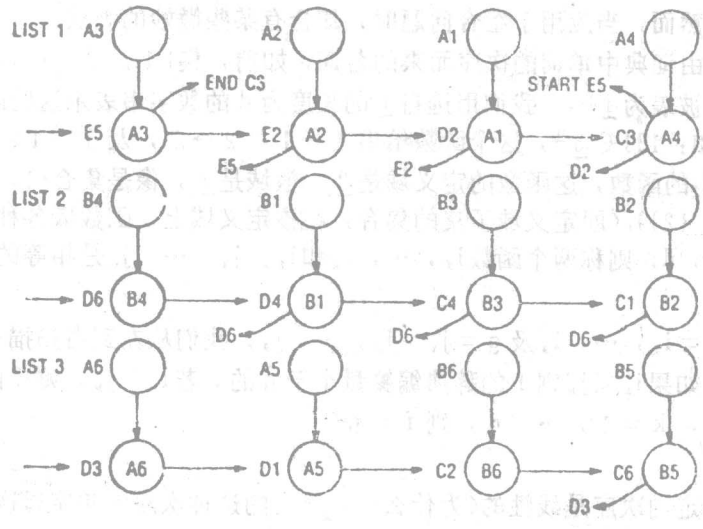
除了上列表外还有一种广义表，该表的元素表示 LIST1，LIST2，LIST3 的第一个条目

表 1

	1	2	3	4	5	6
A	D_2 LIST 1	E_2 LIST 1	E_3 LIST 1 (START)	C_3 LIST 1	D_1 LIST 3	D_3 LIST 3 (START)
B	D_4 LIST 2	C_1 LIST 2	C_4 LIST 2	D_5 LIST 2 (LTART)	C_5 LIST 3	C_2 LIST 3
C	B_2 LAST ENTRY START D_4	B_6 AHEAD C_5	A_4 BACK D_2 LAST ENTRY START E_3	B_3 AHEAD C_1 START D_5		B_5 START D_3
D	A_3 AHEAD C_2	A_1 BACK E_2 AHEAD C_3	A_5 AHEAD D_1	B_1 AHEAD C_4 START D_5	E_3 (LIST 1) AHEAD E_5	B_4 AHEAD D_4 FIRST ENTRY
E	D_5 (LIST 2) AHEAD D_3	A_2 BACK E_3 AHEAD D_2			A3 FIRST ENTRY AHEAD E_2 END C_3	D_3 (LIST 3) START E_1

(TOP)

STRUCTURAL DIAGRAMS:



(Bottom)

图1.17

地址，广义表在E1开始，并含有地址D6，它是 LIST2 的第一个条目地址， 在E1中有指针 AHEAD 指向广义表中的下一个块即D_5，D5中含有LIST1的起始地址E_5及指针 AHEAD 指向E_6，E6是广义表最后一个块，它含有 LIST3 的第一个条目的地址D_9和一个指向广义表的开始的指针。注意到地址列D6，E5，D3定义了 LIST1，LIST2，LIST3 上的线性次序，如何将新表加进广义表中？如何变换广义表中的次序呢？所选方法的基本优越性是容易地做下述操作：仅移动表中的指针而不改变其条目，这样，读者应考虑到如果表被扩大将会发生什么？这个模型与一个人在编程中的实践有怎样的关系？直接查找法真是无意义的呢，还是给你一些有用的思想？

图1.18给出了一种可行记号的提示，当数据结构很复杂(如为图结构)时，用于数据的好的结构图表的记法是很重要的。

1.18 一般情况的结构图表

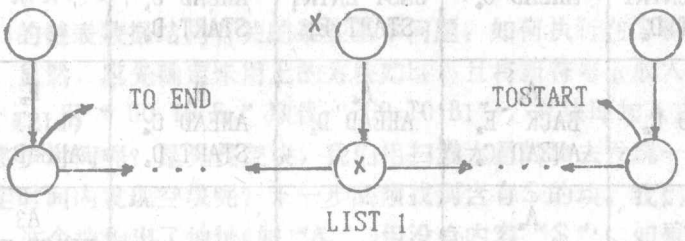


图1.18

1.19 练习

对图1.17按次序作变换：从各种表插入和删去元素及交换表的次序，试详细地讨论这时的数据结构如何样？通常考虑给出很大的整数m和n(长度为n的m个表)，在“固定时间”内作这些交换。

在线性有序集的乘积上构造线性次序的重要方法含有字母次序的观念。这是一种为很多读者熟悉的观念，然而，当应用于组合问题时，却含有某些微妙的观点。

“字母次序”是由词典中单词的次序而来的名词，如前，集{1，2，…，n}被表示为$\underline{n}$，从$\underline{d}$到$\underline{r}$的函数被表为$\underline{r}^{\underline{d}}$，我们用选自$\underline{r}$的长度为d的数字串表示这些函数（“one line”记号法)。例如：$131 \in \underline{3}^{\underline{3}}$，这个函数给出1→1，2→3，及3→1。当然，131也可看作$\underline{r}^{\underline{3}}$，$r \geqslant 3$中的函数，这函数的定义域是$\underline{3}$，值域是$\underline{r}$，像是集合{1，3}，coimage是$\underline{3}$的划分{{1，3}，{2}}，(即定义域子集的集合，在该定义域上，函数取各种值。)，如果$i_k = j_k$，$k = 1, 2\cdots, d$，则称两个函数i_1，…，i_d和j_1，j_2，…，j_d是相等的。

1.20 定义

已知两个函数$f = i_1$，…，i_d及$g = j_1$，j_2，…，j_d，我们从左到右扫描一遍，找到第一个k满足$i_k \neq j_k$，如果$i_k < j_k$则f的辞典编纂量小于g的。若$i_k > j_k$，则f的辞典编纂量大于g的，若$i_k = j_k$，$k = 1$，…，d，则$f = g$。

1.21 评论

由定义1.20所确定的次序是线性的(为什么？)，$\underline{r}^{\underline{d}}$上的这种次序名叫字母次序，简称字典序，在定义1.20中把扫描改为自右至左后所得次序称为伴同字典序。

用字典序我$\underline{3}^{\underline{2}}$中33131231后面的函数，我们将可以增加值的最右边一位加1，并在该值右边的所有位置上放1，于是3313131是下一个函数，在伴同字典序中，将可以增加值的最左

边一位加1，并在该值左边所有位置上放1，于是1123123是伴同字典序中的下一个函数。$\underline{3}^{\underline{3}}$中的函数用字典序的开头部分：111，112，113，121，122，123，131，等等，函数用伴同字典序的开始部分：111，211，311，121，221，321，131等等。用字典序的排列(或单射或一对一映射)是123，132，213，231，312，321.用伴同字典序的排列是321，231，312，…，月有趣的是不减函数用字典序排为：111，112，113，122，123，133，222，223，233，333，用伴同字典序排得：111，112，122，222，113，123，223，133，233，333。可知任一线性有序集的子集按某种次序自动地排成线性次序，(如：像$\underline{3}^{\underline{3}}$的不减函数那样的子集)。

1.22 练习

(1) 用字典序与伴同字典序将$\underline{3}^{\underline{3}}$中的27个函数排成表。在每张表中寻不减函数，并导出这些子集上的次序。什么是这些不同表的反射映象？(用颠倒次序地写出每个元素而获得的表称为表的反射映象)。

(2) 写出$\underline{n}$上所有排列按字典序列表的计算机程序式，对n＝4，5的执行程序。

(3) 用字典序及伴同字典序将$\underline{5}^{\underline{3}}$中的严格递增函数排成表。这些表对应于$\underline{5}$的元素个数为3的子集。

字典序和伴同字典序的观念容易地扩展到线性有序集的乘积$A_1\times\cdots\times A_n$。例如，假设$\underline{2}$上用自然次序且a＜b，α＜β，用字典序及伴同字典序将$\underline{2}\times\{a,b\}\times\{\alpha,\beta\}$的元素列成表，开始部分为1aα，1aβ，1bα，1bβ等等以及1aα，2aα，3aα，1bα等等。

还可以递归地定义一种 字母次序，考虑线性有序集A_i的乘积$A=A_1\times A_2\times\cdots\times A_n$，如果n＝1，则A上的字典序应该是$A_1$上的次序，一般地，如果用$A_1$上的次序$a_1<a'_1$或者$a_1=a'_1$且$(a_2,\cdots,a_n)$的辞典编纂量小于$A_2\times A_3\times\cdots\times A_n$中的$(a'_2,\cdots,a'_n)$，则$(a_1,\cdots,a_n)$的辞典编纂量小于$(a'_1,a'_n)$。

1.23 练习

(1) 递归地定义伴同字母次序。证明字典序及伴同字典序的递归定义给出了如定义1.20与评论1.21那样的某些线性次序。

(2) 令$(L_1,\preceq)$和$(L_2,\preceq)$是线性有序集，如果L_1中$x\preceq x'$，L_2中$y\leqslant y'$，则通过$(x,y)\preceq(x',y')$定义了$L_1\times L_2$上的关系。利用练习1.13(3)证明这种关系是一种线性关系。考虑$L_1=\underline{2}\times\underline{2}=L_2$，$L_1$和$L_2$中用字母次序，画出$L_1\times L_2$上次序的哈希图。

字典序的这种递归定义给出了怎样做卡片分类工作的清晰而抽象的构思 假定给出了A_n的某些子集B_n，希望使B_n的元素取得字母次序，为了更加精确，关于被包含的数据结构，假定B_n的元素(长为n的序列)被写在卡片上，每个元素一张卡片。为了简化符号，假定$A_1=\cdots=A_n=\underline{m}$，设想有m个桶，卡片可以按适当的次序进入这些桶，用某种次序给出B_n的卡片，他们将按某种规则被分配进入各种桶，这种规则说明如下：关键一点是仔细地将卡片放进桶，致使在每个桶的范围内卡片按保持他们刚刚有的同一关系次那样的法则被堆积，分类算法是按照卡片最右边的符号先将他们放进桶，然后，卡片在桶中被移动并且被连结在一起(连锁)，再根据左邻符号重复这个过程，如此继续，这个过程用图1.24；对n＝2的情况加以说明。

1.24 辞典式桶排序

A.从顶读到底，表按某种次序列出的，但它不是辞典序。

B.表的值根据最右边的值放进“桶”内，每一个桶中的次序与开始时一样，于是，桶1