组合数学

算法与分析

下册

卢开澄

清华大学出版社

内 容 简 介

全书分上下两册，共十一章。上册六章；排列组合，母函数与递推关系，容斥原理与鸽子巢原理，Pólya 定理，线性规划，区组设计。下册五章；搜索技术与整数规划，动态规划，优先策略分治策略与快速算法，分类与查找，NP 理论与近似解法。

本书以研究算法与算法分析为中心内容。理论联系实际。适合于计算机科学系、数学系，以及经济系运筹学专业师生作为教材或参考书。

组合数学—算法与分析（下册）

卢开澄　编著

☆

清华大学出版社出版

北京　清华园

清华大学印刷厂印刷

新华书店北京发行所发行·各地新华书店经售

☆

开本：850×1168　1/32　印张：12　印数：304（千字）

1983年11月第一版　1983年11月第一次印刷

数印 1—35000

统一书号：15235·85　定价1.60元

前　言

快速电子计算机的出现是本世纪的大事。它的蓬勃发展改变了这个世界的面貌，也促进了数学的研究。比如从五十年代初期以来，在它的影响下计算数学所取得的成就，使得许多困难问题有了有效的解决工具。

计算机科学是研究算法的科学。本书的主要目的在于讨论不同于“计算方法”的一类算法，称之为“组合算法”。问题来源于计算机、数学、运筹学等方面的离散对象，近年来发展速迅，其态势颇类似于五十年代的计算方法。

计算方法研究的是连续变量的离散化，从而引起了对差分格式的收敛性和稳定性研究。与之相应的，组合算法除了讨论算法之外，还要研究算法的复杂性，即对算法所需的时间和存储单元作出估计（也就是所谓的“时间复杂性”和“空间复杂性”）。

本书的第一部分是讨论“组合分析”，它是基础理论。组合分析和组合算法的关系十分类似于“数学分析”之与“计算方法”。

当然，组合算法还很年轻。随着它的发展，许多带有普遍性的规律将逐步形成。本书仅就组合数学研究些什么问题作个介绍。大致可分为以下几部份：

(1) 组合分析（第 1—4 章），这一部份是全书的基础，为后面的讨论作准备。主要研究计数与枚举。

(2) 组合优化（第 5、7 章），讨论线性规划及整数规划问题。

(3) 组合设计（第 6 章），这一部分内容不仅在科学的实验中有重大的意义，还和编码理论有密切关系。

(4) 算法与算法分析（第 7—11 章）第 7 章介绍搜索法，特别

是 DFS 算法与分支定界法，并结合整数规划深入讨论。第 8 章动态规划，它实际上是算法的一种，这一点与线性规划和整数规划不同。第 9 章优先策略与分治策略。第 10 章分类与查找，重点在于算法的分析。第 11 章讨论 NP 理论及近似算法。

“图论”本是“组合数学”这个“家族”的一个主要成员，但它已成长壮大，独立出去，故这里不再重复。

本书是作者在清华大学计算机工程与科学系，先后对研究生及本科生讲授该课的基础上写成的。由于作者水平有限，错误在所难免，望读者指正。

目　录

第七章　搜索技术与整数规划

有许多实际上属于整数规划问题的求解要借助于穷举，但为了提高效率，更快地找到答案，这一章里将通过实例介绍几种常用的搜索技术，特别是 DFS 算法与分支定界法．最后讨论如何把这些方法用之于一般的整数规划问题．

§ 1　DFS 搜索法举例

DFS 是 Depth First Search 的缩写，有向纵深搜索，或深度优先搜索法的意思．举例说明如下．

例 1．设有一 4×4 的棋盘（即每行每列都有 4 个正方格的棋盘），用 4 个棋子布到格子上，要求满足以下两个条件：

（a）任意两个棋子不在同一行和同一列上；

（b）任意两个棋子不在一对角线上．

试问有多少种的布局？

如若采用穷举法，先不考虑（a）、（b）两个条件，则布到第 1 行的棋子可能有第 1 列到第 4 列的 4 种选择；布到第 2 行的棋子也有第 1 列到第 4 列的 4 种选择；第 3 行、第 4 行也都如是．故共有 $4^4=256$ 种方案．问题是对这 256 种方案逐个检查，从中找出合乎要求的布局来．设 i、j、k、l 分别为棋盘的 4 行；下面是一种算法．

```
begin
    for i:=1 step 1 until 4 do
```

```
    for j:=1 step 1 until 4 do
    begin
        if (i-j)*(abs(i-j)-1)≧0 then
        begin
            for k:=1 step 1 until 4 do
            begin
            if (i-k)*(j-k)*(abs(k-j)-1)*
            (abs(i-k)-2)≧0 then
                begin
                    for l:=1 step 1 until 4 do
                    begin
                    if (l-k)*(l-j)*(l-i)*
                    (abs(l-i)-3)*(abs(l-j)-2)*
                    (abs(l-k)-1)≧0 then
                        print (i, j, k, l)
                    end
                end
            end
        end
    end
end
```

读者可很快发现完全没必要逐一穷举，因为第 1 行的棋子若布到第 1 列，则排除了第 2 行的棋子布到第 1 列的可能性．若用 (h, i, j, k) 表示第 1 行的棋子布到第 h 列，第 2 行的棋子布到第 i 列，第 3 行的棋子布到 j 列，第 4 行的棋子布到第 k 列，其中 $1 \leqslant h、i、j、k \leqslant 4$．由于不允许 $h=i=1$，故排除了 $(1, 1, j, k)$ 的一切可能（共 16 种）．

当然 $h=1$、$i=2$ 也是不允许的，故又排除了 $(1, 2, j, k)$ 的

16 种可能。

$h=1$，$i=3$ 时未出现第 1 行和第 2 行的棋子同列或在一对角线上，则进而考虑 (1，3，j，k) 的可能性。

搜索过程形象表达如图 7—1—1。

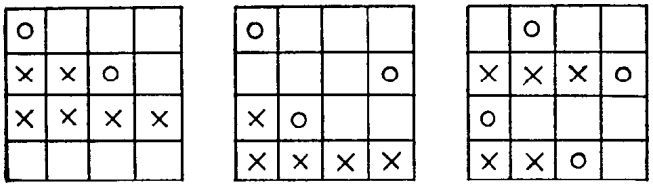


图 7—1—1

○为棋子，×表示不符合条件的格子。

这种一旦发现前面已是“此路不通”，立即回头，改换路径，而不是一条道走到底的策略就是所谓的 DFS 算法的基本思想。它可形象地概括为一句话“向前走，碰壁回头”。现在用 Algol 形式的语言描述 $n\times n$ 的棋盘问题的 DFS 算法如下，其中

$$abs(a[i]-a[j])-abs(i-j)=0$$

是第 i 行的棋子所在的位置和第 j 行棋子所在的位置正好在一对角线上的充要条件。

```
procedure dfs(n);
integer n;
begin integer i, j, k, l;
  array a[1:n];              /* a 记录各行棋子的位置*/
  for i:=1 step 1 until n do a[i]:=1;
  i:=1;
k1: j:=a[i];
  if j≤n then
  begin
```

```
  for k:=1 step 1 until i-1 do
  begin
  if (a[k]-j)*(abs(a[k]-j)-abs(k-i))=0
  then begin a[i]:=a[i]+1; goto k1 end
  end;                    /* 不合格，移一位 */
  i:=i+1;                 /* 向前推进一步 */
  if i≤n then goto k1;
  print a;                /* 打印一种布局 */
  a[n]:=a[n]+1;
  i:=n; goto k1
  end else
    begin
    a[i]:=1; i:=i-1;          /* 碰壁回头 */
    if i<1 then stop;         /* 搜索完毕 */
    a[i]:=a[i]+1; goto k1
    end
end
```

例 2. DFS 搜索法是一种用途很广的算法，尤其在人工智能上，其基本思想经常用到. 例如下面一种 33 个方格的棋盘，开始时仅中央一方格空着，其它 32 个格子上都布着棋子. 每一格子用一个数作为标志：

		1	2	3		
		4	5	6		
7	8	9	10	11	12	13
14	15	16	○	18	19	20
21	22	23	24	25	26	27
		28	29	30		
		31	32	33		

一棋子在棋盘上移动服从以下的规则，它可以跨越过在水平或垂直方向的相邻的棋子进入另侧相邻的空格，并“吃”掉该相邻的棋子。比如格子 5 的棋子跨越过格子 10 进入空格 17，表以 5—7，结果格子 5 和 10 空着，得

```
          1   2   3
          4   ○   6
  7   8   9   ○  11  12  13
 14  15  16  17  18  19  20
 21  22  23  24  25  26  27
         28  29  30
         31  32  33
```

要求找到一种步骤，使得全盘只剩下一个棋子留在棋盘的中央（即格子 17）。

通过 DFS 搜索法可以搜索到一种步骤，使之达到目的。其中之一如下：

5—17，12—10，3—11，1—3，18—6，3—11，

30—18，27—25，13—27，24—26，27—25，22—24，

31—23，33—31，16—28，31—23，4—16，7—9，

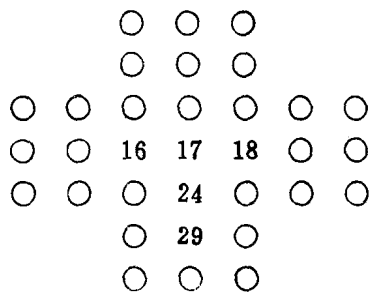
21—7，10—8，7—9。

得如下的图象

```
          ○   ○   ○
          ○   ○   ○
  ○   ○   9   ○  11   ○   ○
  ○  15  16  17  18  19   ○
  ○   ○  23  24  25   ○   ○
          ○  29   ○
          ○   ○   ○
```

继续作

24—22，22—8，8—10，10—12，12—26，26—24。



在此基础上作

17—19，29—17，16—18，19—17

便得所要求的结果。

读者可自行找到不同的步骤。方案可以不同，但用的策略大致相似，只要走得通，就继续往前走，使棋盘上的棋子减少。此路不通时就退回来，改换路径再继续搜索。

例 3. 邮票问题

邮局发行一套票面有 n 种不同值的邮票若限制每封信所贴的邮票个数不得超过 m 枚，存在整数 r，使得用不超过 m 枚的邮票可以贴出以下序列整数值：

$$1、2、3、\cdots、r。$$

比如（1，4，7，8）的邮票，用不超过 3 张邮票可以贴出邮资为

$$1、2、3、4、\cdots、24$$

中的任何一种。如 1，$1+1=2$，$1+1+1=3$，4，$1+4=5$，$1+1+4=6$，7，8，$1+8=9$，$1+1+8=10$，$4+7=11$，$4+8=12$，$1+4+8=13$，$7+7=14$，$7+8=15$，$8+8=16$，$1+8+8=17$，$4+7+7=18$，$4+7+8=19$，$4+8+8=20$，$7+7+7=21$，$7+7+8=22$，$7+8+8=23$，$8+8+8=24$。

问题可以形式地描述如下. 已知非负的整数 n 和 m, 令 $S=\{s_1, s_2, \cdots, s_n\}$ 为不同的邮票集合, 而且

$$s_1<s_2<\cdots<s_n,$$

存在另一个非负整数集合 $T=\{t_1, t_2, \cdots, t_n\}$, 使得

$$r=\sum_{i=1}^{n} t_i s_i, \quad \sum_{i=1}^{n} t_i \leqslant m.$$

令 $val(S, n, m)$ 或简记为 $val(S)$ 为由 S 不能表达的最小整数. 例如对于 $s=\{1, 4, 7, 8\}$, $val(S)=25$.

给定了 m 和 n 后存在集合 S, 使得值 $val(S,n,m)$ 达到了最大值, 设为 $V(n,m)$.

若引进 $s_0=0$, 则问题可以简化为每封信正好贴 m 枚邮票, 而不是不超过 m 枚邮票.

所谓的邮票问题是对于给定了 m 和 n, 存在这样的一组 n 个非负整数的集合 S, 使得

(a) 从 n 个整数中适当取出 m 个来, 允许重复, 使得它的和能表达下列序列中时任何一个整数, 即

$$1, 2, 3, \cdots, N-1.$$

(b) 使得 (a) 中的整数 N 尽可能地大.

设 $S=\{s_0=0, s_1, \cdots, s_n\}$, 其中

$$s_0<s_1<s_2<\cdots<s_n,$$

显然有

$$s_{i-1}<s_i \leqslant val(\{s_0, s_1, \cdots, s_{i-1}\}), \quad 1 \leqslant i \leqslant n,$$

而且

$$s_i \leqslant V(n,i) \leqslant \binom{n+i}{n}.$$

$\binom{n+i}{n}$ 是 $n+1$ 个中取 i 个作允许重复的组合的组合数.

邮票问题可分解为两个部分，一是对于给定的 m、n，列举有关的集合 S；一是对于集合 S 计算 $val(S)$。

对于已知的 m，$n=1,2,\cdots$ 进行搜索时可利用下面的不等式，即

$$s_{i-1}<s_i\leqslant val(\{s_0,\ s_1,\cdots,\ s_{i-1}\}).$$

若 v 为搜索过程中 $val(S,n,m)$ 的最佳结果，剪去

$$s_n<\frac{v-1}{m},\qquad s_{n-1}<\frac{v-1}{m^2}$$

的分枝(Lunnon,1969)．下面是 $V(n,m)$ 表，相应的集合 S 见下页．

$m\diagdown n$	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	5	9	13	17	21
3	4	8	16	25	37	53
4	5	11	27	45	71	109
5	6	15	36	72	127	212
6	7	19	53	115	217	389

例 4. 已知连通的无向图 $G=(V,E)$，利用 DFS 法可生成 G 的一棵树，办法如下：

(a) 设 n 个顶点顺序为 v_1、v_2、$\cdots$、v_n。从 v_1 出发，给 v_1 以标号并进栈。

(b) 设栈顶元素为 v_i，在未给标号的顶点集合中若存在与 v_i 相邻序号最小的点 v_j，给 v_j 点及边 $(v_i,\ v_j)$ 以标号，v_j 进栈；如若不然，v_i 退栈。

(c) 若栈空，则已标号的边便是图 G 的树。否则转 (b)。

m \ n	1	2		3			4				5					6					
1	1	1	2	1	2	3	1	2	3	4	1	2	3	4	5	1	2	3	4	5	6
2	1	1	2	1	3	4	1	3	5	6	1	3	5	7	8	1	2	5	8	9	10
		1	3													1	3	4	8	9	11
																1	3	4	9	11	16
																1	3	5	6	13	14
																1	3	5	7	9	10
3	1	1	3	1	4	5	1	4	7	8	1	4	6	14	15	1	3	7	9	19	24
																1	4	6	14	17	29
4	1	1	3	1	5	8	1	3	11	18	1	3	11	15	32	1	4	9	16	38	49
		1	4													1	5	8	27	29	44
5	1	1	4	1	6	7	1	4	12	21	1	4	9	31	51	1	4	13	24	56	61
							1	5	12	28						1	5	8	33	54	67
6	1	1	4	1	7	12	1	4	19	33	1	7	12	43	52	1	7	11	48	83	115
		1	5																		

m、n 对应的最佳集合 S 表格

§ 2 旅行商问题

设已知 n 个城市间的旅费矩阵

$$D=(d_{ij})_{n\times n}\ .$$

其中 d_{ij} 为从第 i 个城市到第 j 个城市间的旅费。求从某一城市出发，遍历每个城市各一次，最后返回出发地点旅费最少的路径。若采用穷举法则需要对 $(n-1)!$ 种可能方案进行比较。下面利用分支定界法加速搜索过程，举例说明如下：

$$D=\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ \\ \end{array}\begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{pmatrix} \infty & 24 & 34 & 14 & 15 \\ 19 & \infty & 20 & 9 & 6 \\ 7 & 9 & \infty & 6 & 8 \\ 23 & 10 & 22 & \infty & 7 \\ 20 & 8 & 11 & 20 & \infty \end{pmatrix} \end{array}=(d_{ij})_{5\times 5}.$$

对 D 的每行减去该行的最小的元素，或每列减去该列最小元素得一新的矩阵。比如

$$\begin{pmatrix} \infty & 24 & 34 & 14 & 15 \\ 19 & \infty & 20 & 9 & 6 \\ 7 & 9 & \infty & 6 & 8 \\ 23 & 10 & 22 & \infty & 7 \\ 20 & 8 & 11 & 20 & \infty \end{pmatrix}\begin{matrix} -14 \\ -6 \\ -6 \\ -7 \\ -8 \end{matrix}\Rightarrow\begin{pmatrix} \infty & 10 & 20 & 0 & 1 \\ 13 & \infty & 14 & 3 & 0 \\ 1 & 3 & \infty & 0 & 2 \\ 16 & 3 & 15 & \infty & 0 \\ 12 & 0 & 3 & 12 & \infty \end{pmatrix}\Rightarrow \quad b=41$$

$$-1 \qquad -3$$