4.1 常用图论模型与计算

图论中最基本的概念是顶点。在实际问题中，许多对象都可以当作顶点，如公交网络中的各个站点可以是顶点，参加比赛的各队可以看作是顶点，环球旅游时经过的各个城市可以看作是顶点。

顶点组成的集合称为顶点集，记作V；顶点之间的连接称为边，边组成的集合称为边集，记作E。这样一个图就由(V,E)组成，记作$G=(V,E)$。通常G的顶点集用$V(G)$表示，G的边集用$E(G)$表示。如果图中的定点赋予顶点和边具体的含义和权，这样该图就称为网络。这时该网络可表示为$N=(V,E,W)$，W由任意两点之间的距离构成权重矩阵。

如在研究环球旅游时，各城市构成顶点，城市之间的连接构成边，城市之间的距离就是这两点间边的权重。

4.1 图论

如果边的连接是有方向的，称为有向图；如果边之间的连接是无向的，则称为无向图。对一个实际问题，到底应该考虑为无向图还是有向图，需要根据实际问题与背景来区分。如考虑环球旅游问题，各城市之间连接可当作无向图来研究。研究参赛队伍的排名时，两队之间的胜败是需要区分的，这时要考虑为有向图。

在实际问题中，许多问题都可以抽象为图论模型进行求解。在数学建模竞赛中，不少问题最终也可以利用图论中的知识求解。在历年的竞赛中，如全国数学建模竞赛中1993B的足球队排名问题，1994B的锁具开箱问题，1998B的洪灾巡视问题，2011B的交巡警平台问题，还有电工杯数学建模竞赛中2005B的比赛项目排序问题。这些赛题中部分或全部使用到图论的知识进行求解。

本节分为五个部分，每个部分尽量从实用出发，讲解少量的知识，然后利用相关知识求解实际问题，并结合赛题进行讲解。在求解时给出相应的程序，便于读者上手操作。

4.1 图论

目录

4.1.1图论中TSP问题及Lingo求解技巧

问题背景

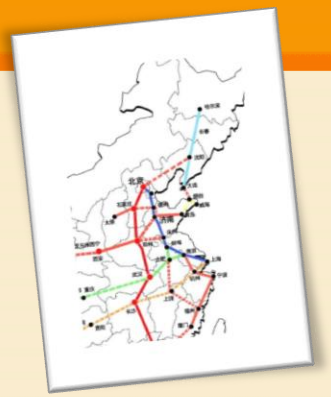
巡回旅行商问题(Traveling Salesman Problem，TSP)也称为货郎担问题。最早可以追溯到1759年Euler提出的骑士旅行问题。1948年，由美国兰德公司推动，TSP成为近代组合优化领域的一个典型难题。它已经被证明属于NP难题。

求解原理

用图论描述TSP，给出一个图G=(V,E),每边$e \in E$上有非负权值w(e),寻找G的Hamilton圈C,使得C的总权$W(C)=\sum_{e\in E(C)} w(e)$最小。

几十年来，出现了很多近似优化算法，如近邻法、贪心算法、最近插入法、最远插入法、模拟退火算法以及遗传算法。这里，我们介绍利用Lingo软件进行求解的方法。

目录



问题一：城市总路程最短问题

问题描述

设有一个售货员从10个城市中的某一个城市出发，去其他9个城市推销产品。10个城市相互距离如下表。要求每个城市到达一次且仅一次后，回到原出发城市。问他应如何选择出行路线，使总路程最短。

表1：10个城市距离表　　（距离：km）

城市 / 距离 / 城市	1	2	3	4	5	6	7	8	9	10
1	0	7	4	5	8	6	12	13	11	18
2	7	0	3	10	9	14	5	14	17	17
3	4	3	0	5	9	10	21	8	27	12
4	5	10	5	0	14	9	10	9	23	16
5	8	9	9	14	0	7	8	7	20	19
6	6	14	10	9	7	0	13	5	25	13
7	12	5	21	10	8	13	0	23	21	18
8	13	14	8	9	7	5	23	0	18	12
9	11	17	27	23	20	25	21	18	0	16
10	18	17	12	16	19	13	18	12	16	0

采用线性规划的方法求解

设城市之间距离用矩阵 来表示， 表示城市 与城市 之间的距离。设0-1矩阵 用来表示经过的各城市之间的路线。设

$$x_{ij}=\begin{cases}0 & \text{若城市i不到城市j}\\ 1 & \text{若城市i到城市j，且在j前}\end{cases}$$

考虑每个城市后只有一个城市，则：

$$\sum_{\substack{j=1\\ j\neq i}}^{n} x_{ij}=1,\quad i=1,L\ ,n$$

考虑每个城市前只有一个城市，则：

$$\sum_{\substack{i=1\\ i\neq j}}^{n} x_{ij}=1,\quad j=1,L\ ,n$$

但仅以上约束条件不能避免在一次遍历中产生多于一个互不连通回路。

如对n=6，下面矩阵表达的方案满足前面两个约束，即每行每列和为1。但该方案结果为1→2→3→1，4→5→6→4。 即节点1,2,3构成子圈； 节点4,5,6构成子圈，不符合条件。

$$\boldsymbol{x}=\begin{bmatrix}0&1&0&0&0&0\\0&0&1&0&0&0\\1&0&0&0&0&0\\0&0&0&0&1&0\\0&0&0&0&0&1\\0&0&0&1&0&0\end{bmatrix}$$

为此我们引入额外变量$u_i(i=1,\mathrm{L}\ ,n)$，附加以下充分约束条件：

$$u_i-u_j+nx_{ij}\le n-1,\quad 1<\mathrm{i}\ne\mathrm{j}\le\mathrm{n};$$

该约束的解释：

如i与j不会构成回路，若构成回路，有：

$x_{ij}=1,x_{ji}=1$,则：

$\mathrm{u}_i-u_j\le-1,u_j-u_i\le-1$,从而有:$0\le-2$，导致矛盾。

如i , j与k不会构成回路，若构成回路，有：

$x_{ij}=1, x_{jk}=1, x_{ki}=1$则：

$u_i-u_j\le -1, u_j-u_k\le -1, u_k-u_i\le -1$从而有：

$0\le -3$，导致矛盾。

其他情况以此类推。

于是我们可以得到如下的模型：

$$\min Z=\sum_{i=1}^{n}\sum_{j=1}^{n}d_{ij}x_{ij}$$

$$s.t.\begin{cases}\sum\limits_{\substack{i=1\\ i\neq j}}^{n}x_{ij}=1 \quad j=1,L,n\\ \sum\limits_{\substack{j=1\\ j\neq i}}^{n}x_{ij}=1 \quad i=1,L,n\\ u_i-u_j+nx_{ij}\le n-1 \quad 1<i\neq j\le n\\ x_{ij}=0\text{或}1 \quad i,j=1,L,n\\ u_i\text{为实数} \quad \mathrm{i}=1,L,n\end{cases}$$

该模型的Lingo程序为：

```
!TSP question;
MODEL:
SETS:
city/1..10/:u;
link(city,city):d,x;
ENDSETS
DATA:
d= 0   7    4   5   8    6    12   13   11   18
   7   0    3  10   9   14    5   14   17   17
   4    3   0   5   9   10   21    8   27   12
   5   10 5     0  14    9   10    9   23   16
   8   9    9  14   0    7    8    7   20   19
   6   14 10    9   7    0   13    5   25   13
  12    5 21   10   8   13    0   23   21   18
  13   14 8     9   7    5   23    0   18   12
  11   17 27   23  20   25   21   18    0   16
  18   17 12   16  19   13   18   12   16   0;
ENDDATA
   MIN=@SUM(link:d*x);
   @for(city(j):@sum(city(i)|j#ne#i:x(i,j))=1); !城市 j 前有一个城市相连;
   @for(city(i):@sum(city(j)|j#ne#i:x(i,j))=1); !城市 i 后前有一个城市相连;
   @for(link(i,j)|i#NE#j#and#i#gt#1:u(i)-u(j)+10*x(i,j)<=9);
@FOR(link:@BIN(x));
End
```

得到的结果为：

X(3,2)=1,　　X(4,1)=1,
X(4,3)=1,　　X(6,5)=1,
X(7,2)=1,　　X(7,5)=1,
X(8,6)=1,　　X(9,1)=1,
X(10,8)=1,　　X(10,9) =1。

其他全为0。其最短路线线为1-4-3-2-7-5-6-8-10-9-1，最短距离为77 km。

问题二：比赛项目排序问题

问题描述

全民健身计划是1995年在国务院领导下，由国家体委会同有关部门、各群众组织和社会团体共同推行的一项依托社会、全民参与的体育健身计划，是与实现社会主义现代化目标相配套的社会系统工程和跨世纪的发展战略规划。现在，以全民健身为主要内容的群众性体育活动蓬勃开展，举国上下形成了全民健身的热潮，人民群众健康水平不断提高，同时也扩大了竞技体育的社会影响，提高了竞技体育水平。现在各级、各类、各种运动比赛比比皆是，这不但提高了全民的身体素质，而且使一批运动员脱颖而出，成为运动健将，为国家争得了荣誉。

在各种运动比赛中，为了使比赛公平、公正、合理地举行，一个基本要求是：在比赛项目排序过程中，尽可能使每个运动员不连续参加两项比赛，以便运动员恢复体力，发挥正常水平。

（1）表2是某个小型运动会的比赛报名表。有14个比赛项目，40名运动员参加比赛。表中第1行表示14个比赛项目，第1列表示40名运动员，表中“＃”号位置表示运动员参加此项比赛。建立此问题的数学模型，并且合理安排比赛项目顺序，使连续参加两项比赛的运动员人次尽可能的少。

（2）文件“运动员报名表”中给出了某个运动比赛的报名情况。共有61个比赛项目，1050人参加比赛。请给出算法及其框图，同时给出合理的比赛项目排序表，使连续参加两项比赛的运动员人次尽可能地少。

（3）说明上述算法的合理性。

（4）对“问题2”的比赛排序结果，给出解决“运动员连续参加比赛”问题的建议及方案。表见P72

表 4-2　某小型运动会的比赛报名表

项目 / 运动员	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		#	#						#				#	
2								#			#	#		
3		#		#						#				
4			#					#				#		
5											#		#	#
6					#	#								
7												#	#	
8										#				#
9		#		#						#	#			
30				#	#									
31						#		#				#		
32							#			#				
33				#		#								
34	#		#										#	#
35					#	#						#		
36				#			#							
37	#								#	#				
38						#		#		#				#
39					#			#	#				#	
40						#	#		#				#	

问题一解答：

若项目i和项目j相邻，可以计算出同时参加这两个项目的人数，作为i和j的距离 d_{ij}。则问题转化为求项目1到项目14的一个排列，使相邻距离和最小。我们采用TSP问题求解。但由于开始项目和结束项目没有连接，可考虑引入虚拟项目15，该虚拟项目与各个项目的距离都为0。

距离矩阵D的求法：

该报名表用矩阵 $A_{40\times14}$ 表示。

$$a_{ij}=\begin{cases}1 & \text{第i个人参加项目j}\\ 0 & \text{第i个人不参加项目j}\end{cases}$$

则 $d_{ij}=\sum_{k=1}^{40}a_{ki}.a_{kj}$ $\quad i\neq j, i, j, =1,2,\cdots,14$ $\quad d_{ij}=0$ $\quad i=1,2,\cdots,14$

另外 $d_{i,15}=0, d_{15,i}=0$ $\quad i=1,2,\cdots,15$

Lingo代码

```
load table1.txt;
a=table1;
[m,n]=size(a);
d=zeros(n+1,n+1);    %定义距离矩阵;
for i=1:n
for j=1:n
    for k=1:m
      d(i,j)=d(i,j)+a(k,i)*a(k,j);    %计算不同项目之间距离
    end
end
end
for i=1:n+1
       d(i,i)=0;
end
    %输出文件
  fid=fopen('d:\lingo12\dat\dis1.txt','w');
for i=1:n+1
  for j=1:n+1
      fprintf(fid,'%1d ',d(i,j));
    end
  fprintf(fid,'\r\n');
end
  fclose(fid);
```

由于问题1中40个运动员参加14个项目的比赛是word表，可将其拷贝到Excel表中，然后将#替换为1，将空格替换为0，形成0-1表，并拷贝到数据文件table1.txt中。问题2中1050个运动员参加的61个项目比赛的Access数据库中的表保存为Excel表，然后在表中将#替换为1，将空格替换为0，形成0-1表，并拷贝到数据文件table2.txt中。

在matlab中编制如下程序B2005.m形成距离矩阵。

输出的距离矩阵D为dis1.txt：

0	2	1	2	0	0	1	0	1	2	1	1	1	1	0
2	0	1	4	1	0	1	1	1	3	1	0	2	1	0
1	1	0	1	0	0	0	3	1	1	0	2	2	1	0
2	4	1	0	1	1	2	1	0	2	1	0	1	1	0
0	1	0	1	0	2	0	1	1	1	0	1	1	2	0
0	0	0	1	2	0	1	2	1	1	1	2	1	2	0
1	1	0	2	0	1	0	1	1	1	0	2	2	1	0
0	1	3	1	1	2	1	0	1	2	1	4	2	2	0
1	1	1	0	1	1	1	1	0	1	1	1	3	1	0
2	3	1	2	1	1	1	2	1	0	1	0	0	3	0
1	1	0	1	0	1	0	1	1	1	0	3	1	1	0
1	0	2	0	1	2	2	4	1	0	3	0	1	0	0
1	2	2	1	1	1	2	2	3	0	1	1	0	4	0
1	1	1	1	2	2	1	2	1	3	1	0	4	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

其中数据文件dis1.txt在MatLab程序B2005.m中输出。

Lingo程序B2005.lg4：

```
!第一个问题的求解的程序：
!比赛项目排序问题;
model:
sets:
   item / 1.. 15/: u;
   link( item, item):dist,x;
endsets
     n = @size( item);
data:     !距离矩阵;
dist=@file('d:\lingo12\dat\dis1.txt'); !文件路径;
  !输出为 1 的变量;
@text()=@writefor(link(i,j)|x(i,j)#GT#0:' x(',i,',',j,')=',x(i,j));
enddata

   MIN=@SUM(link:dist*x);
   @for(item(j):@sum(item(i)|j#ne#i:x(i,j))=1); !点 j 前有一个点相连;
   @for(item(i):@sum(item(j)|j#ne#i:x(i,j))=1); !点 i 后前有一个点;
!保证不出现子圈;
   @for(link(i,j)|i#NE#j#and#i#gt#1:u(i)-u(j)+n*x(i,j)<=n-1);
@FOR(link:@BIN(x));!定义 X 为 0-1 变量;
end
```

Lingo12求解结果为：目标值z=2

x(1,8)=1　x(2,6)=1　x(3,11)=1　x(4,13)=1　x(5,1)=1
x(6,3)=1　x(7,5)=1　x(8,15)=1　x(9,4)=1　x(10,12)=1
x(11,7)=1　x(12,14)=1　x(13,10)=1　x(14,2)=1　x(15,9)=1

由于15是虚拟项，去掉后对应序列为9-4-13-10-12-14-2-6-3-11-7-5-1-8-9，则项目排序如下，其中箭头上所示数字为连续参加相邻两项目的运动员数。

$$9 \xrightarrow{0} 4 \xrightarrow{1} 13 \xrightarrow{0} 10 \xrightarrow{0} 12 \xrightarrow{0} 14 \xrightarrow{1} 2$$
$$\xrightarrow{0} 6 \xrightarrow{0} 3 \xrightarrow{0} 11 \xrightarrow{0} 7 \xrightarrow{0} 5 \xrightarrow{0} 1 \xrightarrow{0} 8$$

即有两名运动员连续参加比赛。

题2解答与题1相同，只是项目变成61个，引入虚拟项目后变为62个，运动员为1050名。模型建立同题1。在题1中的MatLab程序中只需要将表table1.txt改为table2.txt，输出数据文件将dis1.txt改为dis2.txt就可以了。

在Lingo程序中将项目数由15修改为62，使用的数据文件由15改为62，同样可以运行，只是运行时间较长，本程序在Lingo12中大约运行6分钟左右。原始数据文件table2.txt和MatLab输出的距离矩阵dis2.txt，由于数据较大这里不列出。

Lingo程序:

```
!第二个问题的求解的程序：
!比赛项目排序问题;
model:
sets:
  item / 1.. 62/: u;
  link( item, item):dist,x;
endsets
   n = @size( item);
data:   !距离矩阵;

dist=@file('d:\lingo12\dat\dis2.txt'); !文件路径;
 !输出为1的变量;
@text()=@writefor(link(i,j)|x(i,j)#GT#0:' x(',i,',',j,')=',x(i,j));
enddata

MIN=@SUM(link:dist*x);
  @for(item(j):@sum(item(i)|j#ne#i:x(i,j))=1); !点j前有一个点相连;
  @for(item(i):@sum(item(j)|j#ne#i:x(i,j))=1); !点i后前有一个点;
!保证不出现子圈;
  @for(link(i,j)|i#NE#j#and#i#gt#1:u(i)-u(j)+n*x(i,j)<=n-1);
@FOR(link:@BIN(x));!定义X为0-1变量;
end
```

Lingo12求解结果为：
目标值z=5

x(1,19)=1	x(2,44)=1	x(3,50)=1	x(4,25)=1	x(5,20)=1	x(6,15)=1
x(7,42)=1	x(8,59)=1	x(9,35)=1	x(10,3)=1	x(11,54)=1	x(12,21)=1
x(13,32)=1	x(14,41)=1	x(15,40)=1	x(16,57)=1	x(17,22)=1	x(18,9)=1
x(19,60)=1	x(20,6)=1	x(21,10)=1	x(22,37)=1	x(23,14)=1	x(24,51)=1
x(25,13)=1	x(26,27)=1	x(27,29)=1	x(28,17)=1	x(29,24)=1	x(30,58)=1
x(31,12)=1	x(32,56)=1	x(33,47)=1	x(34,23)=1	x(35,46)=1	x(36,45)=1
x(37,30)=1	x(38,49)=1	x(39,31)=1	x(40,48)=1	x(41,1)=1	x(42,52)=1
x(43,38)=1	x(44,4)=1	x(45,7)=1	x(46,62)=1	x(47,55)=1	x(48,34)=1
x(49,26)=1	x(50,36)=1	x(51,16)=1	x(52,18)=1	x(53,39)=1	x(54,43)=1
x(55,5)=1	x(56,11)=1	x(57,53)=1	x(58,61)=1	x(59,28)=1	x(60,8)=1
x(61,2)=1	x(62,33)=1				

由于62是虚拟项，去掉后对应序列为

33—47—55—5—20—6—15—40—48—34—23—14-41—1—19—60—8—59—28—17—22—37—30—58—61—2—44—4—25—13—32—56—11—54—43—38—49—26—27—29—24—51—16—57—53—39—31—12—21—10—3—50—36—45—7—42—52—18—9—35—46

可以验证，其中d(14,41)=1, d(51,16)=1, d(31,12)=1, d(10,3)=1, d(45,7)=1。其余相邻两个项目比赛没有两名运动连续参加比赛。即有5名运动员连续参加比赛。另外，该问题解不唯一，单目标值都为5。

x(1,41)=1 x(2,61)=1 x(3,10)=1 x(4,44)=1 x(5,55)=1 x(6,20)=1 x(7,45)=1
x(8,60)=1 x(9,18)=1 x(10,21)=1 x(11,54)=1 x(12,31)=1 x(13,32)=1
x(14,23)=1 x(15,6)=1 x(16,57)=1 x(17,28)=1 x(18,52)=1 x(19,1)=1 x(20,5)=1
x(21,12)=1 x(22,17)=1 x(23,34)=1 x(24,51)=1 x(25,4)=1 x(26,27)=1
x(27,29)=1 x(28,59)=1 x(29,24)=1 x(30,37)=1 x(31,39)=1 x(32,56)=1
x(33,62)=1 x(34,48)=1 x(35,9)=1 x(36,50)=1 x(37,22)=1 x(38,49)=1
x(39,13)=1 x(40,15)=1 x(41,14)=1 x(42,7)=1 x(43,38)=1 x(44,2)=1
x(45,36)=1 x(46,35)=1 x(47,33)=1 x(48,40)=1 x(49,26)=1 x(50,3)=1
x(51,16)=1 x(52,42)=1 x(53,25)=1 x(54,43)=1 x(55,47)=1 x(56,11)=1
x(57,53)=1 x(58,30)=1 x(59,8)=1 x(60,19)=1 x(61,58)=1 x(62,46)=1

4.1.2 最短路线算法及在建模中的应用

引言

图论中的最短路线问题（包括无向图和有向图）是一个基本且常常碰见的问题。主要的算法有Dijkstra算法和Floyd算法。其Dijkstra算法是求出指定两点之间的最短路线，算法复杂度$O(n^2)$；Floyd算法是求出任意两点之间的最短路线，算法复杂度为$O(n^3)$。其中n为顶点数。

返回

1. Dijkstra算法

假定给出一个网络 $N=(V,E,W)$,现在要求出任意点i到任意点j 之间的最短路线，算法描述如下：

（1）给出初始点集合 $P=\{i\}$,剩余点集合 $Q=\{1,2,...,n\}-P$。初始i点到各点的直接距离 $U_r=w_{ir}(r=1,2,..,n)$。

（2）在Q中寻找到i点距离最小的点k，使得$U_k=\min\limits_{r\in Q}\{U_r\}$ ，并且 $P\cup\{k\}\to P$, $Q-\{k\}\to Q$

（3）对Q中每个 r，如果$U_k+w_{kr}<U_r$ ，则 $U_k+w_{kr}\to U_r$

然后返回（2）直到找到j点为止。

注意：

步骤（2）和（3）实际上是每找到一个到i点距离最小的点k，就更新一次从i点出发，通过集合P中点到达Q中点的最小距离。

该算法经过$n-1$次循环结束。在整个算法过程中，步骤（2）最多作$\frac{1}{2}(n-1)(n-2)$次比较，步骤（3）最多作$\frac{1}{2}(n-1)(n-2)$次加法和比较，总的计算量是$O(n^2)$阶。因此这个算法是一个有效算法。

2. Floyd算法

1) 根据已知的部分节点之间的连接信息，建立初始距离矩阵 $B(i,j)$，

其中没有给出直接距离的赋予一个充分大数值，以便于更新。$(i,j=1,2,\cdots n)$

2) 进行迭代计算。对任意两点 (i,j)，若存在 k，使 $B(i,k)+B(k,j)<B(i,j)$，

则更新 $B(i,j)=B(i,k)+B(k,j)$

3)直到所有点的距离不再更新停止计算。则得到最短路距离矩阵 $B(i,j)$，

$(i,j=1,2,\cdots n)$。

算法程序为：

```
for k=1:n
for i=1 :n
    for j=1:n
        t=B(i,k)+B(k,j);
        if t<B(i,j)  B(i,j)=t; end
        end
    end
end
```

例：灾情巡视路线问题（CUMCM1998B部分）

右图为某县的乡（镇）、村公路网示意图，公路边的数字为该路段的公里数。今年夏天该县遭受水灾，为考察灾情，组织自救，县领导决定，带领有关部门负责人到全县各乡（镇）、村巡视。巡视路线指从县政府所在地出发，走遍各乡（镇）、村，再回到县政府所在地的路线。若分三组（路）巡视，试设计总路程最短且各组尽可能均衡的巡视路线。

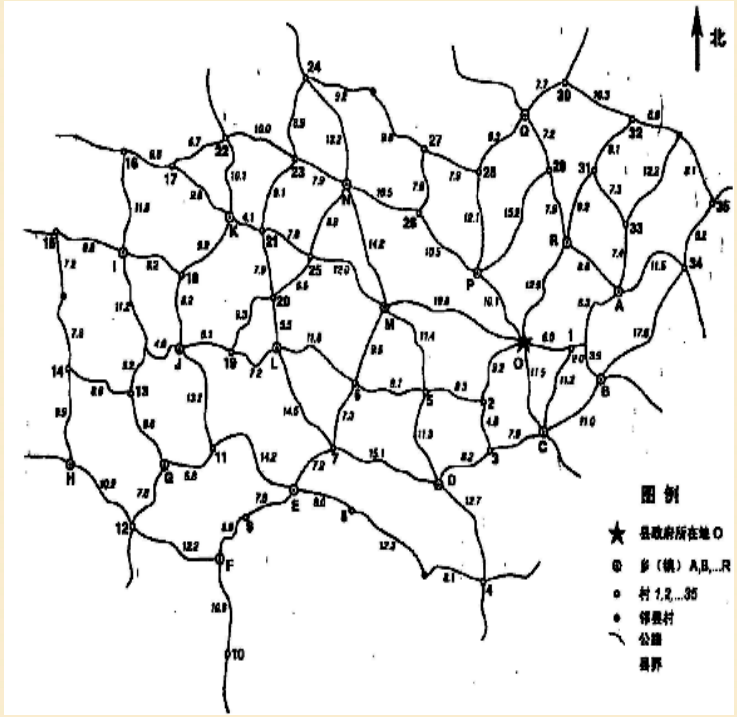


图4-1 某县的乡（镇）、村公路网示意图

问题分析

本题给出了某县的公路网络图，要求的是在不同的条件下，灾情巡视的最佳分组方案和路线。我们可将：

- 每个乡（镇）或村 ⇨ 看作一个图的顶点
- 各乡镇、村之间的公路 ⇨ 看作一个图的对应顶点间的边
- 各条公路的长度 ⇨ 看作一个图对应边上的权

所给公路网就转化为加权网络图，问题就转化为图论中称之为旅行售货员问题，即在给定的加权网络图中寻找从给定点O出发，行遍所有顶点至少一次再回到点O，使得总权（路程或时间）最小。

问题分析

本题是旅行售货员问题的延伸—多旅行售货员问题。

本题所求的分组巡视的最佳路线，也就是m条经过同一点并覆盖所有其他顶点且边权之和达到最小的闭链。该问题是三个旅行售货商问题。众所周知，旅行售货商问题属于NP完全问题，即求解没有多项式时间算法。

鉴于此，一定要针对问题的实际特点寻找简便方法，对于规模较大的问题可使用近似算法来求得近似最优解。

在本题中首先用图论中Floyd算法求出G中任意两个顶点间的最短路线, 并构造出完全图：

$$G' = (V, E'), \forall (x, y) \in E', \omega(x, y) = \min d_G(x, y);$$

问题解答

问题转化为在给定的加权网络图中寻找从给定点0出发，走遍所有顶点至少一次再回到点0，使得总权(路程或时时间)最小，即最佳旅行售货商问题。对三组巡视路线，设计总路程最短且各组尽可能均衡。

在加权图G中求顶点集V的划分V_1，将G分成3个生成子图 $G[V_1]$，$G[V_2]$，$G[V_3]$，使得

① 顶点 $O \in V_i$，$i=1,2,3$ 。

② $\bigcup_{i=1}^{3} V_i = V(G)$

③ $\dfrac{\max\limits_{i,j} |\omega(C_i)-\omega(C_j)|}{\max\limits_{i} \omega(C_i)} \le \alpha$ ，其中 C_i 为 V_i 的导出子图 $G[V_i]$ 中的最佳旅行售货商回路，$\omega(C_i)$ 为 C_i 的权(总路程)，$i, j = 1,2,3$ 。

④ $\sum_{i=1}^{3} \omega(C_i) = \min$ ——总路程最小。

定义 称 $\alpha_0=\dfrac{\max\limits_{i,j}|\omega(C_i)-\omega(C_j)|}{\max\limits_{i}\omega(C_i)}$ 为该分组的实际均衡度. α 为最大容许均衡度。

显然 $0\le\alpha_0\le1$，α_0 越小，说明分组的均衡性越好。取定一个 α 后，α_0 与 α 满足条件③的分组是一个均衡分组，满足条件④表示总巡视路线最短。

另外，此问题包含两方面：a. 对顶点分组，b. 在每组中求(单个售货员)最佳旅行售货员回路。

因单个售货员的最佳旅行售货员回路问题也不存在多项式时间内的精确算法。而图中节点数较多，为53个，我们只能去寻求一种较合理的划分准则，对图4-1进行初步划分后，求出各部分的近似最佳旅行售货员回路的权，再进一步进行调整，使得各部分满足均衡性条件③。

从O点出发去其他点，要使路程较小应尽量走O点到该点的最短路线。

利用Floyd算法中同时求出各点从O点出发的前点，将每点和它对应的前点连接起来，得到如图4-2所示的树状图。该图可以看作从O点出发由6条树干构成的树。

由上述分组准则，我们找到两种分组形式如下。

分组1:（⑥，①），（②，③），（⑤，④）

分组2:（①，②），（③，④），（⑤，⑥）

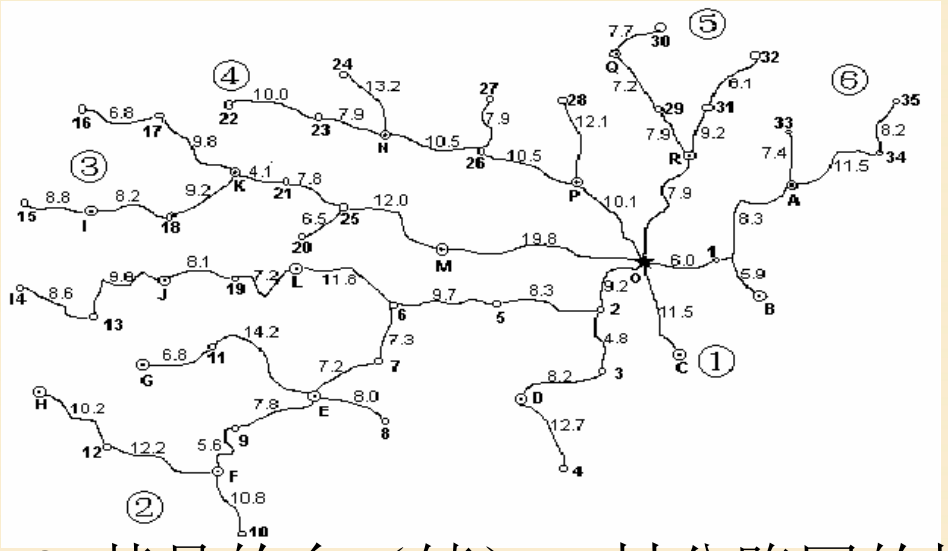


图4-2 某县的乡（镇）、村公路网的树状图

分组1中①和⑥构成的一组点太少，显然这导致与其余两组差异太大，故考虑分组2。

分组2：（①，②），（③，④），（⑤，⑥）

对分组2中每组顶点的生成子图，采用优化算法求出最优解及相应的巡视路线。

（附录1中MatLab程序B98.m(具体程序见P82)输出三条线路的距离矩阵B98G1.txt，B98G2.txt和B98G3.txt，便于Lingo调用进行优化计算）

表4-3　分组2计算结果

分　组	线　路	线路长度	总 长 度
1	O-->C-->3-->D-->4-->8-->E-->9-->F-->10-->12-->H-->14-->13--> G-->11-->J-->19-->L-->7--> 6--> 5--> 2-->O（23 个点）	237.5	554.1
2	O-->P-->28-->27-->26-->N-->24-->23-->22-->17-->16--> 15-->I-->18-->K-->21-->20-->25-->M-->O(19 个点)	191.1	
3	O-->R-->29-->Q-->30-->32-->31-->33-->35-->34-->A-->B--> 1-->O(13 个点)	125.5	

该分组的均衡度

$$\alpha_0 = \frac{\omega(C_1) - \omega(C_2)}{\max\limits_{i=1,2,3} \omega(C_i)} = \frac{237.5 - 125.5}{237.5} = 47.16\%$$

从结果看该分法的均衡性很差。

为改善均衡性，将第1组中的顶点C，2，3，D，4分给第3组，重新分组后的近似最优解见表4-4，调整后的分组树状图如图4-3所示，其中黑色为第1组，黄色为第2组，红色为第3组。

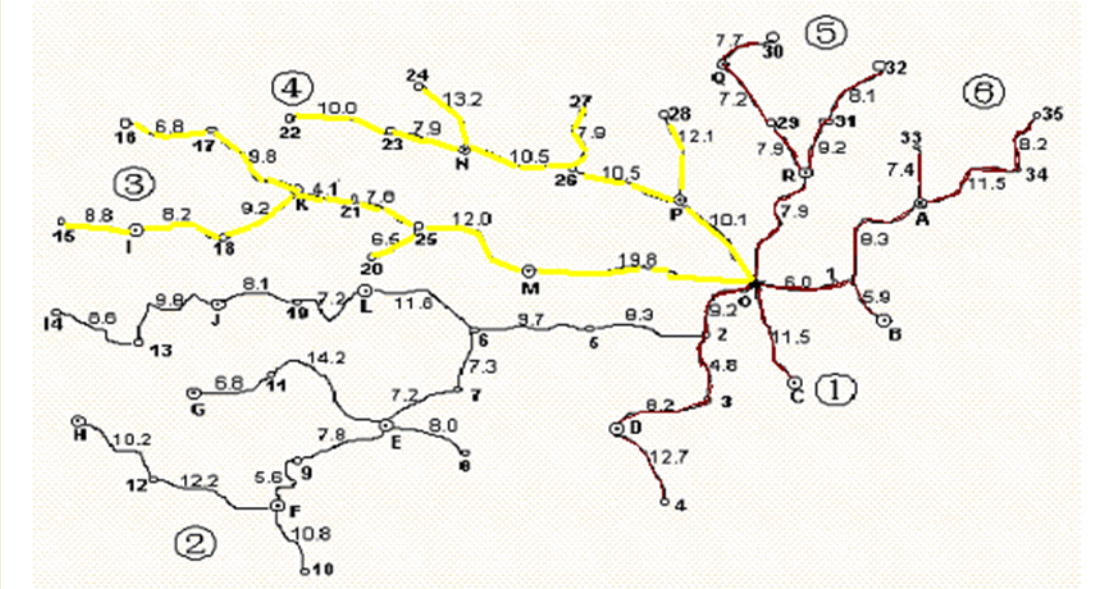


图4-3　调整后的分组树状图

计算n个点之间最优巡回线路的模型

设两个点之间距离用矩阵d来表示，d_{ij}表示点i与点j之间的距离。设0-1矩阵X用来表示经过的各点之间的路线。设

$$x_{ij}=\begin{cases}0 & \text{若}i\text{不到}j\\ 1 & \text{若}i\text{到}j,\text{且}i\text{在}j\text{前}\end{cases}$$

目标函数 $$\min z=\sum_{i,j=1}^{n} d_{ij}x_{ij}$$

约束满足：

考虑每个点后只有一个点，则：$$\sum_{\substack{j=1\\ j\neq i}}^{n} x_{ij}=1,\ i=1,\cdots,n$$

考虑每个点前只有一个点，则：$$\sum_{\substack{i=1\\ i\neq j}}^{n} x_{ij}=1,\ j=1,\cdots,n$$

但仅以上约束条件不能避免在一次遍历中产生多于一个互不连通回路，为此我们引入额外变量$u_i\ (i=1,\cdots,n)$，附加以下充分约束条件：

$$u_i - u_j + nx_{ij} \le n-1,\ 1<i\ne j\le n$$

于是我们可以得到如下的模型（实现程序见附录2的Lingo程序B98.1g4(具体程序见书P89)）：

$$\begin{cases} \min \quad z=\sum_{i,j=1}^{n} d_{ij}x_{ij} \\ \text{s.t.} \begin{cases} \sum_{\substack{i=1\\ i\ne j}}^{n} x_{ij}=1, & j=1,\cdots,n \\ \sum_{\substack{j=1\\ j\ne i}}^{n} x_{ij}=1, & i=1,\cdots,n \\ u_i-u_j+nx_{ij}\le n-1, & 1<i\ne j\le n \\ x_{ij}=0\text{或}1, & i,j=1,\cdots,n \\ u_i\text{为实数}, & i=1,\cdots,n \end{cases} \end{cases}$$

利用该模型可以计算出当选定一组点后的最优路线。

调整后分组1对应距离矩阵为B98N1.txt，分组3对应距离矩阵为B98N3.txt。

表4-4　调整后的划分方案

分　组	线　路	线路长度	总 长 度
1	O→6→7→8→E→9→10→F→12→H→14→13→G→11→J→19→L→5→O（18个点）	216.5	599.9
2	O→P→28→27→26→N→24→23→22→17→16→15→I→18→K→21→20→25→M→O（该线路不变，19个点）	191.1	
3	O→2→3→D→4→C→B→1→A→34→35→33→31→32→30→Q→29→R→O（18个点）	192.3	

因该分组的均衡度

$$\alpha_0 = \frac{\omega(C_3) - \omega(C_1)}{\max\limits_{i=1,2,3} \omega(C_i)} = \frac{216.5 - 191.1}{216.5} = 11.73\%$$

所以这种分法的均衡性较好，故采用表4-4方案。

问题点评

该问题代表了多TSP问题的一般求解方法。

① 先求出各点到达原点0的前点，从而做出树状图。

② 根据树状图进行人为分组，输出对应距离矩阵，采用Lingo优化各组线路。

③ 根据计算结果，调整各部分包含的点，再输出对应距离矩阵，利用Lingo再优化各线路。

4.1.3　状态转移与图论模型的巧妙结合

引言

这里我们通过几个经典的智力问题，讲解状态转移与图论模型的巧妙结合。对这些问题，通常并不需要数学知识进行求解，但我们却可以利用数学知识建立数学模型，然后转化为标准的图论模型进行求解。

返回

问题

- 问题1：人、狼、羊、菜渡河问题
- 问题2：商人过河问题
- 问题3：等分酒问题
- 问题4：等分酒问题的扩展

问题1：人、狼、羊、菜渡河问题

一个摆渡人希望用一条小船把一只狼、一头羊和一篮菜从河的左岸运到右岸去，而船小只能容纳人、狼、羊、菜中的两个，决不能在无人看守的情况下留下狼和羊在一起，也不允许羊和菜在一起，应怎样渡河才能将狼、羊、菜都运过去？

解：采用试探法可以得到两种方法

方法1：

1. 人、羊（去）→2. 人（回）→3. 人、狼（去）→4. 人、羊（回）→5. 人、菜（去）→6. 人（回）→ 7. 人、羊（去）

方法2：

1. 人、羊（去）→2. 人（回）→3. 人、菜（去）→4. 人、羊（回）→5. 人、狼（去）→6. 人（回）→7. 人、羊（去）

然而对于这样的问题，如何采用数学的方法来获得最优解呢？这是我们要解决的问题。

模型建立与求解：

我们用(x_1, x_2, x_3, x_4)作为状态变量表示人、狼、羊、菜在此岸的状态，若x_1 =0表示人在彼岸， x_1 =1表示人在此岸。如(1, 0, 1, 0)表示人和羊在此岸，狼和菜在彼岸。根据问题的要求，我们知道总共有10个状态是安全的。它们集合S为：

S={(1, 1, 1, 1)， (1, 1, 1, 0)， (1, 1, 0, 1)， (1, 0, 1, 1)， (1, 0, 1, 0)，
(0, 0, 0, 0)， (0, 0, 0, 1)， (0, 0, 1, 0)， (0, 1, 0, 0)，(0, 1, 0, 1)}

所有状态之间的转移关系可以通过图4-4表示。如状态(1, 1, 1, 1)转化为(0, 1, 0, 1)，表示人将羊运到对岸，反之也可以。

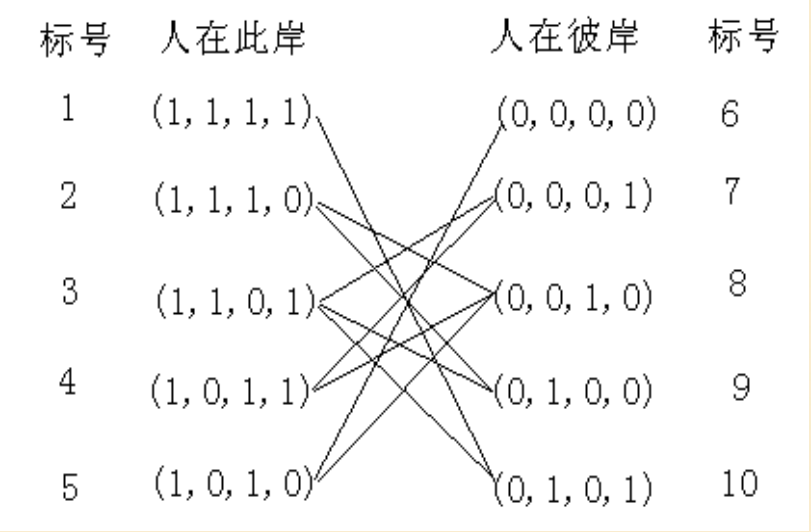


图4-4　状态转移图

由图4-5容易看出，从初始状态(1,1,1,1)到目标状态(0,0,0,0)的最短路线共有两条，都为7步。

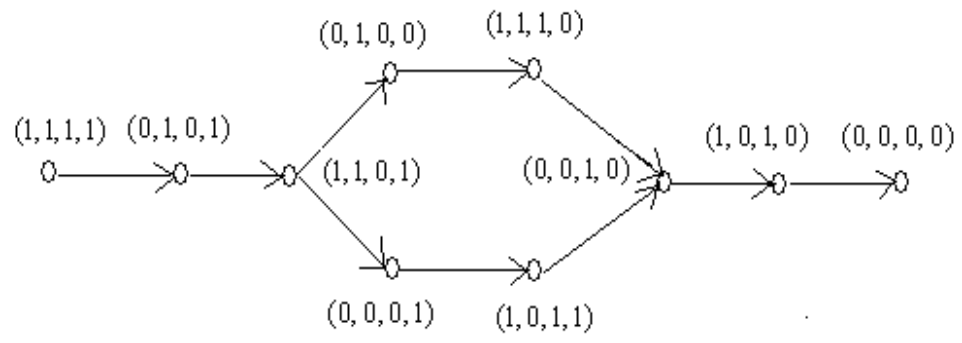


图4-5　重新按照起始状态到目标状态的连接图

对应的解恰好为前面用试探法得到的解

方法1：

1．人、羊（去）→2．人（回）→3．人、狼（去）→4．人、羊（回）→5．人、菜（去）→6.人（回）→ 7．人、羊（去）

方法2:

1．人、羊(去）→2．人（回）→3．人、菜（去）→4．人、羊（回）→5．人、狼（去）→6．人（回）→7．人、羊（去）

问题2：商人过河问题

有3名商人各带一个仆人乘船渡河，小船只能容纳两个人，由他们自己划船。仆人们约定，在河的任一岸，一旦仆人的人数比商人多，就杀人越货。如何乘船的大权掌握在商人们手里。问商人们怎样才能安全渡河？

问题分析：

安全渡河问题是一个多步决策问题。每一步，即船由此岸驶向彼岸或从彼岸回到此岸，都要对船上的商人和仆人的个数做出决策。在保证安全的前提下，在有限步内使全部人员过河。采用状态变量表示某一岸的人员状况，决策变量表示船上人员状况。可以找出状态随决策变化的规律，问题转化为在状态允许范围内（安全渡河的条件），确定每一步的决策，达到安全渡河的条件。

模型建立：

设第k次渡河前此岸的商人数为x_k，仆人数为y_k，$k = 1,2,\ldots$。因此状态变量为(x_k, y_k)，其中x_k，y_k取值为0,1,2,3。安全渡河条件下的状态集合为允许的状态集合，记作S。于是知道该集合为：

$$S=\{(0,0),(0,1),(0,2),(0,3),(3,0),(3,1),(3,2),(1,1),(2,2),(3,3)\} \quad ①$$

S共10种状态。每种安全状态既要满足此岸安全，同时彼岸也要安全。

记第k次渡河船上的商人数为u_k，仆人数为v_k。则决策变量为$d_k=(u_k, v_k)$。允许的决策集合记为D。由船的容量可知D的集合为：

$$D=\{(2,0),(0,2),(1,1),(1,0),(0,1)\} \quad ②$$

每次有5种决策可供选择。

由于k为奇数时船从此岸到彼岸，k为偶数时船从彼岸到此岸，因此状态s_k随决策d_k而变化的规律为：

$$\begin{cases} s_0=(3,3) \\ s_k=s_{k-1}+(-1)^k.d_k \end{cases} \quad k=1,2,3\ldots \quad ③$$

③式称为状态转移律。这样制定安全的渡河方案归结为如下的多步决策模型：求决策$d_k(k=1,2,3\ldots)$，使状态$s_k \in S$按照转移律③，由初始状态(3,3)，经过有限步 n 到达目的状态(0,0)，并且步数 n 尽量小。

模型求解：

（1）手工求解

对该问题，我们可以采用图解的方法简单地用手工方法求解。

在如图4-6所示的平面上，绘制3行3列的格子。每个格子点(x, y)代表一种此岸状态。允许的10个状态在图上用黑点标出。每次决策或者是过去或者是回来。过去时采用向下走1个(d=(0, 1))或2个格子(d=(0, 2))，或者向左走1个(d=(1, 0))或2个格子(d=(2, 0))，或者向左下走两个格子(d=(1, 1))。回来时采用向上走1个(d=(0, 1))或2个格子(d=(0, 2))，或者向右走1个(d=(1, 0))或2个格子(d=(2, 0))，或者向右上走两个格子(d=(1, 1))。每过去一次还需要回来一次，直到从初始状态(3, 3)点到达(0, 0)状态为止。

求解过程如同下棋，过程见图4-6标出的箭头方向。

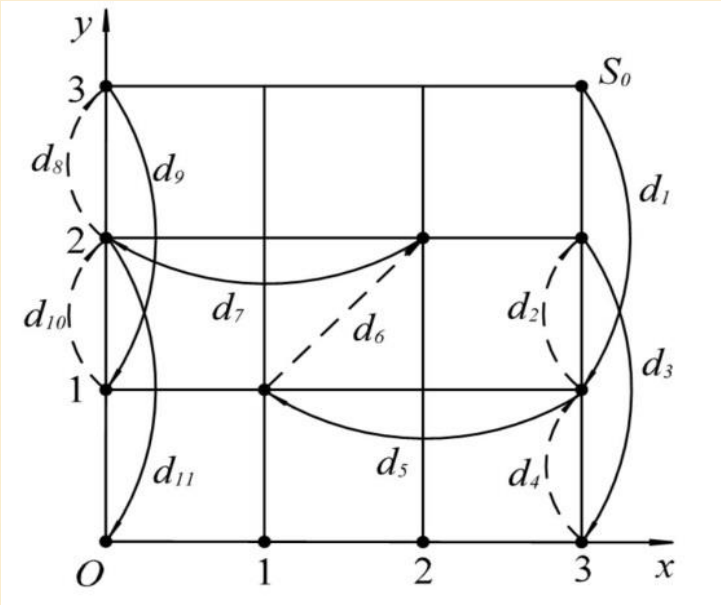


图4-6　安全渡河示意图

（2）图论求解

前面的方法1采用的是如下棋一样的手工操作。通常只能找到可行解，不能保证是最优解，而且通常不具有一般性。这里我们采用图论的方法进行一般求解。

由图4-6知，此岸共有10种状态。每种状态用二维向量（x,y）来表示。我们考虑每种状态可以转化的其他状态，由此构成一个图。但一种状态能否转化为另一种状态，不但跟该状态是否安全有关，而且跟船在此岸还是彼岸有关。如(3,1)状态，当船在此岸时（偶数次渡河后），可以转化为(3,0)；当船在彼岸时（奇数次渡河后）就不能转化为(3,0)。因此，为统一考虑状态之间的转化，我们采用三维向量(x,y,z)来表示。x表示此岸商人数(x=0,1,2,3)，y表示此岸仆人数 y=0,1,2,3，z表示船在此岸还是彼岸(z=1表示船在此岸，z=0表示船在彼岸)。总共有20个状态，我们可以建立20个顶点的图。

根据船每次最多乘两个人的条件，比较容易建立状态之间的转移关系。这里得到20种状态之间的转移关系如图4-7。

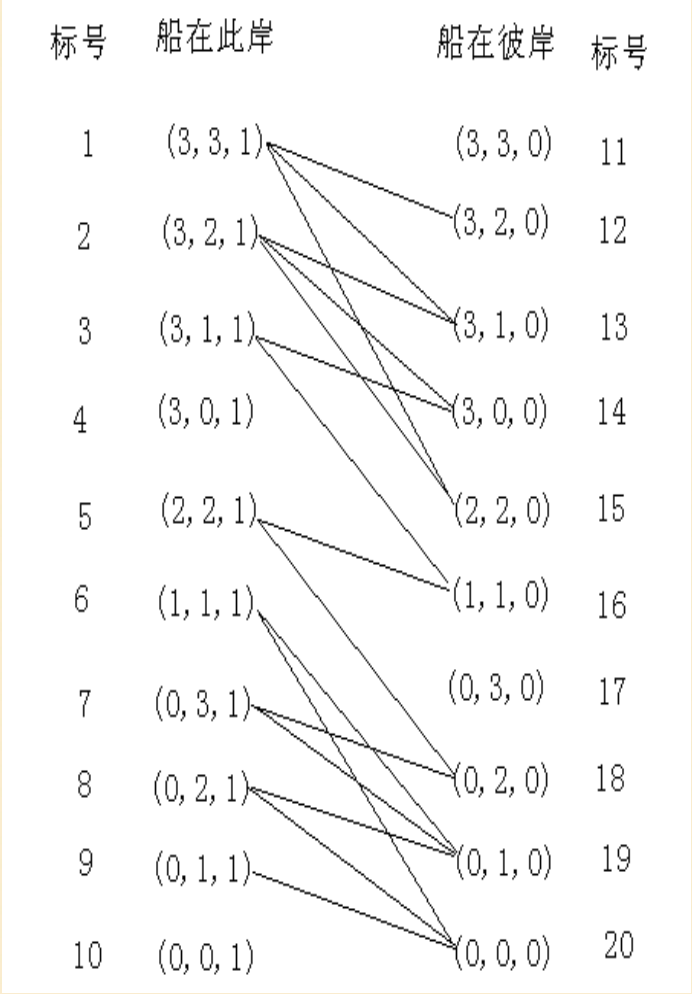


图4-7　状态转移图

该图对20个状态分别标号，同时给出了各状态之间的转移。两个状态之间的转移是相互的。该图为无向图。对该图进行重新标号连接，得到如图4-8所示的连接图（相连的两点表示两状态可互相转化）。我们的问题转化为求顶点1到顶点20的最短路线。

由图4-8可知，最短路线为1→13→2→14→3→16→5→18→7→19→8→20或1→15→2→14→3→16→5→18→7→19→8→20。其最短路线长度为11。即最少经过11步可以由(3,3)转移到(0,0)，安全渡河。

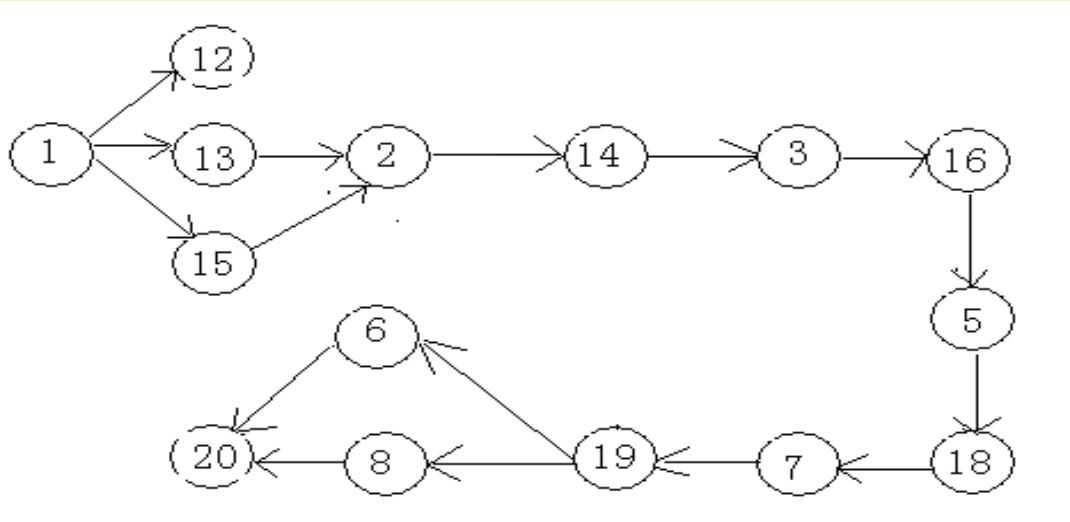


图4-8　用标号表示的从起始状态到目标状态的连接图

问题3：等分酒问题

现有一只装满8千克酒的瓶子和两只分别装5千克和3千克酒的空瓶，如何才能将这8千克酒分成两等份？

解：手工操作法

设状态向量(a, b, c)，其中a代表可装8千克酒的瓶子，b代表可装5千克酒的瓶子，c代表可装3千克酒的瓶子。

该问题转化为如何将初始状态(8, 0, 0)达到目标状态(4, 4, 0)。

其操作过程必须满足条件：任何两瓶之间操作必须满足其中一个瓶子清空为0或另一个装满。

我们可以采用状态转移的方法，利用图论知识进行求解。

我们从初始状态(8, 0, 0)开始，依此推出新出现的状态，由新出现的状态再推出更新的状态，直到不能推出新状态为止。

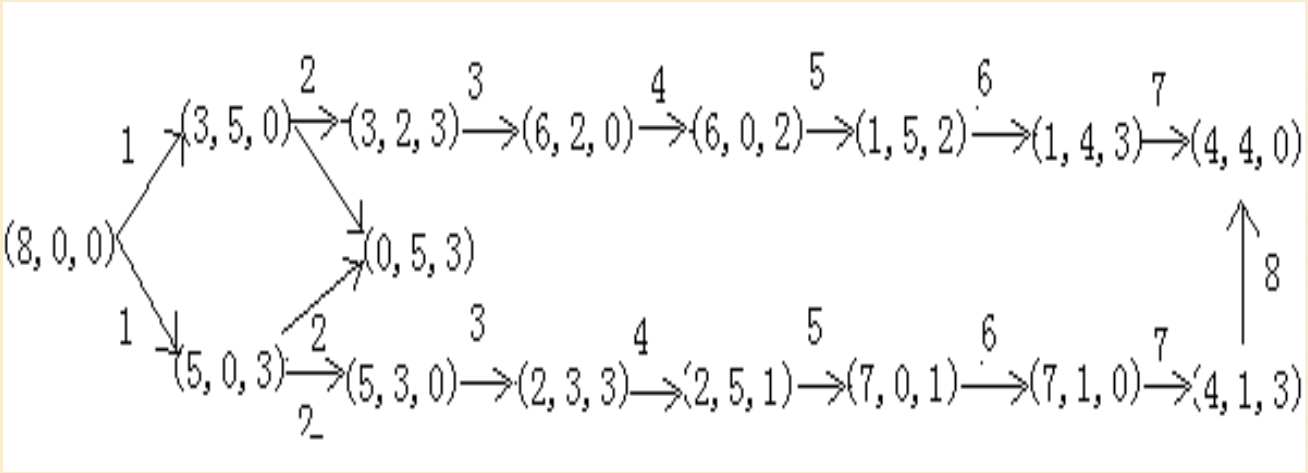


图4-9　分酒问题的状态转移图

从图4-9看出，从(8,0,0)状态可以有两种方式实现平分酒。一种方式是采用上面的方式，通过7步实现。另一种方式是下面的方式，通过8步实现。当然采用上面的方式是最少步数的方式。实现过程为：

(8,0,0)→(3,5,0)→(3,2,3)→(6,2,0)→(6,0,2)→(1,5,2)→1,4,3)→(4,4,0)

问题4：等分酒问题的扩展

上面的分酒问题规模小，手工就可以完成。如果规模扩大，手工无法完成，如何设计算法和建立模型，利用计算机求解呢？这里考虑一个实例，从中学习并掌握一般性的实现方法。

现有一只装满12千克酒的瓶子和三只分别装10千克、6千克和3千克酒的空瓶，如何才能将这12千克酒平分成三等份。如果进行四等份呢，结果如何？如果4个瓶子分别要求装5千克、3千克、2千克、2千克，又能否实现？试建立数学模型并设计算法，求最少经过多少步操作完成，且有多少种方式可采用最少步数完成。要求对实现方式给出详细操作步骤。

对这四只酒瓶，需要最多实现步数的方式是多少（依然采用最短路线方式）？

我们采用如下方法完成该问题。

算法设计步骤。

（1）计算并存储所有状态。

设满状态为（$c_1, c_2, \ldots, c_n$），初始状态为$x_0 = (x_{10}, x_{20}, \ldots, x_{n0}$）。计算符合条件的所有状态。枚举所有符合条件的状态，并存储在矩阵G(m,4)中。m为状态总数，每行代表一个状态。

程序示意：

```
for x1=0:c1
  for x2=0:c2
……
   for xn=0:cn
   T=sum(x0);  if x1+x2+...+xn=T 记录该状态
end
```

2）下一状态的计算。

算法步骤：

假定满状态为（$c_1, c_2, \ldots, c_n$），某状态为 $(x_{10}, x_{20}, \ldots, x_{n0})$ 。计算从该状态出发生成的所有不同状态

① 从n个初始状态中任选两个状态 i和j记 $x_1 = x_{i,0}$， $x_2 = x_{j,0}$ 。

对应的满状态为 $a = c_i$， $b = c_j$ 。

② 生成从（x_1, x_2）可以到达的所有状态。生成方式有4种规则：

- x_1清空，若$x_1 + x_2 \le b$ ，则有新状态$(y_1, y_2) = (0, x_1 + x_2)$。
- x_2清空，若$x_1 + x_2 \le a$ ，则有新状态$(y_1, y_2) = (x_1 + x_2, 0)$。
- x_1倒满，若$x_1 + x_2 - a \ge 0$ ，则有新状态$(y_1, y_2) = (a, x_1 + x_2 - a)$
- x_2倒满，若$x_1 + x_2 - b \ge 0$ ，则有新状态$(y_1, y_2) = (x_1 + x_2 - b, b)$

将每个新状态嵌入原状态中，则有新状态。

$$(x_{10},\cdots,x_{i-1,0},y_1,\ldots,x_{j-1,0},y_2,\ldots x_{n0})$$

③ 重复步骤①和②，生成从初始状态出发的所有新状态。删除与初始状态相同的状态，同时将新状态中相同状态只保留一个。

（3）连接图的生成

我们需要得到图的邻接矩阵表达$T(m,m)$。$T(i,j)=1$表示状态i可以到达状态j；$T(i,j)=0$表示状态i不能到达状态j。注意$T(m,m)$为非对称矩阵。

程序示意：

```
%2.计算图G的邻接矩阵TU
    for k=1:kp  %初始状态序号k，kp图G中所有状态
      cs=G(k,:); %获得初始状态
      P=GetNextAllState(x,cs); %获得初始状态下所有状态
      [Lp,Np]=size(P);
      %获得P中各状态在图G中标号j
       for i=1:Lp  %图P中序号i
         for j=1:kp  %图G中序号j
              if(sum(P(i,:)==G(j,:))==n) TU(k,j)=1; end
       %图G中的状态k与G中状态j相连，令TU(k,j)=1
         end %end for j
       end %end for i
    end
```

（4）计算从初始状态到达各状态的步数

计算从初始状态x0到达各状态的步数H(m)。H(j)=k表示从初始状态i到达第j个状态需要经过k步完成。H(j)=-1 表示从初始状态i不能到达第j个状态。

程序实现：

```
%计算从x0开始，各状态与x0距离
H=-1*ones(kp,1);
%存储G中各状态与x0距离,未标号则为-1,最后仍为-1表示从x0不能到达该状态
  for i=1:kp
     if(sum(x0==G(i,:))==n) H(i)=0; st=i; break; end  %对初始态标号
  end
  for d=0:15  %标注各距离状态
  for i=1:kp
   if H(i)==d
      for j=1:kp
       if TU(i,j)==1 && H(j)==-1 H(j)=d+1; end
      end
   end
  end
  end
```

（5）求目标状态的前状态

寻找目标状态xd的序号d。依次找出目标状态的前一状态r，再寻找r的前一状态……直到找到初始状态x0。

实现程序：

```
%输出与目标状态相连各状态及前点
  for i=1:kp
    if(sum(xd==G(i,:))==n) dh=i; dp=H(i);  break; end  %获得目标状态标号dh及与x0距离dp
  end
 fprintf('目标状态序号%2d 距离%2d\n',dh,dp);
 Map=zeros(kp,dp); %存储从x0出发到达xd，中间各步经过的状态
 Map(dh,dp)=1;
 for d=dp:-1:2
    for i=1:kp
      if Map(i,d)==1 %若第i个状态在链上
    for k=1:kp
       if TU(k,i)==1 &&H(k)==d-1  Map(k,d-1)=1; end  %找到i的前点
    end
      end
    end %end for i
end   % end for d  %kp为总状态数。
```

（6）构造从初始状态到目标状态的连接子图。

这里每个状态都为最短路线上的状态。由该图得到从初始状态到达目标状态的所有最短路线方式。

MatLab程序实现：

1.计算下一状态的函数GetNextAllState.m（详细代码见书P97）

2. 主程序fenjiu.m(详细代码见书P99）

计算结果：

考虑满状态x=[12,10,6,3]，初始状态x0=[12,0,0,0]， 目标状态xd=[4,4,4,0]，总共状态234。

输出与x0相连的各状态、与x0距离、邻接点：（详细见书P102）

（1）目标态(4,4,4,0)(序号125)

目标状态序号125，与初始状态距离10

不同距离的状态为：

距离10:125

距离 9:29

距离 8:31

距离 7:51

距离 6:158 231

距离 5:138 233

距离 4:142 205

距离 3:53 160 214

距离 2:62 71

距离 1:80

初始状态序号=234

根据以上结果，我们可以做出从初始态到目标态的连接图见图4-10，该图上所有状态都为最短路线上状态。

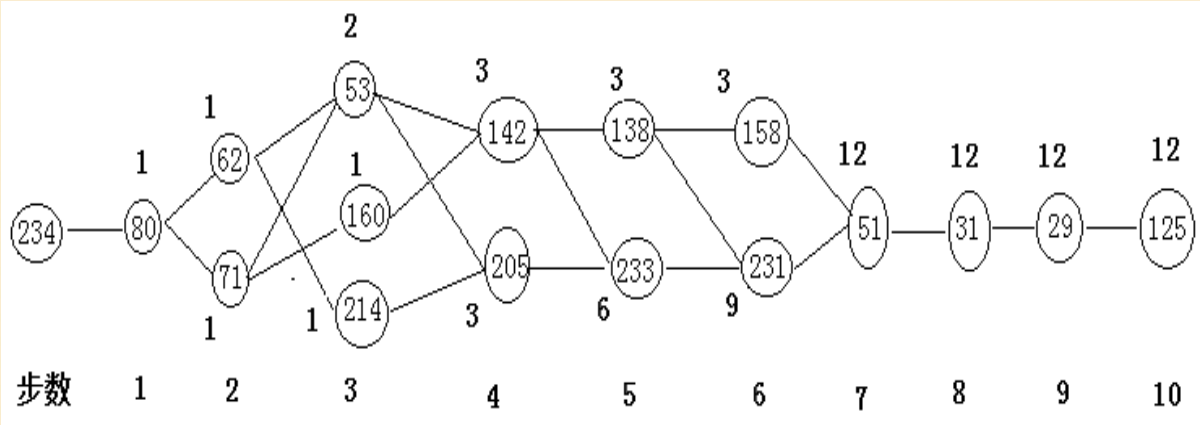


图4-10 从初始态到目标态示意图（图中圈外数字为到达该状态的路径数）

从图上容易得到，共有12种方式。最短路线为10步。

其中状态需要意义如下：

234(12,0,0,0) 80(2,10,0,0) 62(2,4,6,0) 71(2,7,0,3) 53(2,1,6,3) 160(5,7,0,0)

214(8,4,0,0) 142(5,1,6,0) 205(8,1,0,3) 138(5,0,6,1) 231(11,0,0,1) 51(1,10,0,1)

31(1,4,6,1) 29(1,4,4,3) 125(4,4,4,0)

（2）目标态(3,3,3,3)（序号87）。

目标状态序号87，与初始状态距离为4

不同距离的状态为：

距离 4:87

距离 3:90 176

距离 2:81 161 224

距离 1:164 215

初始状态序号=234

根据以上结果，我们可以做出从初始态到目标态的连接图见图4-11，该图上所有状态都为最短路线上状态。

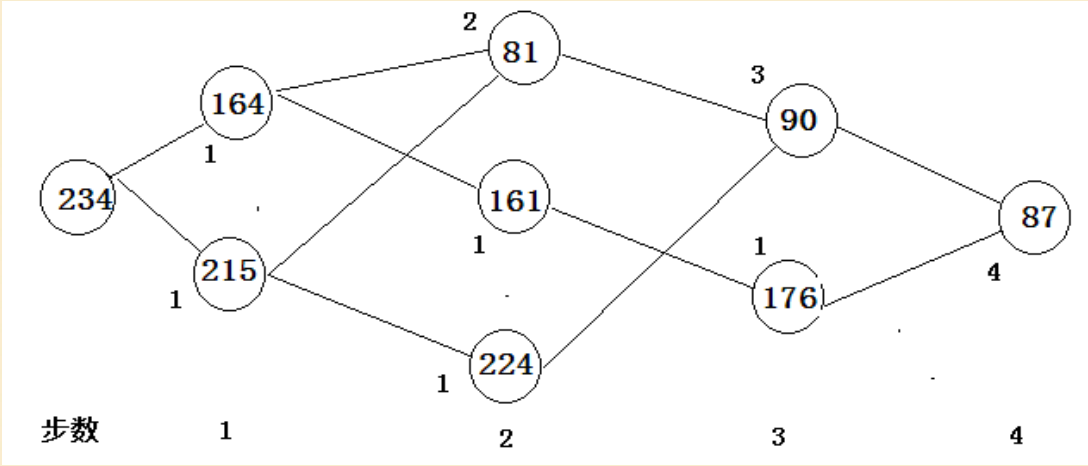


图4-11 从初始态到目标态示意图（图中圈外数字为到达该状态的路径数）

从图上容易得到，共有14种方式，最短路线为4步。

其中状态需要意义如下：

234(12,0,0,0) 164(6, 0, 6, 0) 215(9, 0, 0,3) 81(3, 0, 6, 3) 161(6,0,3,3)

224 (9,3, 0, 0)

90(3,3, 6, 0) 176(6,3,3,0) 87(3,3,3,3)

（3）目标态(5,3,2,2)，计算得到从初始状态无法到达该状态。

（4）要寻找这四只酒瓶，从一状态到另一状态的步数，可利用Floyd算法计算任意两点最短路线。得到最大距离为13。

包括以下方式：

(3,0,6,3)→(4,1,4,3)，(3,0,6,3)→(7,1,4,0)，(3,3,3,3)→(4,1,4,3)，

(3,3,3,3)→(7,1,4,0)

(6,0,3,3)→(4,1,4,3)，(6,0,3,3)→(7,1,4,0)

具体实现方式与前面方法相同。

实现方式总结如下：

① 计算得到状态的矩阵G(n,4)。n为状态总数，每行代表一个状态。

② 得到图的矩阵表达T(m,m)。T(i,j)=1表示状态i可以到达状态j；T(i,j)=0表示状态i不能到达状态j。T(m,m)为非对称矩阵。

③ 计算从初始状态x0到达各状态的步数H(m)。H(j)=k表示从初始状态i到达第j个状态需要经过k步完成。H(j)= -1 表示从初始状态i不能到达第j个状态。

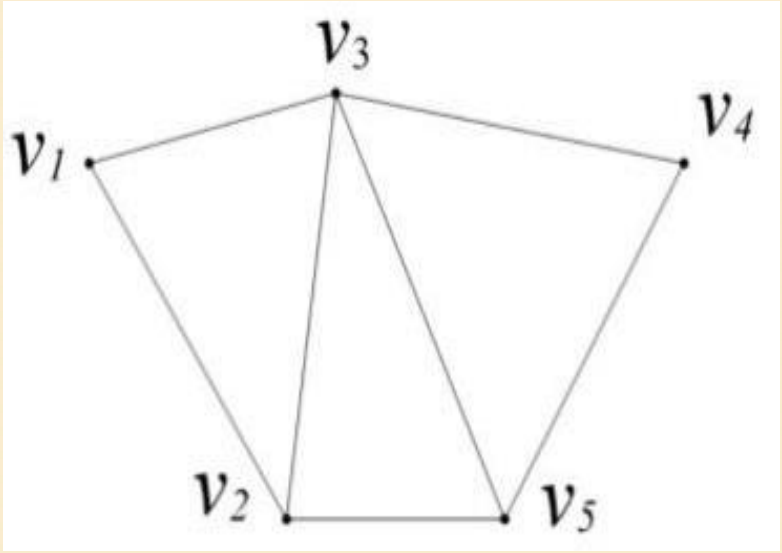
④ 寻找目标状态xd的序号d。依次找出目标状态的前一状态r，再寻找r的前一状态……直到找到初始状态x0。

⑤ 构造从初始状态到目标状态的连接子图。其中每个状态都为最短路线线的状态。

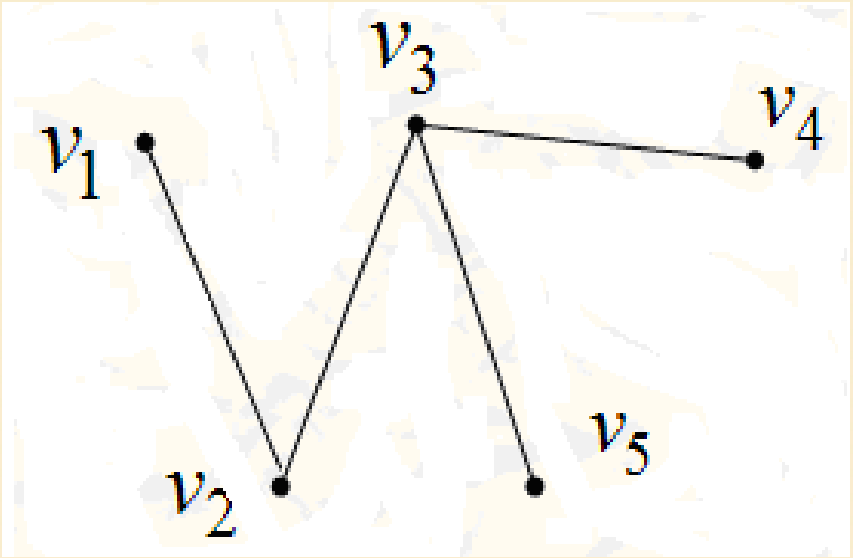
4.1.4最优树问题及Lingo求解

返回

树： 连通且不含圈的无向图称为树，常用T表示。树中的边称为树枝，树中度为1的顶点称为树叶。



树的示例



生成树的实例

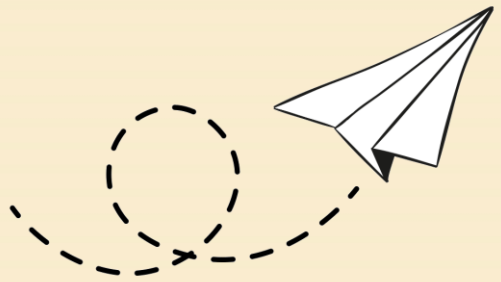
生成树： 若T是包含图G的全部顶点的子图，它又是树，则称T是G的生成树。

在实际问题中，如在许多城市间建立公路网、输电网或通信网络，都可以归结为赋权图的最优树问题。如在一个城市中，对若干居民点要供应自来水，已经预算出连接各点间的直接管道的造价，要求给出一个总造价最小的铺设方案等。

图论中最优树的求解通常有两种算法：Kruskal算法（或避圈法）和Prim算法（破圈法）。

现求根节点1到各节点生成的最优树，要求各线路上的权值和最小。其线性规划模型为：

决策变量：设 $x_{ij}=\begin{cases}1 & \text{节点}i\text{与节点}j\text{连通}\\ 0 & \text{节点}i\text{与节点}j\text{不连通}\end{cases}$



目标函数为寻找一条从起始点1到各节点生成的最优树，要求各线路上的权值和最小，故目标函数为：

$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}.x_{ij}$$

（1）对起始点1至少有一条路出去，故有：

$$\sum_{j=2}^{n} x_{ij} \geq 1 \qquad i = 2,3,...,n$$

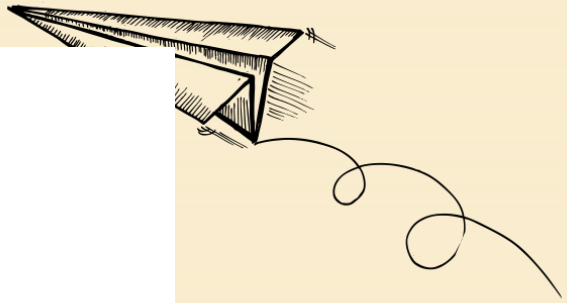
（2）对其余各节点，恰有一条路进入，故有：

$$\sum_{\substack{k=1 \\ k\neq i}}^{n} x_{ki} = 1$$

（3）所有节点不出现圈，约束为：

$$u_i - u_j + n.x_{ij} \le n-1 \qquad i,j = 1,2,\cdots,n$$

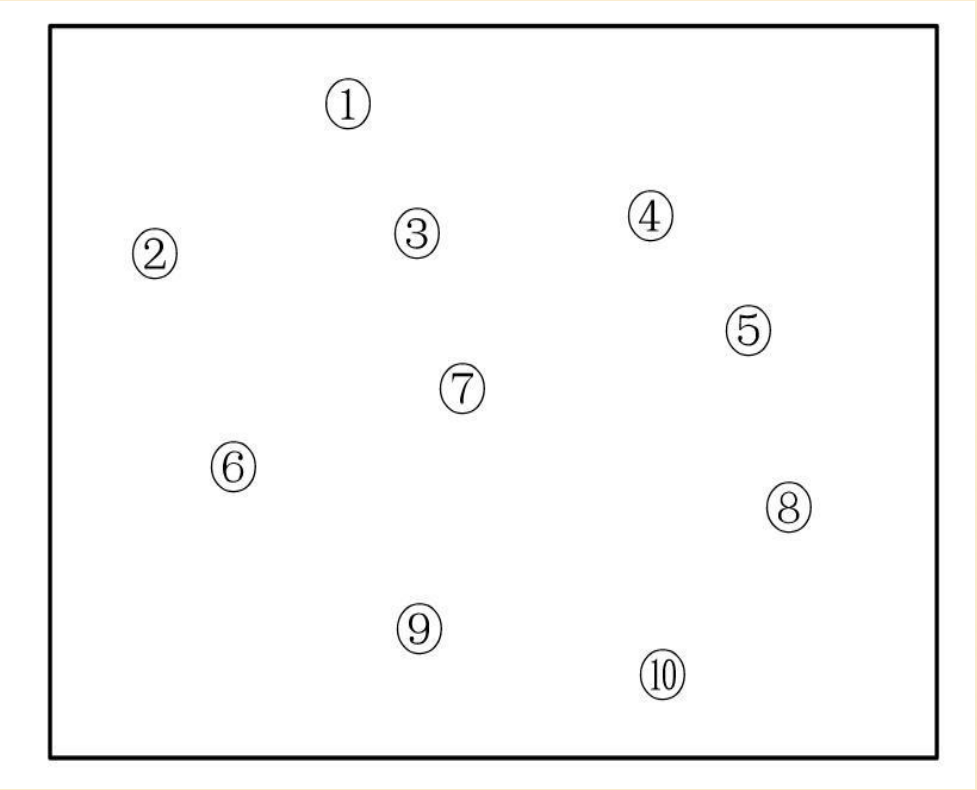
总的线性规划模型为：



$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij}$$

$$s.t.\begin{cases} \sum_{j=2}^{n} x_{1j} \ge 1 \\ \sum_{\substack{k=1 \\ k\ne i}}^{n} x_{ki} = 1 \qquad i = 2,3,\cdots,n \\ u_i - u_j + nx_{ij} \le n-1 \qquad i,j = 1,2,\cdots,n \\ x_{ij} = 0\text{或}1 \end{cases}$$

问题1有10个城镇分布,它们之间的距离见表4-5。城镇1处有一条河流，现需要从各城镇之间铺设管道，使城镇1处的水可以输送到各城镇，求铺设管道最少的设计方式。



该问题实际上是求从点1出发的最优树问题。其Lingo实现程序见下。

表4-5：10个地区之间的距离（单位：km）

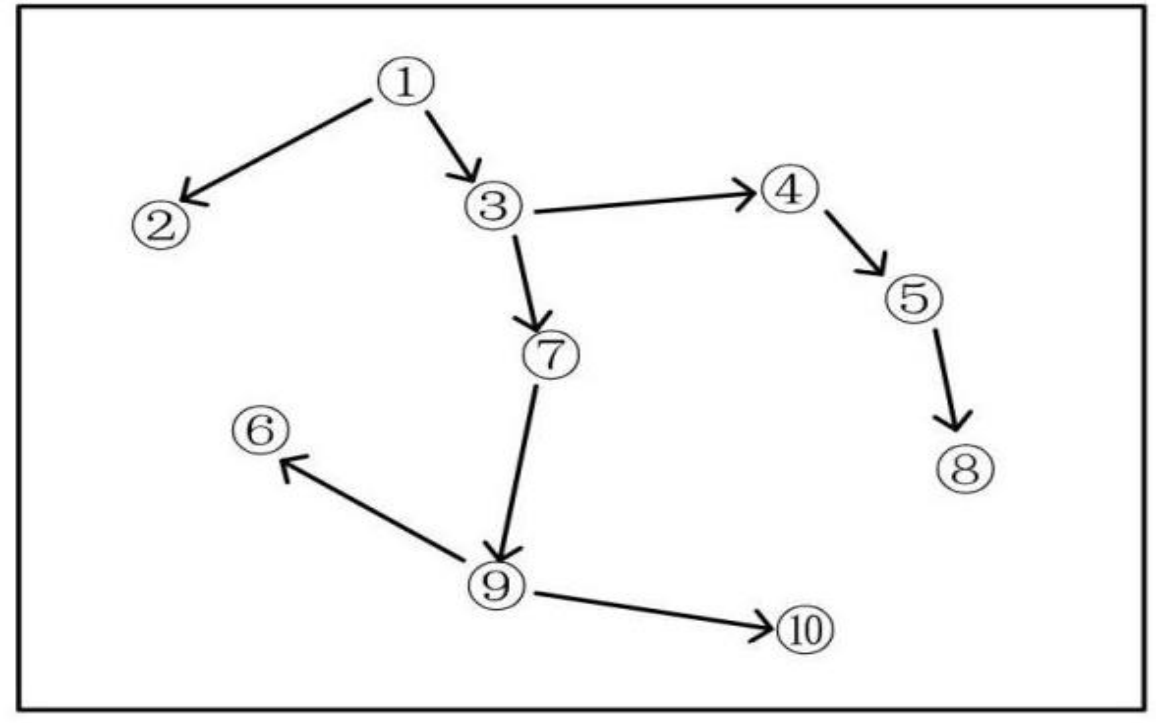
	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩
①	0	8	5	9	12	14	12	16	17	22
②	8	0	9	15	16	8	11	18	14	22
③	5	9	0	7	9	11	7	12	12	17
④	9	15	7	0	3	17	10	7	15	15
⑤	12	16	9	3	0	8	10	6	15	15
⑥	14	8	11	17	8	0	9	14	8	16
⑦	12	11	7	10	10	9	0	8	6	11
⑧	16	18	12	7	6	14	8	0	11	11
⑨	17	14	12	25	15	8	6	11	0	10
⑩	22	22	17	15	15	16	11	11	10	0

```
model:
sets:
point/1..10/:u;
link(point,point):d,x;
endsets
data:
d=0,8,5,9,12,14,12,16,17,22,
  8,0,9,15,16,8,11,18,14,22,
  5,9,0,7,9,11,7,12,12,17,
  9,15,7,0,3,17,10,7,15,15,
  12,16,9,3,0,8,10,6,15,15,
  14,8,11,17,8,0,9,14,8,16,
  12,11,7,10,10,9,0,8,6,11,
  16,18,12,7,6,14,8,0,11,11,
  17,14,12,25,15,8,6,11,0,10,
  22,22,17,15,15,16,11,11,10,0;
@text()=@writefor(link(i,j)|x(i,j)
#GT#0:'x(',i,',',j,')=',x(i,j),' ');
enddata
min=@sum(link(i,j)|i#ne#j:d(i,j)
*x(i,j));
n=@size(point);
@sum(point(j)|j#gt#1:x(1,j))>=1;
!从起始点出来至少1条路;
@for(point(i)|i#ne#1:@sum(poin
t(j)|j#ne#i:x(j,i))=1);!除起始点
外，每点只有一路进入;
@for(link(i,j):@bin(x(i,j)));
@for(link(i,j)|i#ne#j:u(i)-
u(j)+n*x(i,j)<=n-1); !不构成圈;
end
```

程序运算结果为minZ=60

x(1,2)=1　x(1,3)=1　x(3,4)=1　x(4,5)=1
x(9,6)=1　x(3,7)=1　x(7,9)=1　x(5,8)=1　x(9,10)=1

故最优树（最佳铺设管道的方式）见下图。



4.1.5 竞赛图与循环比赛排名问题

返回

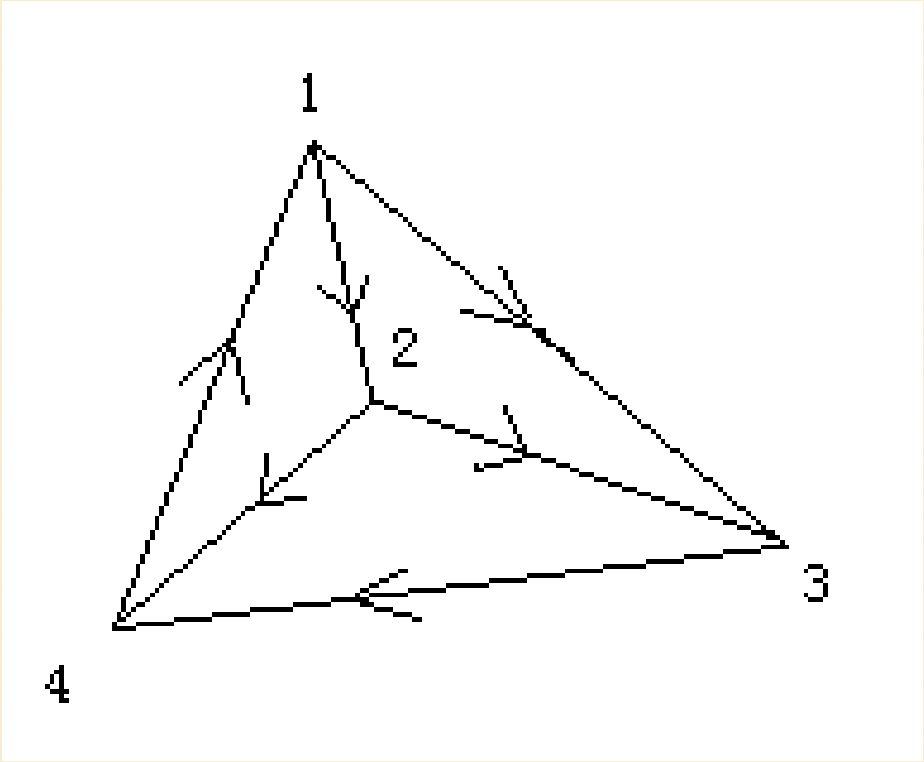
在图论中，完全图的定向图称为竞赛图。在实际问题中，可用于球队竞赛排名，论文引用的排名等。若干支球队参加单循环比赛，两两交锋。假设每场比赛只计胜负，不计比分，在比赛结束后排名次。

下面对只进行一次比赛的情况进行讨论。

1. 双向连通竞赛图

对于任何一对顶点，存在两条有向路径，使两顶点可以互相连通，这种有向图称为双向连通竞赛图。

4支队伍比赛结果的双向连通竞赛图：



其对应的邻接矩阵为：

$$A=\begin{bmatrix}0&1&1&0\\0&0&1&1\\0&0&0&1\\1&0&0&0\end{bmatrix}$$

设 $s_0=(1,1,1,1)^\tau$，则 $s_1=A.s_0=(2,2,1,1)$。该式表明各队胜的场次数。

$s_2=As_1=(3,2,1,2)$。该式是各队的2级得分，其意义是他战胜的各个球队的得分之和

可以作为排名的依据，但无法排出2和3的名次，可继续进行下去，得到结果如下：

$$s_3 = As_2 = (3,3,2,3) \qquad s_4 = As_3 = (5,5,3,3)$$

$$s_5 = As_4 = (8,6,3,5) \qquad s_6 = As_5 = (9,8,5,8)$$

$$s_7 = As_6 = (13,13,8,9) \qquad s_8 = As_7 = (21,17,9,13)$$

s_k各分量代表各队的第k级得分，其意义是他战胜的各个球队的前一级得分之和，因此更可以作为排名的依据。我们容易得出其排名为：1→2→4→3。

对一般性，记 $s_1 = A.s_0$ ，$s_2 = A.s_1, \cdots, s_k = A.s_{k-1}$ 则有：

$$s_k = A.s_{k-1} = A^k.s_0 \quad (k = 1,2,\cdots)$$

当迭代次数越多，名次排定顺序越稳定。可将其较高级的得分作为排名的依据。对其他双向连通竞赛图也可以采用类似方法迭代计算得到。

但这里有一个问题，是否双向连通竞赛图都一定要按照（1）式的方法排出确定的名次，另外是否还有更简单的方法？

为了回答这个问题，我们先给出素阵的定义：

素阵： 对于 $n(n \geq 4)$ 个顶点的双向连通竞赛图的邻接矩阵A，一定存在正整数r，使得 $A^r > 0$，这样的A就称为素阵。

Perron—Frobenius定理：

素阵A的最大特征根为正单根 λ，λ对应正特征向量s且有

$$\lim_{k\to\infty} \frac{A^k.e}{\lambda^k} = s$$

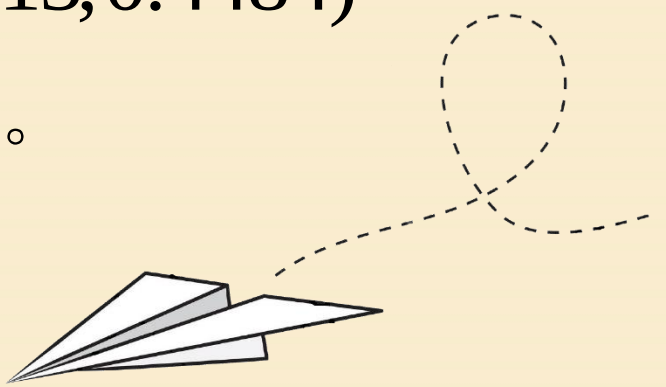
上式说明k级得分分量 S_k，当 $k \to \infty$ 时（归一化后）将趋向于A的对应最大特征根的特征向量s。因此特征向量s可作为排名次依据的得分向量。

我们求出前面矩阵A的最大特征值及对应特征向量有：

$$\lambda_{\max} = 1.3953$$

对应特征向量为$(0.6256, 0.5516, 0.3213, 0.4484)$

特征向量中分量大小与排名次序一致。

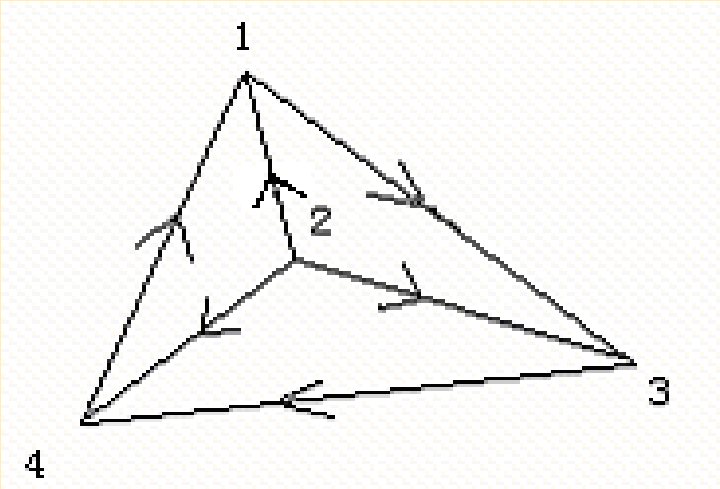


2. 非双向连通竞赛图

对于非双向连通竞赛图，则没有此结论，如：

$$\boldsymbol{A}=\begin{bmatrix}0&0&1&0\\1&0&1&1\\0&0&0&1\\1&0&0&0\end{bmatrix}$$

对应的竞赛图如下：



按照前面的方法计算得到：

$$s_1 = \boldsymbol{A}.e = (1,3,1,1) \qquad s_2 = \boldsymbol{A}.s_1 = (1,3,1,1) \qquad s_3 = \boldsymbol{A}.s_2 = (1,3,1,1)$$

其最大特征值对应特征向量为：

$$(0.2887,\ 0.8660,0.2887,0.2887)$$

从结果看无法对1, 3, 4进行排名。

下面我们将这种傲视胜败的0-1邻接矩阵扩展到一般的胜率矩阵。

设n支球队或队员比赛，第i支球队与第j支球队的比赛表现的能力表示为：

$$a_{ij} = p_{ij}, a_{ji} = 1 - p_{ij}, (i = 1, 2, \cdots, n-1; j = i+1)$$

其中 p_{ij} 表示第i支球队胜第j支球队概率。

且设 $a_{ii} = 0$，则第i支球队胜其余n−1支球队的能力表示为

$$s_i = \sum_{j=1}^{n} a_{ij} \qquad (i = 1, 2, \cdots, n)$$

则各球队的排名根据$\{s_i\}$的大小进行。s_i越大越靠前，越小越靠后。

但实际中 p_{ij} 并不知道，只有根据比赛进行估计。

（1）当进行m次比赛，第i支球队胜l次，则估计

$$p_{ij}=\frac{l}{m} \qquad p_{ji}=\frac{m-l}{m}$$

当只进行一次比赛时，若第i支球队胜，则记

$$a_{ij}=1 \qquad a_{ji}=0$$

若第j支球队胜，则记

$$a_{ij}=0 \qquad a_{ji}=1$$

这里给出利用胜率矩阵进行排名的两个实例。

实例1　足球比赛排名问题(CUMCM1993B)[5]

表4-6给出了我国12支足球队在1988～1989年足球甲级联赛中的成绩，要求：

① 设计一个依据这些成绩排出诸队名次的算法，并给出用该算法排名次的结果。

② 把算法推广到任意N队的情况。

③ 讨论：数据应具备什么样的条件，用你的方法才能够排出诸队的名次。

说明：

- 12支球队依次记作T1，T2，…T12。
- 符号X表示两队未曾比赛。
- 数字表示两队比赛结果，如T3 行与T8 列交叉处的数字表示：T3 与T8 比赛了2场；T3 与T8 的进球数之比为0:1和3:1。

表4-6比赛结果表

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
T1	×	0:1 1:0 0:0	2:2 1:0 0:2	2:0 3:1 1:0	3:1	1:0	0:1 1:3	0:2 2:1	1:0 4:0	1:1 1:1	×	×
T2		×	2:0 0:1 1:3	0:0 2:0 0:0	1:1	2:1	1:1 1:1	0:0 0:0	2:0 1:1	0:2 0:0	×	×
T3			×	4:2 1:1 0:0	2:1	3:0	1:0 1:4	0:1 3:1	1:0 2:3	0:1 2:0	×	×
T4				×	2:3	0:1	0:5 2:3	2:1 1:3	0:1 0:0	0:1 1:1	×	×
T5					×	0:1	×	×	×	×	1:0 1:2	0:0 1:1
T6						×	×	×	×	×	×	×
T7							×	1:0 2:0 0:0	2:1 3:0 1:0	3:1 3:0 2:2	3:1	2:0
T8								×	0:1 1:2 2:0	1:1 1:0 0:1	3:1	0:0
T9									×	3:0 1:0	1:0	1:0
T10										×	1:0	2:0
T11											×	1:1 1:2 1:1
T12												×

分析与解答：

1. 建立邻接矩阵A

对i队和j队，若i队胜j队的场次多，则令

$$a_{ij}=1,\ a_{ji}=0$$

若i队和j队的胜的场次一样，但i队比j队的净胜球多，则令

$$a_{ij}=1,\ a_{ji}=0$$

若i队和j队胜的场次一样，净胜球也一样，或者i队和j队没有交战，则令

$$a_{ij}=-1,\ a_{ji}=-1$$

表示i队和j队的胜败待定。

由此得到尚为完善的建立邻接矩阵A如表4-7所示。

表4-7　初始邻接矩阵

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
T1	0	-1	0	1	1	1	0	0	1	-1	-1	-1
T2	-1	0	0	1	0	1	0	-1	1	0	-1	-1
T3	1	1	0	1	1	1	0	1	-1	1	-1	-1
T4	0	0	0	0	0	0	0	0	0	0	-1	-1
T5	0	-1	0	1	0	0	-1	-1	-1	-1	-1	0
T6	0	0	0	1	1	0	-1	-1	-1	-1	-1	-1
T7	1	-1	1	1	-1	-1	0	1	1	1	1	1
T8	1	-1	0	1	-1	-1	0	0	-1	-1	1	-1
T9	0	0	-1	1	-1	-1	0	-1	0	1	1	1
T10	-1	1	0	1	-1	-1	0	-1	0	0	1	1
T11	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0
T12	-1	-1	-1	-1	1	-1	0	-1	0	0	1	0

2. 修正邻接矩阵A

计算各队一级和二级得分：

$$a_1 = (4,3,7,0,1,2,8,3,4,4,0,2)\ a_2 = (7,6,17,0,0,1,24,4,6,5,0,1)$$

其中，一级得分为各队所胜队数，二级得分为各队所胜队的一级得分和。

根据一级得分，修正邻接矩阵A中尚未确定的比赛胜败。当$a_{ij}=-1,\ a1_i > a1_j$时，令$a_{ij}=1,\ a_{ji}=0$，表示i队胜j队。

经过一级得分修正A后，对尚未确定胜败的队，采用二级得分修正，当$a_{ij}=-1,\ a2_i > a2_j$时，令$a_{ij}=1,\ a_{ji}=0$，表示i队胜j队。

对尚未确定胜败的队伍，采用抽签（随机）方式确定胜败，得到修正的邻接矩阵见表4-8。

表4-8　修正后邻接矩阵A

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
T1	0	1	0	1	1	1	0	0	1	1	1	1
T2	0	0	0	1	1	1	0	1	1	0	1	1
T3	1	1	0	1	1	1	0	1	1	1	1	1
T4	0	0	0	0	0	0	0	0	0	0	0	0
T5	0	0	0	1	0	0	0	0	0	0	1	0
T6	0	0	0	1	1	0	0	0	0	0	1	0
T7	1	1	1	1	1	1	0	1	1	1	1	1
T8	1	0	0	1	1	1	0	0	0	0	1	1
T9	0	0	0	1	1	1	0	1	0	1	1	1
T10	0	1	0	1	1	1	0	1	0	0	1	1
T11	0	0	0	1	0	0	0	0	0	0	0	0
T12	0	0	0	1	1	1	0	0	0	0	1	0

3. 排名

根据修正后的邻接矩阵A，计算其各级得分或最大特征值对应的特征向量，作为排名的依据。

如12支队伍的20级得分为：

$$a_{20}=(3336894,2072949,6087933,0,0,0,9354474,1790388,\ 2072949,2072949,0,0)$$

邻接矩阵A的最大特征值为1.8637，对应的特征向量为：

$$w=(0.1246,0.0774,0.2273,0,0,0,0.3492,0.0668,0.0774,0.0774,0,0)$$

最大特征值对应的特征向量与12支队伍的20级得分大小顺序一致，可以作为排名的依据。从特征向量来看，T7排第1，T3排第2，T1排第3；T2，T9，T10排名相同，居第4至第6；T8排第7；T4，T5，T6，T11，T12排名相同，居第8至第12。

对根据特征向量排名相同的队伍，我们需要根据其他信息进行确定。

考察T9，T10，T2。

由于一级得分 $a_1 = (4,3,7,0,1,2,8,3,4,4,0,2)$，其中T2，T9，T10都交战8场，T2胜3场，T9和T10胜4场，因此T9和T10排在T2前。再从二级得分来看，T9得分为6，T10得分为5，T9应排在T10之前。因此T2，T9，T10的排名顺序为：T9，T10，T2。

考察T4，T5，T6，T11，T12：

由于一级得分T6和T12最高，都为2，二级得分都为1。因此T6和T12靠前。T5的一级得分为1，T11的一级得分为0，则T5排在T11前。T11和T4的一级和二级得分都为0。但从比赛成绩来看，T11比赛6场，平1场，输5场；而T4比赛9场，输9场，T11排在T4前，T4排最后。再考虑T6和T12，他们都各胜两场，除都胜了T5外，T12胜了T11，而T6胜了排名最后的T4，将T12排在T6前。因此T4，T5，T6，T11，T12的排名顺序为：T12，T6，T5，T11，T4。

最后得到的总排名为：（MatLab实现程序见书117页）

T7，T3，T1，T9，T10，T2，T12，T6，T5，T11，T4

输出结果为：

序号	得分	特征向量
1	3336894	0.125
2	2072949	0.077
3	6087933	0.227
4	0	0.000
5	0	0.000
6	0	0.000
7	9354474	0.349
8	1790388	0.067
9	2072949	0.077
10	2072949	0.077
11	0	0.000
12	0	0.000

实例2　乒乓球循环比赛排名问题

2007年5月23至27日，第49届世界乒乓球单项锦标赛在萨格勒布进行，本次单项赛包括男、女单打，男、女双打和混双五个项目,每队可派出男女各12名选手。国家乒乓球球队在世乒赛等重大国际比赛前，往往进行队内大循环比赛，然后选出参赛队员。

其中男单选拔规则如下：

男单的比赛共16人参加，比赛采用11分制，每场为5局3胜。在比赛中如出现伤病和其他不可预测的原因而中途退出比赛者，在此前的比赛成绩有效。

根据规定，两次队内选拔赛积分相加获得前三名的运动员将获得参加第49届世乒赛男子单打比赛的资格，获得四至六名的运动员将获得第49届世乒赛的参赛资格。表4-9和表4-10分别是两次大循环的比赛成绩，表格中1表示横向运动员赢了纵向运动员，反之则为0。请根据该成绩对所有队员进行排名。

其中第二轮周斌顶替第一轮因伤不能参加的单明杰，其第一轮的成绩按单明杰的算，因此这两人的成绩按周斌的计算。

表4-9　第一阶段循环比赛成绩

第一轮	郝帅	马琳	张超	王励勤	王皓	马龙	陈玘	雷振华	李平	单明杰	张继科	邱贻可	王建军	许昕	李虎	侯英超
郝帅	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1
马琳	1	0	1	1	0	1	1	1	0	1	0	0	1	1	1	1
张超	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1
王励勤	1	0	1	0	1	0	0	1	1	0	1	1	1	0	1	1
王皓	0	1	1	0	0	1	0	1	1	0	0	1	1	1	1	1
马龙	0	0	0	1	0	0	1	0	1	1	1	1	0	1	1	1
陈玘	0	0	0	1	1	0	0	0	0	1	1	0	1	1	1	0
雷振华	0	0	1	0	0	1	1	0	1	1	0	0	1	1	1	1
李平	0	1	0	0	0	0	1	0	0	1	1	1	1	0	1	1
单明杰	0	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1
张继科	0	1	0	0	1	0	0	1	0	1	0	1	0	0	0	1
邱贻可	0	1	0	0	0	0	1	1	0	0	0	0	1	1	1	0
王建军	0	0	0	0	0	1	0	0	0	0	1	0	0	1	1	1
许昕	0	0	0	1	0	0	0	0	1	0	1	0	0	0	0	1
李虎	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
侯英超	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0

表4-10　第二阶段循环比赛成绩

第二轮	郝帅	马琳	张超	王励勤	王皓	马龙	陈玘	雷振华	李平	周斌	张继科	邱贻可	王建军	许昕	李虎	侯英超
郝帅	0	1	1	1	0	1	1	0	1	1	0	0	1	0	1	1
马琳	0	0	1	1	1	0	1	1	0	0	1	1	1	1	1	1
张超	0	0	0	0	0	0	1	1	1	0	0	0	1	1	0	1
王励勤	0	0	1	0	1	1	1	1	1	1	1	0	1	1	1	1
王皓	1	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1
马龙	0	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1
陈玘	0	0	0	0	0	0	0	1	1	0	0	1	1	0	1	1
雷振华	1	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1
李平	0	1	0	0	1	0	0	1	0	1	0	1	1	1	1	1
周斌	0	1	1	0	0	0	1	0	0	0	0	0	1	0	0	0
张继科	1	0	1	0	0	0	1	0	1	1	0	1	1	0	0	0
邱贻可	1	0	1	1	0	0	0	1	0	1	0	0	1	1	1	0
王建军	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
许昕	1	0	0	0	0	0	1	0	0	1	1	0	1	0	1	1
李虎	0	0	1	0	0	0	0	0	0	1	1	0	1	0	0	1
侯英超	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0

求解：

由第一阶段循环比赛成绩可得到邻接矩阵 $\boldsymbol{A}_1$，其中 $a1_{ij}=1$ 表示i胜j，$a1_{ij}=0$ 表示i输给j。

同样由第二阶段循环比赛成绩可得到邻接矩阵 $\boldsymbol{A}_2$，其中 $a2_{ij}=1$ 表示i胜j，$a2_{ij}=0$ 表示i输给j。

由两次循环比赛的成绩得到综合矩阵 A，其中：

$$a_{ij}=(a1_{ij}+a2_{ij})/2$$

这样当 $a1_{ij}=1,\ a2_{ij}=1$，则 $a_{ij}=1$，表示i胜j。

当 $a1_{ij}=1, a2_{ij}=0$，或 $a1_{ij}=0, a2_{ij}=1$，则 $a_{ij}=0.5$ 表示i与j胜率相同，都为0.5。

当 $a1_{ij}=0$，$a2_{ij}=0$，则 $a_{ij}=0$，表示i输给j。

这样综合矩阵A各元素取值为0或0.5或1，表示胜率。是0-1邻接矩阵的扩展。

最后得到的综合矩阵*A*见表4-11。

求得矩阵*A*的最大特征值为6.38，对应的归一化的特征向量为：

w=(0.101322,0.095904,0.060594,0.095582,0.087542,0.093150,0.050794,0.058527,0.067374,0.046691,0.061483,0.059527,　0.024322,0.045926,0.025564,0.025698)

计算各人的10级得分，其归一化后的向量与归一化的特征向量w相同。因此w可作为排名的依据。得到结果见表4-12。

表4-11　综合矩阵A

第二轮	郝帅	马琳	张超	王励勤	王皓	马龙	陈玘	雷振华	李平	周斌	张继科	邱贻可	王建军	许昕	李虎	侯英超
郝帅	0	0.5	1	0.5	0.5	1	1	0.5	1	1	0.5	0.5	1	0.5	1	1
马琳	0.5	0	1	1	0.5	0.5	1	1	0	0.5	0.5	0.5	1	1	1	1
张超	0	0	0	0	0	0.5	1	0.5	1	0.5	0.5	0.5	1	1	0.5	1
王励勤	0.5	0	1	0	1	0.5	0.5	1	1	0.5	1	0.5	1	0.5	1	1
王皓	0.5	0.5	1	0	0	0.5	0.5	1	0.5	0.5	0.5	1	1	1	1	1
马龙	0	0.5	0.5	0.5	0.5	0	1	0.5	1	1	1	1	0.5	1	1	1
陈玘	0	0	0	0.5	0.5	0	0	0.5	0.5	0.5	0.5	0.5	1	0.5	1	0.5
雷振华	0.5	0	0.5	0	0	0.5	0.5	0	0.5	1	0.5	0	0.5	1	1	1
李平	0	1	0	0	0.5	0	0.5	0.5	0	1	0.5	1	1	0.5	1	1
周斌	0	0.5	0.5	0.5	0.5	0	0.5	0	0	0	0	0.5	1	0.5	0.5	0.5
张继科	0.5	0.5	0.5	0	0.5	0	0.5	0.5	0.5	1	0	1	0.5	0	0	0.5
邱贻可	0.5	0.5	0.5	0.5	0	0	0.5	1	0	0.5	0	0	1	1	1	0
王建军	0	0	0	0	0	0.5	0	0.5	0	0	0.5	0	0	0.5	0.5	0.5
许昕	0.5	0	0	0.5	0	0	0.5	0	0.5	0.5	1	0	0.5	0	0.5	1
李虎	0	0	0.5	0	0	0	0	0	0	0.5	1	0	0.5	0.5	0	0.5
侯英超	0	0	0	0	0	0	0.5	0	0	0.5	0.5	1	0.5	0	0.5	0

表4-12　选拔赛两轮比赛综合排名

运动员	归一化特征向量	综合排名
郝帅	0.101322	1
马琳	0.095904	2
张超	0.060594	8
王励勤	0.095582	3
王皓	0.087542	5
马龙	0.093150	4
陈玘	0.050794	11
雷振华	0.058527	10
李平	0.067374	6
周斌	0.046691	12
张继科	0.061483	7
邱贻可	0.059527	9
王建军	0.024322	16
许昕	0.045926	13
李虎	0.025564	15
侯英超	0.025698	14

官方最后确定的男单名单是：王励勤、马琳、王皓、马龙、郝帅、陈玘、侯英超。其中，陈玘是在第三阶段通过三轮PK胜出获得名额的，侯英超是教练组最后综合考虑加入到名单中的。而男队除陈玘、侯英超外其他几名本身就是前五名，所以按综合排名确定名单前五名是可行的，因为要照顾一些特殊打法的运动员，所以还要教练组综合评定决定。

实现的MatLab程序见书123页。

程序输出结果为：

序号	两轮总积分
1	23
2	22
3	16
4	22
5	21
6	22
7	13
8	15
9	17
10	11
11	13
12	14
13	6
14	11
15	7
16	7

序号	得分	特征向量
1	0.101322	0.101322
2	0.095904	0.095904
3	0.060594	0.060594
4	0.095582	0.095582
5	0.087542	0.087542
6	0.093150	0.093150
7	0.050794	0.050794
8	0.058527	0.058527
9	0.067374	0.067374
10	0.046691	0.046691
11	0.061483	0.061483
12	0.059527	0.059527
13	0.024322	0.024322
14	0.045926	0.045926
15	0.025564	0.025564
16	0.025698	0.025698

从计算结果看，10级别得分归一化后与归一化的特征向量相同，都可以作为排名的依据。