1.1

这类规划问题，为了方便计算，我们采取变量替换



然后就转变为了常规的线性规划问题，此时有



此时题目的问题变为，在(1.2)的约束条件下，求解



代码如下：

clc,clear

c=[1:4]';

b=[0,1,-1/2]';

a=[1,-1,-1,1;1,-1,1,-3;1,-1,-2,3];

prob=optimproblem;

u=optimvar('u',4,'LowerBound',0);

v=optimvar('v',4,'LowerBound',0);

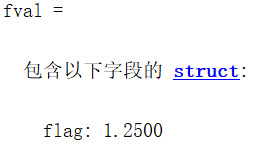
prob.Objective=sum(c'\*(u+v));

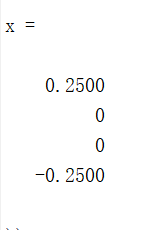
prob.Constraints=a\*(u-v)==b;

[sol,fval.flag,out] =solve(prob)

x=sol.u-sol.v

得出的结果如下：





1.2

这是题基于实际问题的线性规划我们可以这样假设：

不妨设x1，x2为分别用A1,A2加工产品Ⅰ的件数，x3，x4，x5为分别用B1,B2，B3加工产品Ⅰ的件数，x6，x7为分别用A1,A2加工产品Ⅱ的件数，由题意，x6+x7为用B1加工产品Ⅱ的件数，此时我们题目便转化为了求



使得



代码如下：

clc, clear

c=[1-5\*300/6000,1-321\*7/10000,-6\*250/4000,-4\*783/7000,-200\*7/4000,1.65-0.5-8\*250/4000,1.65-321\*9/10000-8\*250/4000,2.3-321\*12/10000-11\*783/7000]';

b=[6000,10000,4000,7000,4000]';

a=[5,0,0,0,0,10,0,0;

0,7,0,0,0,0,9,12;

0,0,6,0,0,8,8,0;

0,0,0,4,0,0,0,11;

0,0,0,0,7,0,0,0];

prob = optimproblem('ObjectiveSense','max')

x = optimvar('x',8,'LowerBound',0);

prob.Objective = sum(c.\*x);

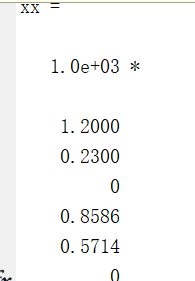
prob.Constraints.con1 = a\*x<=b;

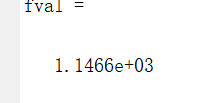
prob.Constraints.con2 = x(1)+x(2)-x(3)-x(4)-x(5)==0;

[sol,fval,flag,out]=solve(prob)

xx=sol.x

结果如下：





1.3

容易求得



由此我们可以求得g关于t的函数以及r关于t的函数



以及



此时我们可以画出图像，程序如下：

r=1.5:0.2:3;

g=0.06:0.01:0.2;

figure;

plot(r, (40\*r-60)./r, 'r-');

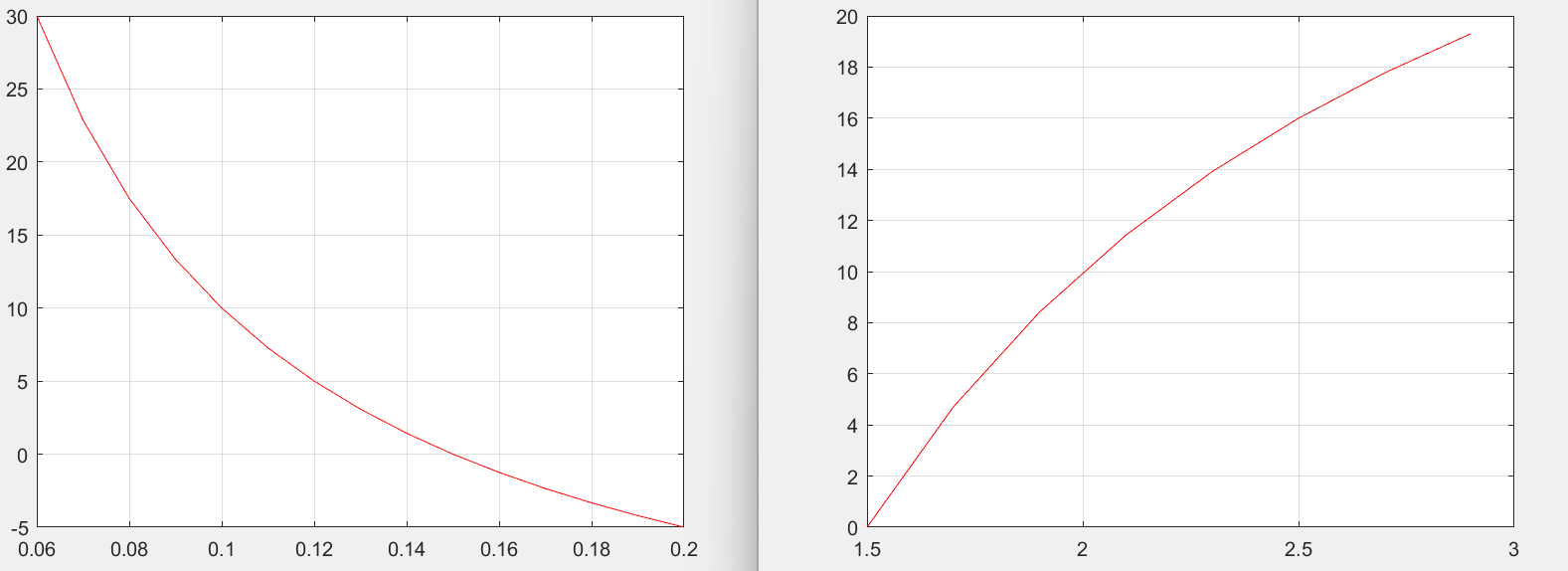
grid on;

figure;

plot(g, (3-20\*g)./g, 'r-');

grid on;

结果如下：



灵敏度分析：

我们如下定义t对r的灵敏度



此时我们带入(1.8)，即可得到r=3时，



即猪的体重r每天增加1.5%，出售时间推迟0.5%

同理，我们可以定义t对s的灵敏度



此时我们带入(1.9)，即可得到g=0.1时，



即猪的价格r每天增加1%，出售时间提前3%

**问题分析**

该问题是部门在面对投资时经常遇到的问题，考虑到在限制时间内用10万元的投资得到最大的回报。

**符号说明**

:第i年(i=1,2,3,4,5)对分别对A,B,C,D(j=1,2,3,4)四个项目的投资额

**模型假设**

假设部门每年将钱全部花出去，不留任何的钱

**模型建立**

在第一年，我们有如下投资



在第二年的年初，我们有



在第三年的年初，我们有



在第四年的年初，我们有



在第五年的年初，我们有



此时，我们的目标便转化为求解



于是乎，数学模型如下



由于求解器的限制，我们将新元素重新排列成一个列向量



**代码如下：**

clc,clear

prob=optimproblem('ObjectiveSense','max');

x = optimvar('x',11,'LowerBound',0);

prob.Objective = 1.15\*x(9)+1.40\*x(4)+1.25\*x(7)+1.06\*x(11);

prob.Constraints.con1 = x(1)+x(2)==10;

prob.Constraints.con2 =x(3)+x(4)+x(5)-1.06\*x(2)==0;

prob.Constraints.con3 = x(6)+x(7)+x(8)-1.15\*x(1)-1.06\*x(5)==0;

prob.Constraints.con4 = x(9)+x(10)-1.15\*x(3)-1.06\*x(8)==0;

prob.Constraints.con5 =1.15\*x(6)+1.06\*x(10)-x(11)==0;

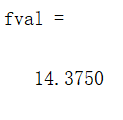
prob.Constraints.con6 =x(7)<=4;

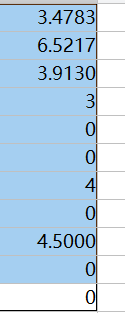
prob.Constraints.con7=x(4)<= 3;

[sol,fval,flag,out]=solve(prob),sol.x;

xx=sol.x

**结果如下**：





此时



最大收益为**14.3750**万元

2.2

**问题分析**

该问题是典型的非线性转线性问题，我们需要将非线性转为线性

考虑到



于是题目的条件即可转化为



**2.2**

**问题分析**

这是常见的0-1决策问题，需要我们在最小建校时能覆盖最大的区域

**符号说明**

:对第i(i=1,2,3,4,5,6)个区域的选取

**模型建立**

对小区A1我们有



对小区A2，我们有



对小区A3，我们有



对小区A4，我们有



对小区A5，我们有



对小区A6，我们有



对小区A7，我们有



对小区A8，我们有



于是，我们只要求解



**代码如下**：

clc,clear

prob=optimproblem;

x=optimvar('x',6,'Type','integer','LowerBound',0,'UpperBound',1);

prob.Objective = sum(x);

prob.Constraints.con1=x(1)+x(2)+x(3)>=1;

prob.Constraints.con2=x(2)+x(4)>=1;

prob.Constraints.con3=x(3)+x(5)>=1;

prob.Constraints.con4=x(4)+x(6)>=1;

prob.Constraints.con5=x(6)+x(5)>=1;

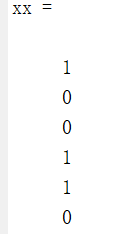
prob.Constraints.con6=x(1)>=1;

prob.Constraints.con7=x(4)+x(2)+x(6)>=1;

prob.Constraints.con8=x(2)+x(5)>=1;

[sol,fval,flag]=solve(prob),xx=sol.x

**结果如下**：



即此时只要对B1, B4, B5,建设即可

2.3

**问题分析**

这是常见的0-1决策问题，需要我们在限定设备时做出能提供最大收益的决策

**符号说明**

:对第i(i=1,2,3,4)个企业对第j(j=1,2,3,4)个工厂的选取

：:所对因的盈利

**模型建立**

每个企业至少有一个设备，我们有



每个设备都有一台，我们有



于是问题转化为



**代码如下**：

clc,clear

prob=optimproblem('ObjectiveSense','max');

b=[4,2,3,4;6,4,5,5;7,6,7,6;7,8,8,6;7,9,8,6;7,10,8,6]

x=optimvar('x',6,4,'Type','integer','LowerBound',0,'UpperBound',1);

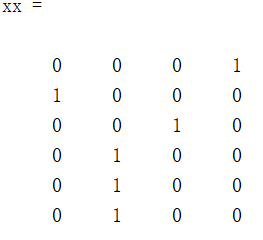
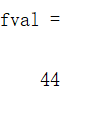
prob.Objective = sum(b.\*x,'all');

prob.Constraints.con1=sum(x,2)==1;

prob.Constraints.con2=sum(x,1)>=1;

[sol,fval,flag]=solve(prob),xx=sol.x

**结果如下：**



即最大收益为44千万

1.7

**问题分析**

这是常见的规划问题，让我们求解变量的最大值

**模型建立**

由题，我们有



代码如下：

clc,clear

prob=optimproblem('ObjectiveSense','max');

a=randi([0,10],100,150);

v=optimvar('v',1,'LowerBound',0)

x=optimvar('x',100,150,'LowerBound',0);

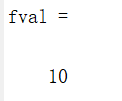
prob.Objective =v;

prob.Constraints.con1=sum(a.\*x,1)>=v;

prob.Constraints.con2=sum(x,1)==1;

[sol,fval,flag]=solve(prob),xx=sol.x;

解得



即v的最大值为10

2.7

**问题分析**

这是常见的规划问题，需要我们求出获利最大时的选取

**符号说明**

:对家电Ⅰ的选取数

：对家电Ⅱ的选取数

**模型建立**

对于设备A，我们有



对于设备B我们有



对于调试工序，我们有



于是问题可转化为



代码如下：

clc,clear

prob=optimproblem('ObjectiveSense','max');

v=optimvar('v',1,'LowerBound',0);

x=optimvar('x',1,'LowerBound',0);

prob.Objective =2\*x+v;

prob.Constraints.con1=5\*v<=15;

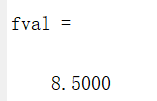
prob.Constraints.con2=6\*x+2\*v<=24;

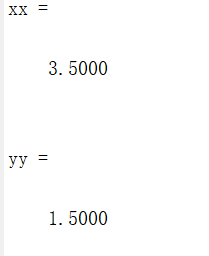
prob.Constraints.con3=x+v<=5;

[sol,fval,flag]=solve(prob)

xx=sol.x,yy=sol.v

结果如下：





即对Ⅰ选3.5台，对Ⅱ选1.5台时，我们有最大值8.5

3.2

**问题分析**

这是一题典型的同余方程组问题，我们只需求出最小解即可

**符号假设**

x：鸡蛋的数量

**模型建立**

由题，可构建如下方程



代码如下

clc,clear

x = 1;

while true

if rem(x, 2) == 1 &&rem(x,4)==1&&rem(x,3)==0&& rem(x, 9) == 0&& rem(x,5)==4&&rem(x,6)==3&&rem(x,7)==4&&rem(x,8)==1;

break;

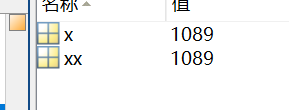
end

x = x + 1;

end

xx = x;

结果如下



即最小值为1089

3.7

**问题分析**

这是一道组合投资问题，需要我们求出在给定投资数下时的投资收益

**符号说明**

**：**购买股票i的数量

：A,B,C相关收益的标准差

：i和j的相关系数

**模型建立**

由题目所给信息，我们首先能求出收益的协方差矩阵



此时风险**X**就能表示为



此时我们的收益**Z**可表示为



则问题就转化为



代码如下：

clc,clear

prob=optimproblem;

R=[4,2.5,-10;2.5,36,-15;-10,-15,100]

x=optimvar('x',3,'LowerBound',0);

prob.Objective =x'\*R\*x;

x0.x=rand(3,1);

for n=0:100

prob.Constraints.con1=5\*x(1)+8\*x(2)+10\*x(3)>=0.01\*n\*500000;

prob.Constraints.con2=20\*x(1)+25\*x(2)+30\*x(3)<=500000;

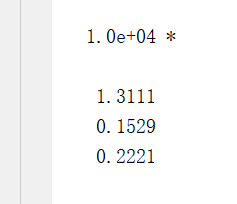
[sol,fval,flag,out]=solve(prob,x0)

y=fval;

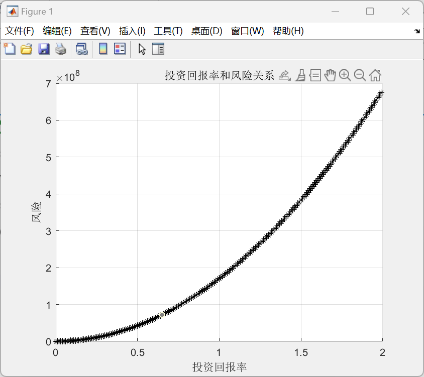
plot(0.01\*n,y);

end

**结果如下**

第一题：

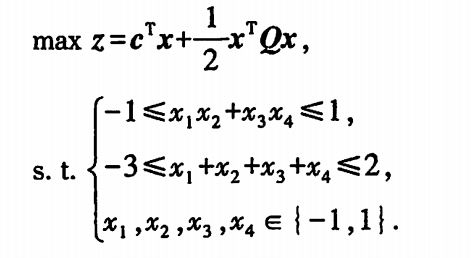
A投13111，B投1529，C投2221

第二题：

从图像上显示，投资回报率越高，风险越高

3.9

问题如下



代码如下

clc,clear

prob=optimproblem('ObjectiveSense','max');

x=optimvar('x',4,'LowerBound',-1,'UpperBound',1);

c=[6,8,4,2];

Q=[-1,2,0,0;2,-1,2,0;0,2,-1,2;0,0,2,-1];

prob.Objective = c\*x+0.5\*x'\*Q\*x;

prob.Constraints.con1=x(1)\*x(2)+x(3)\*x(4)>=-1;

prob.Constraints.con2=x(1)\*x(2)+x(3)\*x(4)<=1;

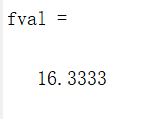
prob.Constraints.con3=x(1)+x(2)+x(3)+x(4)>=-3;

prob.Constraints.con3=x(1)+x(2)+x(3)+x(4)<=2;

x0.x=rand(4,1);

[sol,fval,flag,out]=solve(prob,x0),xx=sol.x;

结果如下：



**4.1**

**代码如下：**

clc,clear,close all

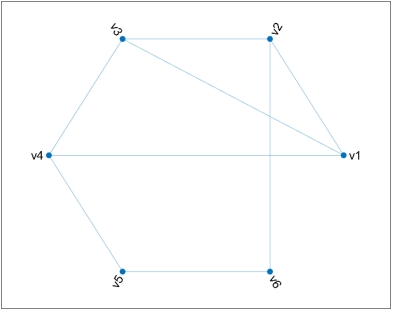
a1=zeros(6);

a1(1,[2:4])=1;a1(2,[3,6])=1;a1(3,4)=1;a1(4,5)=1;a1(5,6)=1;

s=cellstr(strcat('v',int2str([1:6]')));

G1=graph(a1,s,'upper');

plot(G1,'Layout','circle')



a2=zeros(6);

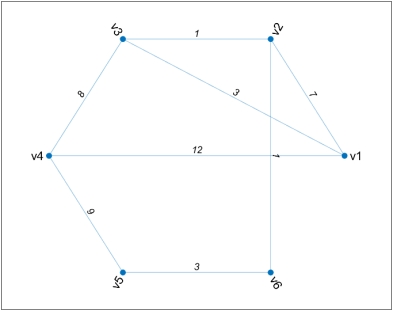
a2(1,[2:4])=[7,3,12];a2(2,[3,6])=[1,1];

a2(3,4)=8;a2(4,5)=9;a2(5,6)=3;

s=cellstr(strcat('v',int2str([1:6]')));

G2=graph(a2,s,'upper');

plot(G2,'Layout','circle','EdgeLabel',G2.Edges.Weight)



a3=zeros(6);

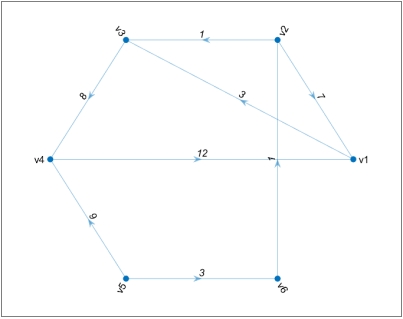
a3(1,3)=3;a3(2,[1,3])=[7,1];a3(3,4)=8;

a3(4,1)=12;a3(5,[4,6])=[9,3];a3(6,2)=1;

s=cellstr(strcat('v',int2str([1:6]')));

G3=digraph(a3,s);

plot(G3,'EdgeLabel',G3.Edges.Weight,'Layout','circle')



**4.4**

**问题分析：**

这是一个最短路径的问题，可以使用Dijkstra标号算法求解。

**符号说明：**

分别用p,d表示最短路径和最短距离。

**模型建立：**

1. 首先从v1出发，v1到v1的最短距离为0，标记节点1
2. 从v1出发，到v2的距离为20，到v5的距离为15，节点2和节点5更新，其前面点均为节点1，标记节点5
3. 从v1出发，经过v5,到v2的距离为40，到v3的距离为33，到v4的距离为50，到v6的距离为30，节点3、节点4和节点6更新，其前面点均为节点5，标记节点2
4. 从v1出发，经过v2,到v3的距离为40，到v4的距离为80，到v5的距离为45，不更新任何节点，标记节点6
5. 从v1出发，经过v6,到v4的距离为40，节点4更新，其前面点为节点6，标记节点3
6. 从v1出发，经过v3,到v4的距离为63，不更新任何节点，标记节点4，结束。

求得从v1到v4的最短路径为v1→v5→v6→v4，最短距离为40。

**代码：**

clc,clear,close all

a=zeros(6);

a(1,[2,5])=[20,15];a(2,[3:5])=[20,60,25];

a(3,[4,5])=[30,18];a(4,[5,6])=[35,10];a(5,6)=15;

s=cellstr(strcat('v',int2str([1:6]')));

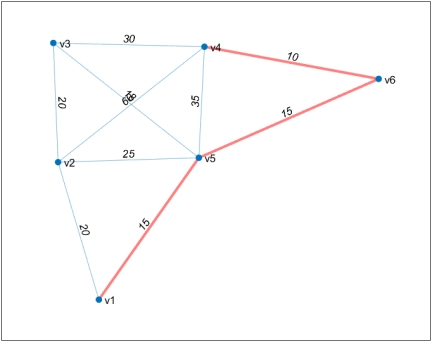
G=graph(a,s,'upper');

[p,d]=shortestpath(G,1,4)

h=plot(G,'EdgeLabel',G.Edges.Weight);

highlight(h,p,'EdgeColor','r','LineWidth',2)

disp('(d)赋权无向图');



**运行结果：**

p = 1×4

1 5 6 4

d = 40

4.2

这题是一题典型的求最短路径问题，我们可以用到Dijkstra算法来求解

**代码如下：**

clc, clear;

L= {'A','B1',2;'A','B2',4 ;'B1','C1',3

'B1','C2',3;'B1','C3',1;'B2','C1',2

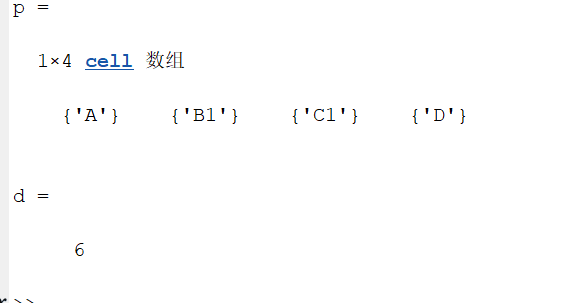
'B2','C2',3;'B2','C3',1;'C1','D',1

'C2','D',3 ;'C3','D',4};

G=digraph (L(:,1),L(:,2),cell2mat (L(:,3)));

plot (G), [p,d]=shortestpath(G,'A','D')

**结果如下：**



4.7

这题也是一题典型的求最短路径问题，同样的，我们可以用到Dijkstra算法来求解

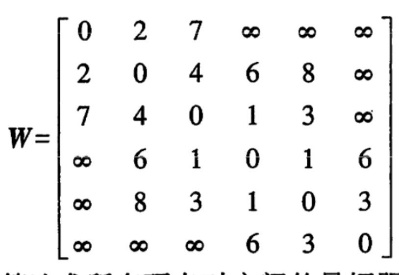
**符号说明**

：到j村所需要的距离

：j村有的小学生人数

**模型建立**

我们首先可以得到对应的邻接矩阵



然后我们可以调用Dijkstra算法来求出所有顶点对的距离，最后问题便转化为求



**代码如下：**

clc,clear,w=zeros(6);

w(1,[2,3])=[2,7];w(2,[3:5])=[4,6,8];

w(3,[4,5])=[1,3];w(4,[5,6])=[1,6];

w(5,6)=3;

G=graph(w,'Upper');

d=distances(G)

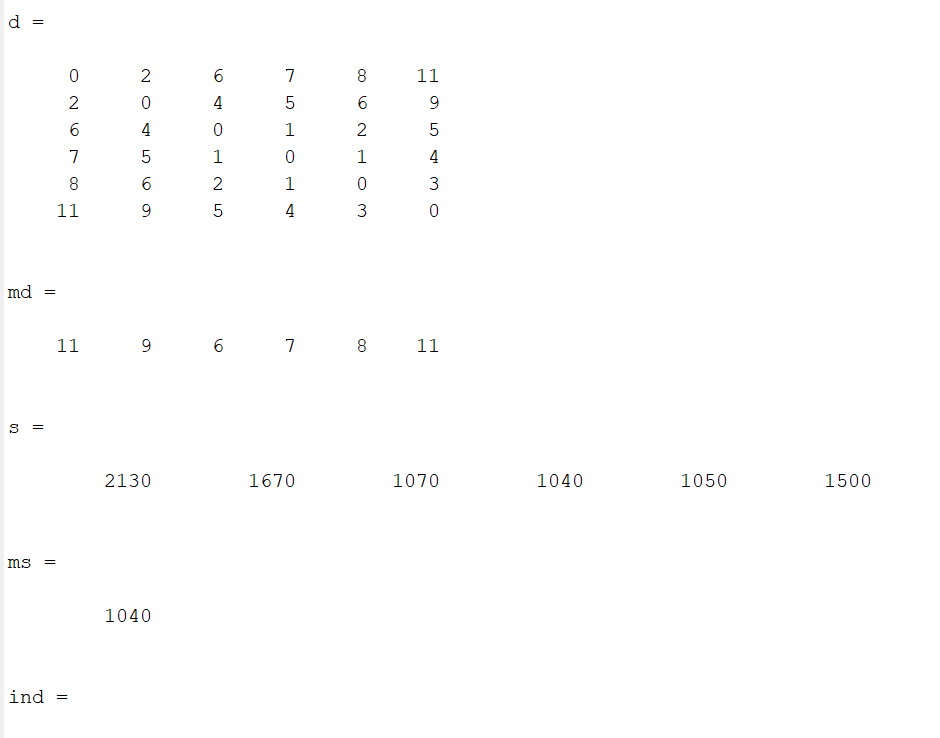
md=max(d)

c=[50 40 60 20 70 90];

s=c\*d

[ms,ind]=min(s)

**结果如下：**

结果

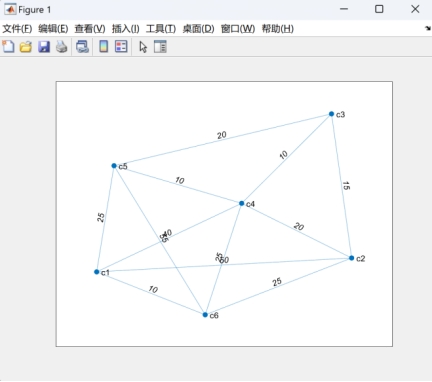
结果表明，选村庄四最好

补充题：

1. 问题分析

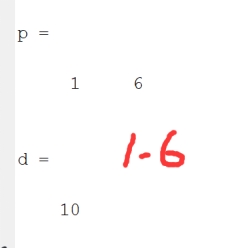
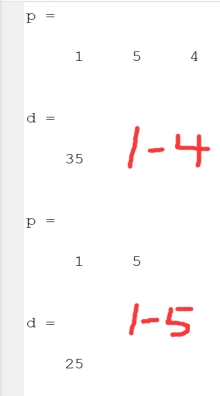
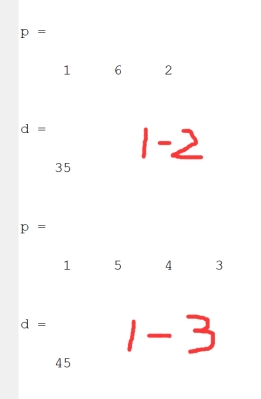
某公司在六个城市c1, c2, ... , c6中有分公司,从ci到cj,的直接航程票价记在下述矩阵的(i,j)位置上（∞表示无直接航路)。画出该矩阵对应的赋权图（顶点和边都要有标注），并帮助该公司设计一个简便的算法，能快速得到一张城市c1到其它城市间的票价最便宜的路线图。

1. 赋权图

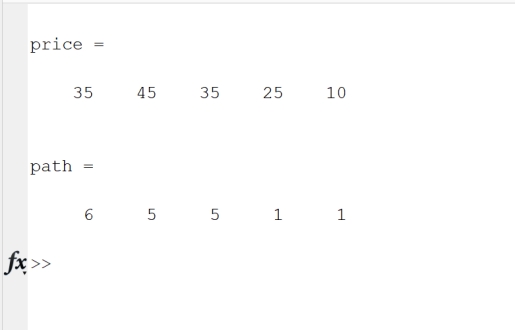


1. 从求解器确定答案：

从c1到其他城市的最短路径以及最短距离如下所示



4.从Floyd算法求解



有图可知:

c1 -c2:1-6-2 ,35

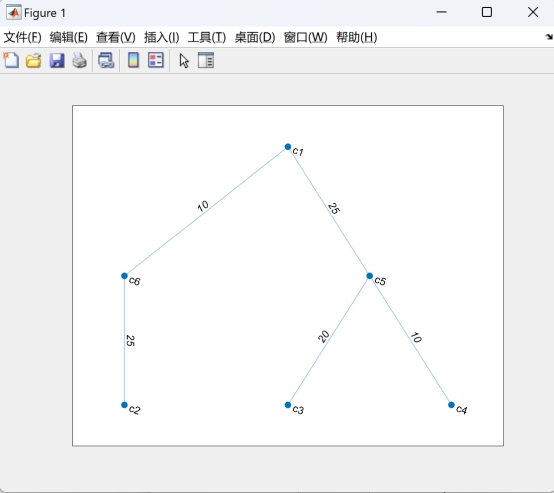
c1 -c3:1-5-3 ,45

c1 -c4:1-5-4 ,35

c1 -c5:1-5 ,25

c1 -c6:1-6 ,10

票价最便宜路线图



代码如下：

clc,clear,w = zeros(6);

w(1,[2,4,5,6]) = [50,40,25,10];w(2,[1,3,4,6]) = [50,15,20,25];

w(3,[2,4,5]) = [15,10,20];w(4,[1,2,3,5,6]) = [40,20,10,10,25];

w(5,[1,3,4,6]) = [25,20,10,55];w(6,[1,2,4,5]) = [10,25,25,55];

%构造完整的邻接矩阵

s = cellstr(strcat('c',int2str([1:6]')));%顶点字符串

G = graph(w,s,'upper');%利用邻接矩阵的上三角元素构造无向图

plot(G,'EdgeLabel',G.Edges.Weight);%画无向图

[price,path] = Floyd(w) %显示c1至其他地方票价最便宜价钱，路线矩阵

p = zeros(6);

p(1,5:6) = [25,10];p(5,3:4) = [20,10];p(6,2) = 25;

p = p+p';

G1 = graph(p,s,'upper');%利用邻接矩阵的上三角元素构造无向图

plot(G1,'EdgeLabel',G1.Edges.Weight);%画最短航线图

function [price,path] = Floyd(w)

%输出矩阵为两两顶点间最短距离矩阵，输入矩阵为待求邻接矩阵

n = length(w);

w(w == 0) = inf; %把零元素换成无穷大

w(1:n+1:end) = 0; %把对角线元素换成0

path = -ones(n);%初始化path矩阵

for i=1:n

for j=1:n

if i~=j && w(i,j)~=inf

path(i,j)=i;

end

end

end %完成path矩阵未更新的建立

for k = 1:n

for i = 1:n

for j = 1:n

if w(i,k)+w(k,j) < w(i,j) %Floyd算法核心，更新

w(i,j) = w(i,k)+w(k,j);

path(i,j) = k; %对w以及path矩阵更新

end

end

end

end

price = w(1,2:n);

path = path(1,2:n);%只考虑c1至其他城市，故只取部分

4.3

这个就只是画出赋权图，我们直接调用prim算法

**代码如下：**

clc,clear

a = zeros(6);

a(1,[2,5]) = [20,15];a(2,[3,4,5]) = [20,60,25];

a(3,[4,5]) = [30,18];a(4,[5,6]) = [35,10];

a(5,6) = 15; a = a+a'; %建立邻接矩阵

s = cellstr(strcat('v',int2str([1:6]')));

G = graph(a,s,'upper');%画出无向赋权图

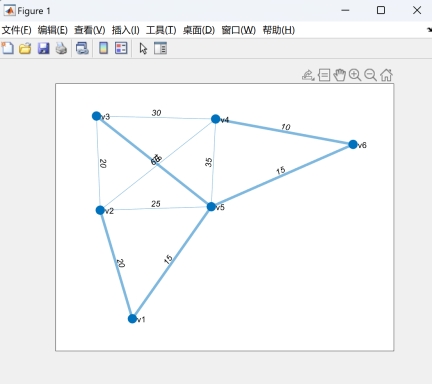
p = plot(G,'EdgeLabel',G.Edges.Weight)

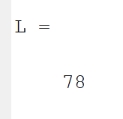
T = minspantree(G)%画出最小生成树

L = sum(T.Edges.Weight)%找出最小生成树的路径，并计算总和

highlight(p,T)%着重标记最小生成树

**结果如下：**





我们可知**最小生成树的长度为78**

4.8

由于我们无法做到如题所示的0.6概率的随机，我们可以调用随机函数矩阵来表示随机的概念，

**代码如下：**

clc,clear

a = rand(10);%构造概率矩阵

a = triu(a,1);%我们取上三角元素

w = randi(10,10);%构造了权重矩阵

W = (a>=0.4).\*w%生成无向赋权图邻接矩阵的上三角部分

W = W +W'%生成完全邻接矩阵

G = graph(W,'upper')

subplot(121),plot(G,'EdgeLabel',G.Edges.Weight)

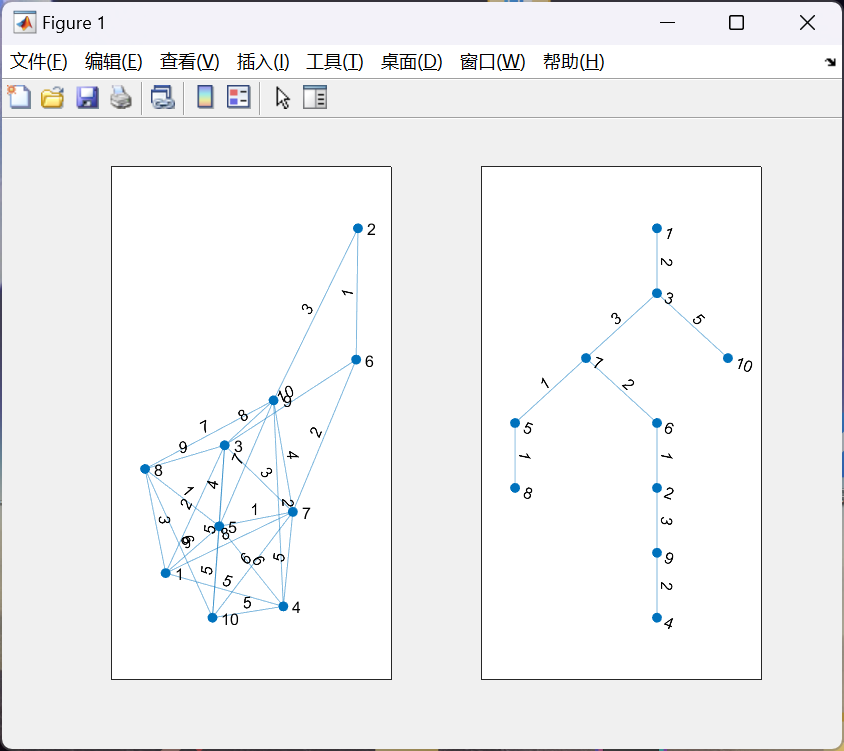
T = minspantree(G)%使用Prim算法求得最小生成树

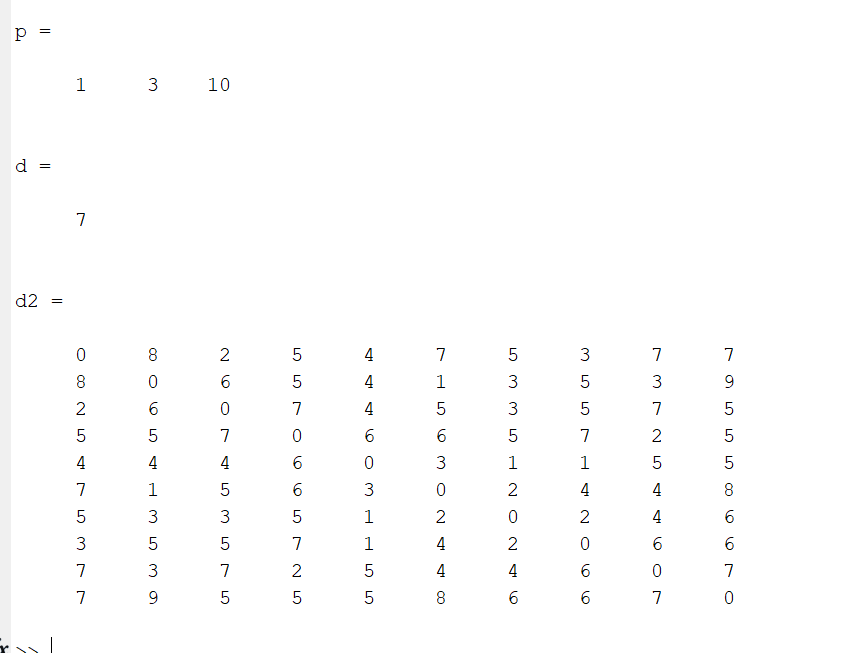
subplot(122),plot(T,'EdgeLabel',T.Edges.Weight)

[p,d] = shortestpath(G,1,10)%q求得1-10的最短距离及最短路径；

d2 = distances(G)

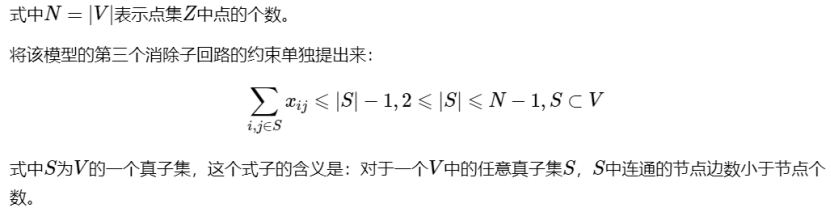
**结果如下：**





1. 最小生成树如上图所示
2. 路径为1→3→10，最短路径长度为7
3. 每个点的最短距离如上

4.13



该问题可以转化为0—1整数规划类问题，具体问题可以转化为如下



**代码如下：**

clc, clear, close all, n = 9;

nod =cellstr(strcat('v',int2str([0:n-1]')));%运用cellstr进行标号

G = graph(); G = addnode(G,nod); %定好无序图

ed ={ 'v0','v1',2;'v0','v2',1;'v0','v3',3;'v0','v4',4

'v0','v5',4;'v0','v6',2;'v0','v7',5;'v0','v8',4

'v1','v2',4;'v1','v8',1;'v2','v3',1;'v3','v4',1

'v4','v5',5;'v5','v6',2;'v6','v7',3;'v7','v8',5};

G = addedge(G,ed(:,1),ed(:,2),cell2mat(ed(:,3)));%无序图确认

p = plot(G,'EdgeLabel',G.Edges.Weight) %做出9个村庄道路及道路长度图

w = full(adjacency(G,'weighted')); %做出边权矩阵

w(w==0) = 1000000; %充分大的正实数，让所有不能直接到达的两个村庄改为足够大

prob = optimproblem;

x = optimvar('x',n,n,'Type','integer','LowerBound',0,'UpperBound',1);

prob.Objective = sum(sum(w.\*x));

prob.Constraints.con1 = 1<=sum(x(1,:));%条件1

prob.Constraints.con2 = sum(x(:,2:end))'==1; %条件2

con3 = [];

for q = 2:n-1

a = zeros(q);

for m = 1:100 %100次足够精度

b = randperm(n);%随机对n数进行排序

c = b(1:q); %相当于从n中随机抽取p个数

a = x(c,c);

con3 = [sum(sum(a)) <= q-1;con3];

end

end

prob.Constraints.con3 = con3;%条件3

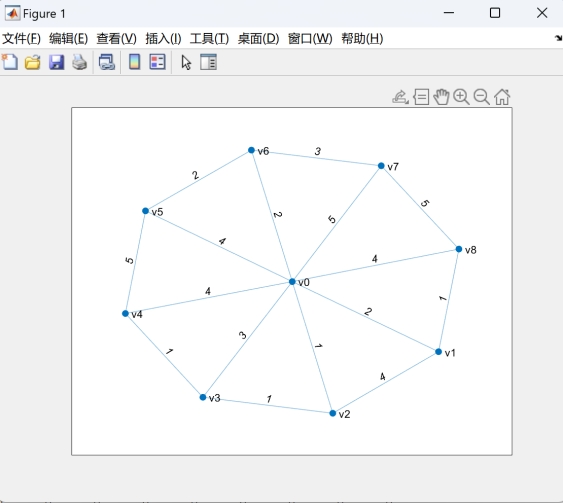
[sol,fval,flag,out] = solve(prob)

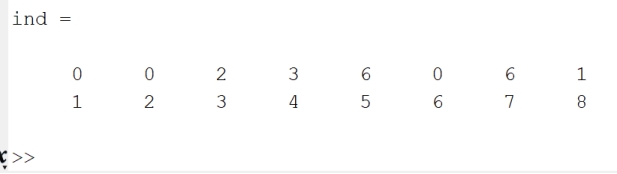
xx = sol.x

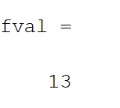
[i,j]=find(sol.x);

ind = [(i-1)'; (j-1)'] %输出树的顶点编号

**结果如下：**





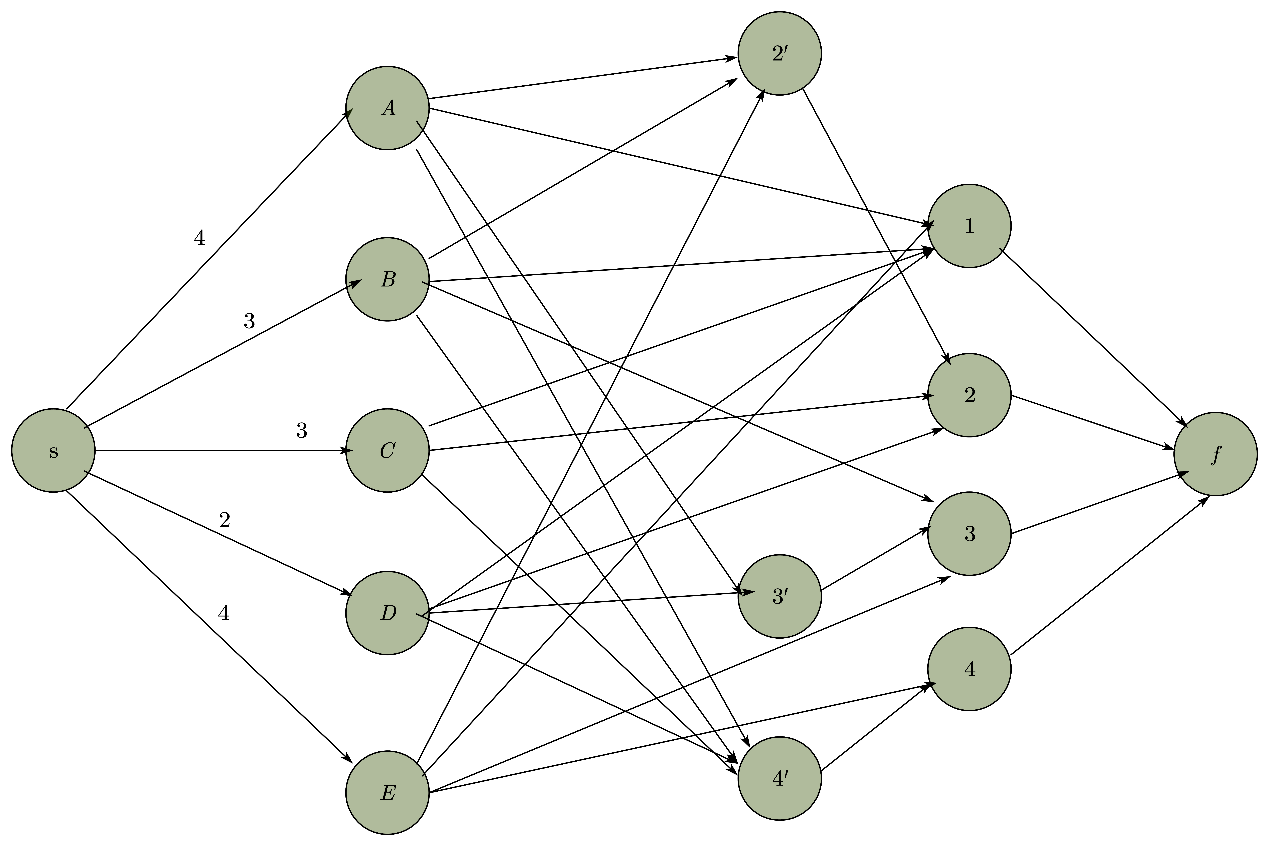


**即最小生成树的坐标和权重都可得到**

4.10

这是个最大流问题。分别有五个专业作为多源，四个公司作为多汇。为此我们需要虚拟假设一个源点和一个汇点。将上述问题转为单对单模型。

由题意可知，转化原问题，可得如下模型：



**代码如下：**

clc,clear

a = zeros(14);%总共有5个专业，4个公司，1个源点，1个汇点，3个中转点

a(1,[2:6]) = [4,3,3,2,4];a(2,[7:10]) = 1;

a(3,[7,9,10,12]) = 1;a(4,[9:12]) = 1;

a(5,[8:12]) = 1;a(6,[7,10,12,13]) = 1;

a(7,11) = 2;a(8,12) = 1;a(9,13) = 2;

a([10:13],14) = [5,4,4,3];%将权赋好

s = cellstr(strcat('v',int2str([1:14]')));%命名序号

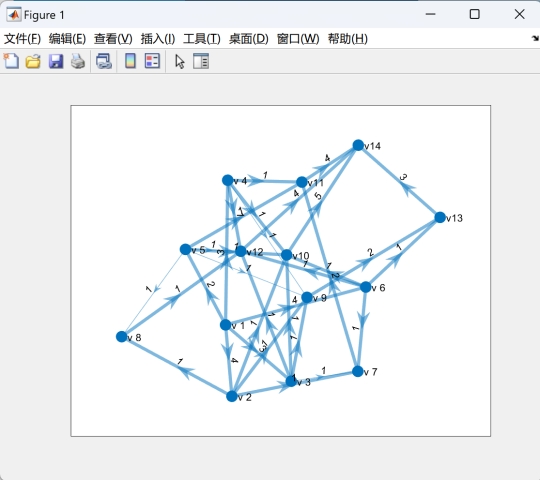
G = digraph(a,s);%确定赋权图

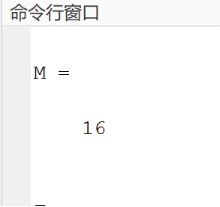
[M,F] = maxflow(G,1,14) %使用默认searchtrees方法求最大流

p = plot(G,'Layout','force','EdgeLabel',G.Edges.Weight);

highlight(p,F)%显示最大流并画出最大流

**结果如下：**





所以最大流量是16，正好与五个专业一共十六名毕业生想对应，所以所有公司都能招聘到各自需要的专业人才。