randi.mlx

side=1000; %区域边长

numjob=10000; %发达地区线下工作数量

numljob=10000; %欠发达地区线下工作数量

numujob=1000; %全部地区远程工作数量

D1Q11X=randi([floor(side\*1/6),floor(side\*2/3)],floor(0.1535\*numljob),1); %欠发达地区Low Requirement(q=1)随机数

D1Q11Y=randi([0,side],floor(0.1535\*numljob),1);

D1Q11W=randi([1,3],floor(0.1535\*numljob),1);

D1Q11H=randi([20,40],floor(0.1535\*numljob),1);

D1Q11=[D1Q11X,D1Q11Y,D1Q11W,D1Q11H];

D1Q21X=randi([floor(side\*1/6),floor(side\*2/3)],floor(0.4386\*numljob),1); %欠发达地区Mid-Low Requirement(q=2)随机数

D1Q21Y=randi([0,side],floor(0.4386\*numljob),1);

D1Q21W=randi([3,8],floor(0.4386\*numljob),1);

D1Q21H=randi([30,40],floor(0.4386\*numljob),1);

D1Q21=[D1Q21X,D1Q21Y,D1Q21W,D1Q21H];

D1Q31X=randi([floor(side\*1/6),floor(side\*2/3)],floor(0.3289\*numljob),1); %欠发达地区Medium Requirement(q=3)随机数

D1Q31Y=randi([0,side],floor(0.3289\*numljob),1);

D1Q31W=randi([8,10],floor(0.3289\*numljob),1);

D1Q31H=randi([30,50],floor(0.3289\*numljob),1);

D1Q31=[D1Q31X,D1Q31Y,D1Q31W,D1Q31H];

D1Q41X=randi([floor(side\*1/6),floor(side\*2/3)],floor(0.0768\*numljob),1); %欠发达地区Mid-High Requirement(q=4)随机数

D1Q41Y=randi([0,side],floor(0.0768\*numljob),1);

D1Q41W=randi([10,20],floor(0.0768\*numljob),1);

D1Q41H=randi([30,50],floor(0.0768\*numljob),1);

D1Q41=[D1Q41X,D1Q41Y,D1Q41W,D1Q41H];

numljobh=numljob-floor(0.1535\*numljob)-floor(0.4386\*numljob)-floor(0.3289\*numljob)-floor(0.0768\*numljob); %欠发达地区High Requirement(q=5)随机数的数量

D1Q51X=randi([floor(side\*1/6),floor(side\*2/3)],numljobh,1); %欠发达地区High Requirement(q=5)随机数

D1Q51Y=randi([0,side],numljobh,1);

D1Q51W=randi([20,25],numljobh,1);

D1Q51H=randi([40,50],numljobh,1);

D1Q51=[D1Q51X,D1Q51Y,D1Q51W,D1Q51H];

D1Q12X1=randi([0,floor(side\*1/6)],ceil(0.0400\*numjob\*1/3),1); %发达地区Low Requirement(q=1)随机数

D1Q12X2=randi([floor(side\*2/3),side],floor(0.0400\*numjob\*2/3),1);

D1Q12X=[D1Q12X1;D1Q12X2];

D1Q12Y=randi([0,side],0.0400\*numjob,1);

D1Q12W=randi([3,5],0.0400\*numjob,1);

D1Q12H=randi([20,40],0.0400\*numjob,1);

D1Q12=[D1Q12X,D1Q12Y,D1Q12W,D1Q12H];

D1Q22X1=randi([0,floor(side\*1/6)],ceil(0.1000\*numjob\*1/3),1); %发达地区Mid-Low Requirement(q=2)随机数

D1Q22X2=randi([floor(side\*2/3),side],floor(0.1000\*numjob\*2/3),1);

D1Q22X=[D1Q22X1;D1Q22X2];

D1Q22Y=randi([0,side],0.1000\*numjob,1);

D1Q22W=randi([3,10],0.1000\*numjob,1);

D1Q22H=randi([20,30],0.1000\*numjob,1);

D1Q22=[D1Q22X,D1Q22Y,D1Q22W,D1Q22H];

D1Q32X1=randi([0,floor(side\*1/6)],ceil(0.2600\*numjob\*1/3),1); %发达地区Medium Requirement(q=3)随机数

D1Q32X2=randi([floor(side\*2/3),side],floor(0.2600\*numjob\*2/3),1);

D1Q32X=[D1Q32X1;D1Q32X2];

D1Q32Y=randi([0,side],0.2600\*numjob,1);

D1Q32W=randi([10,15],0.2600\*numjob,1);

D1Q32H=randi([20,50],0.2600\*numjob,1);

D1Q32=[D1Q32X,D1Q32Y,D1Q32W,D1Q32H];

D1Q42X1=randi([0,floor(side\*1/6)],ceil(0.3600\*numjob\*1/3),1); %发达地区Mid-High Requirement(q=4)随机数

D1Q42X2=randi([floor(side\*2/3),side],floor(0.3600\*numjob\*2/3),1);

D1Q42X=[D1Q42X1;D1Q42X2];

D1Q42Y=randi([0,side],0.3600\*numjob,1);

D1Q42W=randi([20,30],0.3600\*numjob,1);

D1Q42H=randi([40,50],0.3600\*numjob,1);

D1Q42=[D1Q42X,D1Q42Y,D1Q42W,D1Q42H];

D1Q52X1=randi([0,floor(side\*1/6)],ceil(0.2400\*numjob\*1/3),1); %发达地区High Requirement(q=5)随机数

D1Q52X2=randi([floor(side\*2/3),side],floor(0.2400\*numjob\*2/3),1);

D1Q52X=[D1Q52X1;D1Q52X2];

D1Q52Y=randi([0,side],0.2400\*numjob,1);

D1Q52W=randi([30,50],0.2400\*numjob,1);

D1Q52H=randi([40,50],0.2400\*numjob,1);

D1Q52=[D1Q52X,D1Q52Y,D1Q52W,D1Q52H];

D1Q1=[D1Q11;D1Q12]; %全部地区Low Requirement(q=1)随机数

D1Q2=[D1Q21;D1Q22]; %全部地区Mid-Low Requirement(q=2)随机数

D1Q3=[D1Q31;D1Q32]; %全部地区Medium Requirement(q=3)随机数

D1Q4=[D1Q41;D1Q42]; %全部地区Mid-High Requirement(q=4)随机数

D1Q5=[D1Q51;D1Q52]; %全部地区High Requirement(q=5)随机数

D0Q3XY=randi([0,side],floor(0.34\*numujob),2);

D0Q3W=randi([15,20],floor(0.34\*numujob),1);

D0Q3H=randi([20,50],floor(0.34\*numujob),1);

D0Q3=[D0Q3XY,D0Q3W,D0Q3H]; %全部地区远程工作Medium Requirement(q=3)随机数

D0Q5XY=randi([0,side],numujob-floor(0.34\*numujob),2);

D0Q5W=randi([25,50],numujob-floor(0.34\*numujob),1);

D0Q5H=randi([20,40],numujob-floor(0.34\*numujob),1);

D0Q5=[D0Q5XY,D0Q5W,D0Q5H]; %全部地区远程工作High Requirement(q=5)随机数

D1Q1(:,5)=1; %标记工作难度

D1Q2(:,5)=2;

D1Q3(:,5)=3;

D1Q4(:,5)=4;

D1Q5(:,5)=5;

D0Q3(:,5)=3;

D0Q5(:,5)=5;

Findjob.m

function [] = Findjob(A,S,X,Y,D)

randijob; %随机生成全部工作

d=0;

if D>0 %提取出符合条件的线下工作

switch S(1,1) %将满足矩阵S中第一个点的工作赋给d

case 1

d=D1Q1;

case 2

d=D1Q2;

case 3

d=D1Q3;

case 4

d=D1Q4;

case 5

d=D1Q5;

end

while size(S,2)>1 %每赋值一次，S列数减一

S=S(1,2:size(S,2));

switch S(1,1) %将满足矩阵S各点的工作放入d

case 1

d=[d;D1Q1];

case 2

d=[d;D1Q2];

case 3

d=[d;D1Q3];

case 4

d=[d;D1Q4];

case 5

d=[d;D1Q5];

end

end

else %提取出符合条件的远程工作

switch S(1,1)

case 3

d=D0Q3;

case 5

d=D0Q5;

end

if size(S,2)>1

d=[D0Q3;D0Q5];

end

end

for i=1:size(d,1) %计算学生坐标与每个工作的距离

B(i,1)=ceil(((X-d(i,1))^2+(Y-d(i,2))^2)^(1/2));

end

d=[d,B];

if(D>0)

o=1;

for i=1:size(d,1) %留下在大学生可接受距离内的工作

if d(i,6)<=D

d(o,:)=d(i,:);

o=o+1;

end

end

end

if o==1

error('There is no work of this type in this scope\n');

end

d=d(1:(o-1),:);

B1(10,10)=0;

B2(10,10)=0;

B3(10,10)=0;

c=0;

d(1,1)=0;

if(D>0) %选择线下工作的情况下，其工作重要性权重与时薪、时长、距离有关

while(size(d,1)>1) %循环，直到筛选出最适合此学生的工作

if size(d,1)>10

for j=1:10

for k=1:10

m=max(d(1:10,3))-min(d(1:10,3)); %十个工作中时薪极差

B1(10,10)=0;

if d(j,3)>d(k,3)

B1(j,k)=ceil((d(j,3)-d(k,3))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,3)<d(k,3)

B1(j,k)=1/ceil((d(k,3)-d(j,3))\*9/m); %去量纲化 构造成对比较矩阵

else

B1(j,k)=1;

end

m=max(d(1:10,4))-min(d(1:10,4)); %十个工作中时长极差

B2(10,10)=0;

if d(j,4)>d(k,4)

B2(j,k)=1/ceil((d(j,4)-d(k,4))\*9/m); %去量纲化,构造成对比较矩阵

elseif d(j,4)<d(k,4)

B2(j,k)=ceil((d(k,4)-d(j,4))\*9/m); %去量纲化 构造成对比较矩阵

else

B2(j,k)=1;

end

m=max(d(1:10,6))-min(d(1:10,6)); %十个工作中距离极差

B3(10,10)=0;

if d(j,6)>d(k,6)

B3(j,k)=1/ceil((d(j,6)-d(k,6))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,6)<d(k,6)

B3(j,k)=ceil((d(k,6)-d(j,6))\*9/m); %去量纲化 构造成对比较矩阵

else

B3(j,k)=1;

end

end

end

[W, Lmax, CI, CR] = AHP(A); %用户输入的A 时薪、时长、距离的权重

[W1, Lmax1, CI1, CR1] = AHP(B1); %各工作时薪权重

[W2, Lmax2, CI2, CR2] = AHP(B2); %各工作时长权重

[W3, Lmax3, CI3, CR3] = AHP(B3); %各工作距离权重

rlts3 =[W1 W2 W3; %合并结果

Lmax1 Lmax2 Lmax3;

CI1 CI2 CI3;

CR1 CR2 CR3];

a10 = rlts3(1:10,:) \* W;

c=find(a10==max(a10(:))); %找到权重最大的工作所在的位置

f=d(c,:); %f存储权重最大的工作所在的位置

d=[f;d(11:size(d,1),:)]; %滤除前十个工作中较不合适的九个矩阵

else

n=size(d,1);

B1=0;B2=0;B3=0;

B1(n,n)=0;

B2(n,n)=0;

B3(n,n)=0;

for j=1:n

for k=1:n

m=max(d(1:n,3))-min(d(1:n,3)); %最后少于十个工作中时薪极差

B1(n,n)=0;

if d(j,3)>d(k,3)

B1(j,k)=ceil((d(j,3)-d(k,3))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,3)<d(k,3)

B1(j,k)=1/ceil((d(k,3)-d(j,3))\*9/m); %去量纲化 构造成对比较矩阵

else

B1(j,k)=1;

end

m=max(d(1:n,4))-min(d(1:n,4)); %最后少于十个工作中时长极差

B2(n,n)=0;

if d(j,4)>d(k,4)

B2(j,k)=1/ceil((d(j,4)-d(k,4))\*9/m);

elseif d(j,4)<d(k,4)

B2(j,k)=ceil((d(k,4)-d(j,4))\*9/m);

else

B2(j,k)=1;

end

m=max(d(1:n,6))-min(d(1:n,6)); %最后少于十个工作中距离极差

B3(n,n)=0;

if d(j,6)>d(k,6)

B3(j,k)=1/ceil((d(j,6)-d(k,6))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,6)<d(k,6)

B3(j,k)=ceil((d(k,6)-d(j,6))\*9/m); %去量纲化 构造成对比较矩阵

else

B3(j,k)=1;

end

end

end

[W, Lmax, CI, CR] = AHP(A); %用户输入的A

[W1, Lmax1, CI1, CR1] = AHP(B1); %各工作时薪权重

[W2, Lmax2, CI2, CR2] = AHP(B2); %各工作时长权重

[W3, Lmax3, CI3, CR3] = AHP(B3); %各工作距离权重

rlts3 =[W1 W2 W3; %合并结果

Lmax1 Lmax2 Lmax3;

CI1 CI2 CI3;

CR1 CR2 CR3];

a10 = rlts3(1:n,:) \* W;

c=find(a10==max(a10(:))); %找到少于n个工作中权重最大的矩阵

d=d(c,:); %留下最合适的工作

end

end

else %选择远程工作的情况下，其工作重要性权重与时薪、时长有关

while(size(d,1)>1) %循环，直到筛选出最适合此学生的工作

if size(d,1)>10

for j=1:10

for k=1:10

m=max(d(1:10,3))-min(d(1:10,3)); %十个工作中时薪极差

B1(10,10)=0;

if d(j,3)>d(k,3)

B1(j,k)=ceil((d(j,3)-d(k,3))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,3)<d(k,3)

B1(j,k)=1/ceil((d(k,3)-d(j,3))\*9/m); %去量纲化 构造成对比较矩阵

else

B1(j,k)=1;

end

m=max(d(1:10,4))-min(d(1:10,4)); %十个工作中时长极差

B2(10,10)=0;

if d(j,4)>d(k,4)

B2(j,k)=1/ceil((d(j,4)-d(k,4))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,4)<d(k,4)

B2(j,k)=ceil((d(k,4)-d(j,4))\*9/m); %去量纲化 构造成对比较矩阵

else

B2(j,k)=1;

end

end

end

[W, Lmax, CI, CR] = AHP(A); %用户输入的A

[W1, Lmax1, CI1, CR1] = AHP(B1); %各工作时薪权重

[W2, Lmax2, CI2, CR2] = AHP(B2); %各工作时长权重

rlts2 =[W1 W2; %合并结果

Lmax1 Lmax2;

CI1 CI2;

CR1 CR2];

a10 = rlts2(1:10,:) \* W;

c=find(a10==max(a10(:))); %找到权重最大的工作所在的位置

f=d(c,:); %存储此工作

d=[f;d(11:size(d,1),:)]; %滤除前十个工作中较不合适的九个矩阵

else

n=size(d,1);

B1=0;B2=0;

B1(n,n)=0;

B2(n,n)=0;

for j=1:n

for k=1:n

m=max(d(1:n,3))-min(d(1:n,3)); %n个工作中时薪极差

B1(n,n)=0;

if d(j,3)>d(k,3)

B1(j,k)=ceil((d(j,3)-d(k,3))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,3)<d(k,3)

B1(j,k)=1/ceil((d(k,3)-d(j,3))\*9/m); %去量纲化 构造成对比较矩阵

else

B1(j,k)=1;

end

m=max(d(1:n,4))-min(d(1:n,4)); %n个工作中时长极差

B2(n,n)=0;

if d(j,4)>d(k,4)

B2(j,k)=1/ceil((d(j,4)-d(k,4))\*9/m); %去量纲化 构造成对比较矩阵

elseif d(j,4)<d(k,4)

B2(j,k)=ceil((d(k,4)-d(j,4))\*9/m); %去量纲化 构造成对比较矩阵

else

B2(j,k)=1;

end

end

end

[W, Lmax, CI, CR] = AHP(A); %用户输入的A

[W1, Lmax1, CI1, CR1] = AHP(B1); %各工作时薪权重

[W2, Lmax2, CI2, CR2] = AHP(B2); %各工作时长权重

rlts3 =[W1 W2; %合并结果

Lmax1 Lmax2;

CI1 CI2;

CR1 CR2];

a10 = rlts3(1:n,:) \* W;

c=find(a10==max(a10(:))); %找到权重最大的工作所在的位置

d=d(c,:); %留下最符合条件的工作

end

end

end

fprintf('A most suitable job:\n');

fprintf(' position :(%d,%d)\n',d(1,1),d(1,2)); %最符合条件工作的坐标

fprintf(' wage : %d\n',d(1,3)); %最符合条件工作的时薪

fprintf(' hours : %d\n',d(1,4)); %最符合条件工作的每周工作时长

fprintf('difficulty index: %d\n',d(1,5)); %最符合条件工作的难度指数

fprintf(' distance : %d\n',d(1,6)); %最符合条件工作的距离

AHP.m

function [W, Lmax, CI, CR] = AHP(A)

% 实现单层次结构的层次分析法

% 输入: A为成对比较矩阵

% 输出: W为权重向量, Lmax为最大特征值, CI为一致性指标, CR为一致性比率

[V,D] = eig(A);

[Lmax,ind] = max(diag(D)); % 求最大特征值及其位置

W = V(:,ind) / sum(V(:,ind)); % 最大特征值对应的特征向量做标准化

Lmax = mean((A \* W) ./ W); % 计算最大特征值

n = size(A, 1); % 矩阵行数

CI = (Lmax - n) / (n - 1); % 计算一致性指标

% Saaty随机一致性指标值

RI = [0 0 0.58 0.90 1.12 1.24 1.32 1.41 1.45 1.49 1.51];

CR = CI / RI(n); % 计算一致性比率

if CR>=0.1

error('未通过一致性检验，程序退出');

end