**南京理工大学计算机科学与技术学院**

**计算机科学中的建模技术大作业**

|  |  |
| --- | --- |
| **班级** |  |
| **学号** |  |
| **姓名** |  |

目录

[1](#_Toc124719089)

[1 问题分析 3](#_Toc124719090)

[1.1 问题重述 3](#_Toc124719091)

[1.1.1 要求 3](#_Toc124719092)

[1.1.2 前情概要 3](#_Toc124719093)

[1.1.3 前提假设 3](#_Toc124719094)

[1.2 模型假设 4](#_Toc124719095)

[1.2.1 问题一分析 4](#_Toc124719096)

[1.2.2 问题二分析 5](#_Toc124719097)

[1.3 符号说明 5](#_Toc124719098)

[2 建模过程 5](#_Toc124719099)

[2.1 确定影响因子权重向量 5](#_Toc124719100)

[2.1.1 得到成对比较阵 5](#_Toc124719101)

[2.1.2 计算权重向量并进行一致性检验 6](#_Toc124719102)

[2.1.3 计算风险值 6](#_Toc124719103)

[2.1.4 获得邻接矩阵 7](#_Toc124719104)

[2.1.5 使用Dijkstra算法计算最安全路径 7](#_Toc124719105)

[2.1.6 选择最实用的防护措施 8](#_Toc124719106)

[3 相关代码 8](#_Toc124719107)

[3.1 Init.m 8](#_Toc124719108)

[3.2 cal.m 9](#_Toc124719109)

[3.3 main.m 9](#_Toc124719110)

[4 求解答案 13](#_Toc124719111)

# 问题分析

## 问题重述

### 要求

建立一个通用的数学模型，可以帮助“搜索者”制定一个可用的针对阳性患者追踪的搜索方案，辅助选择确定每天上班/上学的最安全路线和最实用的防护方式。

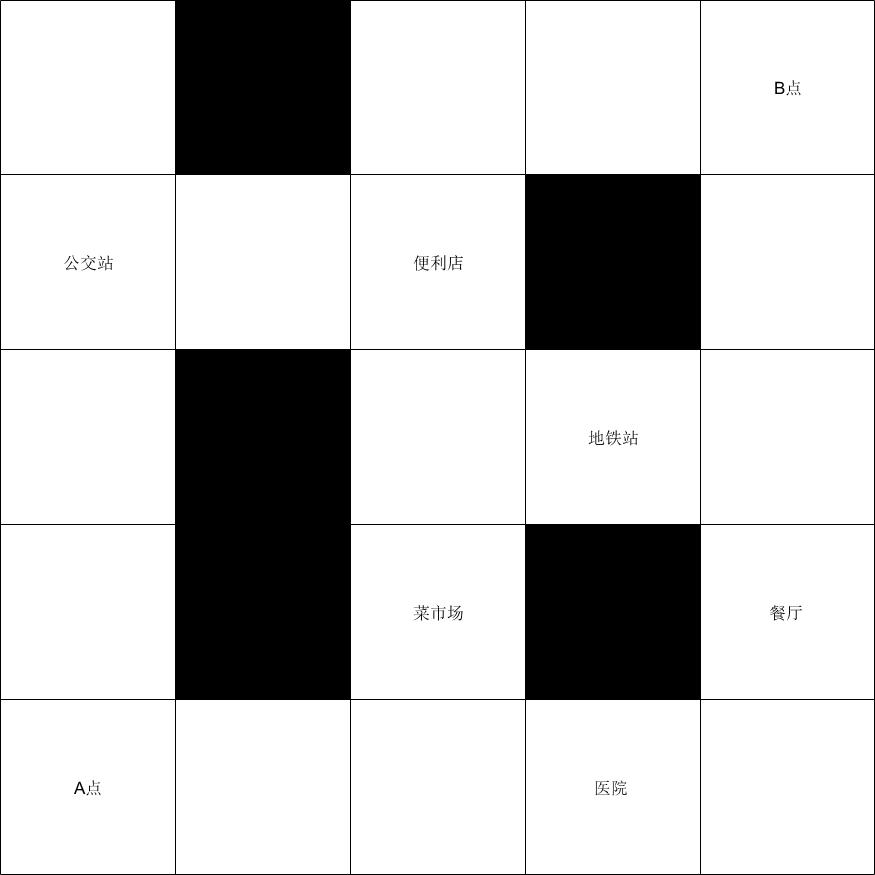
### 前情概要

所有人每天在从 A 点（家里）到 B 点（公司）工作或者学习，中途可能停留在便利店、公交站、地铁站、医院、菜市场、餐厅等区域。

### 前提假设

无法提前获取到所有人的阳性/阴性信号，而且有许多不同类型的感染者（老人、儿童、孕妇、青年等），每个人有不同类型的防护措施（未带口罩、带普通口罩、带N95口罩、穿防护服等），同时不同防护措施的价格和方便程度也不同。

## 模型假设



图表 1地图

假设地图为5x5的矩阵，黑色方格表示不可经过区域，包含六个特殊区域如上。

### 问题一分析

问题一希望给出最安全路径，将求最安全路径转变为求最短路径即可。我们将求解最短路径中的路径映射为求解此问题中的风险值，每个可经过坐标对应一个风险值，表示到达此地点需要经受的感染病毒的风险，然后使用Dijkstra算法求出风险总和最小的路径，即最安全路径。

计算坐标点风险值方面，我们假定所有可经过坐标点的实时人流量在求解过程中固定不变，计算每个路人的风险值并将其相加得到总和即为该坐标点的风险值，最后将其归一化。

计算个人风险值方面，我们考虑了三个影响因子：是否为阳性、类型、防护措施，量化各个影响因子。对于第一个影响因子，由于无法提前获取到所有人的阳性/阴性信号，认为当前数据并不完全准确，我们将影响因子取值范围定为0.9（阳性）和0.1（阴性），而不是1.0（阳性）和0.0（阴性）；对于第二个影响因子，查阅资料可得：儿童和青年新冠病毒传播能力更强，我们将影响因子取值范围定为4（儿童）、3（青年）、2（中年）、1（老年），并将其归一化；对于第三个影响因子，我们将取值范围定为4（未带口罩）、3（带普通口罩）、2（带N95口罩）、0.1（穿防护服），并将其归一化。然后将三种影响因子的值与其权重相乘，得到个人的风险值。

计算影响因子权重方面，我们采用层次分析法计算影响因子权重。

### 问题二分析

问题二希望给出最实用的防护措施。如果单纯给出最安全的防护措施，我们可总是选择穿防护服，但是考虑到防护服的价格和穿戴方便程度，这是不合理的。为了选择出最实用的防护措施，我们在风险值0-1上确定了最佳防护措施区间，根据最安全路径上的最大风险值与所有坐标点的最大风险值的比值的所在区间与最安全路径上的最大风险值，查找相应最佳防护措施。

## 符号说明

C1、C2、C3分别表示是否阳性、类型、防护措施三个影响因子。

A、A\_、、w、w\_、CI、RI、CR分别表示一般成对比较阵（非餐厅成对比较阵）、餐厅成对比较阵、最大特征根、非餐厅特征向量（权向量）、餐厅特征向量（权向量）、一致性指标、随机一致性指标、一致性比率。

filed、Inf、G分别表示地图上的风险值矩阵、无限大数值、风险值矩阵转换的邻接矩阵。

id、x、y分别表示结点、结点的x坐标、结点的y坐标。

V、VV分别表示最安全路径上的最大风险值、所有路径上的最大风险值。

# 建模过程

## 确定影响因子权重向量

### 得到成对比较阵

|  |  |  |  |
| --- | --- | --- | --- |
|  | C1 | C2 | C3 |
| C1 | 1 | 5 | 2 |
| C2 | 1/5 | 1 | 1/3 |
| C3 | 1/2 | 3 | 1 |

图表 2一般成对比较阵

特别，考虑到餐厅吃饭需要摘下口罩等，所以餐厅的防护措施影响较小；

|  |  |  |  |
| --- | --- | --- | --- |
|  | C1 | C2 | C3 |
| C1 | 1 | 4 | 5 |
| C2 | 1/4 | 1 | 3 |
| C3 | 1/5 | 1/3 | 1 |

图表 3餐厅成对比较阵

### 计算权重向量并进行一致性检验

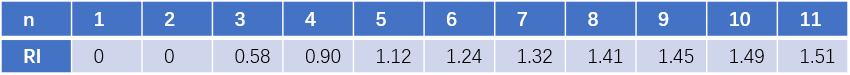


计算得：，，



代入公式(1)得





图表 4一致性指标RI表

根据图表1可得：，代入公式(2)得：

由于CR<0.1，所以A通过一致性检验。

同理可得：A\_也通过一致性检验, 。

### 计算风险值

假设不同坐标种类的实时人流量范围，如下：

普通坐标点：1-10人

公交站：5-20人

菜市场：30-50人

便利店：1-20人

医院：40-60人

地铁站：50-70人

餐厅：10-30人

对于每个个体，拥有三个属性即影响因子，根据权重和属性值计算个体的风险值，非餐厅区域使用,餐厅区域使用。再将坐标点的所有个人风险值相加得到坐标风险值，最后将所有坐标风险值归一化得到风险值矩阵filed。

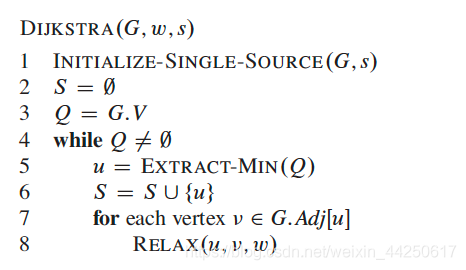
### 获得邻接矩阵

因为Dijkstra算法是用于求图的最短路径算法，所以我们需要将风险值矩阵field转换为邻接矩阵G，转换公式如下。

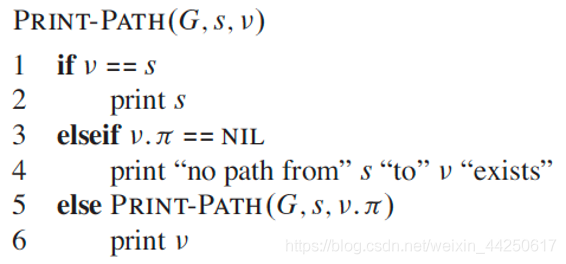


其中id1、id2、x2、y2分别表示结点1、结点2、结点2的x坐标、结点2的y坐标。

### 使用Dijkstra算法计算最安全路径



图表 5Dijkstra伪代码



图表 6打印路径算法伪代码

根据上述算法可得：从A点到B点的最安全路径

### 选择最实用的防护措施

我们在风险值0-1上确定了最佳防护措施区间，根据最安全路径上的最大风险值与所有坐标点的最大风险值的比值的所在区间与最安全路径上的最大风险值，查找相应最佳防护措施。

当最安全路径上最大风险值V和所有路径上最大风险值VV均为0时，即可不带口罩；

当V/VV>=0.8时，若V>0.7，则选择穿防护服，否则选择带N95口罩；

当V/VV<0.8时，选择普通口罩即可。

# 相关代码

## Init.m

function [ field ] = Init(n,w,w\_,locationPos,barrierPos)

field=ones(n,n);

humanRange=[1 10;5 20;30 50;1 20;40 60;50 70;10 30];

values = ones(1,size(locationPos,2));

for i=2:size(humanRange,1)

number=randi([humanRange(i,1),humanRange(i,2)]);

for j=1:number

positive=1;

if rand(1,1)<0.5

positive=0.1;

end

individuality=[positive randi([1,4])/(1+2+3+4) randi([1,5])/(1+2+3+4+5)];

if i==7

values(i-1)=values(i-1) + individuality\*w\_;

else

values(i-1)=values(i-1) + individuality\*w;

end

end

end

for i=1:(n\*n)

if isempty(find(locationPos==i))==1

number=randi(humanRange(1,1),humanRange(1,2));

for j=1:number

positive=1;

if rand(1,1)<0.5

positive=0.1;

end

individuality=[positive randi([1,4]) randi([1,5])];

field(i)=field(i) + individuality\*w;

end

else

field(i)=values(find(locationPos==i));

end

end

field(barrierPos) = Inf;

end

## cal.m

function [ lambad,w,n ] = cal( A )

%计算特征向量和最大特征值

n=size(A,1);

%列向量归一化%

B=repmat(sum(A),n,1);

B=A./B;

%横向求和%

C=sum(B')';

%归一化%

w=C./sum(C);

%最大特征值%

lambad = mean((A\*w)./w);

end

## main.m

clc,clear,close all;

%计算权重

clc

A=[1 5 2;1/5 1 1/3;1/2 3 1];

%餐厅的成对比较表

A\_=[1 4 5;

1/4 1 3;

1/5 1/3 1;];

[lambad,w,n] = cal(A);

[lambad\_,w\_,n\_] = cal(A\_);

re=[lambad,n;lambad\_,n\_;];

isOK=1;

%一致性检验%

for i=1:size(re,1)

RI=0.58;

if ((re(i,1)-re(i,2))/(re(i,2)-1))/RI>=0.1

sprintf('%d 不通过一致性检验',i)

isOK=0;

else

sprintf('%d 通过一致性检验',i)

end

end

if isOK==0

return

end

%创建地图

n = 5;

startposind=1;

goalposind=25;

barrierPos=[7 8 10 17 19];

locationPos=[4 12 14 16 18 22];

locationName=['公交站' '菜市场' '便利店' '医院 ' '地铁站' '餐厅 '];

field = Init(n,w,w\_,locationPos,barrierPos);

field(startposind) = 0; field(goalposind) = 0; %把矩阵中起始点和终止点处的值设为0

all = sum(field(field<Inf));

field=field/all;

%进行地图的绘制

if isempty(gcbf)

figure('Position',[460 65 700 700]);

axes('position', [0 0 1 1]);%设置坐标轴的位置

else

gcf; cla;

end

pcolor(1:n+1,1:n+1,[field field(:,end); field(end,:) field(end,end)]);

cmap = flipud(colormap('summer'));

cmap(1,:) = zeros(3,1); cmap(end,:) = ones(3,1);

colormap(flipud(cmap)); %进行颜色的倒转

hold on;

[goalposy,goalposx] = ind2sub([n,n],goalposind);

[startposy,startposx] = ind2sub([n,n],startposind);

text(goalposx+0.5,goalposy+0.5,'B点','HorizontalAlignment','center');

text(startposx+0.5,startposy+0.5,'A点','HorizontalAlignment','center');

for i=1:size(locationPos,2)

[y,x] = ind2sub([n,n],locationPos(i));

text(x+0.5,y+0.5,locationName((i-1)\*3+1:(i-1)\*3+3),'HorizontalAlignment','center');

end

%转换成邻接矩阵

p=n\*n;

G = ones(p,p)\*Inf;

for i=1:n

for j=1:n

if field(i,j)<Inf

ind = sub2ind([n n],i,j);

if i>1 && field(i-1,j)<Inf

ind\_=sub2ind([n n],i-1,j);

G(ind,ind\_)=field(i-1,j);

end

if j>1 && field(i,j-1)<Inf

ind\_=sub2ind([n n],i,j-1);

G(ind,ind\_)=field(i,j-1);

end

if i<n && field(i+1,j)<Inf

ind\_=sub2ind([n n],i+1,j);

G(ind,ind\_)=field(i+1,j);

end

if j<n && field(i,j+1)<Inf

ind\_=sub2ind([n n],i,j+1);

G(ind,ind\_)=field(i,j+1);

end

end

end

end

for i=1:p

G(i,i)=0;

end

% 初始化操作

S(1) = goalposind;

U = 1:p;

U(goalposind) = [];

Value = zeros(2,p);

Value(1,:) = 1:p;

Value(2,1:p) = G(goalposind,1:p);

new\_Distance = Value;

D = Value;

D(:,goalposind) = [];

path = zeros(2,p);

path(1,:) = 1:p;

path(2,Value(2,:)~=inf) = goalposind;

% 寻找最安全路径

while ~isempty(U)

index = find(D(2,:)==min(D(2,:)),1);

k = D(1,index);

S = [S,k];

U(U==k) = [];

new\_Distance(2,:) = G(k,1:p)+Value(2,k);

D = min(Value,new\_Distance);

%更新路径

path(2,D(2,:)~=Value(2,:)) = k;

Value = D;

D(:,S) = [];

end

%总共风险值

All = Value(2,startposind);

% 绘制安全路径

pxs=[1];

pys=[1];

v=[0];

while startposind ~= goalposind

next = path(2,startposind);

startposind = next;

[py,px]=ind2sub([n n],next);

pxs=[pxs;px];

pys=[pys;py];

v=[v;field(py,px)];

end

plot(pxs+0.5,pys+0.5,'Color','red','LineWidth',4);

drawnow;

drawnow;

V = max(v);

VV = max(field(field<Inf));

result='';

if V==0 && VV==0

result='选择不带口罩';

else

if V/VV>0.8 && V>0.7

result='选择穿防护服';

else

if V/VV>0.8

result='选择带N95口罩';

else

result='选择带普通口罩';

end

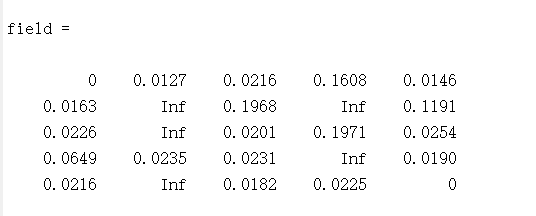
end

end

hold on;

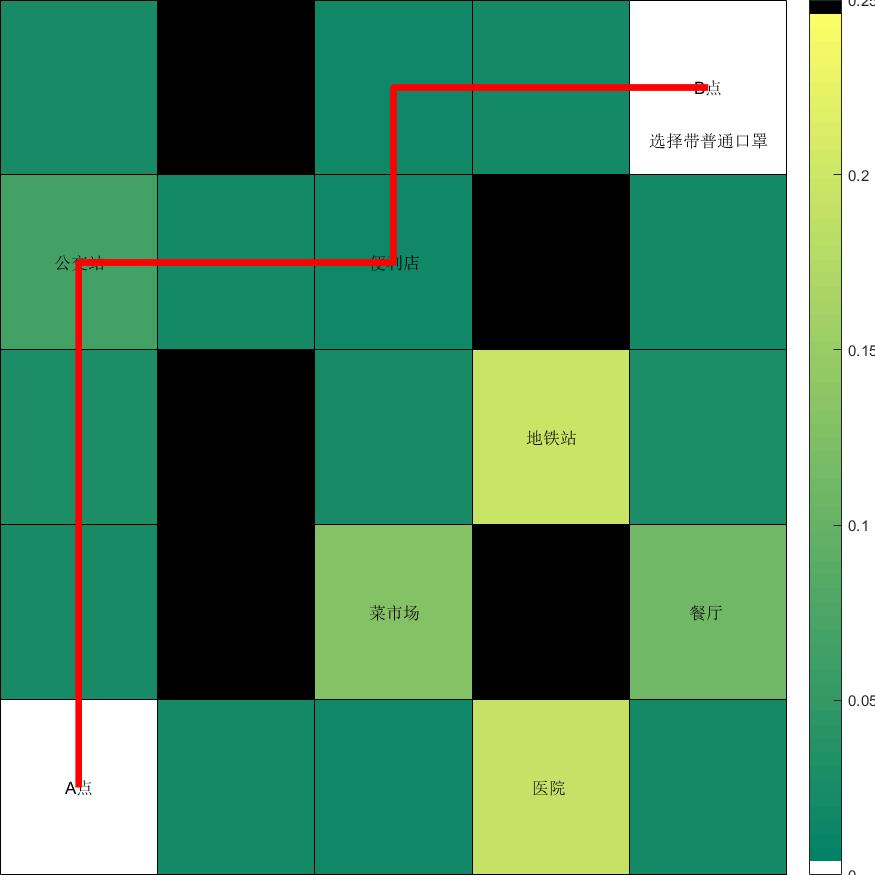
text(goalposx+0.5,goalposy+0.2,result,'HorizontalAlignment','center');

# 求解答案



图表 7风险值矩阵

注：与绘制的地图不同，field矩阵左上角为A点，右下角为B点。



图表 8答案