智能优化方法课程作业

课题名称：基于改进蚁群算法的机器人路径规划

姓名：王伍臣

学号：SC12009057

**基于改进蚁群算法（蚁群系统）的机器人路径规划**

**一、该算法要解决的问题及原理：**

机器人路径规划问题简单说就是从机器人起点到终点寻找一条无障碍的最优路径，不光要避障而且要找到最优路径，在该算法中将机器人路径图简化为一个0—1矩阵，0表示无障碍，1表示有障碍，应用改进的蚁群算法中的蚁群系统思想对其路径进行优化。

**二、蚁群算法及改进蚁群算法中的蚁群系统思想简介**

蚁群算法(Ant Colony Algorithm, ACA) 是Dorigo M等人于1991年提出的。经观察发现, 蚂蚁个体之间是通过一种称之为信息素的物质进行信息传递的。在运动过程中, 蚂蚁能够在它所经过的路径上留下该种信息素, 而且能够感知信息素的浓度, 并以此指导自己的运动方向 。蚁群的集体行为表现出一种信息正反馈现象: 某一路径上走过的蚂蚁越多, 则后来者选择该路径的概率就越大。蚂蚁个体之间就是通过这种信息的交流达到搜索食物的目的。它充分利用了生物蚁群通过个体间简单的信息传递，搜索从蚁巢至食物间最短路径的集体寻优特征，以及该过程与旅行商问题求解之间的相似性。同时，该算法还被用于求解二次指派问题以及多维背包问题等，显示了其适用于组合优化问题求解的优越特征。

Dorigo M等在提出基本蚁群算法后不久，又提出了一种新的蚁群算法，并将其命名为Ant-Q system。为了避免出现停滞现象，Dorigo M等在该算法中采用了确定性选择和随机性选择相结合的选择策略，并在搜索过程中动态调整状态转移概率。

**三、应用改进蚁群算法进行机器人路径规划的主要步骤及原理描述如下**：

（1） 输入由0和1组成的矩阵表示机器人需要寻找最优路径的地图（其中0表示此处可以通过的，1表示此处为障碍物），初始化各参数，输入循环次数N，每轮蚂蚁数M，令各路径初始信息量为相等的常数，初始时刻信息量增量为0。输入信息素、期望因子等其他各参量。

（2） 编写子函数生成（1）中0-1地图的加权邻接矩阵，此矩阵存储（1）中地图上两任意方格的连通情况，在这里两方格连通表示蚂蚁可以先后通过这两个方格即为两相邻或者对角的无障碍方格，对于连通的两方格在邻接矩阵中以元素值存储它们之间的距离，用元素的行、列下标分别存储两方格在（1）中地图中的位置，该算法中邻接矩阵与禁忌表一起决定了所有蚂蚁下一步能去的方格。

（3）开始N轮蚂蚁的搜索，设定循环结束条件，初始化禁忌表。

（4）状态转移。状态转移包括两小步：首先根据（2）中求得的邻接矩阵以及禁忌表求出蚂蚁下一步能前往的方格；然后根据蚁群系统状态转移规则求蚂蚁下一步所前往的方格，并且更新禁忌表。

蚁群系统的状态转移规则如下：为避免停滞现象的出现，蚁群系统采用了确定性选择和随机性选择相结合的选择策略，并在搜索过程中动态调整状态转移概率。其规则可以用下面两个式子描述，对位于地点i的蚂蚁随机数q小于等于算法参数q0按式（2.1）选择下一个地点j，当q大于q0时按照式（2.2）的概率，利用轮盘赌法确定蚂蚁下一个要去的地点：

（3.1）

 （3.2）

其中arg max f（k） 的结果是使f（k）达到最大值的k值，为路径上的信息素浓度。表示有方格i转移到方格j的期望程度，表示信息启发式因子，表示期望启发式因子， 表示此时蚂蚁k下一步允许 选择的方格。

（5） 更新蚂蚁爬行路径，以及路径长度。

(6) 重复（4）（5）过程，直到蚂蚁到达终点或者陷入死胡同。

（7） 重复（4）（5）（6），直到所有m只蚂蚁搜索结束。

（8） 更新信息素矩阵，在这里不再更新不再用于所有蚂蚁，而是只对每一次循环中找到本轮最短路径的蚂蚁使用。更新规则如下式：

 （3.3）

 （3.4）

其中为信息素挥发系数。为信息量增加强度。表示第k只蚂蚁在本次循环中所走路径的总长度。

（9）重复（4）~（8），直到满足（3）设定的循环结束条件时循环结束，输出结果。

**四、运行结果（图、表等）**

将如下程序段输入到matlab命令窗口：

L=[0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;

0 0 0 1 1 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 ;

0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;

0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0;

0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0;

0 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0;

0 1 1 1 0 0 1 1 1 0 0 0 1 1 1 1 0 0 0 0 ;

0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 ;

0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 ;

0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0;

0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0;

0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0;

0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 ;

0 0 1 1 0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 1 ;

0 0 1 1 0 0 1 1 1 0 0 0 0 0 0 1 1 1 1 0;

0 0 0 0 0 0 1 1 1 0 1 1 0 0 0 0 0 1 1 0;

0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1 0;

0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0;

0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;];

N=100;

M=60;

a=1;

b=6;

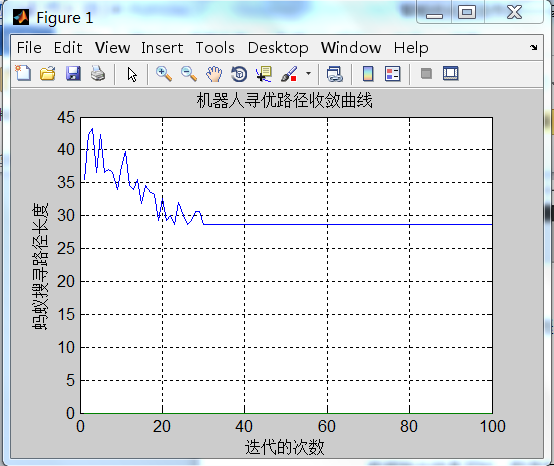
p=0.3 ;

Q=1;

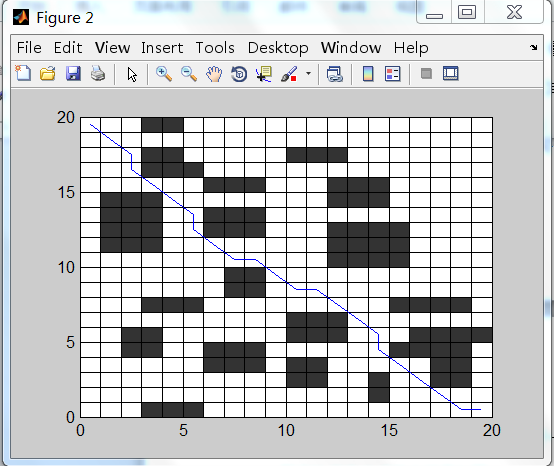
q1=0.1;

lujingyouhua\_1(L,N,M,a,b,p,Q,q1)

执行一次后程序输出如下结果：

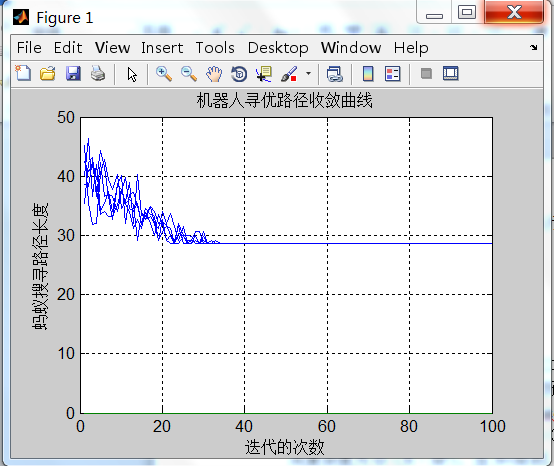


图一 程序执行一次输出的收敛曲线图

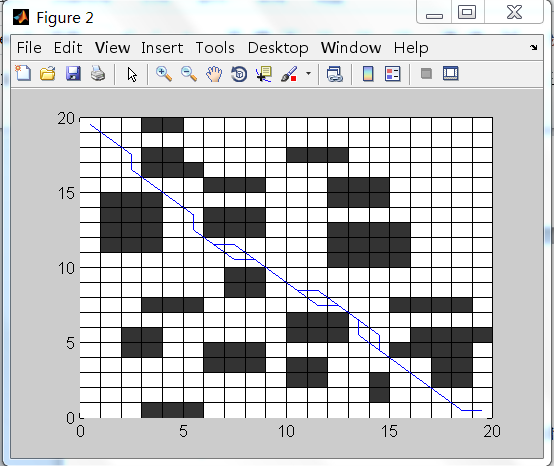


图二 程序执行一次得到的本次最优路径图

程序执行N次后输出结果如下：



图三 程序执行N次输出的收敛曲线图



图四 程序执行N次输出的各次最优路径图

**五、实验结果分析**

根据输出结果可知，程序在经过三十次左右循环后就逐渐趋于收敛，一方面原因是所输入的0-1矩阵规模较小，使得蚂蚁爬行路线较短，觅食时间较短；另一方面也反映出该算法收敛性能较好，而且搜索到的最优路径正确性较好。程序运行时间适中，在本次实验中100代蚂蚁搜索平均运行时间为25s左右。

该程序计算邻接矩阵的route\_distance子函数时间复杂度为O（M4），M是蚂蚁种群规模，主函数循环体时间复杂度为O（N·M3），N派出的蚂蚁轮数，当N>M时，程序总的时间复杂度为O（N·M3），当N<M时，程序总的时间复杂度为O（M4）。

由于matlab中特殊符号难辨识，用一些简单字母表示各参量，程序中的一些参量：a表示α，即信息启发式因子；b表示β，即期望启发式因子；y1表示η，即启发式信息矩阵；p表示ρ，即信息素挥发系数。Q 为常量，表示蚂蚁循环一周或一个过程在经过的路径上所释放的信息素总量。

在该算法中参量α、参量β，参量Q、参量ρ对程序的收敛速度影响较大，适当选择大一点的α、Q、ρ同时选择适当的β值，可提高收敛速度。同时在判断蚂蚁转移方式时的p1参量要适当取得小一点，以提高算法优化性能。经过实验分析，α取默认值1，Q也取默认值1时，参量β取值为6左右并且p1取0.1~0.2时算法性能较好。

**六、MATLAB源程序**

function lujingyouhua\_1(L,N,M,a,b,p,Q, q1)

% L 地形图为01矩阵，如果为0表示机器人能通过，如果为1表示障碍物，机器人不能通过

% N 迭代的次数

% M 蚁群中蚂蚁的个数

% a 表示信息启发式因子，反映了蚁群在运动过程中所残留的信息量的相对重要程度

% b 表示期望启发式因子，反映了期望值的相对重要程度

% p 表示信息素挥发系数

% Q 常量，表示蚂蚁循环一周或一个过程在经过的路径上所释放的信息素总量

%q1是（0,1）间的一个算法参量，在状态转移时作为一个判断基准

M1=size(L,1);

Set=1; % Set 为地图中路径起始点位置

En=M1\*M1; % En为地图中终止点位置

H= route\_distance (L); %通过调用子函数route\_distance.m，产生地图中方格的邻接矩阵H用此矩

%阵存储蚂蚁可以先后通过的两个方格的位置及距离，这两个方格都是无障碍方格并且处于相

%邻或对角位置。H矩阵是蚂蚁机器人移动的根据。

K=size(H,1); %K表示问题的规模，也表示L地图上的方格总数

t1=ones(K,K); % t1 初始信息素矩阵，

t1=9.\*t1; %将每条路径上的信息素数设为常数9；

d\_t1=zeros(K,K);% d\_t1 信息素增量矩阵

aa=1; %aa方格的边长

Enx=M1-0.5; %Enx 终止点的横坐标

Eny=0.5; %终止点纵坐标

y1=zeros(K); %启发式信息矩阵，表示蚂蚁在方格间转移的期望程度

%%下面构造启发式信息矩阵,每个节点的启发式信息素数量取为至目标点（在此为终止点）的

%直线距离的倒数

for i=1:K

ix=aa\*(mod(i,M1)-0.5);

if ix==-0.5 %当i等于M1的整数倍时，ix=-0.5，即每一行的最后一个方格

ix=M1-0.5;

end

iy=aa\*(M1+0.5-ceil(i/M1));

if i~=En

y1(i)=1/((ix-Enx)^2+(iy-Eny)^2)^0.5; %构造启发式信息矩阵

else

y1(i)=100;

end

end

Rs=cell(N,M); %用细胞结构存储每一代的每一只蚂蚁的爬行路线

Pr=zeros(N,M); %用矩阵存储每一代的每一只蚂蚁的爬行路线长度

kn=inf; k2=0; n2=0; %初始化过程参数

%N轮蚂蚁觅食开始，每轮派出的蚂蚁数为M

for k=1:N

for m=1:M

%% 第一步：初始化参数

S1=Set; %当前节点初始化为起始点

rt\_1=Set; %rt\_1存储当前蚂蚁的爬行路线

rd=0; %rd 爬行路线长度

tabuk=ones(K); %初始化禁忌表

tabuk(Set)=0; %排除起始点

H1=H; %邻接矩阵初始化

%% 第二步：搜寻蚂蚁下一步可以前往的方格

H2=H1(S1,:); % H2为H1矩阵的第一行，表示L矩阵第S1个元素（按行排列）与其他元素的

%连通关系

H3=find(H2); %返回H2中不为零元素的序号，按顺序排列

for j=1:length(H3)

if tabuk(H3(j))==0

H2(H3(j))=0;

end

end

L1=find(H2);

Len\_L1=length(L1);%可选节点的个数

while S1~=En&&Len\_L1>=1 % 觅食停止条件：蚂蚁到终点或者陷入死胡同

%% 第三步：确定蚂蚁下一步将要转移的位置

q=rand;

if q<=q1

t12=zeros(1,K);

for i=1:Len\_L1

t12(i)= (t1(S1,L1(i))^a)\*((y1(L1(i)))^b);

end

m\_t12=max(t12);

select=find(t12>=m\_t12);

to\_visit=L1(select(1));

else

P1=zeros(Len\_L1); %初始化状态转移概率矩阵

for i=1:Len\_L1

P1(i)=(t1(S1,L1(i))^a)\*((y1(L1(i)))^b);

end

sump1=sum(P1);

P1=P1/sump1;%计算状态转移概率

%%利用轮盘赌方式选择蚂蚁下一步要去的位置

P2(1)=P1(1);

for i=2:Len\_L1

P2(i)=P2(i-1)+P1(i);

end

Select1=find(P2>=rand);

to\_visit=L1 (Select1(1));

end

%% 第四步：更新状态并做有关记录

rt\_1=[rt\_1,to\_visit];%路径增加

rd=rd+H1(S1,to\_visit);%路径长度增加

S1=to\_visit;%蚂蚁移到下一个节点位置

for k12=1:K

if tabuk(k12)==0

H1(S1,k12)=0;

H1(k12,S1)=0;

end

end

tabuk(S1)=0;%已访问过的节点从禁忌表中删除

H2=H1(S1,:);

H3=find(H2);

for j=1:length(H3)

if tabuk(H3(j))==0

H2(j)=0;

end

end

L1=find(H2);

Len\_L1=length(L1);%可选节点的个数

end

%% 第五步：记录蚂蚁的爬行路线并且记录搜索到本轮最短路径的蚂蚁及路径长度

Rs{k,m}=rt\_1; %存储第k轮第m个蚂蚁的路线

if rt\_1(end)==En

Pr(k,m)=rd;

if rd<kn

k2=k;

n2=m;

kn=rd;

end

else

Pr(k,m)=0;

end

end

%% 第六步：执行全局更新

d\_t1=zeros(K,K);

for m=1:M

if Pr(k,m)

Rt=Rs{k,m};

TS=length(Rt)-1;

PL\_d=Pr(k,m);

for s=1:TS

x=Rt(s);

y=Rt(s+1);

d\_t1(x,y)=d\_t1(x,y)+Q/PL\_d;

d\_t1(y,x)=d\_t1(y,x)+Q/PL\_d;

end

end

end

t1=(1-p).\*t1+d\_t1;%信息素挥发一部分，新增加一部分

end

%%绘图

%绘制机器人寻优路径收敛曲线

minrt=zeros(N);

rt11=zeros(1,M);

for i=1:N

rt11=Pr(i,:);

Nzero=find(rt11);

rt12=rt11(Nzero);

minrt(i)=min(rt12);

end

figure(1)

plot(minrt);

hold on

grid on

title('机器人寻优路径收敛曲线');

xlabel('迭代的次数');

ylabel('蚂蚁搜寻路径长度');

%绘蚂蚁爬行图

figure(2)

axis([0,M1,0,M1])

for i=1:M1

for j=1:M1

if L(i,j)==1

x1=j-1;y1=M1-i; x2=j;y2=M1-i;

x3=j;y3=M1-i+1; x4=j-1;y4=M1-i+1;

fill([x1,x2,x3,x4],[y1,y2,y3,y4],[0.2,0.2,0.2]);

hold on

else

x1=j-1;y1=M1-i; x2=j;y2=M1-i;

x3=j;y3=M1-i+1; x4=j-1;y4=M1-i+1;

fill([x1,x2,x3,x4],[y1,y2,y3,y4],[1,1,1]);

hold on

end

end

end

hold on

Rt=Rs{k2,n2};

LEN\_Rt=length(Rt);

Rx=Rt;

Ry=Rt;

for ik=1:LEN\_Rt

Rx(ik)=aa\*(mod(Rt(ik),M1)-0.5);

if Rx(ik)==-0.5

Rx(ik)=M1-0.5;

end

Ry(ik)=aa\*(M1+0.5-ceil(Rt(ik)/M1));

end

plot(Rx,Ry)

end

%% route\_distance.m子函数

%下面的子函数产生表示蚂蚁可先后通过的两方格及其距离的矩阵H，在H矩阵中元素行下标

%代表任意先后两节点第一个节点在L中位置，列下标代表第二个节点在L中的位置，若这两

%个节点为蚂蚁可先后通过则元素值为这两个无障碍节点之间的距离，否则元素值0。

%，蚂蚁可先后通过的两方格为L地形图中两个无障碍方格并且这两个方格处于相邻或对

%角位置。通过对矩阵L两次嵌套的搜索遍历，寻找两个相邻或者对角的无障碍方格的所有可

%能并计算蚂蚁通过它们的距离，在这里若两无障碍方格相邻，则距离设为1，相应的若两无障

%碍方格对角，则距离为2^0.5≈1.4142。

function H=route\_distance(L)

h=size(L,1);

H=zeros(h\*h,h\*h);

%用i表示H矩阵的行下标，j表示H矩阵的列下标，这是第一个嵌套循环的外部循环

%用m表示L矩阵的行下标，n表示L矩阵的列下标，这是嵌套循环的内循环

for i=1:h

for j=1:h

if L(i,j)==0

for m=1:h

for n=1:h

if L(m,n)==0

im=abs(i-m);jn=abs(j-n); %分别计算在L矩阵搜寻到的两个零元素即L地图中的两个无

%障碍方格在L地图中x方向距离和y方向距离

if im+jn==1||(im==1&&jn==1)

H((i-1)\*h+j,(m-1)\*h+n)=(im^2+jn^2)^0.5;

%用矩阵H中部分元素存储两相邻或对角的无障碍方格在L地图中位置和路径距

%离，其他元素值为0

end

end

end

end

end

end

end

end