# 蚁群算法以及 matlab 实现

蚁群算法是一种仿生算法，作为通用型随机优化方法，它吸收了蚂蚁的行为特性，通过其内在的搜索机制，在一系列困难的组合优化问题中取得了成效。

## 蚁群算法原理

1、 蚁群算法基本思想

蚁群算法的基本原理来自于自然界蚂蚁觅食的最短路径原理。

2、 蚁群算法数学模型

3、 蚁群算法流程

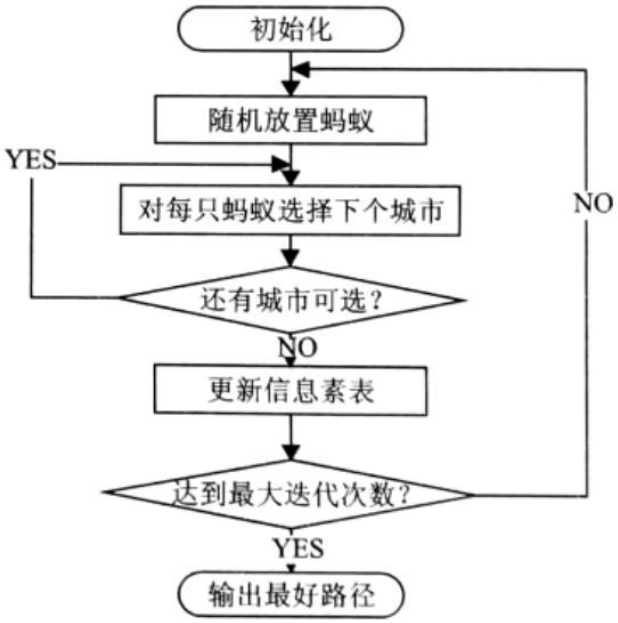
a) 对相关参数进行初始化，包括蚁群规模、信息素因子、启发函数因子、信息素挥发因子、信息素常数、最大迭代次数等等，以及将数据读入程序，并对数据进行基本的处理。

b) 随即将蚂蚁放在不同的出发点，对每个蚂蚁计算其下一个访问城市，直至所有蚂蚁访问完所有城市

c) 计算各个蚂蚁经过的路径长度 Lk，记录当前迭代次数中的最优解，同时对各个城市连接路径上的信息素浓度进行更新。

d) 判断是否达到最大迭代次数，若否，则返回步骤 2，若是，则终止程序

e) 输出程序结果，并根据需要输出程序寻优过程中的相关指标，如运行时间、收敛迭代次数等。



# 解决 TSP 问题 Matlab 代码

```
clear all;
clc;


t0=clock;


%导入数据
A1=xlsread('E:\顾子涵专用文件夹\学习\matlab 学习\matlab 与数学模型\TSP 数
据.xlsx',1,'A2:C19');
A2=xlsread('E:\顾子涵专用文件夹\学习\matlab 学习\matlab 与数学模型\TSP 数
据.xlsx',1,'D2:F19');
A3=xlsread('E:\顾子涵专用文件夹\学习\matlab 学习\matlab 与数学模型\TSP 数
据.xlsx',1,'G2:I17');
citys=[A1;A2;A3];
amount=size(citys,1);


citys(:,1)=[];
x1=citys(:,1)*ones(1,amount);
x2=x1';
y1=citys(:,2)*ones(1,amount);
y2=y1';
dist_matrix=sqrt((x1-x2).^2+(y1-y2).^2);


%修正矩阵对角的值
for i=1:amount
    dist_matrix(i,i)=1e-4;
end


m=80; %蚂蚁数量
Alpha=1;   %信息素重要程度因子
beta=5;    %启发函数重要程度因子
vol=0.2;   %信息素挥发因子
Q=10;      %常系数
Heu_f=1./dist_matrix;%启发函数
Tau=ones(amount,amount);%信息素矩阵
Table=zeros(m,amount);%路径记录表
iter=1;%迭代次数初始值
iter_max=100;%迭代次数最大值
Route_best=zeros(iter_max,amount);%各代最佳路径
```

```
Length_best=zeros(iter_max,1);%各代最佳路径长度
Length_ave=zeros(iter_max,1);%各代平均路径长度
Limit_iter=0;     %收敛时程序迭代次数


%%寻找最佳路径
while iter<=iter_max
    %随机产生各个蚂蚁的初始城市
    start=zeros(m,1);
    for i=1:m
        temp=randperm(amount);
        start(i)=temp(1);
    end
    Table(:,1)=start;
    %构建解空间
    citys_index=1:amount;
    for i=1:m
        for j=2:amount
            tabu=Table(i,1:(j-1));%第 i 行第 1 至 j-1 号元素
            %检查哪个城市访问过，哪个城市未访问过
            allow_index=~ismember(citys_index,tabu);
            %直接取得未访问过的城市的编号
            allow=citys_index(allow_index);
            P=allow;
            %计算城市间转移概率
            for k=1:length(allow)

P(k)=Tau(tabu(end),allow(k))^Alpha*Heu_f(tabu(end),allow(k))^beta;
            end
            P=P/sum(P);
            %轮盘赌法选择下一个访问城市
            Pc=cumsum(P);
            target_index=find(Pc>=rand);
            target=allow(target_index(1));
            Table(i,j)=target;
        end
    end
    %计算各个蚂蚁的路径距离
    Length=zeros(m,1);
    for i=1:m
        Route=Table(i,:);
        for j=1:(amount-1)
            Length(i)=Length(i)+dist_matrix(Route(j),Route(j+1));
        end
        Length(i)=Length(i)+dist_matrix(Route(amount),Route(1));
```

```
    end
    %计算最短路径距离及平均距离
    if iter==1
        [min_Length,min_index]=min(Length);
        Length_best(iter)=min_Length;
        Length_ave(iter)=mean(Length);
        Route_best(iter,:)=Table(min_index,:);
        Limit_iter=1;
    else
        [min_Length,min_index]=min(Length);
        Length_best(iter)=min(Length_best(iter-1),min_Length);
        Length_ave(iter)=mean(Length);
        if Length_best(iter)==min_Length
            Route_best(iter,:)=Table(min_index,:);
            Limit_iter=iter;
        else
            Route_best(iter,:)=Route_best((iter-1),:);
        end
    end
    %更新信息素
    Delta_Tau=zeros(amount,amount);
    %逐个蚂蚁计算
    for i=1:m
        %逐个城市计算
        for j=1:(amount-1)

Delta_Tau(Table(i,j),Table(i,j+1))=Delta_Tau(Table(i,j),Table(i,j+1))+Q
/Length(i);
        end

Delta_Tau(Table(i,amount),Table(i,1))=Delta_Tau(Table(i,amount),Table(i
,1))+Q/Length(i);
    end
    Tau=(1-vol)*Tau+Delta_Tau;
    %迭代次数加 1，清空路径记录表
    iter=iter+1;
    Table=zeros(m,amount);
end


%%结果显示
[Shortest_Length,index]=min(Length_best);
Shortest_Route=Route_best(index,:);
Time_Cost=etime(clock,t0);
disp(['最短距离：',num2str(Shortest_Length)]);
```

```
disp(['最短路径：',num2str([Shortest_Route Shortest_Route(1)])]);
disp(['收敛迭代次数：',num2str(Limit_iter)]);
disp(['程序执行时间：',num2str(Time_Cost) '秒']);


%%绘图
figure(1);
plot([citys(Shortest_Route,1);citys(Shortest_Route(1),1)],...
    [citys(Shortest_Route,2);citys(Shortest_Route(1),2)],'o-');
grid on;
for i=1:size(citys,1)
    text(citys(i,1),citys(i,2),['' num2str(i)]);
end
text(citys(Shortest_Route(1),1),citys(Shortest_Route(1),2),'       起点
');
text(citys(Shortest_Route(end),1),citys(Shortest_Route(end),2),'
终点');
xlabel('城市横坐标');
ylabel('城市纵坐标');
title(['ACA 最优化路径(最短距离:' num2str(Shortest_Length) ')']);
figure(2);
plot(1:iter_max,Length_best,'b');
legend('最短距离');
xlabel('迭代次数');
ylabel('距离');
title('算法收敛轨迹');
```

附加函数学习：

1、 cumsum 函数，计算矩阵累计和　A=[1,2,4,5];b=cumsum(A);则 b=[1 3 7 12]
　　若 b 为矩阵，则按照列求和，若如此写：cumsum(b,2)则按行求和

2、 randperm 函数，随机乱序函数 randperm(10)　则将 1~10 打乱输出

运行结果:

```
最短距离：7681.4537
最短路径：8   41   19   45   32   49    1   22   31
收敛迭代次数：88
程序执行时间：98.259秒
```