蚁群算法 Ant Colony Algorithm

蚁群算法是在 20 世纪 90 年代由澳大利亚学者 Marco Dorigo 等人通过观察蚁群觅食的过程，发现众多蚂蚁在寻找食物的过程中，总能找到一条从蚂蚁巢穴到食物源之间的最短路径。随后他们在蚂蚁巢穴到食物源之间设置了一个障碍，一段时间以后发现蚂蚁又重新走出了一条到食物源最短的路径。通过对这种现象的不断研究，最后提出了蚁群算法。蚁群算法在解决旅行商问题（即 TSP 问题）时，取得了比较理想的结果。

1.转移概率：（核心公式）

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[n_{ij}(t)]^{\beta}}{\sum\limits_{s\in J_k(i)}[\tau_{ij}(t)]^{\alpha}[n_{ij}(t)]^{\beta}}, & j\in J_k(i)\\ 0, & j\notin J_k(i)\end{cases}$$

$P_{ij}^{k}(t)$：第t代蚂蚁中第k只蚂蚁选择闯关东呢，还是走西口的概率，即蚂蚁k选择从i~j的概率

α：信息素的重要程度

β：启发因子的相对重要程度

n_{ij}：启发因子

$J_k(i)$：蚂蚁k当期可以选择的城市（注：每个城市只能走一次）

式中的:$n_{ij}=\dfrac{1}{d_{ij}}$

2.信息素更新公式

当所有的蚂蚁完成一次迭代（即每一个蚂蚁都爬遍了所有的城市节点），那么所经路径上的信息素必定会发生改变，因此需及时更新信息素，方便蚂蚁的子子孙孙根据爷爷们留下的信息素去寻找食物，传宗接代！

$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}$　$\tau_{ij}(t)$表示第t时刻(第t代蚂蚁)i~j路径上的信息素

$$\begin{cases}\Delta\tau_{ij}=\sum\limits_{i=1}^{m}\Delta\tau_{ij}^{k} & m\text{只蚂蚁对在}i\text{~}j\text{这条路径上留下的信息素总和}\\ \Delta\tau_{ij}^{k}=\dfrac{Q}{L_k} & \text{第}k\text{只蚂蚁在}i\text{~}j\text{这条路径上留下的信息素(}Q\text{表示一只蚂蚁一生所拥有的信息素)}\end{cases}$$

k：蚂蚁编号

m：蚂蚁数量

ρ：信息素稀释程度(信息素随着时间减少的比率)

参数设置：（https://blog.csdn.net/zuochao_2013/article/details/71872950）

也可以参考文献[1]:

α: $[1.0,2.0]$　（一般取1）

β: $[4.0,6.0]$

Q: $[10,700]$　（一般取100）

ρ: $[0.5,0.8]$

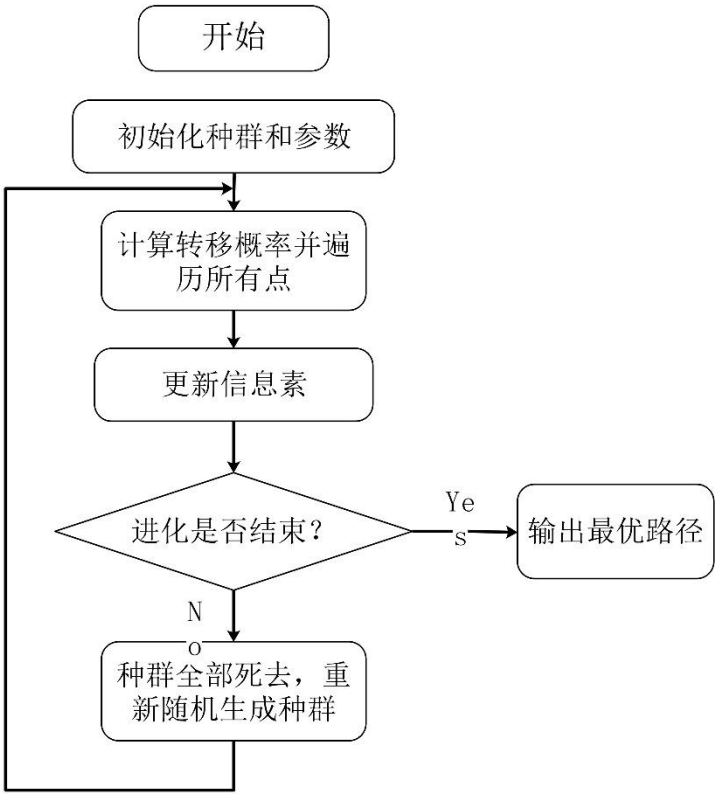
算法步骤：

（1）初始化$\tau_{ij}(t)=C$(在第一代蚂蚁出生的时候，女娲已经在路径上撒了很多信息素)

（2）初始化第一代具有开辟精神的蚂蚁所在城市的位置（生死在天，出生不是由我决定嗒，随机分配，看天意）；第一个禁忌表诞生。

（3）取第一只蚂蚁，计算转移概率$P_{ij}^{k}(t)$，按轮盘赌的方式选择下一个要去的城市

（4）当第一只蚂蚁走完所有的城市，它的一生结束，计算她最后的杰作：$\Delta\tau_{ij}^{k}$
第二只蚂蚁开始工作。
（5）所有的蚂蚁迭代结束，第一代蚂蚁光荣牺牲。开始第二代蚂蚁的开辟道路。
（6）到达了要求迭代的蚂蚁代数，或者出现停滞现象（所有的蚂蚁在啃老本，路线一直不变），算法终止!!!



蚁群算法流程图

实例分析：

经典的 TSP 问题，有 5 个城市，从一点出发经过所有的城市一次，然后返回起始点，求解最短的路径！

	A	B	C	D	E
A	0	2	10	8	3
B	1	0	2	5	7
C	9	1	0	3	6
D	10	4	3	0	2
E	2	7	5	1	0

假设初始参数$m=5$（有5只蚂蚁），$\alpha=1$，$\beta=1$，$\rho=0.5$，$\tau_{ij}(0)=2$，$Q=100$

第一代第一只蚂蚁：（开始选择的城市为 A）

i	tabu$_k$	J$_k$(i)	$\tau_{ij}(t)$	$P_{ij}^k(t)$	L_k	$\Delta\tau_{ij}^k$
A	A	B C D E	2 2 2 2	0.47 0.095 0.118 0.315	11	9.1
B	A B	C D E	2 2 2	0.593 0.237 0.169		
C	A B C	D E	2 2	0.67 0.33		
D	A B C D	E	2	1.0		
E	A B C D E					

$P_{ij}^k(t)$的计算步骤：

$$P_{12}^1=\frac{2\times0.5}{2\times0.5+2\times0.1+2\times0.125+2\times1/3}=0.472$$

$$P_{13}^1\frac{2\times0.2}{2\times0.5+2\times0.1+2\times0.125+2\times1/3}=0.095$$

……

$$P_{23}^1=\frac{2\times0.5}{2\times0.5+2\times0.2+2\times1/7}=0.593$$

$$\Delta\tau_{ij}^1=\frac{Q}{L_k}=\frac{100}{11}=9.1$$

禁忌表：k=1（第一只蚂蚁）

A	B	C	D	E
1	0	0	0	0
1	1	0	0	0
1	1	1	0	0
1	1	1	1	0
1	1	1	1	1

路线: A→B→C→D→E

第一代第二只蚂蚁：（开始选择的城市为 B）

i	tabu$_k$	J$_k$(i)	$\tau_{ij}(t)$	$P_{ij}^k(t)$	L_k	$\Delta\tau_{ij}^k$
B	B	A C D E	2 2 2 2	0.54 0.27 0.11 0.02	9	11.1
A	B A	C D E	2 2 2	0.18 0.22 0.60		
E	B A E	C D	2 2	0.17 0.83		
D	B A E D	C	2	1		
C	B AE D C					

$$P_{21}^{2}=\frac{2\times1}{2\times1+2\times0.5+2\times0.2+2\times1/7}=0.54$$

禁忌表：k=2（第二只蚂蚁）

A	B	C	D	E
0	1	0	0	0
1	1	0	0	0
1	1	0	0	1
1	1	0	1	0
1	1	1	1	1

路线：B→A→E→D→C

第一代第三只蚂蚁：（开始选择的城市为 C）

i	$tabu_k$	$J_k(i)$	$\tau_{ij}(t)$	$P_{ij}^{k}(t)$	L_k	$\Delta\tau_{ij}^{k}$
C	C	A B D E	2 2 2 2	0.069 0.62 0.207 0.103	9	11.1
B	C B	A D E	2 2 2	0.745 0.149 0.106		
A	C B A	D E	2 2	0.273 0.727		
E	C B A E	D	2	1		
D	C B A E D					

禁忌表：k=3（第三只蚂蚁）

A	B	C	D	E
0	0	1	0	0
0	1	1	0	0
1	1	1	0	0
1	1	1	0	1
1	1	1	1	1

线路：C→B→A→E→D

第一代第四只蚂蚁：（开始选择的城市为 D）

i	$tabu_k$	$J_k(i)$	$\tau_{ij}(t)$	$P_{ij}^{k}(t)$	L_k	$\Delta\tau_{ij}^{k}$
D	D	A B C E	2 2 2 2	0.084 0.211 0.287 0.422	11	9.1
E	D E	A B C	2 2 2	0.593 0.169 0.237		
A	D E A	B C	2 2	0.83 0.17		
B	D E A B	C	2	1		
C	D E A B C					

禁忌表：k=4（第四只蚂蚁）

A	B	C	D	E
0	0	0	1	0

0	0	0	1	1
1	0	0	1	1
1	1	0	1	1
1	1	1	1	1

线路：D→E→A→B→C

第一代第五只蚂蚁：（开始选择的城市为 E）

i	$tabu_k$	$J_k(i)$	$\tau_{ij}(t)$	$P_{ij}^k(t)$	L_k	$\Delta\tau_{ij}^k$
E	E	A B C D	2 2 2 2	0.271 0.078 0.109 0.543	9	11.1
D	E D	A B C	2 2 2	0.146 0.366 0.488		
C	E D C	A B	2 2	0.1 0.9		
B	E D C B	A	2	1		
A	E D C B A					

禁忌表：k=5（第五只蚂蚁）

A	B	C	D	E
0	0	0	0	1
0	0	0	1	1
0	0	1	1	1
0	1	1	1	1
1	1	1	1	1

线路：E→D→C→B→A

第一代蚂蚁开拓结束，更新信息素矩阵：

	A	B	C	D	E
A	0	9.1+9.1+1=19.2	1	1	11.1+11.1+11.1+1=34.3
B	11.1+11.1+11.1=34.3	0	9.1+9.1+1=19.2	1	1
C	1	11.1+11.1+11.1+1=34.3	0	9.1+9.1+1=19.2	1
D	1	1	11.1+11.1+11.1+1=34.3	0	9.1+9.1+1=19.2
E	9.1+9.1+1=19.2	1	1	11.1+11.1+11.1+1=34.3	0

根据公式$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}$

$\tau_{ij}(1)=(1-0.5)\tau_{ij}(0)+\Delta\tau_{ij}$

$\tau_{12}(1)=(1-0.5)\times 2+\Delta\tau_{12}=1+9.1+9.1=19.2$

$\Delta\tau_{12}=\Delta\tau_{12}^{1}+\Delta\tau_{14}^{1}=9.1+9.1$

注意：这里每一个都是一个环，所以计算$\tau_{15}(1)$时应为：

$\tau_{15}(1)=(1-0.5)\times 2+\Delta\tau_{15}=1+11.1+11.1+11.1=34.3$

$\Delta\tau_{15}=\Delta\tau_{15}^{2}+\Delta\tau_{15}^{3}+\Delta\tau_{15}^{5}=11.1+11.1+11.1$

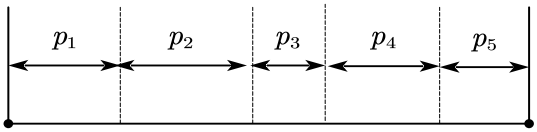
按此规律进行第二代蚂蚁的迭代：
最终结果为：
第二代第一只蚂蚁：A→E→D→C→B
第二代第二只蚂蚁：B→A→E→D→C
第二代第三只蚂蚁：C→B→A→E→D
第二代第四只蚂蚁：D→C→B→A→E
第二代第五只蚂蚁：E→D→C→B→A

我们发现五只蚂蚁走的是同一条路，所以算法收敛结束。

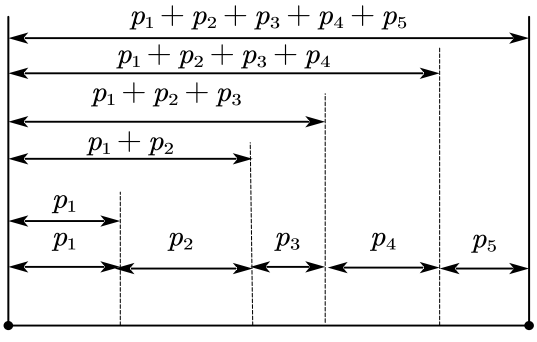
附录：轮盘赌算法：

轮盘赌法又称比例选择方法。其基本思想是：个体被选中的概率与其适应度大小成正比。在 TSP 问题中就是：选择下一个城市时，选到某个城市的概率是根据信息量和启发量确定的。概率虽然有大有小，但事件本身仍然是随机事件。大概率事件只是发生的概率大，其也有可能不发生；小概率事件虽然发生的概率小，其仍有可能发生。

假设有五个可选城市，其概率分别为 p1~p5，这五个城市是一个完备事件组，由概率的归一性，有 P1+P2+P3+P4+P5=1，我们将其依次排列到【0,1】区间内，如下图。



这时，我们随机在【0,1】区间内取一个数 a，a 落在哪个区间内，就选择此区间长度表达的概率所对应的城市。这就完成了轮盘赌法
如何判断随机数 a 落在哪个区间呢？看下图，如果 a 小于 p1 就是第 1 个区间，否则 a 小于 p1+p2 就是第 2 个区间，再否则 a 小于 p1+p2+p3 就是第 3 个区间……依次类推。



问题一：（需要根据的实际问题修改代码的地方标黄，可适当调参的地方标红）
已知 20 个城市的经纬度，计算经典 TSP 问题的最短路径：

Matlab 代码：

```
clear;
clc;

% 程序运行计时开始
t0 = clock;
%导入数据
citys=[5.326,2.558;
    4.276,3.452;
    4.819,2.624;
    3.165,2.457;
    0.915,3.921;
    4.637,6.026;
    1.524,2.261;
    3.447,2.111;
    3.548,3.665;
    2.649,2.556;
    4.399,1.194;
    4.660,2.949;
    1.479,4.440;
    5.036,0.244;
    2.830,3.140;
    1.072,3.454;
    5.845,6.203;
    0.194,1.767;
    1.660,2.395;
    2.682,6.072];%20 个城市
%---------------------------------------------------------------------
%% 计算城市间相互距离
n = size(citys,1);
D = zeros(n,n);
for i = 1:n
    for j = 1:n
        D(i,j) = sqrt(sum((citys(i,:) - citys(j,:)).^2));
    end
end
%---------------------------------------------------------------------
%% 初始化参数
m = 75;                                 % 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(volatilization)因子
Q = 10;                                     % 常系数
Heu_F = 1./D;                               % 启发函数(heuristic function)
Tau = ones(n,n);                            % 信息素矩阵
Table = zeros(m,n);                         % 路径记录表
iter = 1;                                   % 迭代次数初值
iter_max = 100;                             % 最大迭代次数
Route_best = zeros(iter_max,n);             % 各代最佳路径
Length_best = zeros(iter_max,1);            % 各代最佳路径的长度
Length_ave = zeros(iter_max,1);             % 各代路径的平均长度
Limit_iter = 0;                             % 程序收敛时迭代次数
%--------------------------------------------------------------------------
%% 迭代寻找最佳路径
while iter <= iter_max
    % 随机产生各个蚂蚁的起点城市
      start = zeros(m,1);
      for i = 1:m
          temp = randperm(n);
          start(i) = temp(1);
      end
      Table(:,1) = start;
      % 构建解空间
      citys_index = 1:n;
      % 逐个蚂蚁路径选择
      for i = 1:m
          % 逐个城市路径选择
        for j = 2:n
            has_visited = Table(i,1:(j - 1));          % 已访问的城市集合(禁忌表)
            allow_index = ~ismember(citys_index,has_visited);
            allow = citys_index(allow_index);  % 待访问的城市集合
            P = allow;
            % 计算城市间转移概率
            for k = 1:length(allow)
                P(k) = Tau(has_visited(end),allow(k))^alpha * ...
Heu_F(has_visited(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:(n - 1)
            Length(i) = Length(i) + D(Route(j),Route(j + 1));
        end
        Length(i) = Length(i) + D(Route(n),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);   % 比较当前代数与前一代的距离
        Length_ave(iter) = mean(Length);
        if Length_best(iter) == min_Length     % 当前代就是最短路径
            Route_best(iter,:) = Table(min_index,:);
            Limit_iter = iter;
        else                                   % 上一代是最短路径
            Route_best(iter,:) = Route_best((iter-1),:);
        end
    end
    % 更新信息素
    Delta_Tau = zeros(n,n);
    % 逐个蚂蚁计算
    for i = 1:m
        % 逐个城市计算
        for j = 1:(n - 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,n),Table(i,1)) = Delta_Tau(Table(i,n),Table(i,1)) + Q/Length(i);
        % 加上最后一个城市到第一个城市的回路
    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,n);
```

```
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)],...   %三点省略符为 Matlab 续行符
     [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('城市位置横坐标')
ylabel('城市位置纵坐标')
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')
```

问题二：

求函数$f(x,y)=20(x^2+y^2)^2-(1-y)^2-3(1+y)^2+0.3$的最小值，其中$x$的取值范围为$[-5,5]$，$y$的范围也为$[-5,5]$。

Matlab 代码：

```
clear;
clc;

t0=clock();

m=20;                                                    %蚂蚁个数
G=200;                                                   %最大迭代次数
Rho=0.9;                                                 %信息素蒸发系数
```

```
P0=0.1;                                                    %转移概率常数
XMAX=5;                                                    %搜索变量 x 最大值
XMIN=-5;                                                   %搜索变量 x 最小值
YMAX=5;                                                    %搜索变量 y 最大值
YMIN=-5;                                                   %搜索变量 y 最小值
%随机设置蚂蚁初始位置
for i=1:m
    X(i,1)=(XMIN+(XMAX-XMIN)*rand);
    X(i,2)=(YMIN+(YMAX-YMIN)*rand);
    Tau(i)=func(X(i,1),X(i,2));
end
step=0.1;%局部搜索步长
for NC=1:G
    lamda=1/NC;
    [Tau_Best,BestIndex]=min(Tau);
    %计算状态转移概率
    for i=1:m
        P(NC,i)=(Tau(BestIndex)-Tau(i))/(Tau(BestIndex));
    end
    %位置更新
    for i=1:m
        %局部搜索
        if   P(NC,i)<P0
            temp1=X(i,1)+(2*rand-1)*step*lamda;
            temp2=X(i,2)+(2*rand-1)*step*lamda;
        else
            %全局搜索
            temp1=X(i,1)+(XMAX-XMIN)*(rand-0.5);
            temp2=X(i,2)+(YMAX-YMIN)*(rand-0.5);
        end
        %边界处理
        if temp1<XMIN
            temp1=XMIN;
        end
        if temp1>XMAX
            temp1=XMAX;
        end
        if temp2<YMIN
            temp2=YMIN;
        end
        if temp2>YMAX
            temp2=YMAX;
        end
        %判断蚂蚁是否移动
```

```
            if func(temp1,temp2)<func(X(i,1),X(i,2))
                X(i,1)=temp1;
                X(i,2)=temp2;
            end
        end
        %更新信息素
        for i=1:m
            Tau(i)=(1-Rho)*Tau(i)+func(X(i,1),X(i,2));
        end
        [value,index]=min(Tau);
        trace(NC)=func(X(index,1),X(index,2));
        trace_x(NC,1)=X(index,1);
        trace_x(NC,2)=X(index,2);
end
[min_value,min_index]=min(trace);
minX=trace_x(min_index,1);
minY=trace_x(min_index,2);
% minValue=func(X(min_index,1),X(min_index,2));
disp(strcat('函数的最小值为',num2str(min_value)));
disp(strcat('对应的 X 为',num2str(minX)));
disp(strcat('对应的 Y 为',num2str(minY)));
Time_Cost=etime(clock,t0);
disp(['程序执行时间:' num2str(Time_Cost) '秒']);

%%绘图
% figure
% plot(trace)
% xlabel('搜索次数')
% ylabel('适应度值')
% title('适应度进化曲线')
%适应度函数
function value=func(x,y)
value=20*(x^2-y^2)^2-(1-y)^2-3*(1+y)^2+0.3;
end
```

关于蚁群算法的改进[4]：

1.精英蚂蚁系统

该算法一经发现的最好解为$T^{bs}(best-so-far)$,而该路径在修改信息素轨迹时人工释放额

外的信息素，以增强正反馈结果。公式为：$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}+e\Delta\tau_{ij}^{bs}$

式中 e 是调整T^{bs}影响权重的参数，$\Delta\tau_{ij}^{bs}$由下式给出：

$$\Delta\tau_{ij}^{bs}=\begin{cases}\dfrac{1}{L_{bs}}, & (i,j)\in T^{bs}\\ 0,\text{其它}\end{cases}$$

其中L_{bs}是已知最优路径T^{bs}的长度。

2.基于排序的蚁群算法

该算法中，每个蚂蚁释放的信息素按照它们的不同等级进行挥发，类似于精英算法，精英蚂蚁在每次循环中释放更多信息素。在修改信息素路径前，按照它们的旅行长度进行排名，蚂蚁释放信息素的量与他们的排名相乘。每次循环，只有排名前 w-1 只蚂蚁和精英蚂蚁才允许释放信息素，信息素如下所示:

$$\Delta\tau_{ij}=\sum_{r=1}^{w-1}(w-r)\Delta\tau_{ij}^{r}$$

3.自适应蚁群算法

（1）自适应改变 ρ 值:（如果这次的最短距离小于上一次，则需要留下更多的信息素给下一代蚂蚁，如果大于上一次，我们需要更多的稀释，避免子孙们重蹈爷爷们的覆辙）

$$\begin{cases}\rho_{t+1}=\rho_t, L_{\min}(t+1)\leq L_{\min}(t)\\ \rho_{t+1}=0.95\rho_t, L_{\min}(t+1)>L_{\min}(t)\end{cases}$$

参考：

[1]徐红梅,陈义保,刘加光,王燕涛.蚁群算法中参数设置的研究[J].山东理工大学学报(自然科学版),2008(01):7-11.

2.https://cloud.tencent.com/developer/article/1082500

3. https://zhuanlan.zhihu.com/p/63530081

4. https://zhuanlan.zhihu.com/p/113629381