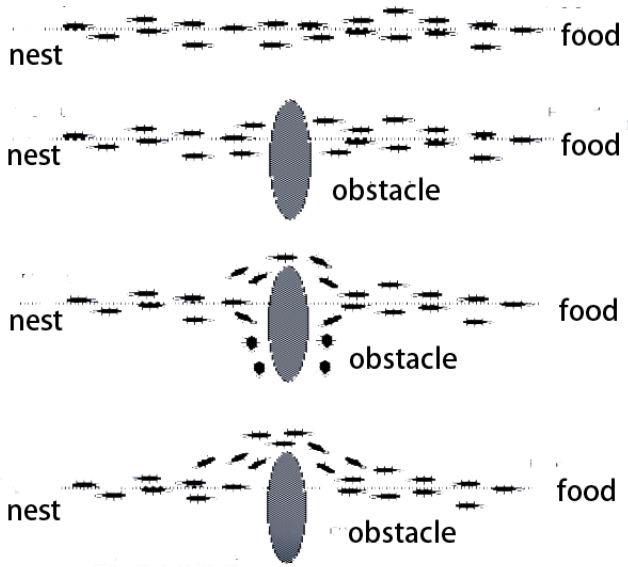
FCM聚类处理
的蚁群算法优化旅行商问题

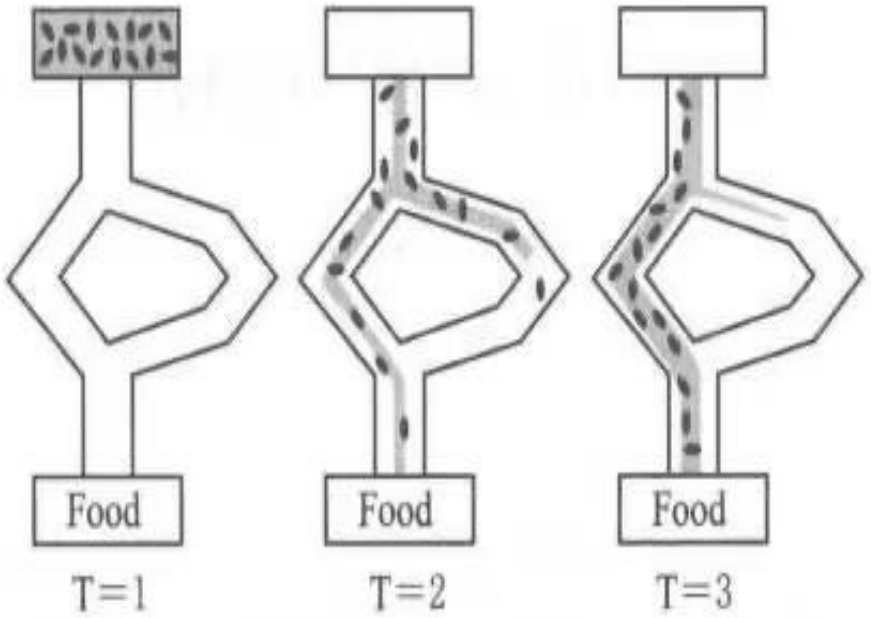
目录

CONTENTS

PART 01

背景及传统算法介绍





• 背景 BACKGROUND

旅行商问题（TSP）是一个经典的组合优化问题，它要求在给定的一组城市之间找到最短的路径，为了解决TSP问题，许多优化算法被提出，其中蚁群算法是一种受到自然界蚂蚁觅食行为启发的元启发式算法。

然而，随着城市规模的大幅度增加，传统的蚁群算法求解的速度较慢，容易陷入局部最优解，求出解的精度不高等缺点，针对此问题，需要提出相应的改进。

传统算法介绍

蚁群算法

蚁群算法是对自然界蚂蚁的寻径方式进行模拟而得出的一种仿生算法。

蚂蚁在运动过程中，能够在它所经过的路径上留下一种称之为外激素(pheromone)的物质进行信息传递，而且蚂蚁在运动过程中能够感知这种物质，并以此指导自己的运动方向，因此由大量蚂蚁组成的蚁群集体行为便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

FCM模糊聚类法

模糊C均值聚类算（Fuzzy C-Means，简称FCM）是一种常用的模糊聚类算法，用于将一组数据点划分为若干个模糊的聚类。

在传统的K均值聚类算法中，每个数据点只能属于一个聚类，而在FCM中，每个数据点可以以一定的隶属度（membership degree）属于多个聚类，表示其与每个聚类的关联程度

精华蚂蚁系统

Elitist Ant System

精华蚂蚁系统（Elitist Ant System）是在普通蚂蚁系统的基础上加以改进，其主要改进方式为更改信息素的迭代方式。在此次迭代中，若有蚂蚁所走的路径为当次最短路径时T^{bs}，为增强其正反馈的效果，人工释放额外的信息素e。信息素修改公式为

$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}+e\Delta\tau_{ij}^{bs}$$

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

T^{bs}即为最优路径L_{bs}的长度

精华蚂蚁系统

Elitist Ant System

MATLAB代码实现

```
        pos=find(L==min(L));
        % 更新信息素
        dt = zeros(n);
        e = 30;
        for k = 1:m
            for j = 1:n-1
                if ismember(k, pos)
                    dt(Tabu(k,j), Tabu(k,j+1)) = dt(Tabu(k,j),
Tabu(k,j+1)) + e * Q / L(k);
                end
                dt(Tabu(k,j), Tabu(k,j+1)) = dt(Tabu(k,j), Tabu(k,j+1)) +
Q / L(k);
            end
            dt(Tabu(k,n), Tabu(k,1)) = dt(Tabu(k,n), Tabu(k,1)) + Q /
L(k);
        end
        Tao = (1 - rho) * Tao + dt
```

PART 02

混合算法改进

FCM-EAS混合算法

FCM-EAS改进算法

将一个大规模的城市，通过FCM聚类分析，转化为K个小规模的子TSP问题，对于每个子TSP问题，独立的使用蚁群算法，求出每个子TSP问题的最优或较优路径，通过一定的规则，把每个子TSP问题的路径相连，进而得到全局最优或较优解。

将之前上机课编写的模糊聚类fcm代码改写为function函数，命名为f_fcm，以便在主程序中调用

```
%% 聚类处理
k = 5; % 设置聚类的群组数量
[idx, centroids] = f_fcm(citys,k,2); % 使用 fcm 聚类算法
```

将传统精华蚂蚁系统代码改写为function函数，命名为f_EAS，以便在函数中调用，传入城市坐标，蚂蚁数量等参数，返回值为最短路径长度与最短路径

```
 % 调用 EAS 算法
[shortestRoute, shortestLength,cov(class,:)] = ...
f_EAS(class_cities, m, max_count,class_D,class_v,class_Tabu);
Short_length(class)=shortestLength;
Short_path{class}= shortestRoute;
```

PART 03

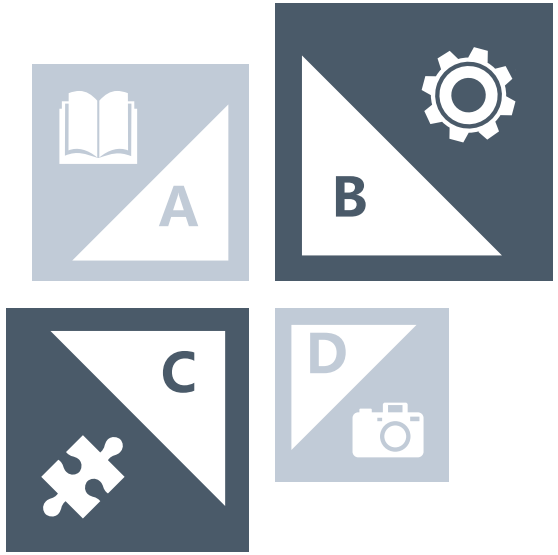
分析与仿真实验

时间复杂度对比

在传统的蚁群算法中，设外层的最大迭代次数为max_count，蚂蚁的数量为m，城市的数量为n，在每次迭代中，需要进行m次蚂蚁的路径选择和更新信息素的操作，路径选择和更新信息素的时间复杂度均为 $O(m\cdot n^2)$

传统蚁群算法的总时间复杂度为

$$O_1 = O\left(\max\_count\cdot m\cdot n\right)$$



相差 k^2 倍

基于FCM聚类的蚁群改进算法中，迭代次数，蚂蚁数量，城市数量，均与传统算法一致，但在分为k类之后，假设每类子TSP问题的蚂蚁数量和城市数量取分类均值，即分别取m/k，n/k，则该问题的时间复杂度为：

$$O_2 = O\left(k\cdot \max\_count\cdot\frac{m}{k}\cdot\left(\frac{n}{k}\right)^2\right)$$

$$= O\left(\max\_count\cdot m\cdot n/k^2\right)$$

仿真结果比较

城市数量	算法	聚类 K	路径长度	运行时间/s
30	AS	5	6846.94	10.25
	EAS		6846.94	9.25
	FCM-EAS		6664.86	0.97
100	AS	5	11902.33	48.59
	EAS		12044.22	32.37
	FCM-EAS		12505.47	13.63
197	AS	5	17921.20	208.59
	EAS		18197.13	81.43
	FCM-EAS		17725.90	62.67

注： 由于作者并未找到好的方法，将聚类处理后的各子 TSP 问题的最优路径连接起来，故这里取各子问题最短路径的和来近似看做整个问题的最短路径长度

从表格中可以看出，当城市的坐标为30时，三种算法的运行速度均很快，AS和EAS均在10秒左右找出了最短路径

当问题的规模增大到100时，三种算法的运行时间便有了较大的差异

当问题的规模增大到197时，传统AS算法与EAS和FCM-EAS有了非常大的差异，时间最大差异为三倍多

结　论

通过前面的分析以及仿真，我们可以得出结论：

在处理大规模的TSP问题时，使用 FCM-EAS混合算法，能够较大的缩短运行时间,提高运行的效率。

PART 04

评价与不足

评价与不足

1. 在选择聚类的K值时，是由自己直接给出，改进措施可使用循环遍历 $k \in \left[2, \sqrt{n}\right]$ ，得出各自的轮廓系数，选取轮廓系数最高相对应的k值作为聚类的数量。但需要注意的是，遍历求取轮廓系数的时间复杂度较高。

2. 在问题规模较大时，也可尝试考虑使用K-means聚类法替代FCM聚类法

3.本文在计算FCM-EAS的最短路径时，只是简单的把各个路径长度相加视为最短路径。可制定规则，当把聚类之后的各个最短路径连接起来时，可将各个类中心作为中心城市，对其再次使用蚁群算法获取其连接顺序（由于聚类之后的中心城市数量不太多，此项步骤或可以用普通的遍历或其他算法所替代）。得到中心城市的连接顺序之后，可遍历相邻两个类中各个点之间的距离，选取最短的将其连接起来，同时原类别中断开的另一点，可向此类的另一相邻类做遍历，连接其最短路径，如此往复直到形成完整回路，即视为最优路径。