# 实验二：遗传算法求解

# 旅行商问题

## 1.1 实验目的

1. 进一步巩固遗传算法的理论知识；
2. 理解旅行商问题，及如何建模为遗传算法求解问题；
3. 熟悉使用python语言编程；

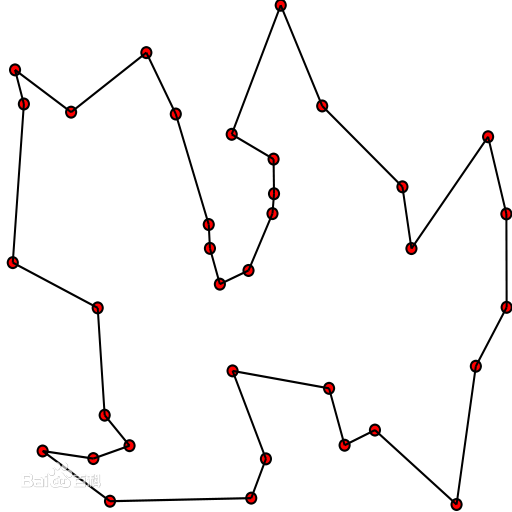
## 1.2 实验内容（4课时）

本实验采用遗传算法方法对旅行商问题进行求解。实验基于pycharm开发环境，用python语言进行编程。

## 1.3实验原理

**旅行商问题**

旅行推销员问题（英语：Travelling salesman problem, TSP）是这样一个问题：给定一系列城市和每对城市之间的距离，求解访问每一座城市一次并回到起始城市的最短回路。它是[组合优化](https://baike.baidu.com/item/%E7%BB%84%E5%90%88%E4%BC%98%E5%8C%96/3314860)中的一个[NP](https://baike.baidu.com/item/NP/7470332)难问题，在[运筹学](https://baike.baidu.com/item/%E8%BF%90%E7%AD%B9%E5%AD%A6/1559)和[理论计算机科学](https://baike.baidu.com/item/%E7%90%86%E8%AE%BA%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6/11034581)中非常重要。



**已知：20个城市的坐标**

*[(88, 16),(42, 76),(5, 76),(69, 13),(73, 56),(100, 100),(22, 92),(48, 74),(73, 46),(39, 1),(51, 75),(92, 2),(101, 44),(55, 26),(71, 27),(42, 81),(51, 91),(89, 54),(33, 18),(40, 78)]*

**采用遗传算法进行多次迭代求解。**

## 1.4 实验步骤

**（1）编码**

编为20\*1的列向量，每个分量是城市序号，如（1,3,6,12,…,19）。也可以直接是城市的坐标，如

*[(88, 16),(42, 76),(5, 76),(69, 13),(73, 56),(100, 100),(22, 92),(48, 74),(73, 46),(39, 1),(51, 75),(92, 2),(101, 44),(55, 26),(71, 27),(42, 81),(51, 91),(89, 54),(33, 18),(40, 78)]*

**（2）初始化**

**设置种群规模（20-100），每个解是随机产生的。**

**（3）目标函数**

回路的总长度最小。

**（4）适应度函数**

适应度函数是当当前回路总长度的倒数**。**

*#适应度函数***def** calFitness(line,dis\_matrix):  
 dis\_sum = 0  
 dis = 0  
 **for** i **in** range(len(line)):  
 **if** i<len(line)-1:  
 dis = dis\_matrix.loc[line[i],line[i+1]]*#计算距离* dis\_sum = dis\_sum+dis  
 **else**:  
 dis = dis\_matrix.loc[line[i],line[0]]  
 dis\_sum = dis\_sum+dis  
 **return** round(dis\_sum,1)

**（5）个体选择**

采用竞标赛的方式选择个体

*#锦标赛法选择***def** tournament\_select(pops,popsize,fits,tournament\_size):  
 new\_pops,new\_fits = [],[]  
 **while** len(new\_pops)<len(pops):  
 tournament\_list = random.sample(range(0,popsize),tournament\_size)  
 tournament\_fit = [fits[i] **for** i **in** tournament\_list]  
 *#转化为df方便索引* tournament\_df = pd.DataFrame([tournament\_list,tournament\_fit]).transpose().sort\_values(by=1).reset\_index(drop=**True**)  
 *#选出获胜者* fit = tournament\_df.iloc[0,1]  
 pop = pops[int(tournament\_df.iloc[0,0])]  
 new\_pops.append(pop)  
 new\_fits.append(fit)

**（6）交叉**

采用单点位交叉方法。

*#顺序交叉***def** crossover(popsize,parent1\_pops,parent2\_pops,pc):  
 child\_pops = []  
 **for** i **in** range(popsize):  
 *#初始化* child = [**None**]\*len(parent1\_pops[i])  
 parent1 = parent1\_pops[i]  
 parent2 = parent2\_pops[i]  
 **if** random.random() >= pc:  
 child = parent1.copy()*#随机生成一个（或者随机保留父代中的一个）* random.shuffle(child)  
 **else**:  
 *#parent1* start\_pos = random.randint(0,len(parent1)-1)  
 end\_pos = random.randint(0,len(parent1)-1)  
 **if** start\_pos>end\_pos:  
 tem\_pop = start\_pos  
 start\_pos = end\_pos  
 end\_pos = tem\_pop  
 child[start\_pos:end\_pos+1] = parent1[start\_pos:end\_pos+1].copy()  
 *# parent2 -> child* list1 = list(range(end\_pos+1,len(parent2)))  
 list2 = list(range(0,start\_pos))  
 list\_index = list1+list2  
 j = -1  
 **for** i **in** list\_index:  
 **for** j **in** range(j+1,len(parent2)):  
 **if** parent2[j] **not in** child:  
 child[i] = parent2[j]  
 **break** child\_pops.append(child)  
 **return** child\_pops

**（7）变异**

以一定变异概率，对新个变异进行单点位变异

*#基本位变异***def** mutate(pops, pm):  
 pops\_mutate = []  
 **for** i **in** range(len(pops)):  
 pop = pops[i].copy()  
 *# 随机多次成对变异* t = random.randint(1, 5)  
 count = 0  
 **while** count < t:  
 **if** random.random() < pm:  
 mut\_pos1 = random.randint(0, len(pop) - 1)  
 mut\_pos2 = random.randint(0, len(pop) - 1)  
 **if** mut\_pos1 != mut\_pos2:  
 tem = pop[mut\_pos1]  
 pop[mut\_pos1] = pop[mut\_pos2]  
 pop[mut\_pos2] = tem  
 pops\_mutate.append(pop)  
 count += 1  
 **return** pops\_mutate

**（8）画出每个解的路线，及对应的回路长度值变换曲线**

*# 画路径图***def** draw\_path(line, CityCoordinates):  
 x, y = [], []  
 **for** i **in** line:  
 Coordinate = CityCoordinates[i]  
 x.append(Coordinate[0])  
 y.append(Coordinate[1])  
 x.append(x[0])  
 y.append(y[0])  
 plt.subplot(1, 2, 1)  
 plt.plot(x, y, **'mo:'**, color=**'#4169E1'**, alpha=0.8, linewidth=0.8)  
 plt.xlabel(**'x'**)  
 plt.ylabel(**'y'**)  
 *# plt.show()*

## 1.5 实验拓展

1. 更换种群规模，看有什么不同效果；

2. 更换变异概率，看有什么不同效果；

3. 可以使用geapy工具来做

4. 把竞标赛法换成轮盘赌方法选择个体

## 1.6 实验要求

1.按照标准格式撰写实验报告

2.要求对实验所用知识、心得体会进行梳理