**中山大学计算机学院**

**人工智能**

**本科生实验报告**

课程名称：Artificial Intelligence

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 |  | 姓名 |  |

# 实验题目

1. 为了方便批改作业, 我们统一用类 GeneticAlgTSP来编写遗传算法的各个模块, 并分析算法性能. 该类需包含以下方法:
2. 构造函数 \_\_init\_\_(), 输入为TSP数据集文件名 filename, 数据类型 str. 例如 "dj38.tsp"是Djibouti的38个城市坐标数据文件; "ch71009.tsp"是China的71009个城市坐标数据文件. 我们需要在构造函数中读取该文件中的数据, 存储到类成员 self.cities中(数据类型自定, 建议存储为 numpy数组). 同时在构造函数中初始化种群, 存储到类成员 self.population中(数据类型自定).
3. 求解方法 iterate(), 输入为算法迭代的轮数 num\_iterations, 数据类型 int. 该方法是基于当前种群 self.population进行迭代(不是从头开始), 返回迭代后种群中的一个较优解, 数据类型 list, 格式为1-n个城市编号的排列. 例如, 对于n=5的TSP问题, 迭代后返回的较优解形如 [1,3,4,5,2], 表示当前较好的游览城市次序为1-3-4-5-2-1.

可以在类中编写其他方法以方便编写并分析遗传算法的性能. 请在代码注释或实验报告中说明每个方法/模块的功能.

# 实验内容

1. 算法原理

遗传算法：

1：总思路：

1. 随机生成10条不同的初始路径作为（初代种群）population
2. 迭代指定次数

* 从（当代种群）population中根据轮盘赌选取10个个体更新当代种群，模拟自然界的优胜劣汰选择过程。
* 生成子代个体，组成子代种群Children：

随机从更新后的当代种群中挑选2个父本parent1,parent2，进行部分映射交叉（PMX），生成两个个体child1,child2

根据一定的概率，对两个子代个体child1,child2进行倒置变异。若两个子代个体相等，为了避免子代种群单一化，再进行倒置变异操作

* 将子代个体chikld1,child2加入到子代种群Children中
  + - * + 子代种群Children和父代种群population合并，根据适应度大小选取出最适应的十个个体作为下一代的父本，继续迭代

1. 从迭代完的当代种群中挑选出适应度最大的个体作为答案

// 初始化参数

POPULATION\_SIZE = 10

ITERATIONS = 指定次数

MUTATION\_RATE = 指定的变异概率

// 初始化种群

population = 初始化种群(POPULATION\_SIZE)

// 开始迭代过程

for iteration in range(ITERATIONS):

// 1. 轮盘赌选择

selected\_parents = 轮盘赌选择(population, POPULATION\_SIZE)

// 2. 生成子代种群

children = []

for i in range(POPULATION\_SIZE ):

parent1, parent2 =//随机挑选父本个体

child1, child2 = PMX交叉(parent1, parent2)

// 变异操作

if (随机生成数() < MUTATION\_RATE):

child1 = 倒置变异(child1)

if (随机生成数() < MUTATION\_RATE):

child2 = 倒置变异(child2)

// 如果两个子代个体相等，则再次进行变异

if (child1 == child2):

child2 = 倒置变异(child2)

// 将子代个体加入子代种群

children.append(child1)

children.append(child2)

// 3. 选择下一代父本

merged\_population = population + children

population = 选择操作(merged\_population, POPULATION\_SIZE)

// 4. 输出最终结果

best\_individual = 获取适应度最大的个体(population)

输出(best\_individual)

2：相关操作：

1. 计算种群中个体中的适应度（fitness）：
   * + - * 计算种群中每个个体对应路径的路径长度distance
         * 挑选出最大的路径长度max
         * 种群中每个个体的相对适应度为max-distance+1，实现路径长度越短，适应度越大
         * 注意这里计算的适应度只是相对于这个种群内部的，两个种群的适应度的尺度不一样，进行两个种群的个体适应度比较时，要统一尺度
2. 对种群轮盘赌实现优胜劣汰（RWS）：
   * + - * 对种群计算总适应度，用于计算累计概率
         * 计算种群每个个体的累计概率数组：（排在前面个体的适应度之和+自己的适应度）/总适应度
         * 进行十次轮盘赌，选出10个个体，组成新的种群

生成10个0-1的随机数，根据累计概率的区间选择符合的个体

1. 对父代个体的部分映射交叉重组（Crossover）:

* 随机生成两个父代个体（P1，P2）交叉重组的范围（s,t）
* P1，P2对（s,t）片段内的父代路径进行交换，并且记录下交换的映射dict\_P1,dict\_P2（旅行商问题，一个城市只经过一次，还要不重复，则交叉片段外的自然要根据交叉片段进行映射）
* P1,P2对交换片段外的个体基因根据dict\_P1,dict\_P2进行映射

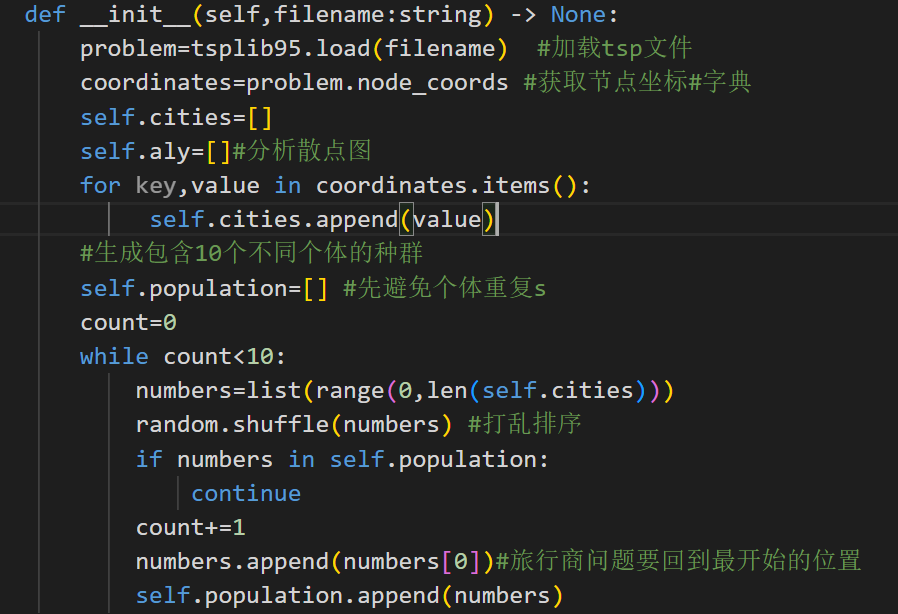
1. 对个体的倒置变异（Mutation）：

* 随机个体生成倒置变异的范围，然后倒置该范围内的路径

1. 关键代码展示（可选）
2. 初始化：从.tsp文件中提取城市坐标信息，并且随机生成10条路径初始化种群

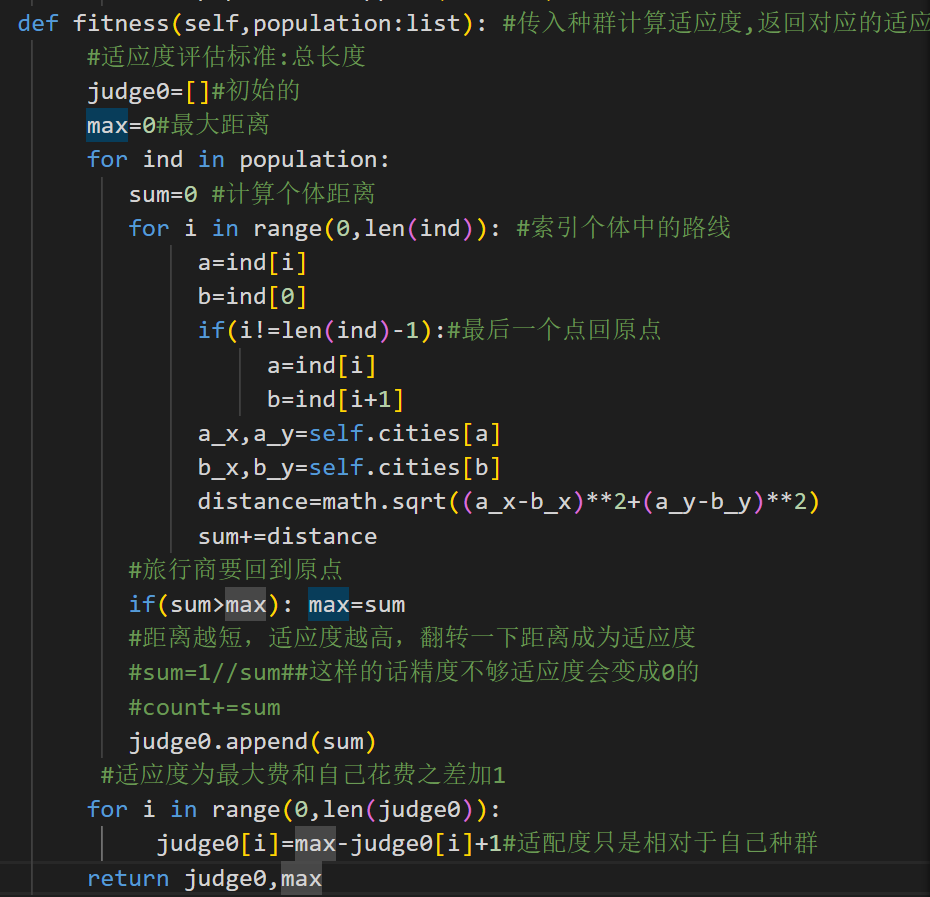
旅行商问题是要回到原点的，随机生成完路径后要加上原点

aly数组：记录每次迭代的最优路径数据，用于绘制散点图分析迭代效果



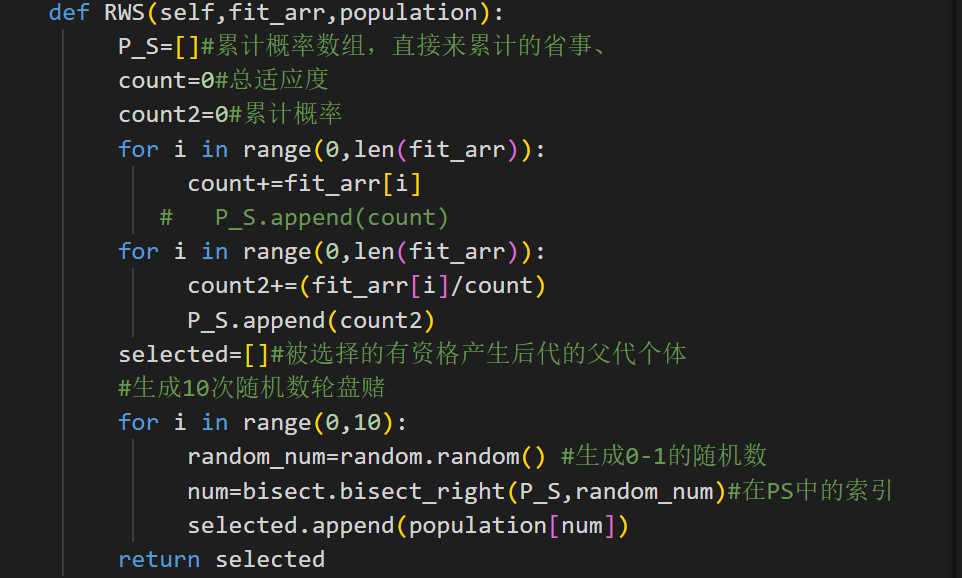
1. 种群相对适应度计算（fitness）：这里没有选择直接求distance的倒数为适应

度，因为distance很大，倒数的精度可能会不够而出错

s

1. 轮盘赌优胜劣汰（RWS）：计算种群中每个个体的累计概率，生成0-1随机数

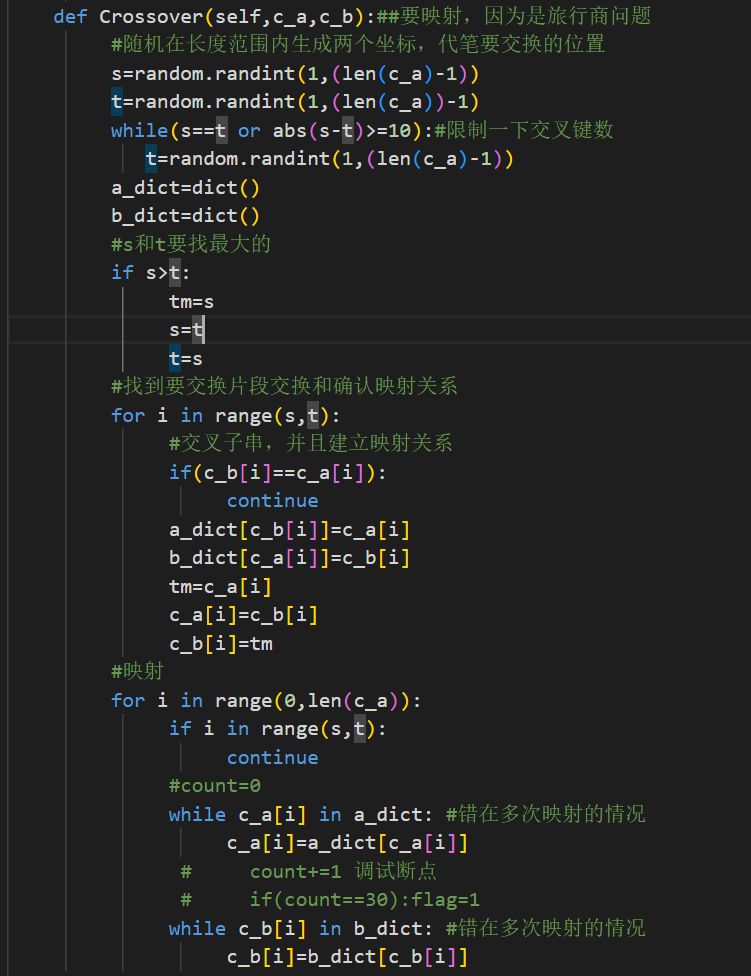
看其落在的累计概率区间选取个体



1. PMS部分交叉重组（Crossover）：

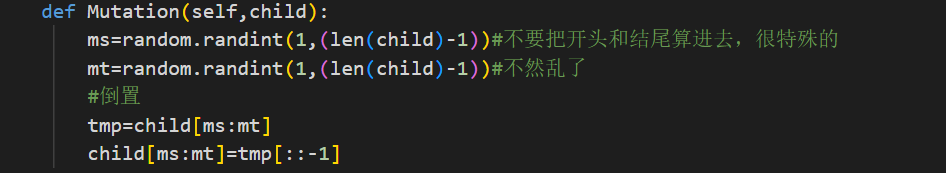
* 交叉限制范围不超过10
* 交叉范围不能包括路径头或者尾，如果交叉路径包括头或者尾，在交叉范围外映射时容易陷入死循环（（为了方便清楚起见））
* 连环映射的目的：完全确保交叉后的路径头尾相同，且中间不经过相同城市且完全经过所有城市
* 问题：容易出现交叉映射字典中自己映射自己无法退出映射的问题

交叉映射字典中相互映射不容易判断何时退出连环映射问题



1. 倒置变异（Mutation）

注意随机生成的变异范围不要包含头和尾



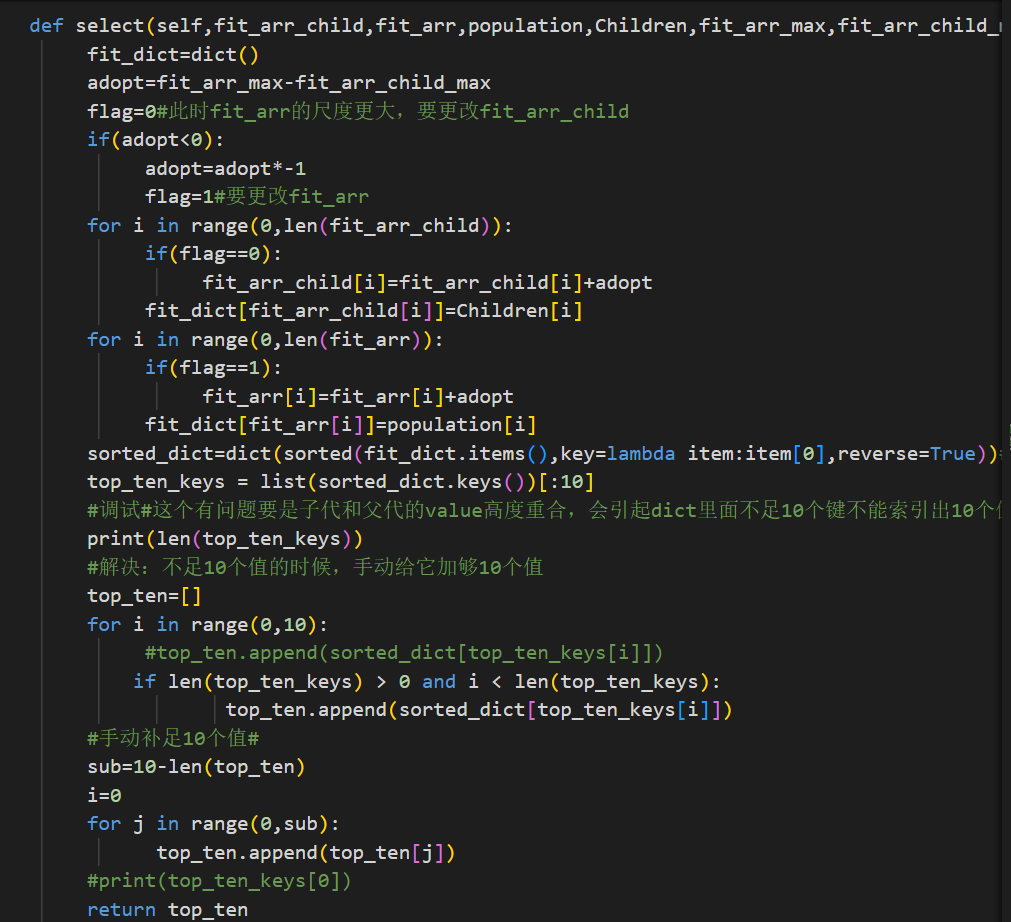
1. 合并父子种群并且得到适应度前十的个体构成新种群（select）：

* 注意：要调整父子种群的种群相对适应度得到父子种群间绝对适应度，找

到父子种群最大适应的差值adopt，如果是父种群的最大适应度大，

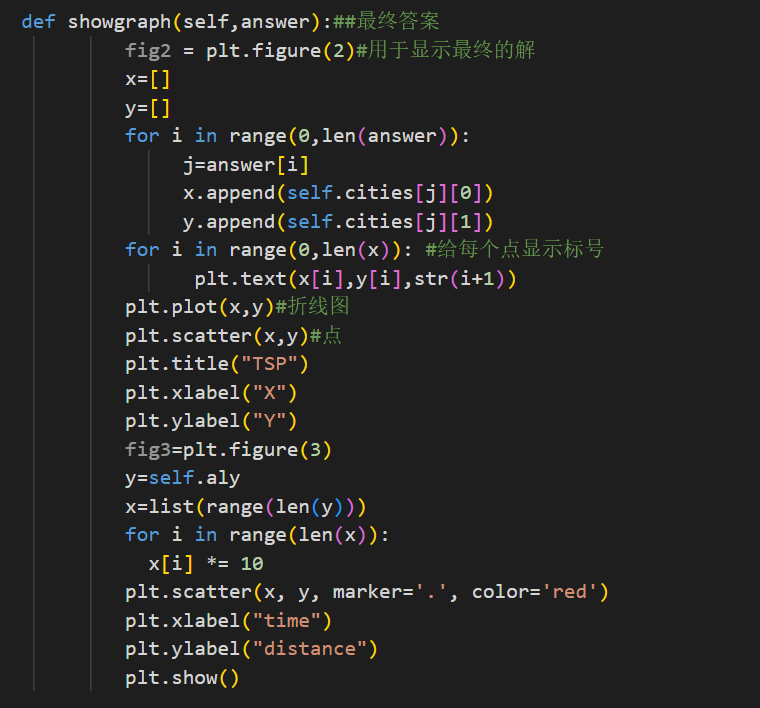
则为子种群加上adopt调整因子，否则就为父种群加上adopt调整

因子

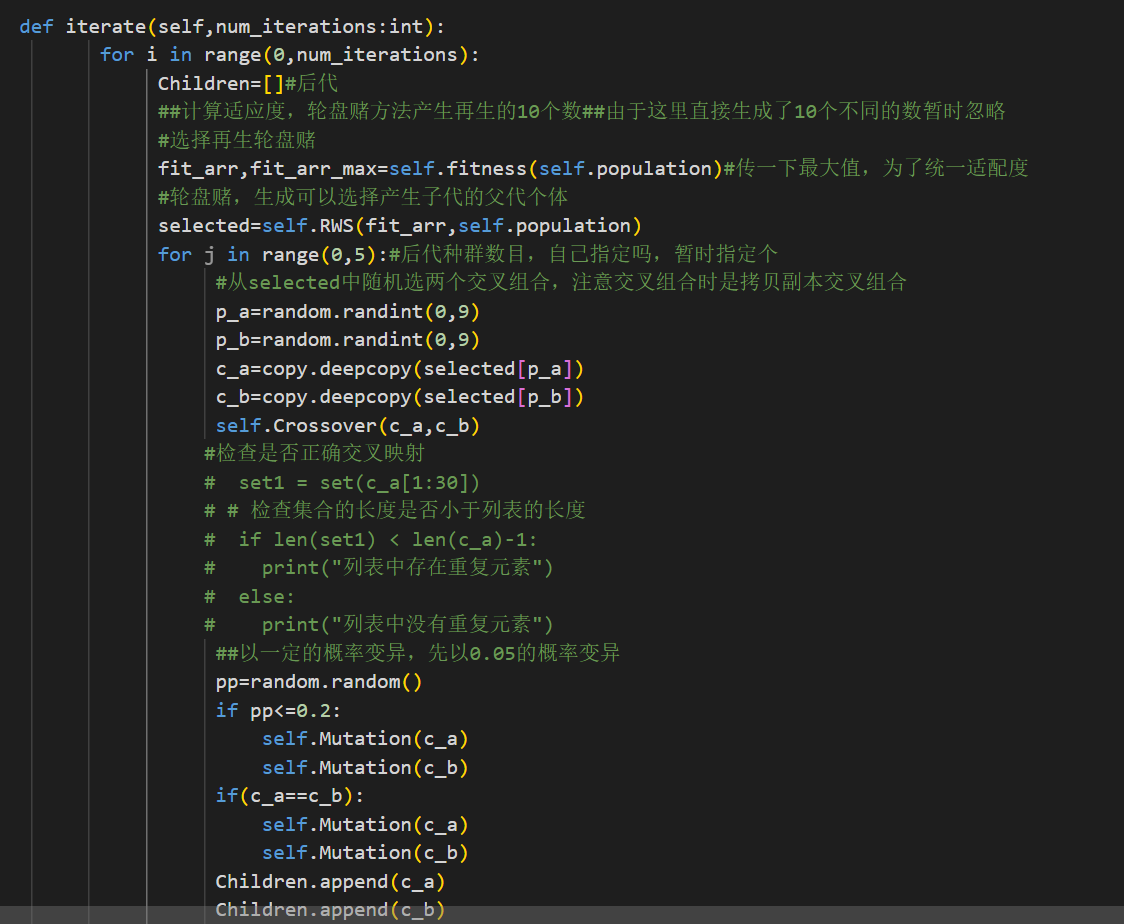


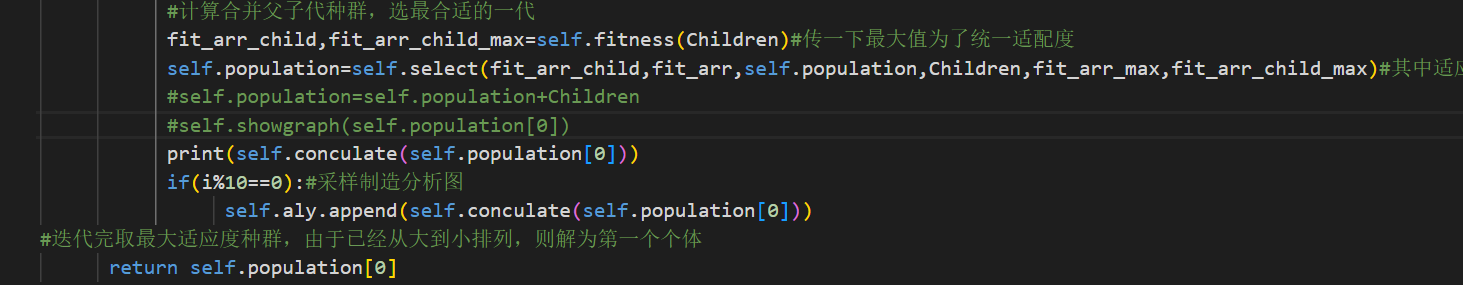
1. 结果可视化（showgraph）：

* 画出最终路径
* 画出每次迭代最优长度的散点分析图



1. 总迭代函数（iterate）：





1. 创新点&优化（如果有）

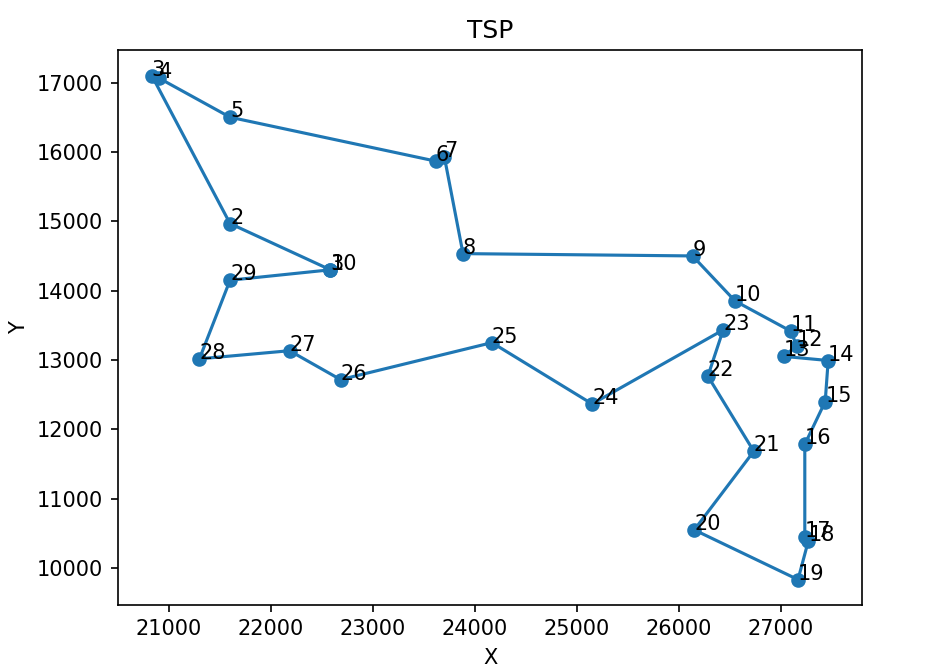
计算的是相对适应度，而非绝对适应度，减少由于计算绝对适应度精度损失带来的误差

# 实验结果及分析

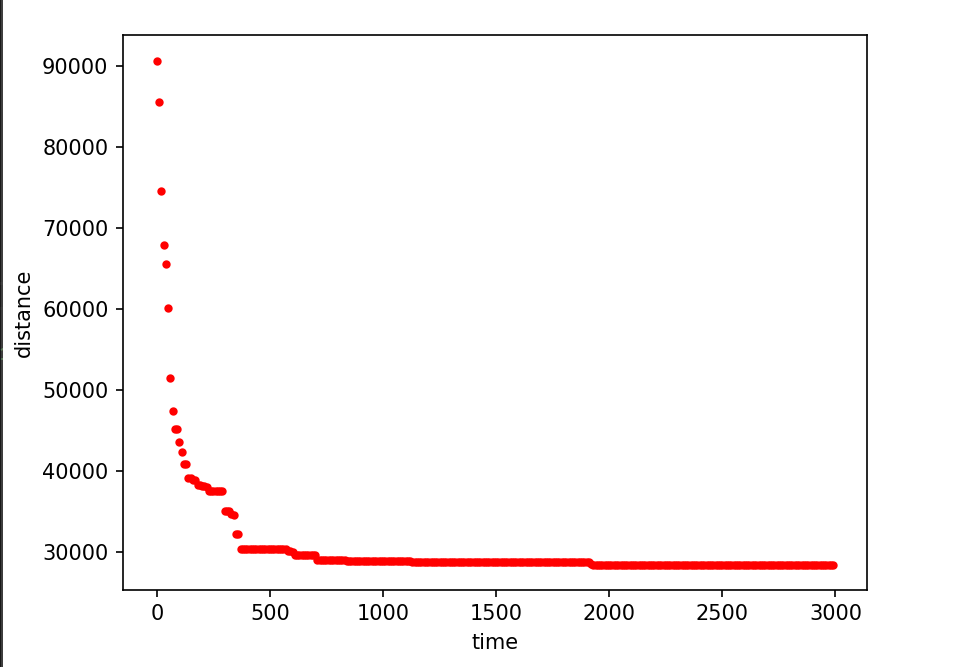
1. 实验结果展示示例

1：为29个城市的旅行商问题迭代3000次得到的结果（wi29.tsp）：

1. 最终路径



1. 迭代分析散点图：每次迭代得到的长度越来越优

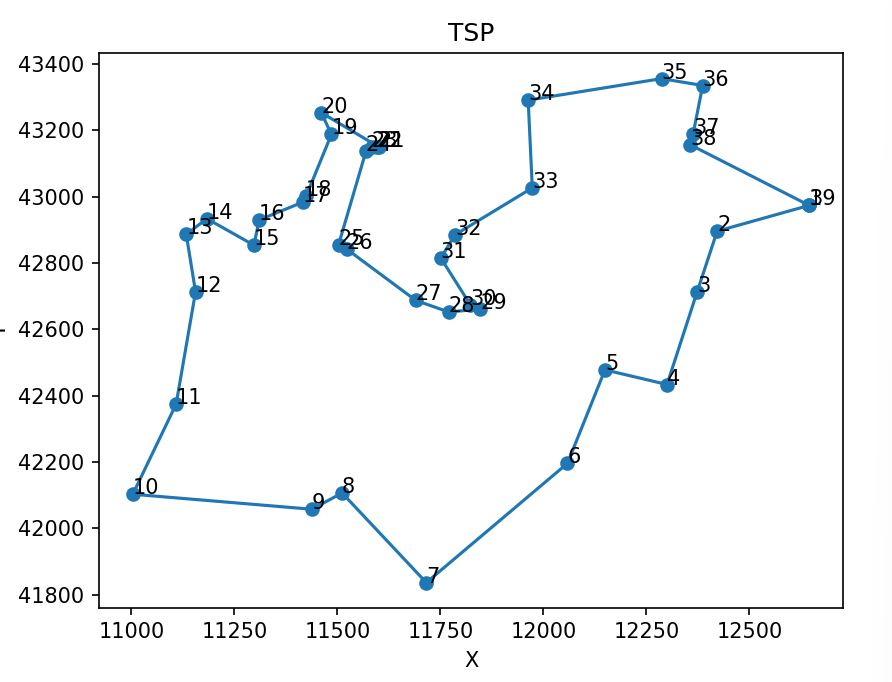


1. 得到的最终数据和<https://www.math.uwaterloo.ca/tsp/world/summary.html>记录的中最优数据接近

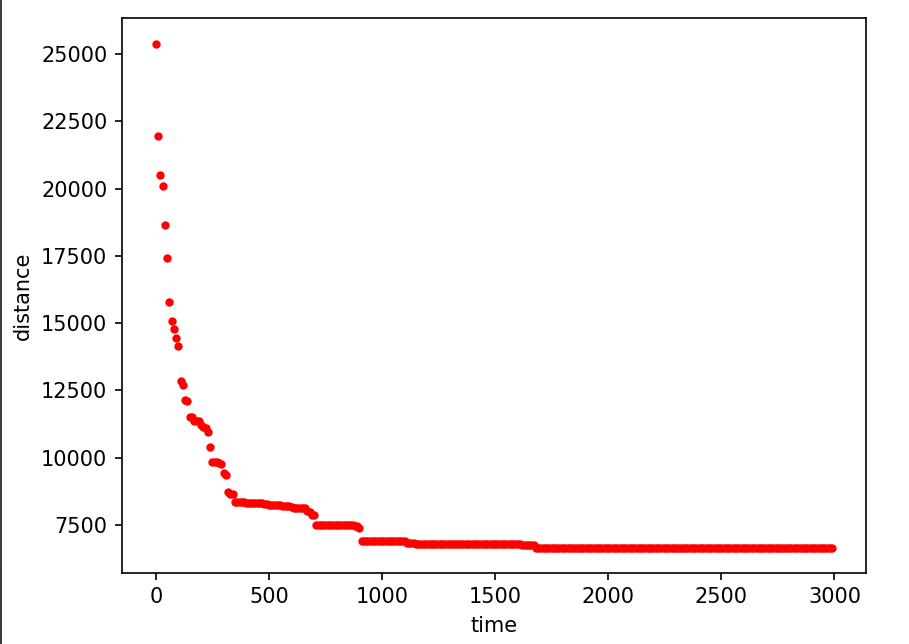




1. 为38个城市的旅行商问题迭代3000次得到的结果(dj38.tsp)：
2. 最终路径



1. 迭代分析散点图：每次迭代得到的长度越来越优



1. 得到的最终数据和<https://www.math.uwaterloo.ca/tsp/world/summary.html>记录的中最优数据几乎一致都为6656

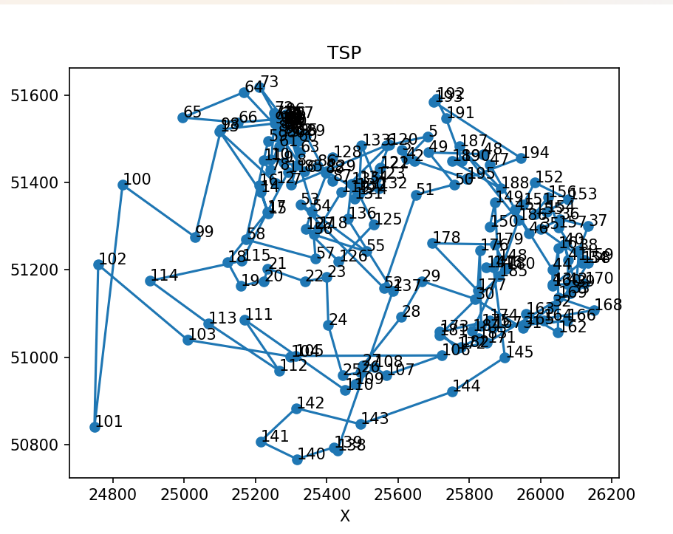




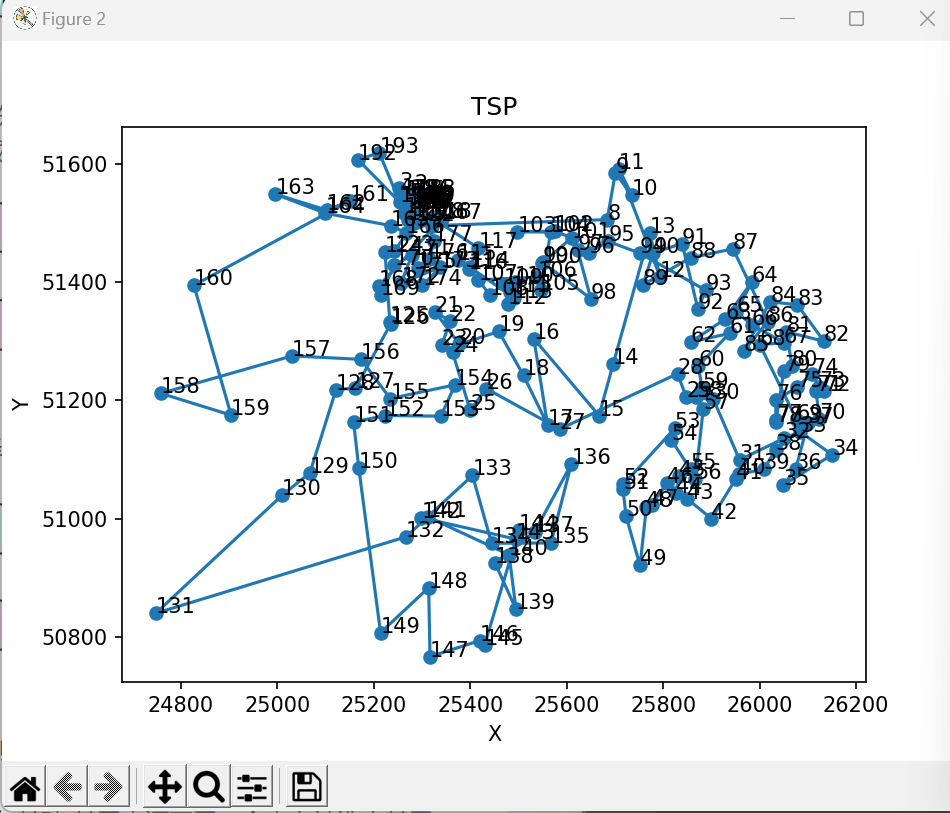
3. 为194个城市的旅行商问题迭代3000/7000次得到的结果(qa194.tsp)：

1) 最终路径

3000次迭代

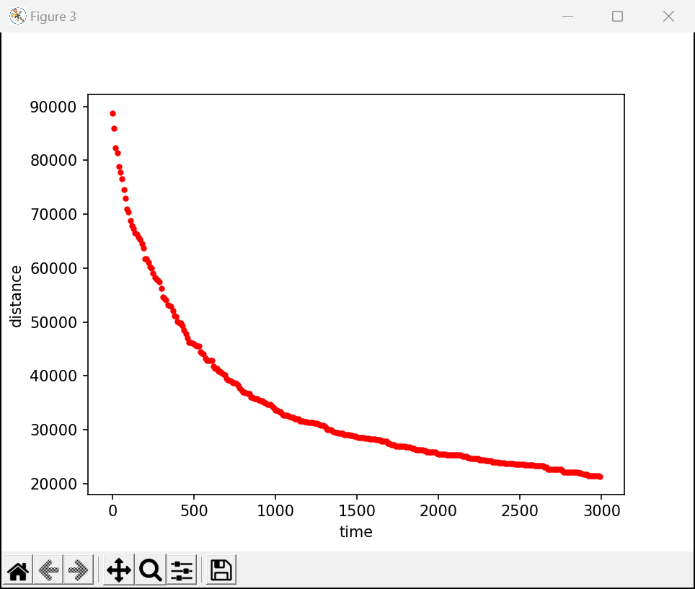


7000次迭代

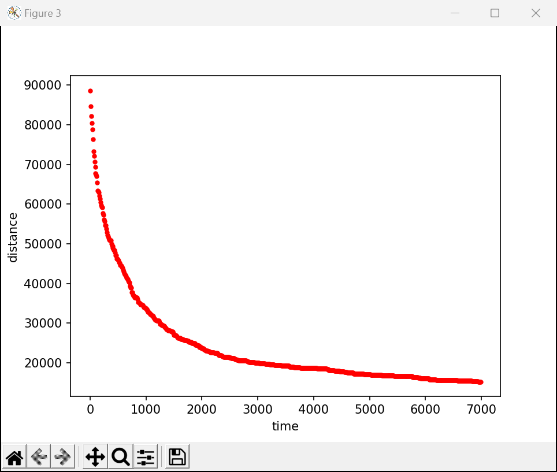


2) 迭代分析散点图：每次迭代得到的长度越来越优

3000次迭代

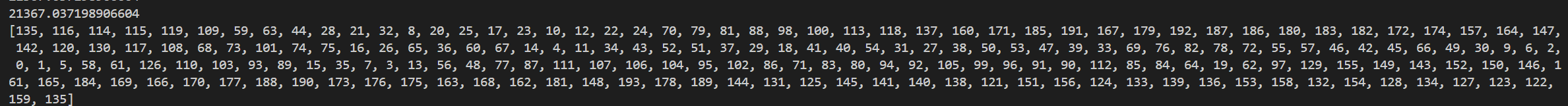


7000次迭代：

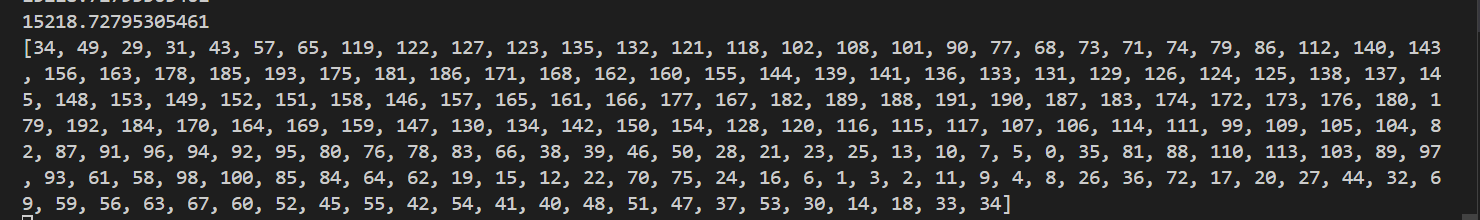


1. 得到的最终数据和https://www.math.uwaterloo.ca/tsp/world/summary.html记录的中最优数据虽然有一定差距，但是可以看见迭代趋势是下降的，且3000次迭代处的梯度还是较大的，进一步迭代肯定能得到更优长度

3000：



7000：





2. 评测指标展示及分析

1. 在对wi29求解时，对变异率参数做出了调整，尝试了0.2，0.25，0.23，0.22，0.21五种变异率，运行多次后，变异率0.21和0.22较优。

* 变异率为0.21时，运行六次路径总长度（精确到个位）为：28082，27601，27601，29044，29006，28434（27601出现可能是运算过程中出现舍入误差）。对多种变异率迭代相同次数，变异率为0.21时能以较大的概率得到相对最优的解，但是不是很稳定，也能得到较大的数据。
* 变异率为0.22时，运行六次路径总长度（精确到个位）为： 28042，28292， 28189，28229， 28189， 28229。变异率为0.22时，能稳定得到较优的解。
* 但是从对路径总长度的收敛程度而言，迭代3000次的情况下，五种变异率率得到的路径总长度都已经趋近收敛了

对dj38和qa194（迭代3000次）：

···

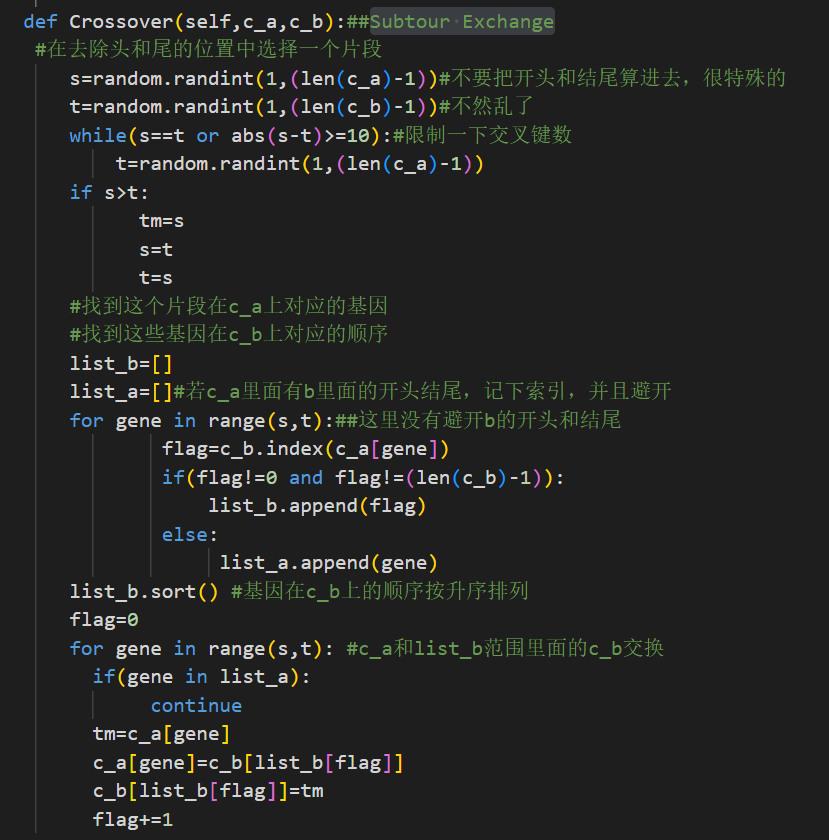
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 变化率 | 实验路径长度 | 实验路径长度 | 实验路径长度 |
| Dj38 | 0.21 | 7282 | 6659 | 6890 |
| Dj38 | 0.22 | 6662 | 7078 | 7179 |
| Qa194 | 0.21 | 21190 | 20793 | 20145 |
| Qa194 | 0.22 | 19811 | 22067 | 20405 |

1. 运行时间：对wi29迭代3000次的时间约为6s，对dj38迭代3000次的时间约为7.5s，对qa194迭代3000次的时间约为24.5s，由于对qa194迭代3000次得到的解离较优解有差距，对其迭代7000次时间为52秒
2. 其他交叉和变异算法：（由于经过实验以下的交叉变异方式比上面提到的效率低和得到的解较大，所以只在下面简略分析）：

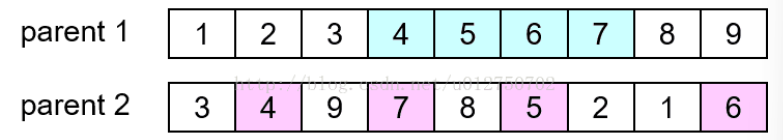
* Subtour Exchange交叉算法：

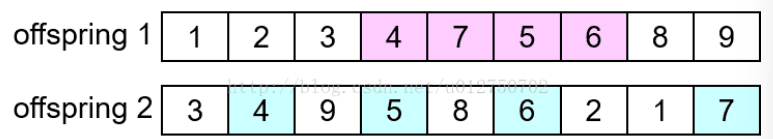
在父代基因1中随机得到一段片段，找到片段里面的基因在父代基因2中的位置，交换两个父代基因中关于这个片段的因子的相对顺序。

* Code：

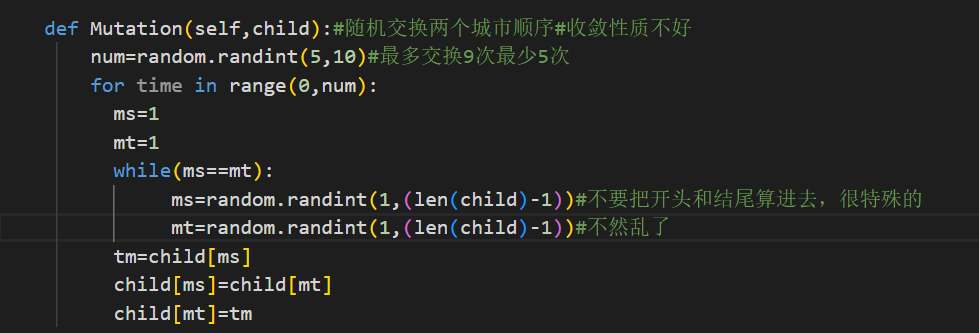


* 图解：来源于<https://blog.csdn.net/ztf312/article/details/82793295>





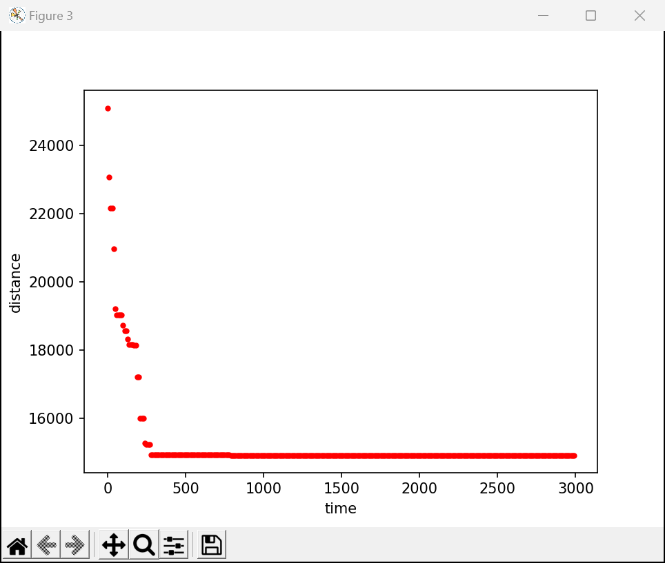
* 简单变异算法：
* 思路：生成随机数代表要变异的因子对数n，生成n对随机数（去除城市的开头和路径索引）作为要交换的因子的索引，交换对应城市在路径里面的位置
* Code:

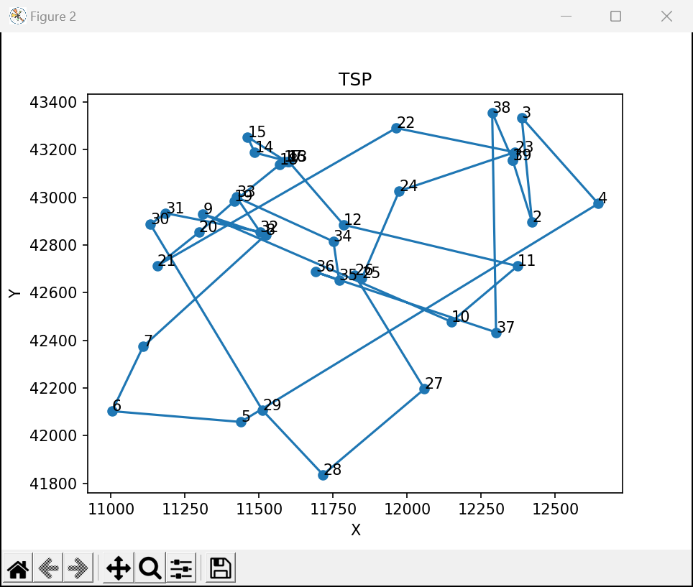


* 分析：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 路径长度 | 路径长度 | 路径长度 | 路径长度 | 路径长度 |
| dj38 | 14059 | 13489 | 14908 | 13401 | 13503 |
| Wi29 | 51874 | 38769 | 43645 | 43083 | 42854 |

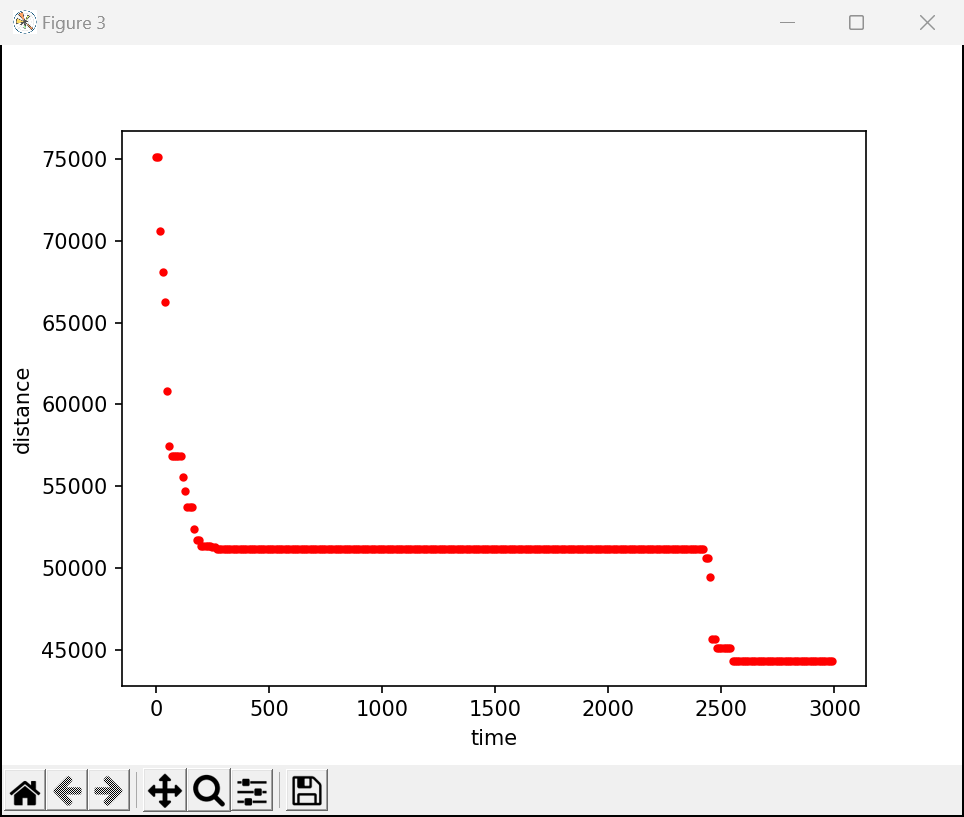
* Dj38可视化结果：

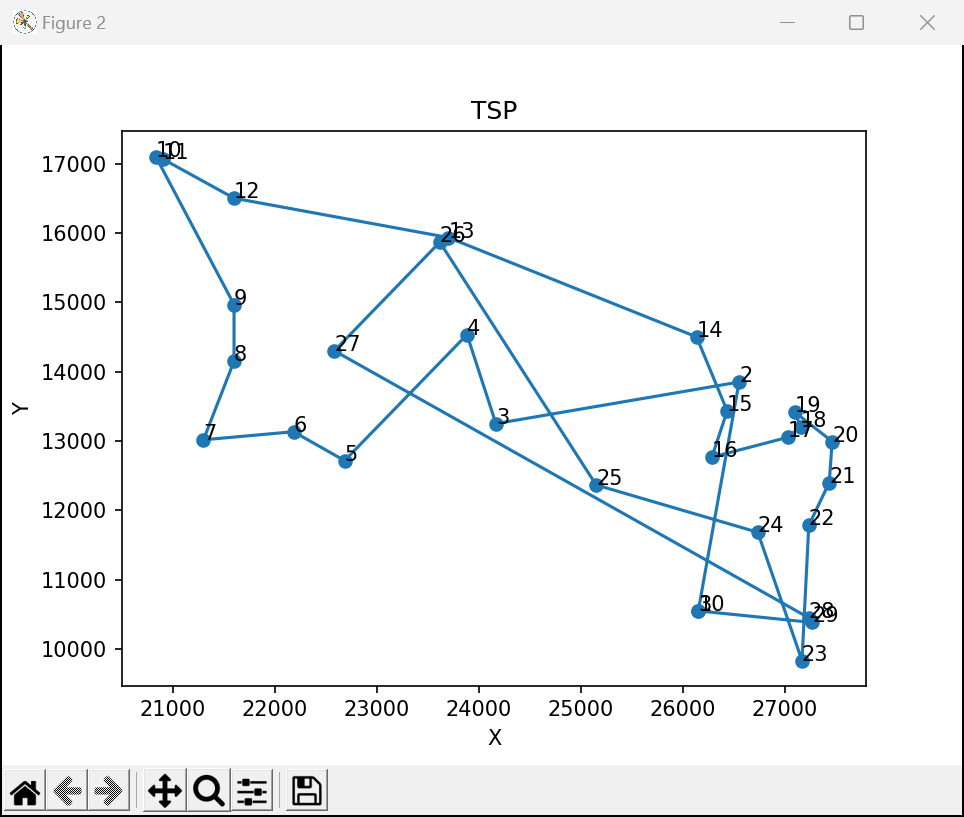






* Wi29可视结果：







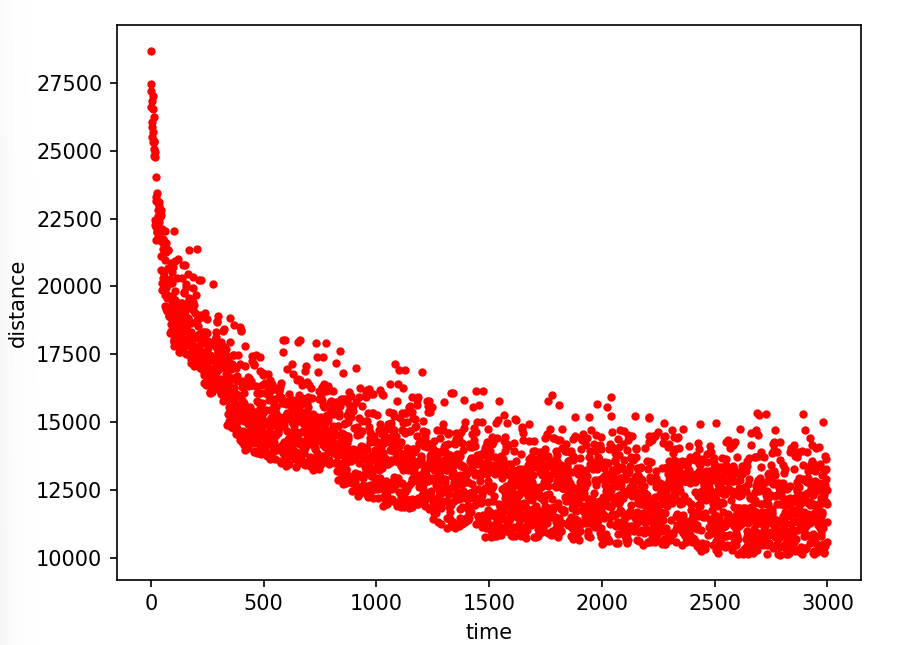
# 与模拟退火算法对比

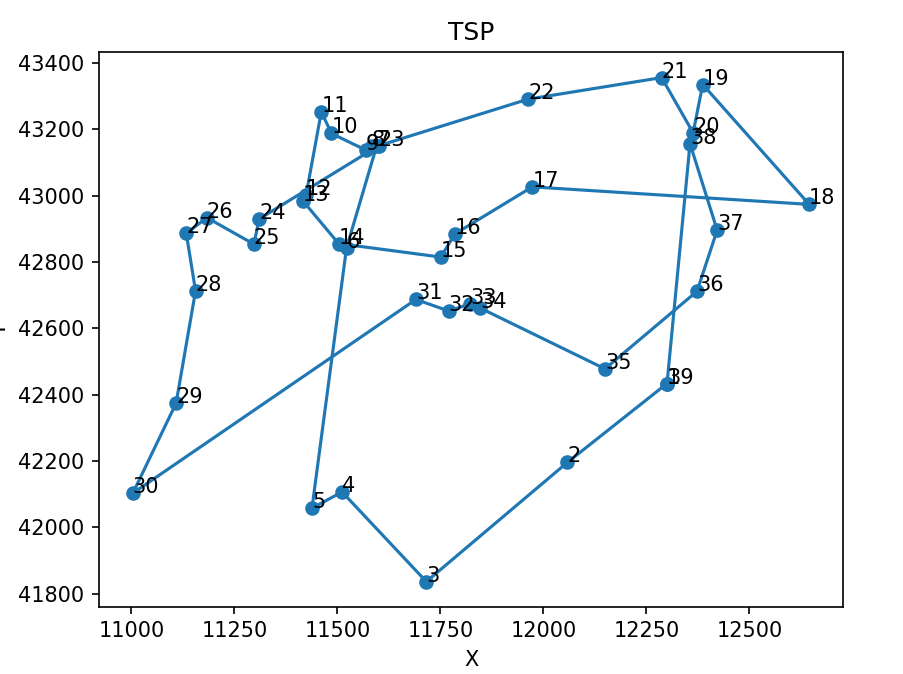
1. 模拟退火算法的实现：

* 温度采用指数型下降
* 新的解的生成采用随机调换路线中两个城市的位置
* 新的解的接受概率分为两种情况：
* 当新的解比原来解优的时候，以概率1接受
* 新的解比原来解差的时候，以exp（-cost\_del/t）接受



模拟退火算法的可视化处理（dj38迭代3000次）





2；遗传算法和模拟退火算法的对比：（都迭代3000次）

相同TSP问题迭代相同次数，遗传算法更为稳定和解更优（在qa194上退火算法的

解也比遗传算法差一点）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Wi29 | 退火 | 37017 | 31319 | 30661 |
| Wi29 | 遗传 | 28042 | 28292 | 28189 |
| Dj38 | 退火 | 10118.3 | 8884 | 11193 |
| Dj38 | 遗传 | 6662 | 7078 | 7179 |

# 参考资料

其他交叉算法来源：<https://blog.csdn.net/ztf312/article/details/82793295>

PMX交叉，倒置变异参考来源：实验课ppt

RWS轮盘赌优胜劣汰参考来源：课本