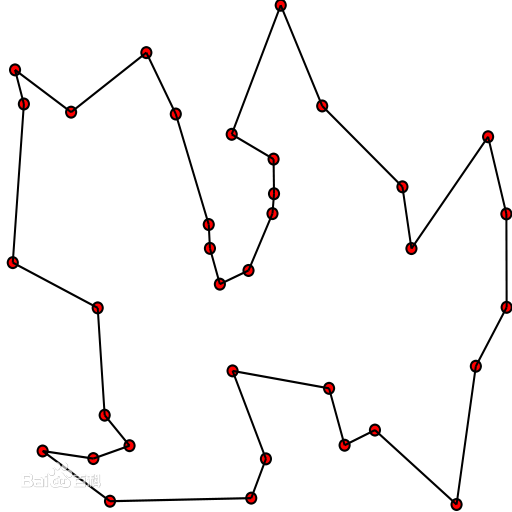
**编程实现改进的遗传算法处理旅行商问题**

1. **问题介绍**

旅行商问题(Traveling Salesman Problem，TSP)是一个经典的组合优化问题。经典的TSP可以描述为：一个商品推销员要去若干个城市推销商品，该推销员从一个城市出发，需要经过所有城市后，回到出发地。应如何选择行进路线，以使总的行程最短。从图论的角度来看，该问题实质是在一个带权完全无向图中，找一个权值最小的哈密顿回路。由于该问题的可行解是所有顶点的全排列，随着顶点数的增加，会产生组合爆炸，它是一个NP完全问题。



1. **算法介绍**
2. **遗传算法来源：**

1885年年，达尔文用自然选择来解释物种的起源和生物的进化。达尔文的自然选择学说包括三个方面：遗传，变异，生存斗争和适者生存。上世纪20年代，一些学者用统计生物学和种群遗传学重新解释达尔文自然选择理论，形成现代综合进化论。

种群遗传学认为：在一定地域中一个物种的全体成员构成一个种群；生物的进化是种群的进化，每一代个体基因型的改变会影响种群基因库的组成，而种群基因库组成的变化就是这一种群的进化。

下面是遗传算法中与生物学相关的概念和术语与优化问题中的描述的关系：

个体：解

种群：解集/解空间

适应度：评价/目标/寻优函数

选择、交叉、变异：产生新解的方法

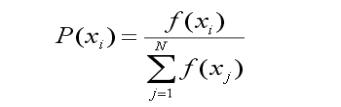
1. **遗传算法实现流程：**

1.计算开始时，随机初始化一定数目的个体，并计算每个个体的适应度值，产生第一代（初始种群）。

2.如果不满足优化准则，开始新一代的计算：按照适应度值选择父代个体，父代按一定概率通过轮盘赌进行交叉操作，产生的子代按一定概率变异，形成新的一代。计算新一代的适应度值并入交叉操作之前的种群中，优胜劣汰。

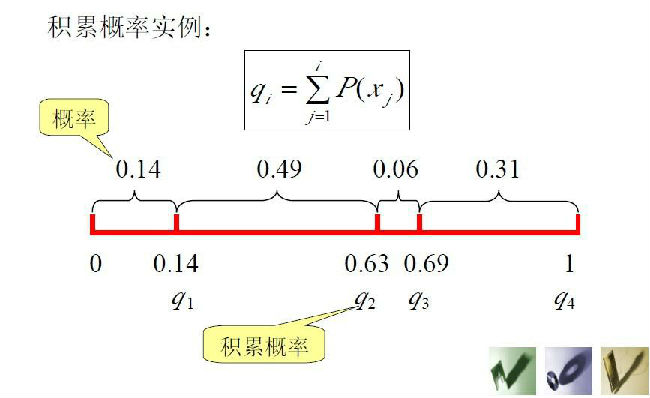
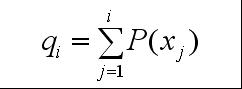
3.这一过程循环执行，直到满足优化准则为止。

4.轮盘赌简介：  
(1)计算出群体中每个个体的适应度f(i=1，2，…，M)，M为群体大小；  
(2)计算出每个个体被遗传到下一代群体中的概率；



(3)计算出每个个体的累积概率；

（q[i]称为染色体x[i] (i=1, 2, …, n)的积累概率）



(4)在[0，1]区间内产生一个均匀分布的伪随机数r；  
(5)若r<q[1]，则选择个体1，否则，选择个体k，使得：q[k-1]<r≤q[k] 成立；  
(6)重复(4)、(5)共M次

1. **把遗传算法应用于TSP问题：**

1.把TSP问题中的城市转化为一个图模型：每一个城市看作一个节点并给出序号（接下来以10个城市为例，这10个城市被标号为0—9），两个城市间的距离看作图中两节点间的边权重，接下来就可以计算出图的邻接矩阵，由邻接矩阵可以得到任意两个城市间的距离，从而对于任意一个旅行路径，可以求出它的对应路径长度。

2.随机出几个可行解作为种群里的最初一代个体，比如0123456789为城市的遍历顺序，也是一个可行解

3.对种群里的个体进行适应度评价，这里的适应度可以和旅行的路径长度相关，长度越短适应度越高，留下适应度高的个体，淘汰适应度低的个体。使用轮盘赌方法对种群里的个体进行交叉变异操作，得到新一代个体，以固定轮次重复这一步。

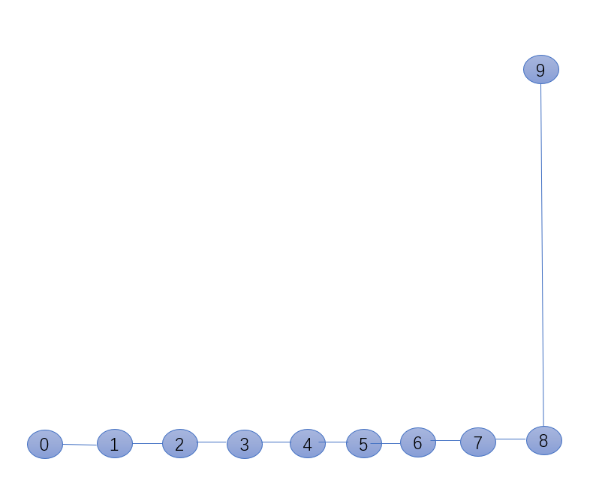
4.在最终的种群中选出适应度最高，即路径最短的解。

5.这种算法得到的解不一定是全局最优解，但当循环轮次足够多时，可以得到代价较小的可行解

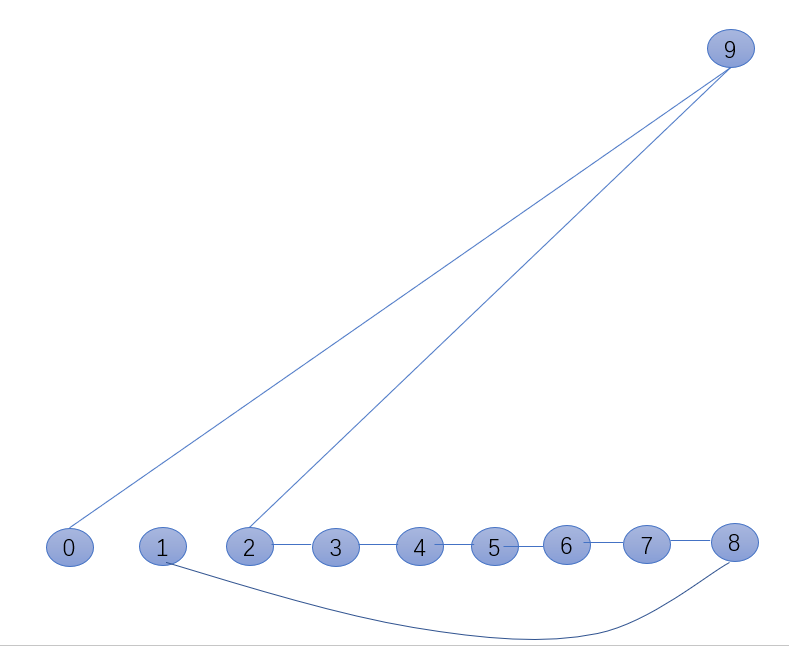
**4）我使用的改进方法：**

因为在TSP问题中可能存在这样一种情况：某个可行解A与全局最优解只有两个基因不同，可能A只经历一次交叉操作就可以达到最优，但是A却因为适应度太低在某一轮循环中产生后就马上被淘汰了，查找最优解就需要更多的循环轮次。举个例子如下图所示：

最优解顺序：0123456789



某个可行解顺序：0923456781



因此改进的方法是，不淘汰低适应度的个体，这样即使适应度低也有一定概率继续交叉变异产生后代。而且父代在产生子代后继续留在种群中，这样做就可以在每轮交叉操作中重复选择某一个个体作为多个子代的父代，相当于增加了并行。

所以经过改进的遗传算法可以更加适应TSP问题，不仅加速了最优解的的查找，还增加了CPU的并行。

1. **实验流程**
2. **实验环境**

操作系统：Windows 10

编程语言：Python

编译环境：Python 3.9.7

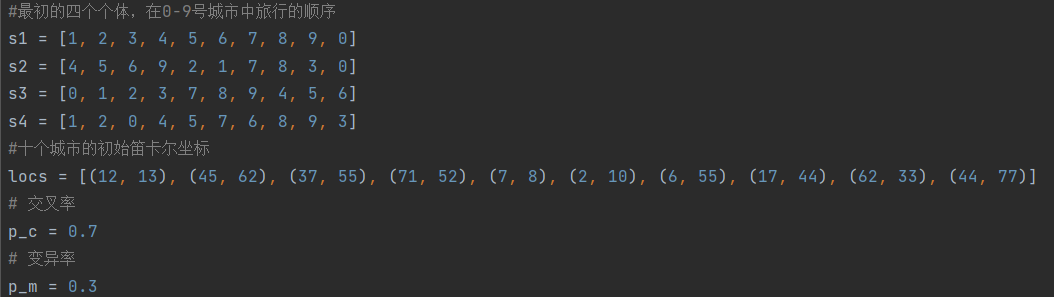
IDE：Pycharm

硬件环境：GTX3060Laptop NVIDIA CUDA 11.4.94 driver

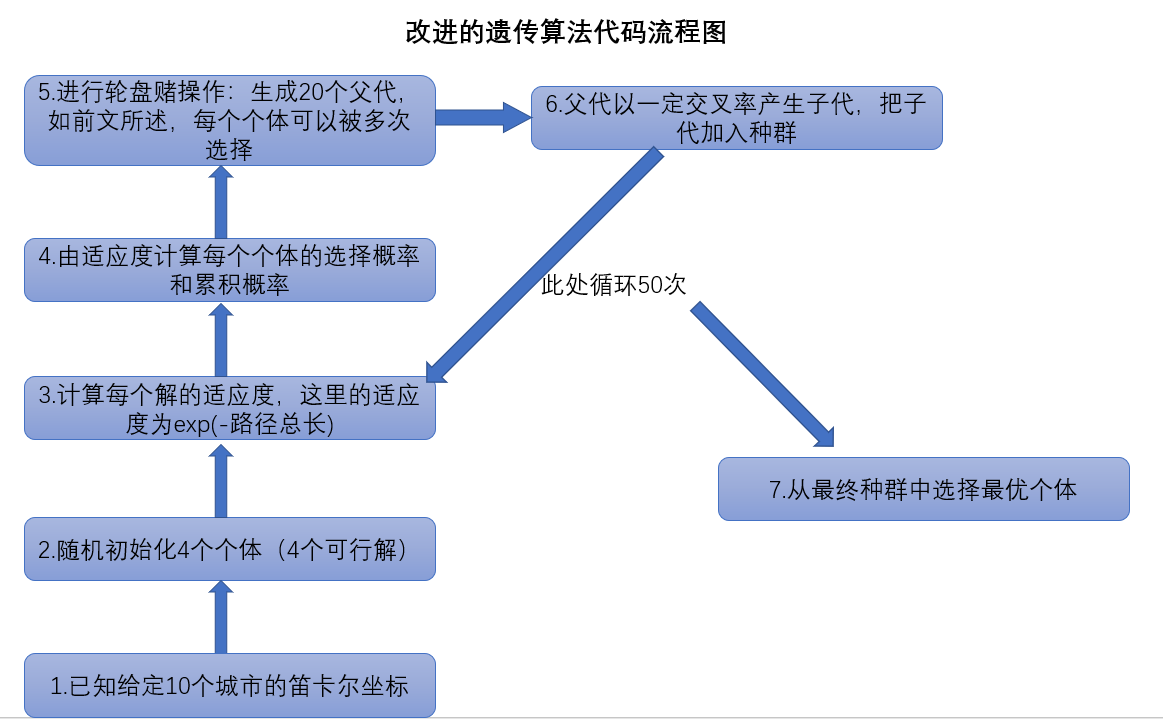
Intel(R) Core(TM) i7-11800H @ 2.30GHz

1. **程序流程介绍**

首先初始种群个体，并对10个城市的笛卡尔坐标进行定义，以此作为程序的输入，如下图所示：



之后改进遗传算法的代码的整体运算逻辑如下图所示，其中步骤3-6被设置成执行50次，其中没有个体被淘汰



其中第3步中取适应度函数为exp(-路径总长)的原因是，这样所有个体适应度都是大于0小于1的数，减小那些适应度大的个体对种群繁衍的支配，相当于对算法增加了一点启发性

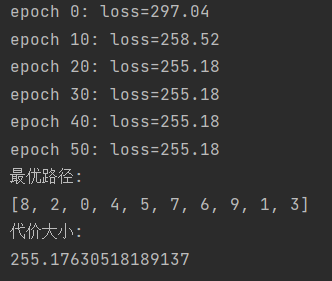
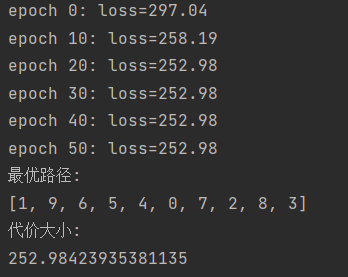
第4步设置了一个二维数组，数组的每一行储存了每个个体的选择概率和累积概率等信息，正如其代码如下图所示

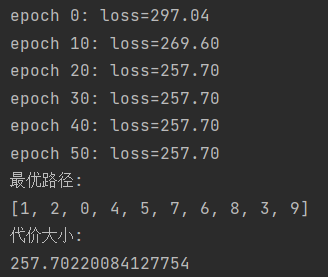
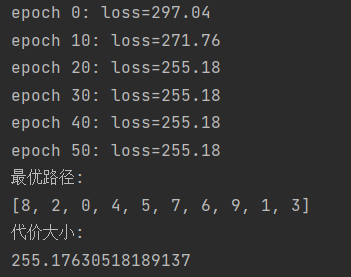


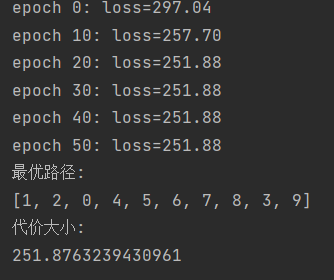
第5，6步通过第4步给出的二维数组进行了轮盘赌产生后代，其中的所用的随机数我没有设置随机数种子，这样可以确保每次产生结果的程序运行路径基本不一样，这样就可以通过观察不同的结果来大致检验算法的准确性。

1. **实验结果及其分析**

我在不改变输入值的情况下，将程序运行了5次，其中得到的结果如下图所示



可以看到因为没有设置轮盘赌的随机数种子，每次的程序运行路径和运行结果确实会有差异，但这5次的结果基本一致，所以说这种改进的遗传算法在极大提高了运算速度的同时，能以较高置信度获得较优的结果。