基于求解tsp问题的多算法集成的进化优化

1. 项目背景和目标：

多算法集成是通过将不同的优化算法组合在一起来提高优化性能的一种方法。本项目旨在设计一个多算法集成的框架，其中包括进化算法、遗传算法、粒子群算法、蚁群算法。用于解决复杂的tsp优化问题。

2. 项目步骤：

2.1 问题定义：

选择tsp问题求解其最短的距离；

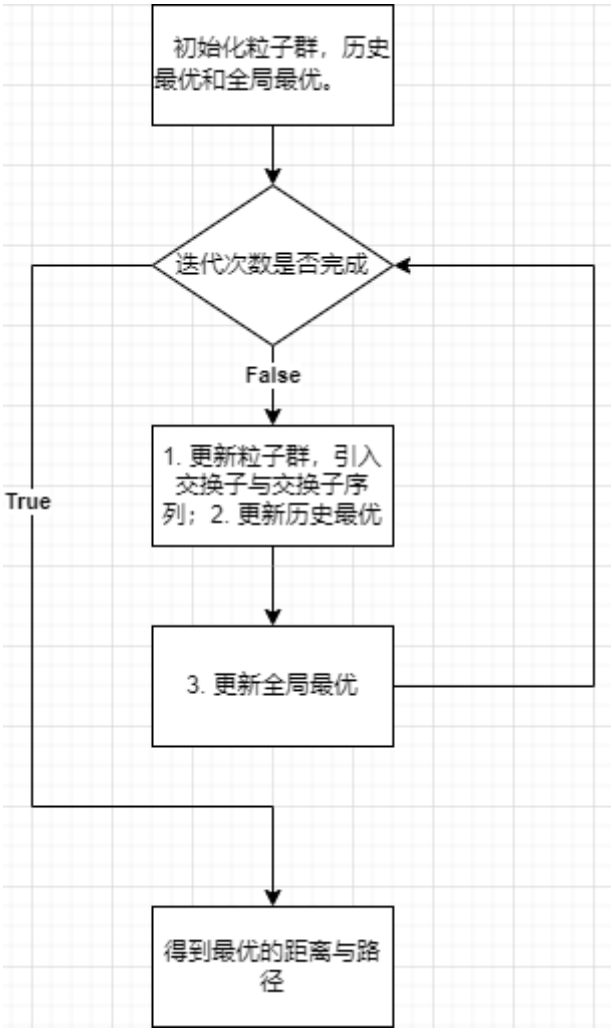
2.2 单算法优化模型设计：

粒子群算法

1. 初始化粒子群，初始化的历史最优和全局最优。
2. 更新粒子群；引入交换子与交换序列来更新粒子群；

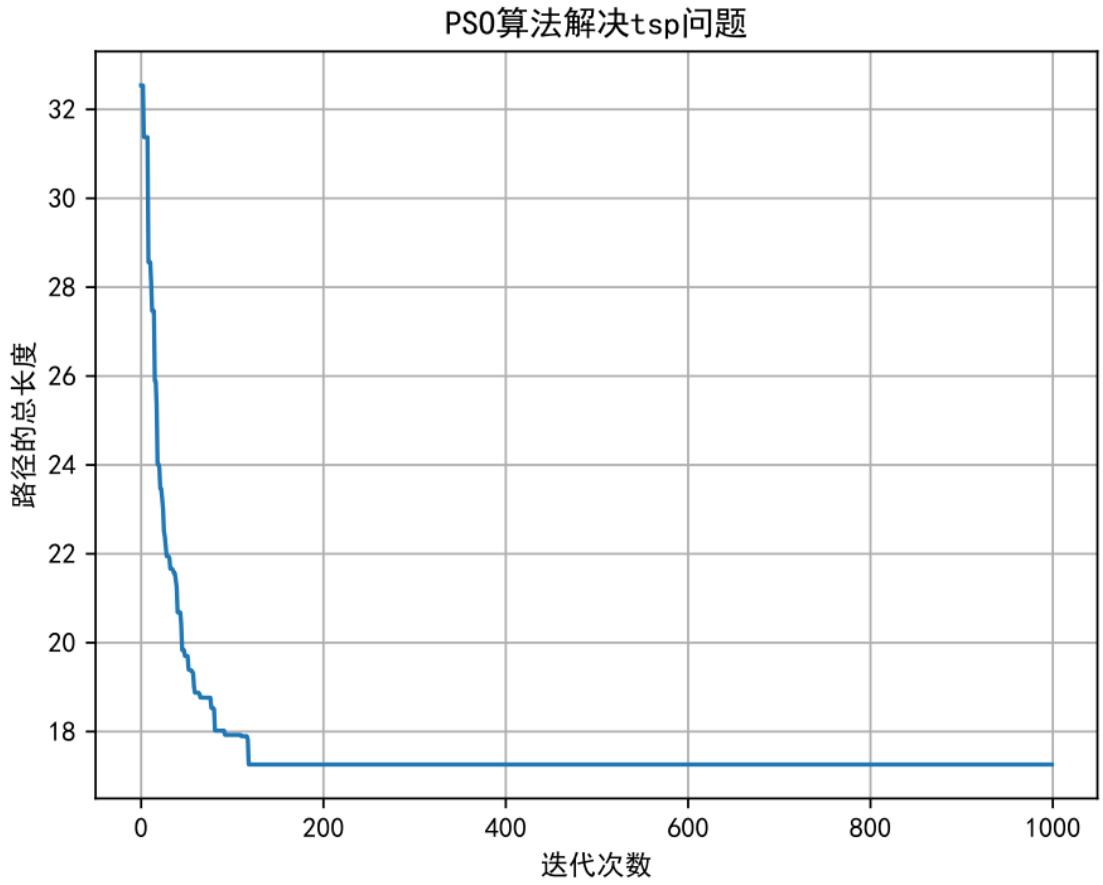
```
ss1=getss(x_i,x_best,r1)
ss2=getss(x_i,x_best,r2)
#ss1与ss2分别是x_i到x_best的交换过程
ss=ss1+ss2 #ss1与ss2分别是([i,j,r],.....)(i,j分别是要交换的顺序，r是交换顺序的概率)
```

3. 与历史最优和全局最优比较；得出新的历史最优和全局最优
4. 循环往复；



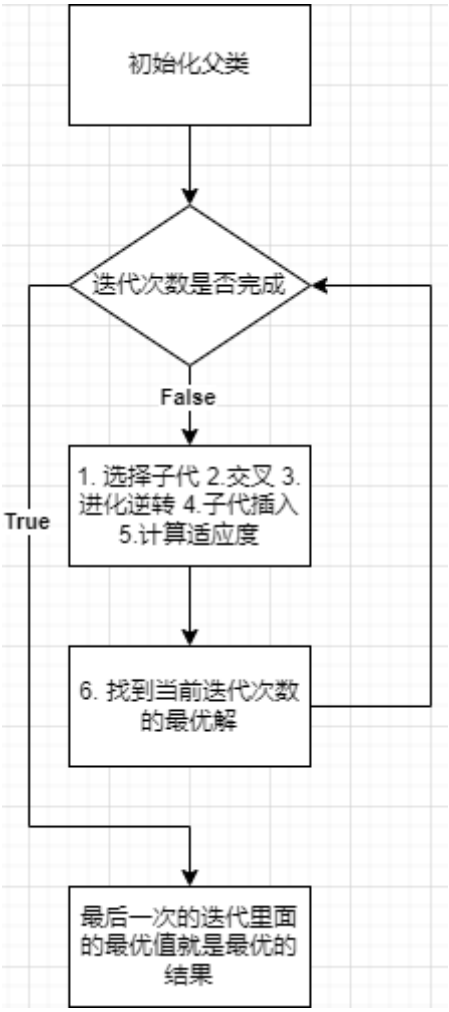
- 算法的效果：

```
global_best_value= 18.661804205984165
global_best_position= [16 18 15  4 22 29 30 28 14  0 26 27 25 20 21  2 19 24 23
10 11 13 12  5
  6  1  3  9  8  7 17]
  pso_tsp.pdf
```

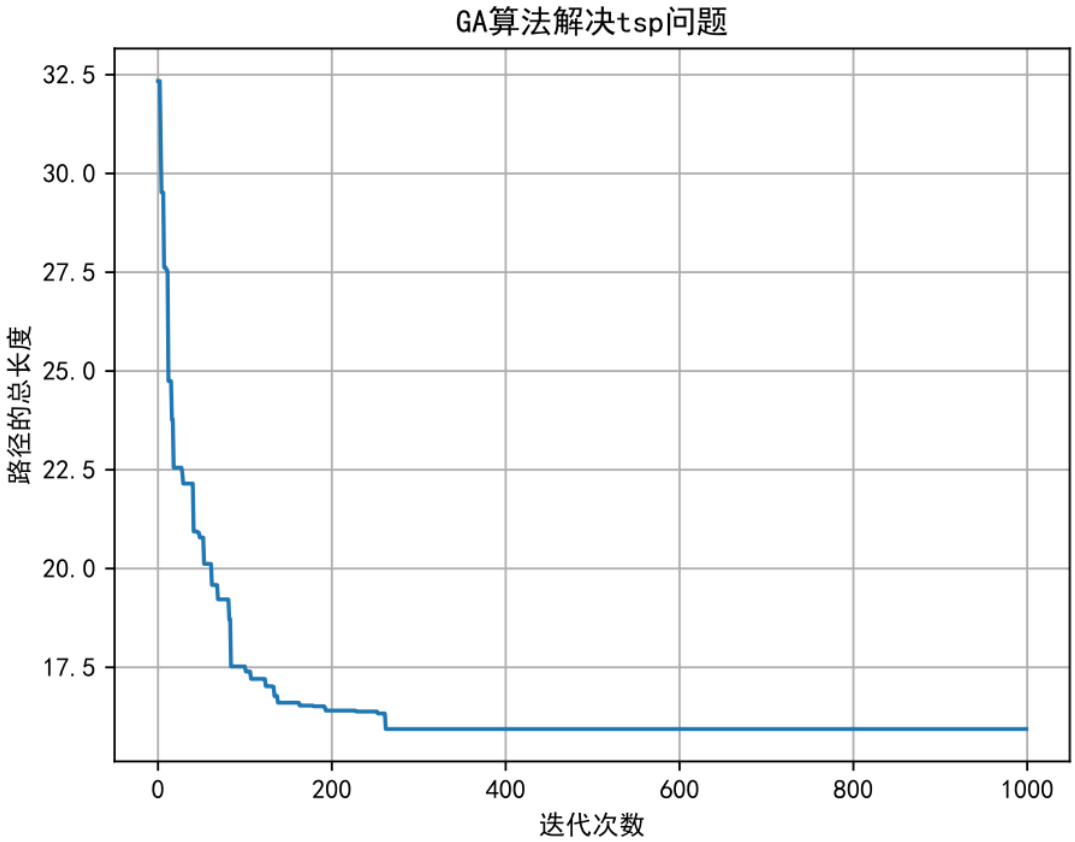


遗传算法

1. 初始化父类
2. 选择子代: 子代选取，根据选中概率与对应的适应度函数，采用随机遍历选择方法
3. 交叉: 依概率对子代个体进行交叉操作
4. 变异: 在变异概率的控制下，对单个染色体随机交换两个点的位置。
5. 进化逆转: 将选择的染色体随机选择两个位置`r1:r2` ，将 `r1:r2` 的元素翻转为 `r2:r1` ，如果翻转后的适应度更高，则替换原染色体，否则不变
6. 子代插入: 子代插入父代，得到相同规模的新群体
7. 计算适应度
8. 迭代；找到最优的路径与距离

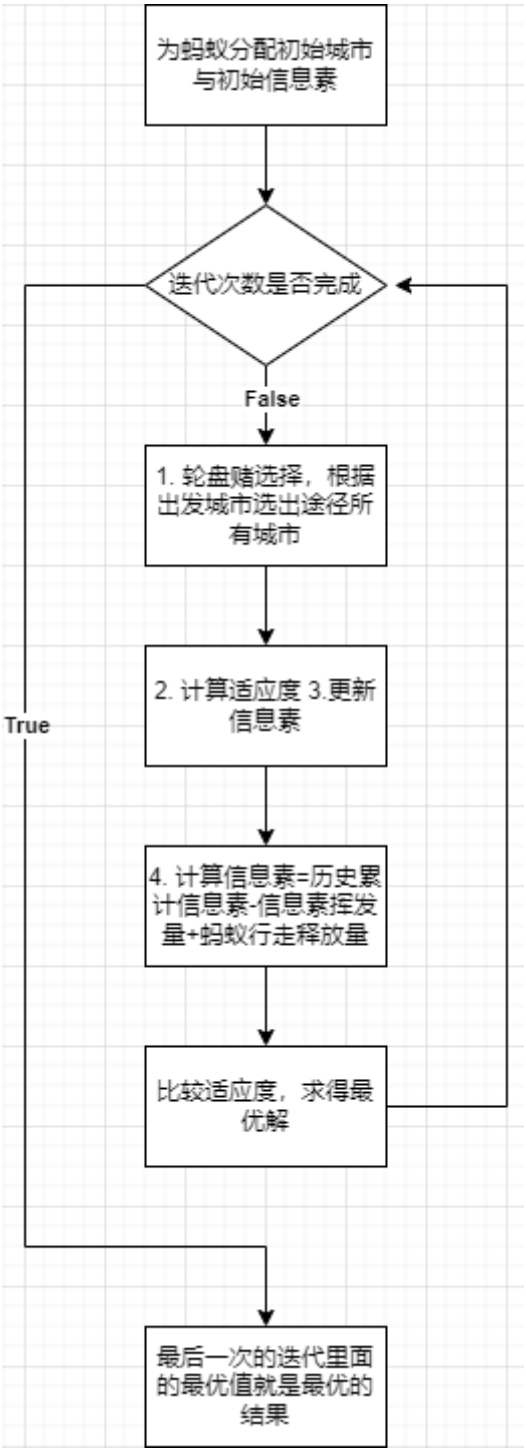



```
tsp_best_value= 15.764572394402471
tsp_best_path= [ 2 21 20 19 24 23 22 10 28 29 25 27 26 30  0 14 13 11 12  6  5  4
3  1
  9  8  7 15 18 16 17]
  ga_tsp.pdf
```

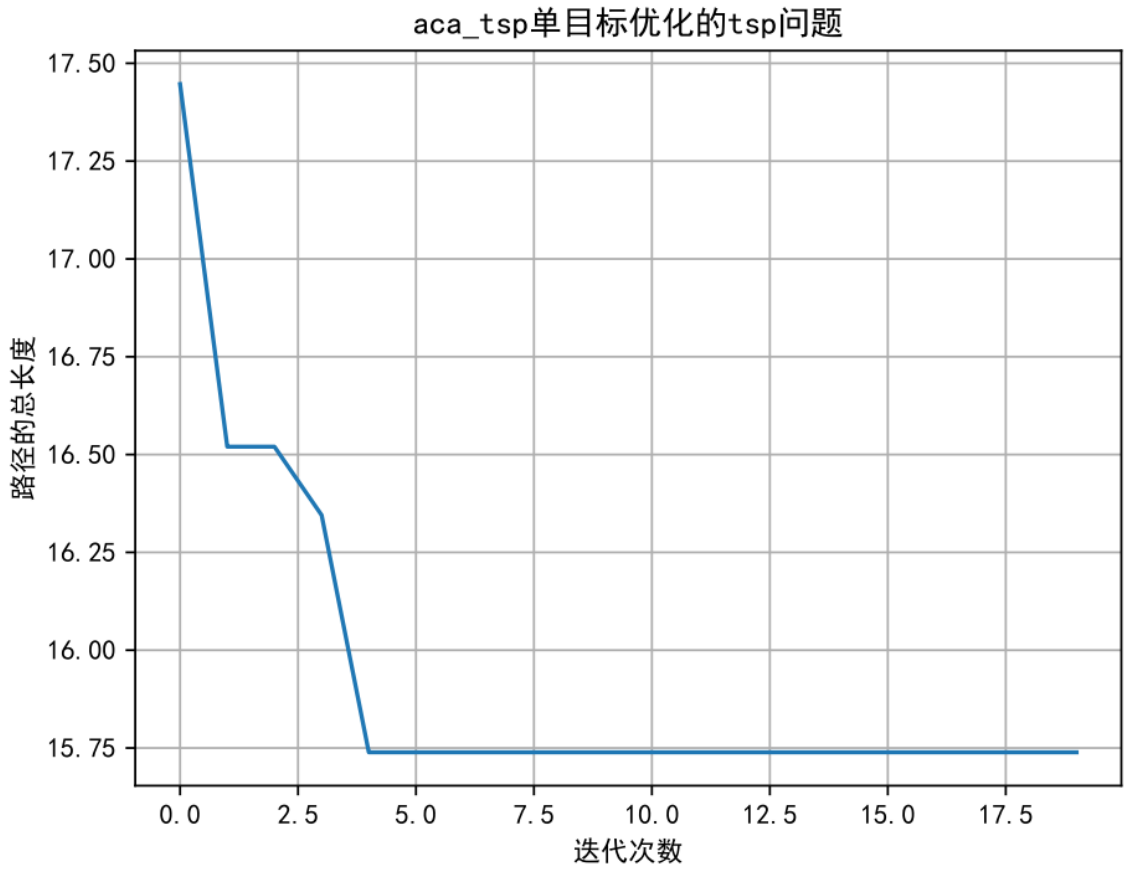


蚁群算法

1. 初始化，为蚂蚁分配初始城市，为蚂蚁初始化信息素；
2. 轮盘赌选择，根据出发城市选出途径所有城市；
3. 计算适应度；
4. 更新信息素
5. 计算信息素=历史累计信息素-信息素挥发量+蚂蚁行走释放量；
6. 迭代；求出最优值路径与解；




```
ACO_value= 15.73904594113959 ACO_path= [14, 0, 30, 28, 29, 26, 27, 25, 24, 23,
19, 20, 21, 2, 16, 17, 18, 15, 4, 5, 6, 1, 3, 7, 8, 9, 22, 10, 12, 11, 13]
```



2.3 总结

从以上的图像与数据可得；

粒子群算法更加容易收敛，因此容易陷入局部最优，效果最差；但是收敛性强

遗传算法不容易陷入收敛并且可深入的搜索，得到最优值。

蚁群算法由于运行时间长，但是它可以快速搜索全局，达到靠近极值的方法；

2.3 多算法集成框架设计：

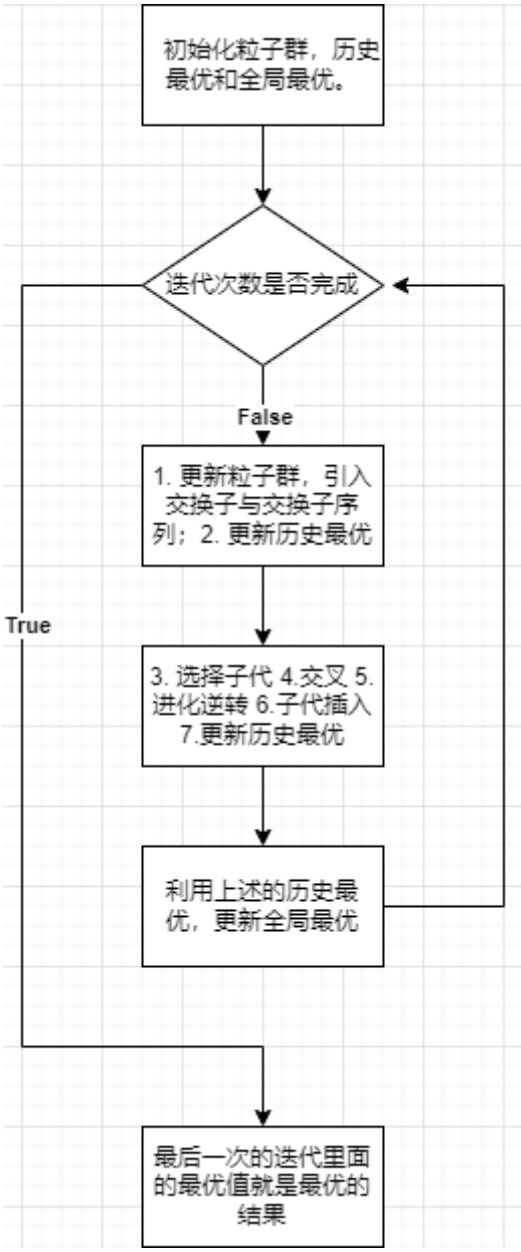
- 设计一个多算法集成框架，可以同时运行并结合多个进化算法的优势，形成一个更强大的优化器。

`GA_PSO`算法

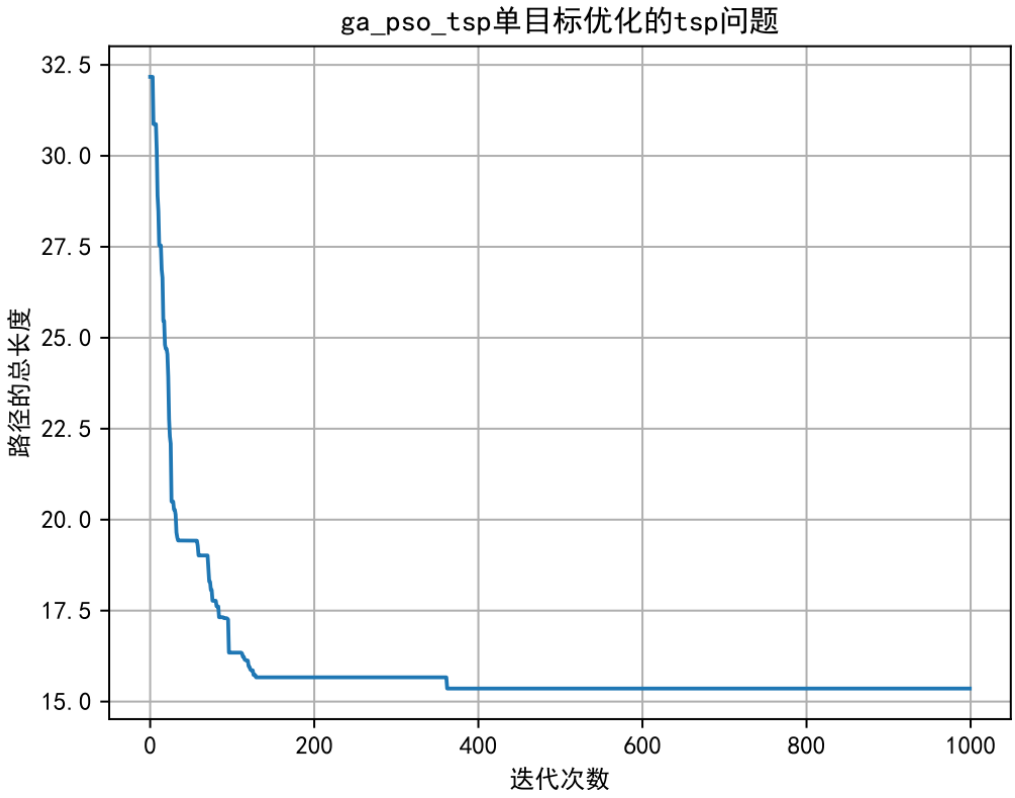
1. 初始化粒子群，历史最优和全局最优
2. 更新粒子群：

```
ss1=getss(x_i,x_best,r1)
ss2=getss(x_i,x_best,r2)
#ss1与ss2分别是x_i到x_best的交换过程
ss=ss1+ss2 #ss1与ss2分别是([i,j,r],.....)(i,j分别是要交换的顺序，r是交换顺序的概率)
```

3. 与历史最优比较；得出新的历史最优
4. 对粒子群进行选择子代，交叉，变异，进化逆转，子代插入的遗传算法的方法；
5. 与历史最优比较；得出新的历史最优
6. 根据上述的两个历史最优求出全局最优的路径与距离；




- ```
  pso_value= 15.358477655757767
  pso_path= [18 17 16  2 21 20 19 23 24 25 27 26 29 30 28  0 14 13 11 10 12  6
  5  4
    3  1  9  8  7 15 22]
  ga_pso1.pdf
  ```
  ```



- 思路

我通过将粒子群算法与遗传算法相结合，使得在遗传算法的基础上更容易收敛；

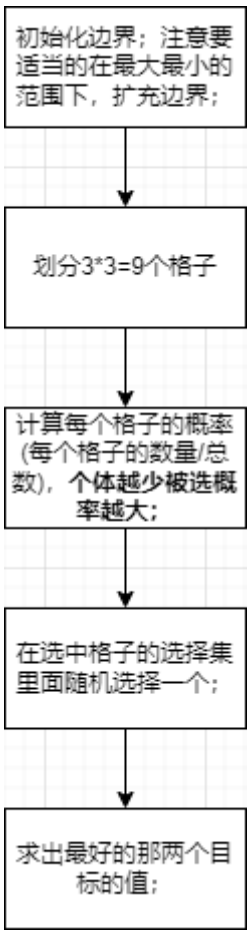
2.4 多目标优化：

- 考虑多目标优化问题，扩展框架以处理多目标优化任务，评估算法在同时优化多个目标时的性能。

在多目标的算法；我采用网格法对多目标进行求解；

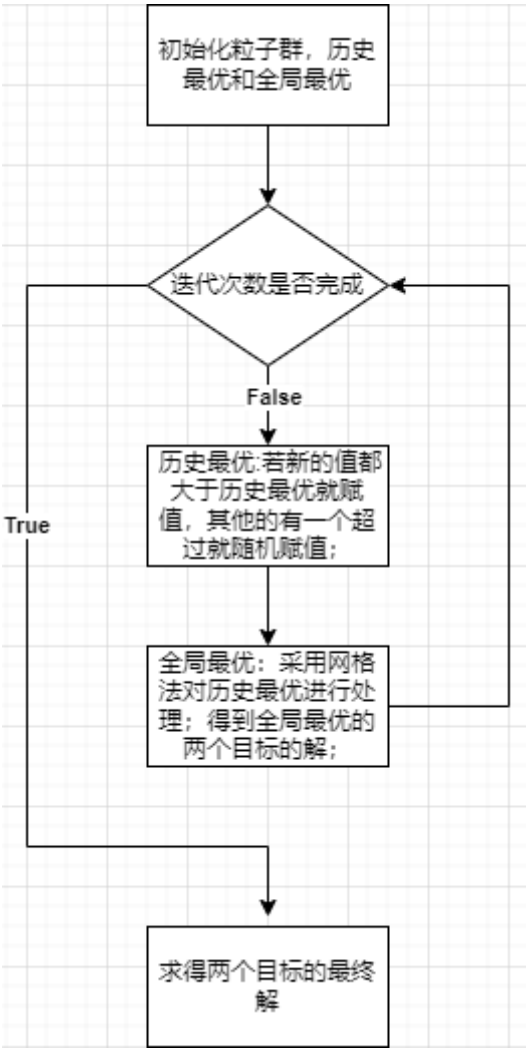
网格法

1. 初始化边界；注意要适当的在最大最小的范围下，扩充边界；
2. 划分3*3=9个格子
3. 计算每个格子的概率(每个格子的数量/总数)；个体越少被选概率越大；
4. 赌盘选择;根据每个格子的概率从9个格子里面选择一个格子;
5. 在选中格子的选择集里面随机选择一个；
6. 求出最好的那两个目标的值；

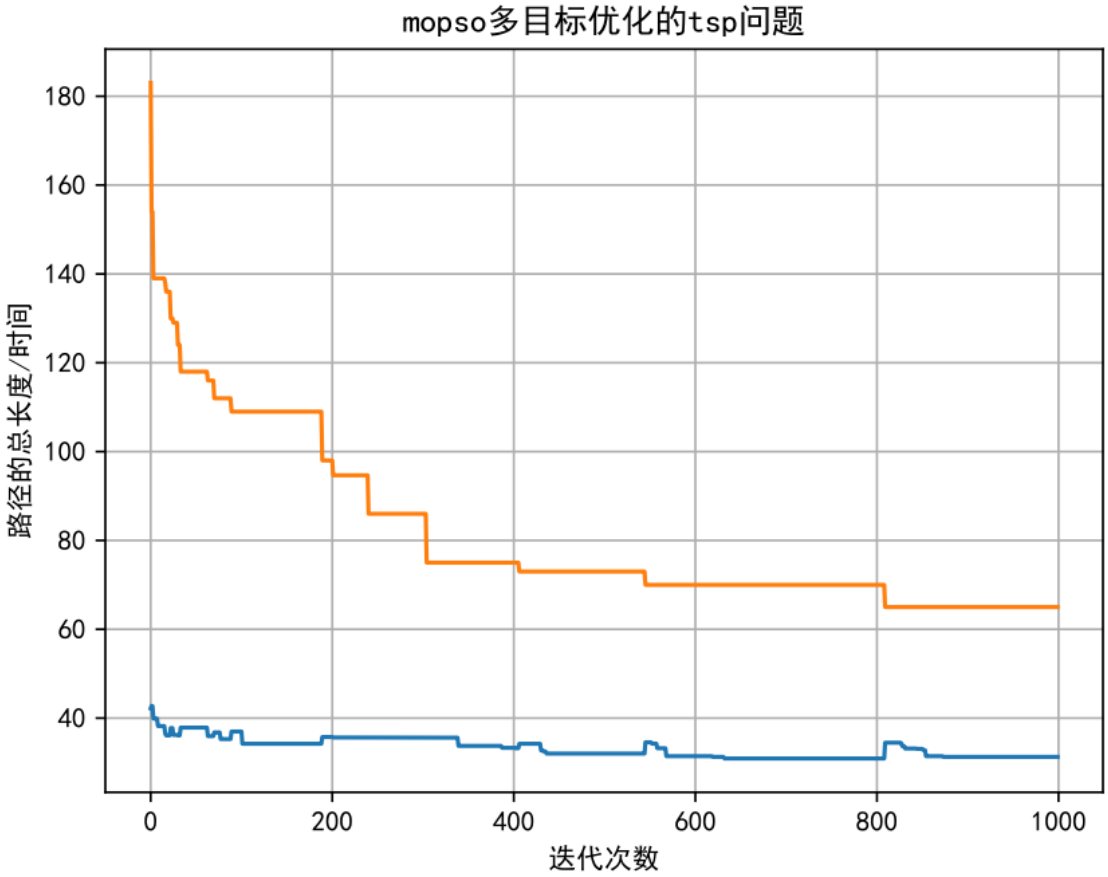


1. MOPSO流程

1. 初始化粒子群，初始化的历史最优和全局最优。
2. 更新粒子群；引入交换子与交换序列来更新粒子群；
3. **历史最优**：若新生成的粒子在两个目标(距离与时间)上都比之前的要好；那就更新。再然后如果其中一个目标比之前的好，那就随机更新。
4. **全局最优**：采用网格法对历史最优进行处理；得到全局最优的两个目标的解；
5. 循环往复；

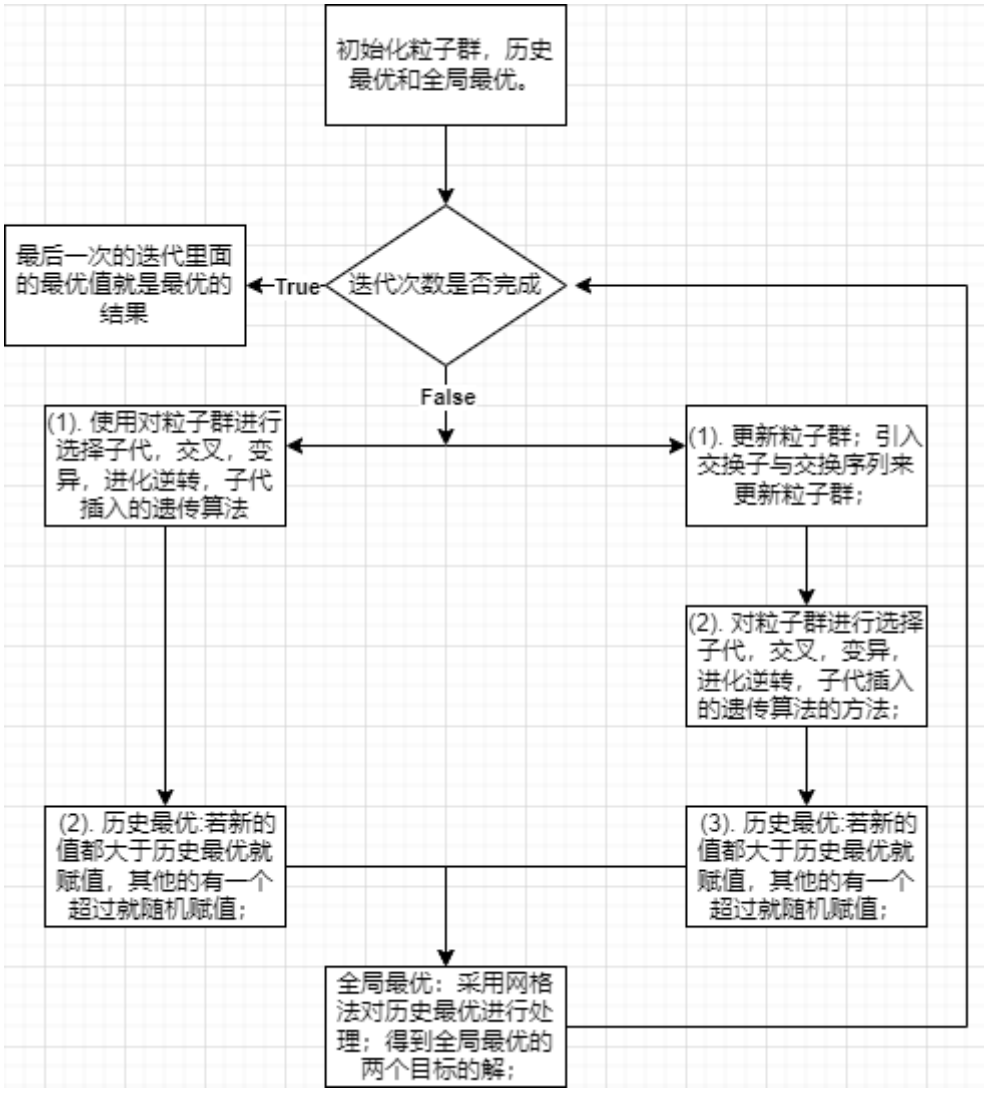



```
gb_path= 30.861929923516833 gb_time= 61.0
gb_position= [24 25  4 20 27 28 22  6  7 23 12 13 11 10 16 30  0 18 29  1 21 17
9  8
 14  5 26 15 19  3  2]
```

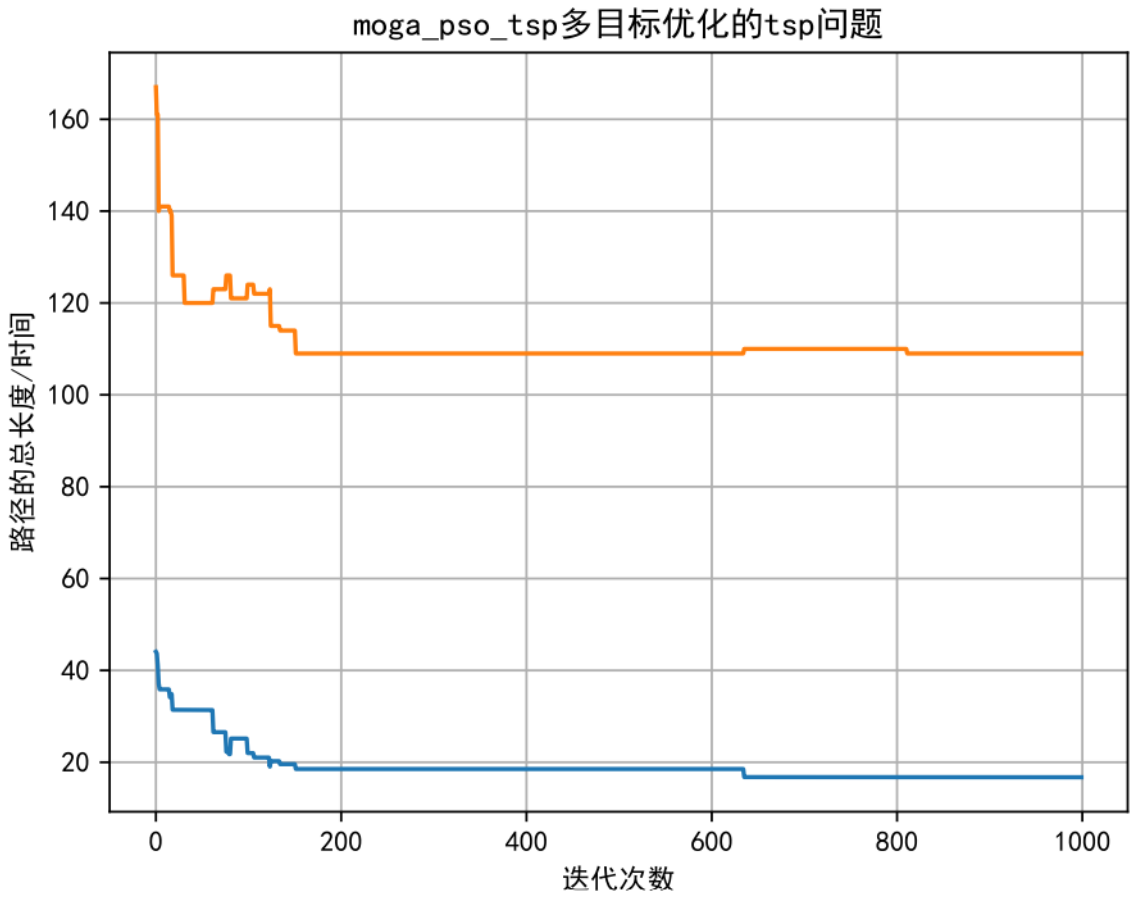


2. MOGA_PSO 流程

1. 初始化粒子群，历史最优和全局最优。
2. 一方面，（1）、直接使用对粒子群进行选择子代，交叉，变异，进化逆转，子代插入的遗传算法的方法；
3. **历史最优**：若新生成的粒子在两个目标(距离与时间)上都比之前的要好；那就更新。再然后如果其中一个目标比之前的好，那就随机更新。
4. 另一方面，（1）、更新粒子群；引入交换子与交换序列来更新粒子群；（2）、对粒子群进行选择子代，交叉，变异，进化逆转，子代插入的遗传算法的方法；
5. **历史最优**：若新生成的粒子在两个目标(距离与时间)上都比之前的要好；那就更新。再然后如果其中一个目标比之前的好，那就随机更新。
6. **全局最优**：采用网格法对历史最优进行处理；得到全局最优的两个目标的解；
7. 循环往复；




```
self.gb_path= 16.738239117665255 self.gb_time 109.0
gb_position= [ 0 14 13 19  4  6 18  1 11 22 10  8 21 12  9 15 17  2  3  7 23 28
16 26
 24 25 29 20  5 27 30]
```



3. 总结

`MOPSO`偏向于时间这一目标，而`MOGA_PSO`偏向于距离这一目标；

2.5 集成效果评估：

- 设计适当的指标来评估多算法集成框架的性能，包括收敛速度、稳健性、以及在不同问题上的适应性。

参考文献

【建模算法】基于粒子群算法求解TSP问题（Python实现）_python粒子群算法实现tsp-CSDN博客

超详细 | 遗传-粒子群自适应优化算法及其实现(Matlab)_自适应遗传粒子群算法-CSDN博客

Matlab_一文搞懂MOPSO多目标粒子群算法 - 知乎 (zhihu.com)