实验报告-实验一

摘要

在本次实验中，我们小组实现了模拟退火算法和遗传算法，并且进行了多个对比实验。模拟退火实验结果显示了区间翻转邻域操作在寻找 TSP 解上具有一定的优越性；并且结果显示，采取单一邻域操作的效果比不上结合先验知识，依据概率选取邻域操作的混合邻域操作。

关键字： 模拟退火　遗传算法　混合邻域

一、 导言

在 TSPLIB[1]中选一个大于 100 个城市数的 TSP 问题，采用模拟退火算法和遗传算法求解。模拟退火算法要求如下：

1. 采用多种邻域操作的局部搜索 local search 策略求解；
2. 在局部搜索策略的基础上，加入模拟退火 simulated annealing 策略，并比较两者效果；
3. 要求求得的解不要超过最优值的 10%，并能够提供可视化。

遗传算法要求如下：

1. 设计较好的交叉操作，并且引入多种局部搜索操作；
2. 和之前的模拟退火算法（采用相同的局部搜索操作）进行比较；
3. 得出设计高效遗传算法的一些经验，并比较单点搜索和多点搜索的优缺点。

二、 实验过程-模拟退火算法

2.1 多种邻域操作的局部搜索

局部搜索算法是在一组可行解的基础上，在当前解的领域内进行局部搜索产生新的可行解的过程。局部搜索是解决最优化问题的一种启发式算法。对于某些计算起来非常复杂的最优化问题，比如各种 NP 完全问题，要找到最优解需要的时间随问题规模呈指数增长，因此诞生了各种启发式算法来退而求其次寻找次优解，是一种近似算法（Approximate algorithm），基于以时间换精度的思想。局部搜索就是其中的一种方法。

[1] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

2.1.1 多种邻域操作

参考文献 [1] 中的多种启发式邻域操作，本次实验中我们小组所使用的邻域操作有：

（1）两点交换：

本邻域操作的动作为将路径中的随机两点进行交换操作：

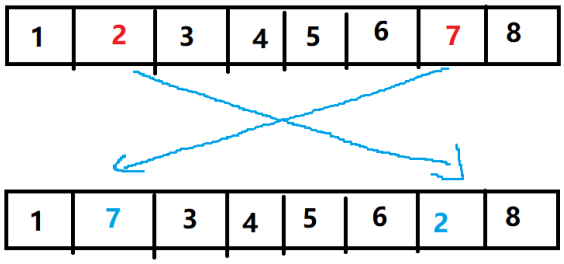


图 **1**　两点交换

（2）区间翻转：

本邻域操作的动作为将路径中的随机区间进行翻转操作：

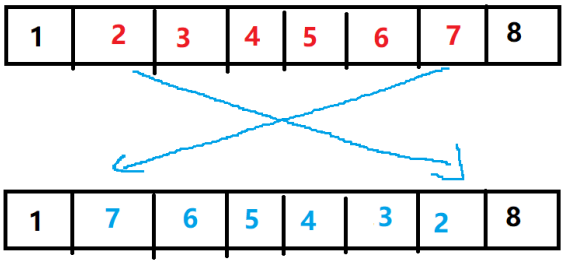


图 **2**　区间翻转

（3）随机置顶：

本邻域操作的动作为随机选取路径中的两点并顶置，其余点顺序不变：

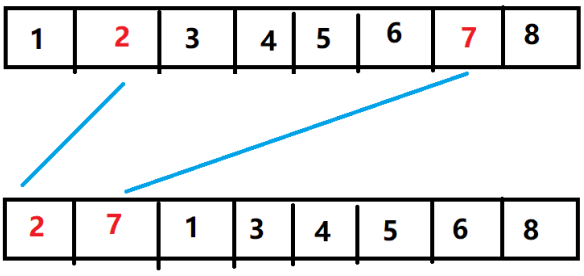


图 **3**　随机置顶

2.1.2 实验设置及结果

随机初始化解，采用早停法，为了与模拟退火对标，局部搜索同样设置内循环，循环次数为解长度的 10 倍，早停法容忍次数设置为 50 次。对不同的邻域操作进行实验，取 5 次随机运行后所得的最好解、最差解、平均值、标准差作为指标显示如下，百分比为相对于最优解的百分比：

表 **1**　局部搜索实验结果

局部搜索	最好解	最差解	平均值	标准差
两点交换	45.38%	66.42%	58.15%	8.26%
区间翻转	**13.47%**	**19.41%**	**15.55%**	**2.23%**
随机置顶	149.45%	181.80%	159.88%	12.36%

从结果可以得到，采用区间翻转邻域操作的局部搜索效果最优，最优解与目前最优值的差值达到了 13.47%，其余邻域操作的效果相比区间翻转操作有大幅度的下降。绘制区间翻转的最优解如下所示：

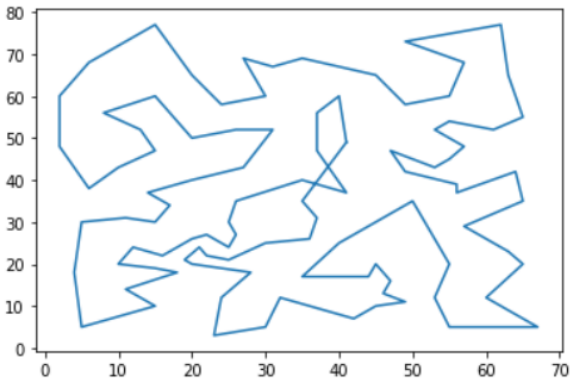
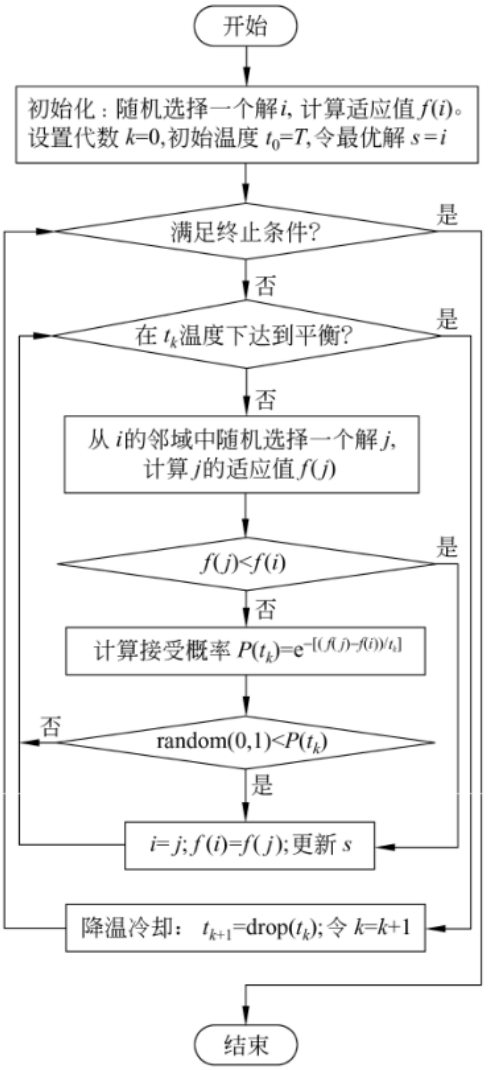


图 **4**　采用区间翻转邻域操作的局部搜索最优解

2.2 模拟退火

模拟退火相比局部搜索优越的地方在于其能够通过接受劣解来达到最终获得更优解的效果。其接受劣解的概率由 Metropolis 准则决定，通过劣解与目前最优解目标值的距离以及当前温度来决定接收劣解的概率，模拟退火的流程图和伪代码图5所示。

]




```
//功能：模拟退火算法伪代码
//说明：本例以求问题最小值为目标
//参数：T为初始温度;L为内层循环次数
procedure SA
  //Initialization
  Randomly generate a solution X0, and
  calculate its fitness value f(X0);
  Xbest=X0; k=0; tk=T;
  while not stop
    //The search loop under the temperature tk
    for i=1 to L // The loop times
       Generate a new solution Xnew based on
       the current solution Xk, and calculate
       its fitness value f(Xnew).
      if f(Xnew)<f(Xk)
        Xk=Xnew;
        if f(Xk)<f(Xbest) Xbest=Xk;
        continues;
      end if
      Calculate P(tk)=e^-[f(Xnew)-f(Xk)/tk];
      if random(0,1)<P
          Xk=Xnew;
      end if
    end for
    //Drop down the temperature
    tk+1=drop(tk);k=k+1;
  end while
  print Xbest
end procedure
```

图 **5**　模拟退火的流程图和伪代码

2.2.1 实验设置及结果

随机初始化解，采用早停法，设置内循环次数为解长度的 10 倍，早停法容忍次数设置为 50 次。对不同的邻域操作进行实验，取 5 次随机运行后所得的最好解、最差解、平均值、标准差作为指标显示如下，百分比为相对于最优解的百分比：

表 **2**　模拟退火实验结果

模拟退火	最好解	最差解	平均值	标准差
两点交换	13.66%	25.04%	17.97%	**3.90%**
区间翻转	**6.08%**	**29.28%**	**16.06%**	8.10%
随机置顶	119.33%	141.59%	125.93%	8.15%

从结果可以得到，与局部搜索结果类似，采用区间翻转邻域操作的效果最优，最优解与目前最优值的差值达到了 6.08%，其余邻域操作的效果相比区间翻转操作有些微的下降，但是相比与局部搜索可得，模拟退火使得效果有了明显的提升。绘制区间翻转的最优解如下所示：

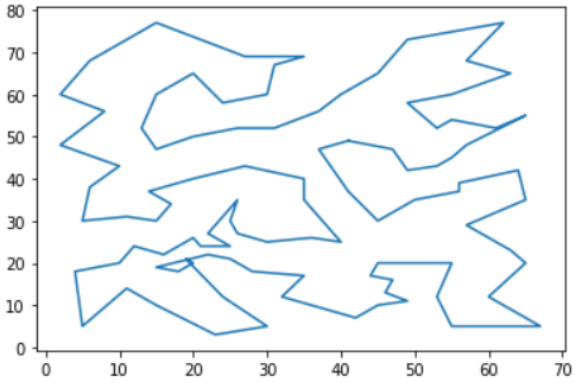


图 **6**　采用区间翻转邻域操作的模拟退火最优解

2.3 改进：混合邻域操作

根据上述结果可得，采用区间翻转的效果相比使用其他的邻域操作效果更好，但是考虑到仅一种邻域操作对解的搜索强度不如随机选取邻域操作，因此结合先验知识，采用混合邻域操作，选取两点交换、区间翻转、随机置顶邻域操作的概率为 [0.01, 0.98, 0.01]，随机选取可以简单地通过 numpy 库的 choice 函数进行实现：

```
func = np.random.choice([self.neighbour_swap, self.neighbour_two_opt_swap,
    self.neighbour_two_h_opt_swap], p=[0.01, 0.98, 0.01])
```

LS 代表局部搜索、SA 代表模拟退火，实验结果对比为：

表 **3**　实验结果

方案	最好解	最差解	平均值	标准差
两点交换-LS	45.38%	66.42%	58.15%	8.26%
区间翻转-LS	13.47%	19.41%	15.55%	2.23%
随机置顶-LS	149.45%	181.80%	159.88%	12.36%
混合邻域-LS	11.25%	15.52%	13.63%	**1.52%**
两点交换-SA	13.66%	25.04%	17.97%	3.90%
区间翻转-SA	6.08%	29.28%	16.06%	8.10%
随机置顶-SA	119.33%	141.59%	125.93%	8.15%
混合邻域-SA	**4.15%**	**16.48%**	**11.24%**	4.76%

从结果可以得到，采用混合邻域操作的模拟退火算法效果最优，达到了 4.15%。采用混合邻域操作后，不管是局部搜索还是模拟退火算法的效果都有显著的提升，显示了混合邻域操作的优越性。绘制最终的得到的最优解如下所示：

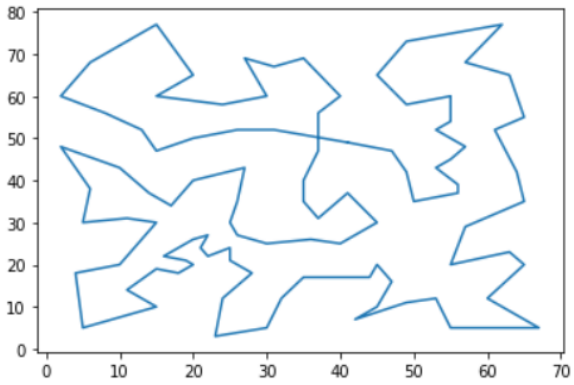


图 **7**　最终得到的最优解

三、 结论-模拟退火算法

在本次实验中，我们小组实现了局部搜索和模拟退火算法，并且就多种邻域操作进行了对比。实验结果显示了区间翻转邻域操作在寻找 TSP 解上具有一定的优越性；并且结果显示，采取单一邻域操作的效果比不上结合先验知识，依据概率选取邻域操作的混合邻域操作；并且混合邻域操作对于局部搜索和模拟退火两个算法的效果都有提升。

四、 实验过程-遗传算法

遗传算法是用于解决最优化的搜索算法，借鉴了进化生物学中的一些现象而发展起来，这些现象包括遗传、突变、自然选择以及杂交。现代遗传算法把每个个体抽象为染色体，使种群向更好的解进化。一般包括以下几个环节：选择、交配池中交叉、变异、交叉产生新后代、新选择，然后一直循环直到满足终止条件。算法框架如下

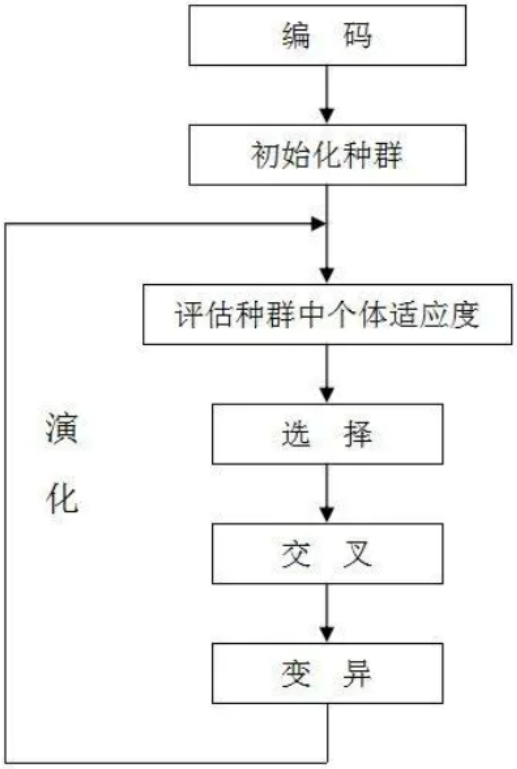


图 **8** 遗传算法流程

本次实验中，我们小组从各个环节进行设计，遗传算法取得比模拟退火算法更优的结果。

4.1 个体的编码

tsp 问题中，设有 N 个城市，视每个城市为一个数字，那么 1 到 $N-1$ 的数字的乱序组合就是一个可能的解，假设这个组合是 $ABC...N$，那么去到城市 A 后下一步去到城市 B，再去到城市 C... 最后城市 N 再去到城市 A，就得到了一条回路，这是 tsp 问题的一个可能最优解。

4.2 贪婪算法的遗传池初始化

传统遗传算法完全从随机的初始种群开始，所以需要很长的时间才能交配出正确解。但在 tsp 问题中，采用贪婪方式进行初始化，已经能够得到一个不错的个体。这些个体前半部分的基因组合十分贴近最优解，在循环的一开始就引入这些基因片段无疑是十分有利的。

算法简要描述如下，对于 N 个点，任选 1 个点 A，然后，在剩余 $N-1$ 个点中找到距离 A 最近的点 B，那么就认为取到点 A 后向点 B 走。之后将 A 从 N 个点剔除，然后寻找点 B 在剩余 $N-2$ 个点中距离最近的点 C，以此类推。这样就得到一个 $ABC...N$ 的初始个体，由于每次起始点 A 都是随机选取的，遗传池中的个体也各有不同，保证了一定的随机性。

贪婪算法虽然有效，但也存在性能的问题，以上描述的算法是 $O(n^2)$ 的，为了找到每个点最近的点，需要遍历所有其他点。为此我们引入了阈值，使得算法早些停下来，在算法中，如果找到两个点之间的距离小于 d，那就视为找到该点最近的点，经测试，d 取为横纵坐标最大跨度的 1% 效果最好。

4.3 PMX 交叉

对于个体的交叉方式，我们使用 PMX 算法，这个算法对于每个个体都寻找随机的一段，然后将两端交换得到两个后代，之后建立起两端之间的数字关联，在后代中补充因为交换缺少的数字，如下图所示:

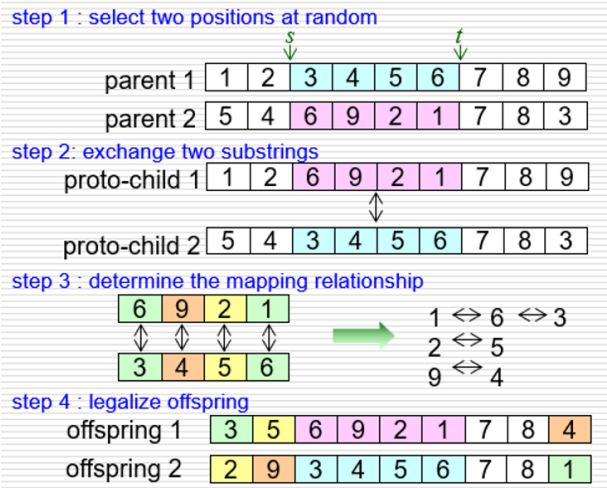


图 **9**　**PMX** 算法图解

这对于 tsp 问题是最优异的一种交叉方式，这是因为它保留了完整的基因片段。在贪婪初始化的过程中，我们得到了很好的起始基因，这种交叉方式恰好能够保留这些基因。所以这种交叉方式很适合我们的遗传算法，在实际的实现中，我们对这个算法进行

了简单的改进，比起随机选择两段基因，我们先随机选出一段基因，然后在另一个个体中找到开头相同的另一端基因。这样做，替换的基因大概率是相近的，能够最大概率的用好的基因替换坏的基因。

4.4 变异方式

常用的变异方式也是前面提到的局部搜索，但我们最终在遗传算法中使用更有效的方式。这种方式是随机顶置的变体。每次变异中，随机选取一个点 A，然后寻找距离 A 最近的点 B，直接把 B 插入到点 A 的后方。采用这种方法最大的好处是结果不会变坏，选取的 B 有两种情况：第一种是 B 本身就在 A 的后方，此时变异个体与原个体一致；第二种是 B 原本不在 A 的后方，设原本个体为 $AC...DBE$，变异后个体变为 $ABC...DE$，那么路径中减少了 AC、DB、BE 增加了 AB、BC、DE，由于三角形性质 $DB + BE \geq DE$，并且 $AC \geq AB$，只要 BC 较小，进行这样的变异就总是更好的。

4.5 选择

常见的选择方式有轮盘赌选择、Boltzmann 选择、繁殖池选择等方法。但经过测试这些方法的效果都差不多，要使遗传算法效果最好，核心还是放在交叉和变异上。所以最终我们采用随机选择法，我们维护大小为 15 的繁殖池，然后每次交叉任意选择两个个体，重复 20 次，取最好的个体。变异的选择也类似，重复变异 20 次，选择其中最好的后代。繁殖池每次都保留最好的 15 个个体，需要注意将父代从池中提出，不然繁殖池会高度相似，个体失去多样性

五、 结果分析-遗传算法

遗传算法也选择 eil101 问题进行求解，迭代 5000 次。

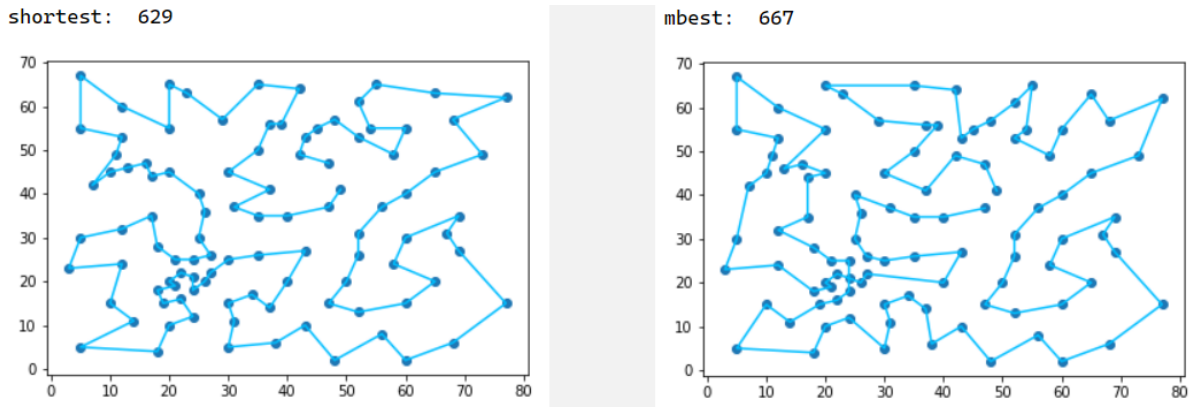


图 **10** **eil101** 最优解和我们的解对比

此时得到的最好解与最优解相差 6%，虽然略弱于模拟退火算法。但在运行时间上，遗传算法只需要 30s，而模拟退火算法需要 10min，使用遗传算法远远快于遗传算法。使用遗传算法得到更优的效果，这是各个环节都优化组合的结果。

本算法还有可以改进的地方，在课堂分享中，有的小组采用多种变异方式混合的方法，取得了更好的结果，这是十分值得尝试和探索的思路。

六、 结论-遗传算法

在本次实验中，我们对 tsp 问题进行了模拟退火算法和遗传算法求解的探索。在遗传算法的设计中，我们设计了较好的交叉操作，并测试了多种局部搜索操作，采用了最优的变异方式。与模拟退火算法比较过程中，我们发现，精心设计的遗传算法能够以极快的速度取得接近模拟退火算法准确率的可接受解。

多点搜索对比单点搜索有一个很大的优势，多点搜索可以更快的速度收敛。因为多点搜索总是能很快的试错，然后经过反复随机搜索，探索更广的搜索空间，找到其中一个最优解，这点单点搜索比不上。但单点搜索允许更细致的搜索，单点搜索虽然比较慢，但是只要给出足够的时间，就能得到更精确的解。我们的实验中也验证了这一点：模拟退火算法速度慢，但是得到更精确的解。

经过这次实验，我们更加深入理解了模拟退火算法和遗传算法，也对于经典 tsp 问题积累了一些经验。

参考文献

[1] Lin S . Computer solutions of the traveling salesman problem[J]. Bell Labs Technical Journal, 1965, 44(10):2245-2269.

[2] Yang W H, Mathur K, Ballou R H. Stochastic vehicle routing problem with restocking[J]. Transportation Science, 2000, 34(1): 99-11

[3] Potvin J Y, Rousseau J M. An exchange heuristic for routeing problems with time windows[J]. Journal of the Operational Research Society, 1995, 46(12): 1433-1446.