目录

1 问题描述 **2**
1.1 类自然语言描述 . . . 2
1.2 一种形式化描述 . . . 2

2 研究现状与对比算法 **3**
2.1 非随机近似算法 . . . 3
2.1.1 最近邻点算法 . . . 3
2.1.2 克里斯托菲德斯算法 . . . 4
2.1.3 2-OPT 改进算法 . . . 7
2.2 随机型近似算法 . . . 9
2.2.1 王磊算法 . . . 9
2.2.2 模拟退火 . . . 10

3 遗传算法及改进策略 **11**
3.1 传统的遗传算法 . . . 11
3.2 改进的遗传算法 . . . 12

4 实验设置与测试结果 **14**
4.1 数据集与超参数 . . . 14
4.2 实验结果 . . . 14

5 结束语 **15**

求解旅行商问题的拟物拟人算法研究

杜睿

摘要

旅行商问题是一个典型的 NP 难度问题，虽易于描述但无法在多项式时间内求得最优解。近年来，国内外研究者设计各种近似算法（尤其是进化算法）期望求解该问题。

对于组合优化问题，有两条主线。第一条是如何表达可行解与解空间，语义（表现型）和存储（基因型）可以有所不同。第二条是如何平衡局部搜索与跳坑策略，平衡开采与探索：如果开采不足，收敛性不好；如果探索不够，容易早熟，陷入局部最优解。

本文提出了改进的遗传算法用于求解旅行商问题：在种群的初始化阶段发扬“继承”策略，减少迭代次数并保留种群多样性；在变异部分，在 K-OPT 的基础上，设计了一种基于“贪婪插入”的算子；同时，在选择操作中弃用轮盘赌方法，改用排位等级法。

选取城市数小于等于 1000 中全部 48 个 benchmark 测试用例对算法进行测试，每个实例计算 10 次。实验表明，提出的算法在求解质量和求解速度上优于所对比的算法。

1 问题描述

1.1 类自然语言描述

给定 n 个城市，对这 n 个城市中的每两个城市来说，从一个城市到另一个城市所走的路程是已知的正实数（符合三角形三边关系定则），其中 n 是已知的正整数，$n \geq 3$。这 n 个城市的全排列共有 $n!$ 个。每一个这 n 个城市的全排列都恰好对应着一种走法：从全排列中的第一个城市走到第二个城市，$\cdots$，从全排列中的第 $n-1$ 个城市走到第 n 个城市，从全排列中的第 n 个城市回到第一个城市。要求给出一个这 n 个城市的全排列 σ，使得在 $n!$ 个全排列中，全排列 σ 对应的走法所走的路程是最短的（严格来讲，由于起点任意、顺逆时针等价，问题复杂度为 $\frac{(n-1)!}{2}$）。

1.2 一种形式化描述

给定一个有向完全图 $G = (V, A)$，其中集合 $V = \{v_1, \ldots, v_n\}$ 是顶点集合，每个顶点代表一个城市，n 是顶点数（$n \geq 3$），集合 $E = \{(v_i, v_j) | v_i, v_j \in V, v_i \neq v_j\}$ 是有向边集合。

c_{ij} 是有向边 (v_i, v_j) 的长度（权值），c_{ij} 是已知的正实数，其中 $(v_i, v_j) \in E$。集合 Σ 是顶点全排列的集合，共有 $n!$ 元素。σ 是所有顶点的一个全排列，$\sigma = (\sigma(1), \ldots, \sigma(n)), \sigma \in \Sigma, \sigma(i) \in V\ 1 \le i \le n$ 对应着一条遍历所有顶点的回路：从顶点 $\sigma(1)$ 走到顶点 $\sigma(2)$，$\cdots$，从顶点 $\sigma(n-1)$ 走到顶点 $\sigma(n)$，从顶点 $\sigma(n)$ 回到顶点 $\sigma(1)$。

全排列 σ 所对应的回路的长度记为 $L(\sigma)$，$L(\sigma) = \sum_{i=2}^{n} c_{\sigma(i-1)\sigma(i)} + c_{\sigma(n)\sigma(1)}$。

目标是给出所有顶点的一个全排列 $\sigma^\star$，使得 $L(\sigma^\star) = \min_{\sigma \in \Sigma} L(\sigma)$。

每一对顶点 v_i 和 v_j 来说，都有 $c_{v_i v_j}$ 成立，那么称问题是对称的（Symmetric traveling salesman problem）；否则称问题是非对称的（Asymmetric traveling salesman problem）。

2 研究现状与对比算法

求解旅行商问题的算法大体可分为两类：确切算法和近似算法。

1. 确切算法保证给出最优解，但由于“组合爆炸”，其仅可用于计算较小规模实例。

2. 近似算法，或许有可能在短时间内，给出相当接近最优解的近似解。其中，非随机性近似算法包括构建式启发/贪婪算法，克里斯托菲德斯算法；随机性近似算法包括随机局域搜索、模拟退火、遗传算法、粒子群算法等。

本节接下来介绍对比算法，包括非随机近似算法（最近邻点算法、克里斯托菲德斯算法以及 2-OPT 改进算法）和随机近似算法（王磊算法、模拟退火算法）。

2.1 非随机近似算法

2.1.1 最近邻点算法

顾名思义，在选定一个启始城市 s 后，每次贪婪地选择距离当前城市最近的未访问城市 v 作为下一站；依次类推，直至将所有城市访问一遍，最后回到出发城市 s。伪码如下：

Algorithm 1: GreedyNearestNeighbor Algorithm

input : $V = \{v_1, \ldots, v_n\}, dist(\cdot,\cdot), L(\cdot), s \in V$

output: σ^*, L

$tour \leftarrow [s]$, $visited \leftarrow \{s\}$, $L \leftarrow 0$;

while $|tour| < n$ **do**

  $v \leftarrow \arg\min_{u \in V \setminus visited} dist(\text{last}(tour), u)$;

  $tour.append(v)$, $visited \leftarrow visited \cup \{v\}$;

$\sigma^* \leftarrow tour, L \leftarrow L(\sigma^*)$;

2.1.2 克里斯托菲德斯算法

可证明最差情况下，该近似算法所得回路长度也不会超过最优回路长度的 1.5 倍。求最小值问题，设 Opt 是最优值，x 表示某近似算法给出的一个值，一般规定，$Opt \leq x \leq \alpha \times Opt$，$\alpha$ 记为该算法的近似比，可用于评价算法优劣。元启发算法虽然有可能得出比较好的近似解，但往往不涉及在最差情况下的效率证明。

首先，引入近似比为 2 的算法（2-Approximation）：

(a) 定义：S 代表一系列边（允许重边），$c(S)$ 代表各边权重（长度）之和。

(b) 定义：H_G^* 为无向多重图 G 上，长度最短的哈密尔顿回路（Hamiltonian Cycle），途中经过所有点且只经过一次。

(c) 构造最小生成树 T，根据最小权生成树定义，$c(H_G^*) \geq c(H_G^* - e) \geq c(T)$。

(d) 按深度优先搜索次序记录回路 C，下探一次，回溯一次，因此 $c(C) = 2 \times c(T)$。

(e) 搭桥（short-cut/bypass）略过重复访问的点得到符合问题描述的新回路 C'（最后回到起点），例如，1，2，3，4，5，6...，1。

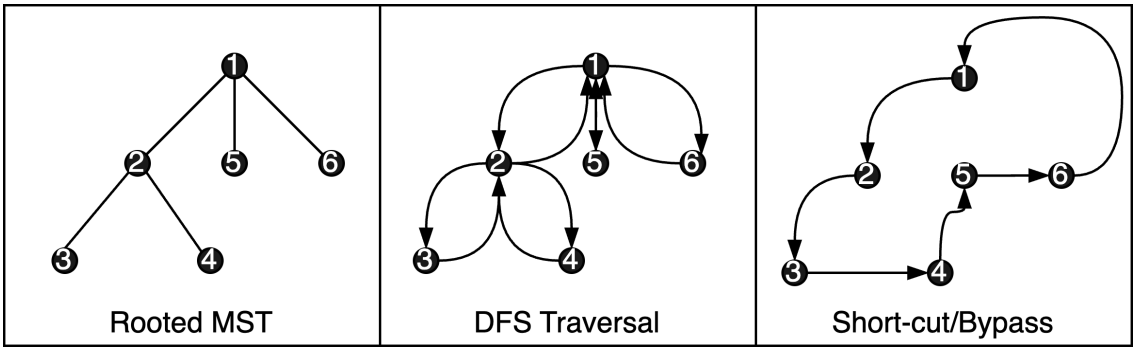

图 1: 近似比为 2 的算法（引入）

证明如下：

- 由 e、三角形三条边关系定则，$c(C') \leq c(C)$；
- 由 c，$c(H_G^*) \geq c(H_G^* - e) \geq c(T)$；
- 由 d，$c(C) = 2 \times c(T)$；
- 故 $c(C') \leq 2c(H_G^*)$；
- 因此，该近似算法所得解，最多也不会超过最优解的 2 倍。

然后，仍基于最小生成树，想方设法减小“每边下探一次，回溯一次”带来的额外开销，导出理论近似比为 1.5 的算法。期待一笔画、不重边地遍历所有顶点，可以将问题转换成“欧拉回路”问题。无向图存在欧拉回路的充要条件为：该图为连通图，且所有顶点度数均为偶数。倘若‘奇度数‘顶点为偶数个（证明见下），那么可以通过将其两两匹配，为每一个顶点都“附赠”一个度，这样便可以满足“顶点度数均为偶数”条件。

(a) 定义：S 代表一系列边（允许重边），$c(S)$ 代表各边权重（长度）之和。

(b) 定义：H_G^* 为无向多重图 G 上，长度最短的哈密尔顿回路（Hamiltonian Cycle），即途中经过所有点且只经过一次。

(c) 定义：假设 S 为无向多重图 G 上的导出子图，在 S 上长度最短的哈密尔顿回路记为 H_S^*。根据三角形三边关系定则易证，$c(H_S^*) \leq c(H_G^*)$。

(d) 构造最小生成树 T，根据最小权生成树定义，$c(H_G^*) \geq c(H_G^* - e) \geq c(T)$。

(e) 分离在 T 上度数为奇数的点，生成导出子图 S（根据握手定理，给定无向图 $G = (V, E)$，一条边贡献 2 度，故有 $\Sigma degG(v) = 2|E|$；除开度数为偶数的顶点所贡献的度数，推论可知，度数为奇数顶点数有偶数个）；

(f) 构造 S 的最小权完美匹配 M，构造多重图 $G' = T \cup M$（此时每个顶点均为偶数度，故存在欧拉回路）；

(g) 生成 G' 的欧拉回路 C，$c(C) = c(T) + c(M)$;

(h) 搭桥（short-cut/bypass）略过重复访问的点（起点终点不删）得到符合问题描述的新回路 C'（最后回到起点）。

证明：

- 由 e、三角形三边关系定则，$c(C') \leq c(C)$；
- 由 d，$c(H_G^*) \geq c(H_G^* - e) \geq c(T)$；
- 由 g，$c(C) = c(T) + c(M)$；
- 由 f、c，$c(M) + c(M) \leq c(M1) + c(M2) = c(H_S^*) \leq c(H_G^*)$；
- 故 $c(C') \leq c(T) + c(M) \leq c(H_G^*) + \frac{1}{2}c(H_G^*)$；
- 即得证。

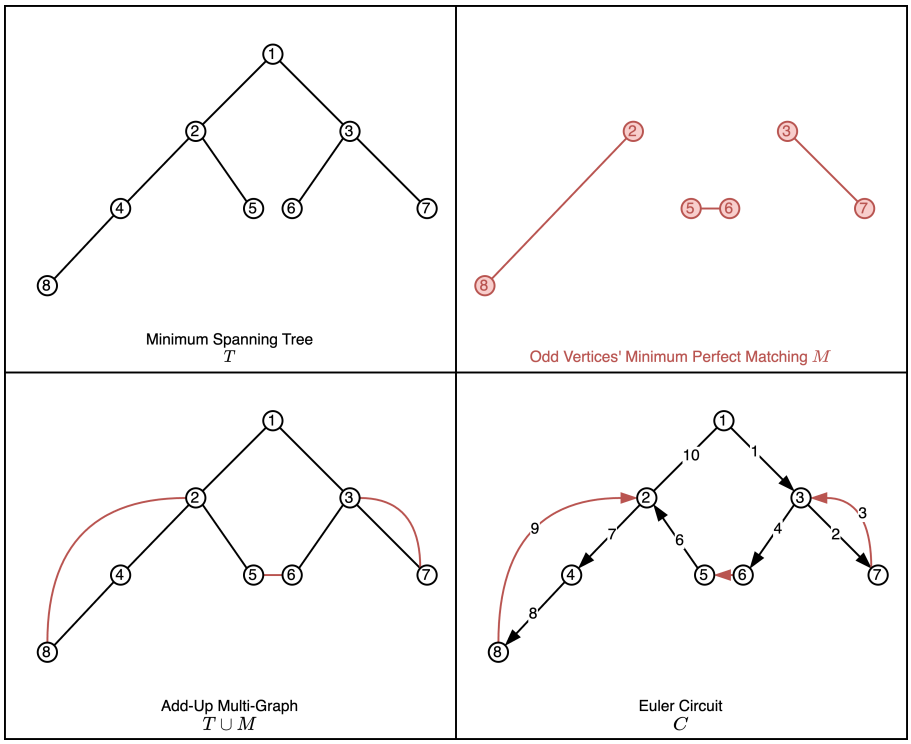

图 2: 克里斯托菲德斯算法（步骤）

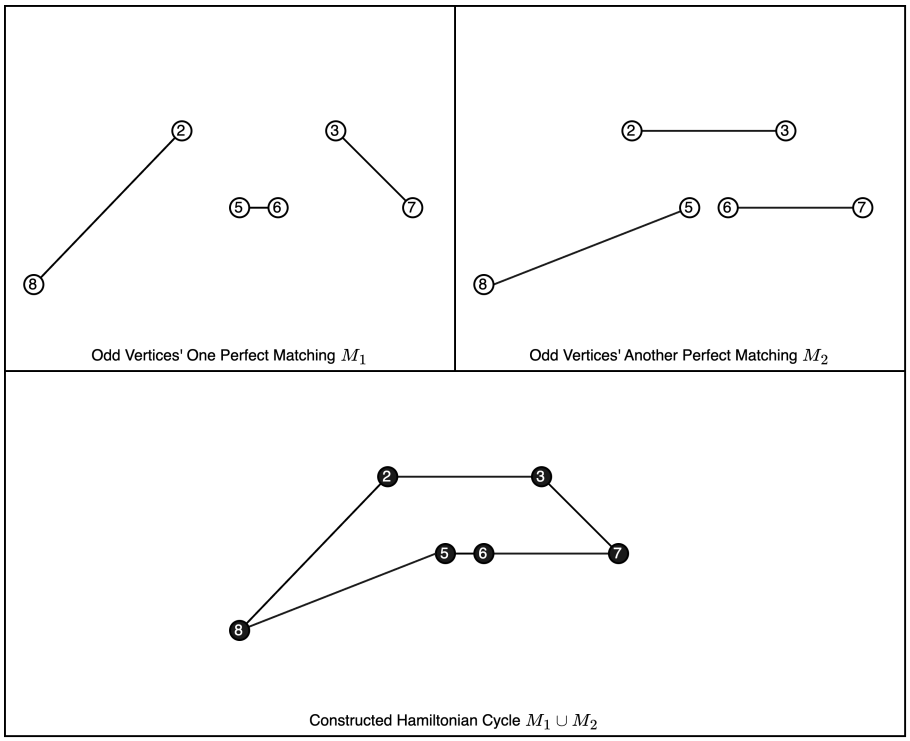

图 3: 最小权完美匹配（举例）

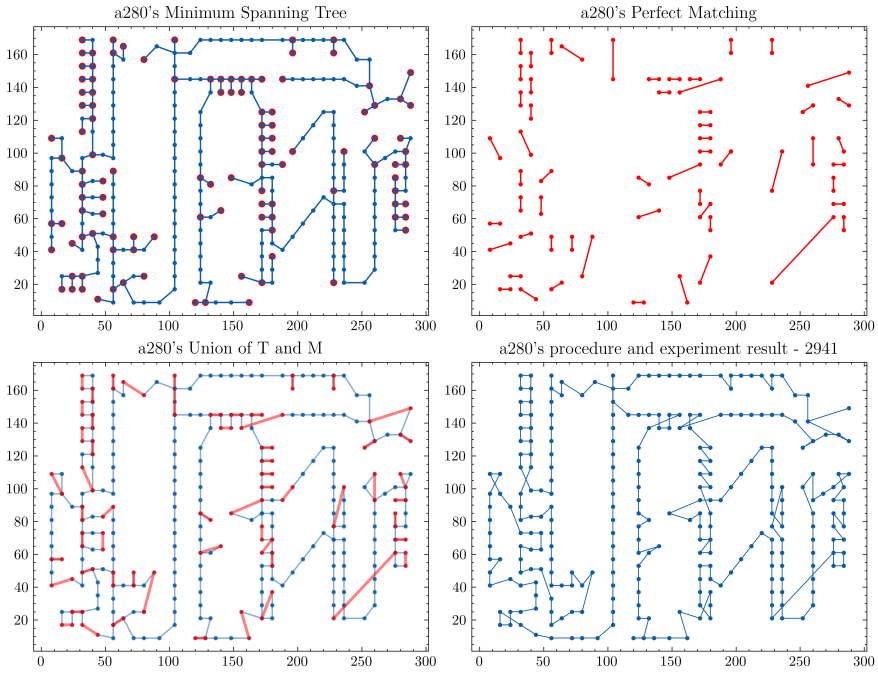

图 4: 克里斯托菲德斯算法（实例）

2.1.3 2-OPT 改进算法

如果题目数据使用欧几里得距离，那么最优路线必定不会自交。基于这一观察，有学者倡导使用“改进”算法，即对于一条可行回路查漏补缺对其进行细微调整。“知错能改，善莫大焉”。“怎么改”对应着一种“操作”，即一种“邻域算子”。

解空间中的一个巡回旅行路线直接或间接对应一个全排列 σ，若将其视作 n 维空间中的一个点，其邻域 σ' 操作有很多种，如插入、块插入、块反转、点对换、块交换、边重组等等。边重组中，最著名的是 2-交换（2-OPT）、3-交换（3-OPT）。2-交换的步骤就是删除路线中的两条边，用另外两条更短的边重新连接，是路径连为一体。反复使用 2-交换算子改进路线，就可以在很大程度上改进“虎头蛇尾”、“目光短浅”的回路路线。

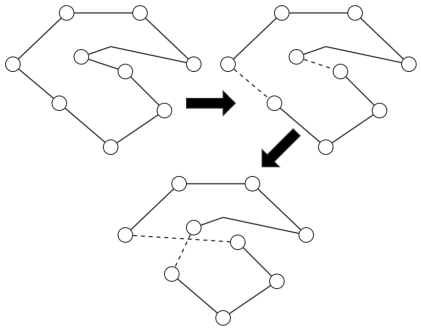

图 5: 2-OPT（图例）

2-OPT 改进算法伪代码如下：

Algorithm 2: 2-OPT Algorithm

input : $V = \{v_1, \ldots, v_n\}, dist(\cdot,\cdot), L(\cdot), \sigma$

output: σ^*

$length \leftarrow L(\sigma)$;
repeat
  $improved \leftarrow$ False;
  for $i \leftarrow 0$ **to** $n-3$ **do**
    for $j \leftarrow i+2$ **to** n **do**
      $\sigma' \leftarrow \sigma$;
      $\sigma'[i+1 \ldots j] \leftarrow \text{reverse}(tour'[i+1 \ldots j])$;
      $length' \leftarrow L(\sigma')$;
      if $length' < length$ **then**
        $\sigma \leftarrow \sigma'$;
        $length \leftarrow length'$;
        $improved \leftarrow$ True;
until $\neg improved$;
$\sigma^* \leftarrow \sigma$;

3-OPT 改进算法与之类似，但是可能的情况更多：

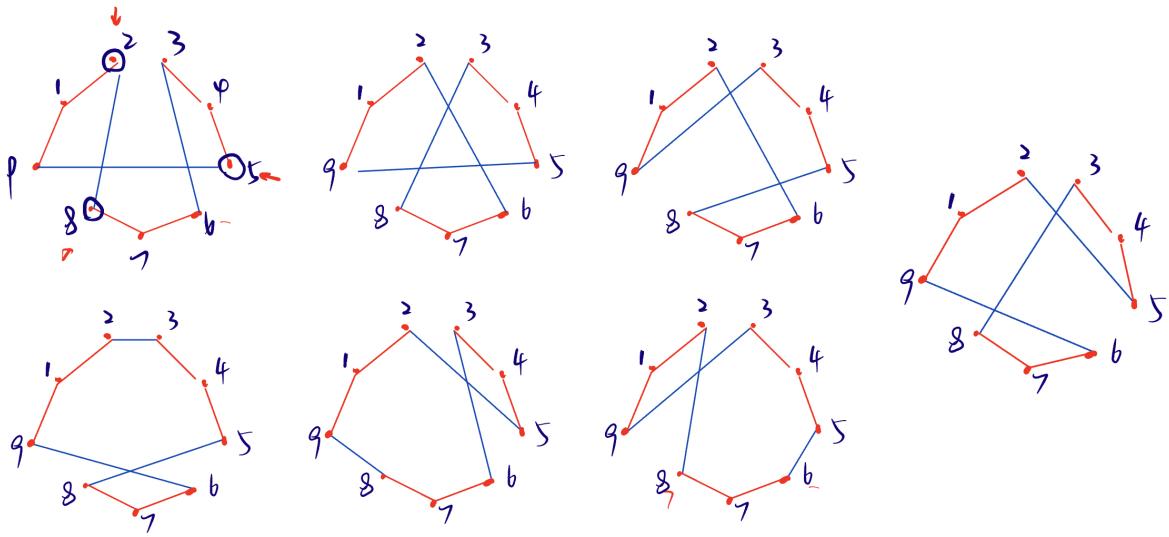

图 6: 3-OPT（图例）

2.2 随机型近似算法

2.2.1 王磊算法

王磊老师在课上跟学生说过一个随机型近似算法（王磊算法），基本算法 A_1 描述如下：

输入：指导序列 γ，γ 是所有顶点的一个全排列；

开局：用前 3 个点绘制外接凸多边形，构成初始的部分回路 $\sigma = (\gamma(1), \gamma(2), \gamma(3))$；

迭代：每次从当前格局向新格局演化时，选择下一个点，按照使得新的部分回路长度尽量短的贪心策略，将其插入至 σ 合适的位置；

停机：直到产生 n 个点的回路 σ，算法结束，输出 σ。

Algorithm 3: Generate Tour from a Conductor

input : $V = \{v_1, \ldots, v_n\}, dist(\cdot,\cdot), \gamma$ a permutation of V

output: σ the tour

$\sigma \leftarrow (\gamma(1), \gamma(2), \gamma(3))$;
for $i \leftarrow 4$ **to** n **do**
  $best\_idx \leftarrow \underset{j \in \{1,\ldots,|\sigma|\}}{\arg\min}\ L(\sigma_{1:j}) + dist(\gamma(i), \sigma(j)) + L(\sigma_{j:|\sigma|}) - L(\sigma)$;
  $\sigma \leftarrow (\sigma_{1:best\_idx}, \gamma(i), \sigma_{best\_idx+1:|\sigma|})$;

对所有指导序列 $\gamma \in \Gamma$，目标是 $\gamma^\star = \underset{\gamma \in \Gamma}{\arg\min}\, L(A_1(\gamma))$。据此，提出算法 A_2：

初始格局：初始化 γ，通过 A_1 算法指导获得回路 $\sigma = A_1(\gamma)$，以及长度 $l = L(\sigma)$；

邻域搜索：邻域变换得到 γ'、σ' 及 l'，若 $l' < l$，依照最陡下降法，更新格局 $\gamma \leftarrow \gamma'$

跳坑策略当 γ 位于局部最优，即几乎尝试所有邻域都无法改善目标函数时，重新随机初始化 γ 或者采用大步长算子（如块移动、块对换、块插入）对 γ 进行变换。

Algorithm 4: WangLei Algorithm

input : $V, dist(\cdot,\cdot), L(\cdot), epoch, early\_stop$
$permutation(\cdot), transform(\cdot), shuffle(\cdot)$

output: σ, l

$\gamma \leftarrow permutation(V)$; $\sigma \leftarrow A_1(\gamma)$; $l \leftarrow L(\sigma)$;
for $e \leftarrow 1$ **to** $epoch$ **do**
  $\gamma' \leftarrow transform(\gamma)$; $\sigma' \leftarrow A_1(\gamma')$; $l' \leftarrow L(\sigma')$;
  if $l' < l$ **then**
    $\gamma \leftarrow \gamma'$; $\sigma \leftarrow \sigma'$; $l \leftarrow l'$;
  if *no improvement for early_stop iterations* **then**
    $\gamma \leftarrow permutation(V)$ or $\gamma \leftarrow shuffle(\gamma)$;

王磊算法的创新和启发意义主要有以下三点：

1. 传统启发算法求解旅行商问题，几乎全部都是直接在回路 σ 上进行邻域扰动，获得新解。而王磊算法则提出了 $\gamma \to \sigma$ 的映射算法 A_1，这相当于对原有解空间进行了“扭曲”，将求“回路”的原问题转化为了求“指导顺序“的新问题。

   在最优化理论中，原始问题很难求解时，往往通过引入对偶问题的方式，简化对原始问题的求解。在机器学习中，也有代替函数、核函数作为例子。我们不禁要问，对于所有的“指导序列”$\gamma \in \Gamma$，它们所生成的所有回路集合 $\Sigma^\star$，是否包含了最优回路 $\sigma^\star$?

   即，通过指导序列将问题转换，问题转换前后是否仍然具有“一致性”?

2. 邻域搜索和跳坑策略思想并不高深。局部极小值的定义来自于函数求极值，跳坑则更有烟火气：如果你已经期末总评满分了，就要跳坑，到更有希望的学府继续深造。

   对于旅行商问题而言，无论是回路 σ，还是指导序列 γ，若邻域中的点所对应的回路长度都不比中心点短，则中心点是局部极小值点，当邻域搜索走到局部极小值点时，就采用跳坑策略，进行随机扰动，跳出局部极小值陷阱以后，继续进行邻域搜索。

   这其中的问题是，随着邻域算子设计的不同，邻域中的“点”随着维度的增大，个数可能比想象中要多得多，因此有时候不得不采用固定次数的方式来执行邻域搜索。邻域搜索对应“变异”、“开采”，而跳坑策略则对应“探索”，可以说所有的最优化算法都要考虑这两者的平衡。

3. 生成回路算法本身也具有烟火气。想象一下，借一个扎头发的橡皮筋，套住几个点；然后采用贪心策略，将其余点加入回路。

   传统的最近邻点贪心策略是，最后一步方能连成回路，这就导致目光浅显、虎猴蛇尾；而如果是在一个成形的“回路”中添加，每次添加评价的都直接是回路的全长，则能一定程度上缓解“短视”问题。

   这启发我们同样是贪心策略，但是如何运用，运用的好不好则是可以评价的。

2.2.2 模拟退火

事实上，人们从物理世界状态演化、自然界各种现象、千百年来生存斗争经验获得启发，以仿生拟人拟物途径设计了各种千奇百怪五花八门的算法。模拟退火是其中一种，具有自然背景而且实现简单。

模拟退火并没有显式地将跳坑策略（探索）和邻域搜索（开采）分成两阶段看待；它的基本思想是，以接受劣解，且接受劣解的概率随迭代次数递减直至无限趋近于零。如果只接受优解，则容易早熟，多样性不足，易于陷入局部最优，因此需要接受劣解；如果一味接受劣解，则无法保证收敛性，因此需要控制接受劣解的概率。

Algorithm 5: SimulatedAnnealing Algorithm

input : $V, dist(\cdot,\cdot), L(\cdot), transform(\cdot)$
$T, \epsilon, \alpha, time\_out, early\_stop$

output: σ^*, L^*

start_time ← current time;
while *current time* − *start_time* < *time_out* **do**
  $\sigma \leftarrow \text{permutation}(V)$;
  $L \leftarrow L(\sigma)$;
  while $T > \epsilon$ **do**
    for *step* ← 1 **to** *early_stop* **do**
      $\sigma' \leftarrow \text{transform}(\sigma)$; $L' \leftarrow L(\sigma')$; $\Delta L \leftarrow L' - L$;
      if $\Delta L < 0$ ***or*** $random(0,1) \leq e^{\frac{-\Delta L}{T}}$ **then**
        $\sigma \leftarrow \sigma'$; $L \leftarrow L'$;
    $T \leftarrow T \times \alpha$;
$\sigma^* \leftarrow \sigma$, $L^* \leftarrow L$;

3 遗传算法及改进策略

3.1 传统的遗传算法

Algorithm 6: Genetic Algorithm for TSP

input : V, epoch, early_stop, population_size, pc, pm

output: σ^*, L^*

初始化种群;
for $e \leftarrow 1$ **to** *epoch* **do**
  初始化当前最佳长度为无穷大;
  for *step* ← 1 **to** *early_stop* **do**
    选择操作：根据适应度选择当前种群中的一些个体;
    交叉操作：根据交叉概率 pc 结合选中的个体产生后代;
    变异操作：根据变异概率 pm 改变某些个体的特征;
    如果找到更优的解，则更新当前最佳长度;
  重新初始化种群;
σ^* ← 找到的最佳解; L^* ← 最佳解的长度;

无论是基于邻域搜索和拟人策略跳坑的王磊算法，还是从淬火物理结晶过程获得启发的模拟退火算法，都是基于“个体”的启发算法。而遗传算法，从生物学获得启发，将“个体”扩展至“种群”；除邻域操作（也成“变异”算子）外，新增了“交叉”操作，将“个体理性”和“群体理性”进行结合。传统的遗传算法求解旅行商问题的具体细节为：

编码 将执行变异操作的个体直接编码为城市序号的全排列 σ；

适应 采用 $\frac{1}{L(\sigma)}$ 表示解的优劣，适应度越大，被选择保留的概率越高；

选择 采用轮盘赌，计算每条染色体的被选择概率和累计概率，再根据一个随机数确定要保留的染色体；选择操作是遗传算法的核心，一方面，要保证收敛质量好，即回路长度短，另一方面，要保证种群有足够的多样性，避免陷入局部最优的困境；

交叉 交叉操作的目的是，集合不同回路的优良顺序特征，常用有顺序交叉和部分映射交叉。

变异 通过邻域变换对种群中的个体（回路）进行扰动；遗传算法中，变异概率通常非常小。

3.2 改进的遗传算法

我对传统遗传算法的初始化、选择、变异操作做出如下改进：

初始 发扬“继承”策略，在初始化阶段，将“2-OPT”和“最近邻点”的结果作为初始化种群的一部分；这样可以极大的减少迭代次数，保证解的收敛性，使得随机的元启发算法依然具有理论保证，回路长度最大不会超过最优回路的 1.5 倍。

选择 在选择过程中，弃用轮盘赌法，轮盘赌法的缺陷是，当适应度相似时，选择概率相近，不一定保证选择当前回路长度最小的解，使得收敛性无法保证；我们采用排位等级法，随着适应度的排序等级该确定选择概率，缓解了适应度相近时，选择困难的问题。

变异 在变异过程中，除了传统的算子外，设计了一个全新的变异算子。从王磊 A_1 算法中获得启发，我们对于一个已知回路 σ，随机剔除 N 个城市，然后依序采取贪心策略将被剔除的点添加到回路中。N 取自一个概率分布，这样能够保证剔除城市个数可以动态变化；而剔除策略，分为单点剔除和随机剔除。

下面给出种群初始化的伪代码：

Algorithm 7: Population Initialization for Genetic Algorithm

input : V, $size$, $init\_population$

output: Initialized population P

$P \leftarrow init\_population$;
while $|P| < size$ **do**
    P.append(permutation(V));

下面给出选择操作的伪代码：

Algorithm 8: Selection Operation in Genetic Algorithm

Function `Select`$(P, L, size, C, operator)$**:**
    $lengths \leftarrow [L(individual)$ for each $individual \in P]$;
    $order \leftarrow$ sort indices of $lengths$ in ascending order;
    $selected \leftarrow$ [best seen tour];
    while $|selected| < size$ **do**
        $idx \leftarrow 0$, $target \leftarrow 1$;
        while $random(0,1) < target \times (1 - C)$ **do**
            $idx \leftarrow idx + 1$;
            $target \leftarrow target \times C$;
        $selected$.append($P[order[idx]]$);
    return $selected$;

下面给出变异算子的伪代码：

Algorithm 9: Greedy Insert Operator for Genetic Algorithm

input : σ, $dist(\cdot,\cdot)$, $times$, $dimension$

output: Modified σ

if $random(0,1) < 0.5$ **then**
    $conductor \leftarrow$ remove $times$ random elements from σ;
else
    $pivot \leftarrow$ random integer$(1, dimension - times - 1)$;
    $conductor \leftarrow$ remove $times$ elements starting at $pivot$ from σ;
foreach $vertex \in conductor$ **do**
    $best\_idx \leftarrow \underset{j \in \{1,\ldots,|\sigma|\}}{\arg\min}\ L(\sigma_{1:j}) + dist(vertex, \sigma(j)) + L(\sigma_{j:|\sigma|}) - L(\sigma)$;
    $\sigma \leftarrow (\sigma_{1:best\_idx}, vertex, \sigma_{best\_idx+1:|\sigma|})$;

4 实验设置与测试结果

4.1 数据集与超参数

TSPLIB（http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/）中公布了旅行商问题的 benchmark 测试数据集。以 EUC-2D 类型的测试数据集中的实例 a280 为例。*a280.txt* 文件开头有一段说明文字，然后是 280（表示点的个数），接下来有 280 行数据，每行数据含有 3 个数，分别是：当前点的序号、当前点的 x 坐标、当前点的 y 坐标。

表 1: 随机近似算法实验超参数设置

对应算法	超参数	缺省值
GreedyNearestNeighbor	*boost*, 是否随机选择一个起始城市	$True$
SimulatedAnnealing	初始温度 t, 终止温度 ϵ, 衰减系数 α	$1000, 10^{-14}, 0.98$
SimulatedAnnealing	重启停机参数 time_out, early_stop	$1, 250$
WangLeiAlgorithm	重启停机参数 epoch, early_stop	$16, 250$
Proposed	种群大小 $size$, 交叉概率 p_c，变异概率 p_m	$50, 1, 0.4$
Proposed	选择系数 C	0.5
Proposed	重启停机参数 epoch, early_stop	$6, 7500$

* 提出改进的遗传算法的初始种群仅来自 2-OPT、GreedyNearestNeighbor。

4.2 实验结果

由于王磊老师是在 C 语言环境下，选择“最快速度”编译选项，在 CPU 主频为 3.4GHz 的微机上进行的测试；而我对算法的实现均采用 Python 语言编程，不具有可比较性，因此，我弃用了王磊老师于《专业方向综合实践验收的问题》中提到的报道结果，而是用 Python 复现的王磊算法进行比较。

基于 Python 语言编程实现了最近邻点算法、克里斯托菲德斯算法、2-OPT 改进的克里斯托菲德斯算法、模拟退火算法，依照课上所述的基本思想对王磊算法进行了复现作为对比算法，对本文提出的改进的遗传算法进行了测试。选取城市数小于等于 1000 中全部 48 个 benchmark 测试用例对算法进行测试，每个实例计算 10 次。

下面是 10 次计算所得回路长度的最小值、平均值和平均计算时间。实验验证了提出的遗传算法的收敛质量和求解速度：在。

5 结束语

算法及复现代码开源在：https://github.com/DURUII/Homework-Algorithm-TSPLIB95。

本次专业方向综合实践，我对进化计算、遗传算法、组合优化问题以及算法思想直觉有了进一步的认识，更深刻地体会到开采和探索、多样和收敛、邻域和跳坑、个体和种群的平衡。