Statistics and Application 统计学与应用, 2025, 14(4), 399-408
Published Online April 2025 in Hans. https://www.hanspub.org/journal/sa
https://doi.org/10.12677/sa.2025.144119

Hans汉斯

基于模拟退火算法的旅行商路径优化

刘　洋，刘　楠，刘　智*

沈阳工业大学化工装备学院，辽宁　辽阳

收稿日期：2025年3月30日；录用日期：2025年4月21日；发布日期：2025年4月30日

摘　要

旅行商问题(TSP)是一个典型的组合优化问题，在物流配送、路径规划等领域具有广泛应用。然而，由于TSP问题属于NP-hard问题，传统的精确算法难以在大规模情况下求得最优解，因此启发式算法成为求解TSP的主要方法之一。本文研究了模拟退火算法(SA)在TSP问题中的应用，详细分析了SA的基本原理、算法流程及其在TSP问题中的求解过程。通过构建城市坐标数据，本文利用模拟退火算法进行路径优化，并在不同温度控制策略下探讨了算法的收敛性。实验结果表明，SA能够有效避免局部最优，逐步收敛至高质量解，并在全局搜索能力和计算效率方面表现出较好的平衡性。最后，本文总结了SA在TSP求解中的优劣势，并探讨了未来改进方向，如结合其他智能优化算法以提升求解效率和精度。

关键词

旅行商问题，模拟退火算法，路径优化，组合优化

Traveler Path Optimization Based on Simulated Annealing Algorithm

Yang Liu, Nan Liu, Zhi Liu*

School of Chemical Equipment, Shenyang University of Technology, Liaoyang Liaoning

Received: Mar. 30th, 2025; accepted: Apr. 21st, 2025; published: Apr. 30th, 2025

Abstract

The traveler's problem (TSP) is a typical combinatorial optimization problem with a wide range of applications in the fields of logistics and distribution, path planning and so on. However, since the TSP problem is an NP-hard problem, traditional exact algorithms are difficult to find the optimal solution in large-scale situations, so heuristic algorithms become one of the main methods to solve the TSP. In this paper, we study the application of simulated annealing algorithm (SA) in TSP problems, and

*通讯作者。

文章引用: 刘洋, 刘楠, 刘智. 基于模拟退火算法的旅行商路径优化[J]. 统计学与应用, 2025, 14(4): 399-408.
DOI: 10.12677/sa.2025.144119

analyze the basic principle of SA, the algorithmic process and its solution process in TSP problems in detail. By constructing city coordinate data, this paper uses the simulated annealing algorithm for path optimization and explores the convergence of the algorithm under different temperature control strategies. The experimental results show that SA can effectively avoid local optimization, gradually converge to a high-quality solution, and exhibit a good balance between global search capability and computational efficiency. Finally, this paper summarizes the advantages and disadvantages of SA in TSP solving, and explores the future improvement directions, such as combining with other intelligent optimization algorithms to enhance the solving efficiency and accuracy.

Keywords

Traveler's Problem, Simulated Annealing Algorithm, Path Optimization, Combinatorial Optimization

1. 引言

旅行商问题(Traveling Salesman Problem, TSP)是一个经典的组合优化问题，它描述了这样一个场景：一个旅行商需要访问若干个城市，每个城市恰好访问一次，并最终回到出发城市，要求找出访问所有城市并返回出发点的最短可能路径[1]。这个问题在实际生活中有着广泛的应用，如物流配送、路径规划、通信网络优化等。因此，研究 TSP 问题不仅具有理论价值，也有着重要的实践意义。

然而，TSP 问题是一个 NP-hard 问题，即随着城市数量的增加，问题的求解难度呈指数级增长，使得传统的精确算法在解决大规模 TSP 问题时变得不切实际。因此，研究者们转向启发式算法来寻找 TSP 问题的近似最优解。而模拟退火算法正是一种有效的启发式搜索算法。因此，本文旨在通过深入研究模拟退火算法在 TSP 问题中的应用，探讨算法的实现细节和性能表现，为实际问题的求解提供有效的算法支持。此外，模拟退火算法的收敛性是其能否有效求解 TSP 问题的关键问题之一。基于马尔可夫链的理论研究表明，在满足一定的温度衰减条件时，SA 算法能够以概率 1 收敛至全局最优解。然而，在实际应用中，温度控制策略对算法的收敛速度和最终解的质量有显著影响。因此，本文将探讨 SA 算法的收敛性，并分析不同温度控制策略对收敛效果的影响。另一方面，SA 算法的计算复杂度主要受迭代次数、邻域搜索策略及温度衰减方式影响，为了评估其求解效率，本文也将对 SA 算法的时间复杂度进行分析，并与其他优化方法进行比较。通过本文的研究，希望能够为 TSP 问题的求解提供新的思路和方法，推动启发式算法在组合优化领域的应用和发展[2]。

2. 模拟退火算法原理

2.1. 模拟退火算法的基本原理

模拟退火算法(Simulated Annealing, SA)借鉴了物理退火过程的思想，通过模拟固体物质在高温下逐渐冷却的过程，寻找全局最优解。为了更好地理解模拟退火算法的核心机制，表 1 展示了物理退火与模拟退火中的状态对应关系。从表中可以看出，物理退火中的温度、能量状态和冷却速率分别对应模拟退火中的控制参数、目标函数值和迭代策略。这种对应关系为算法的设计和参数设置提供了理论依据，同时也揭示了模拟退火算法在解决复杂优化问题时的有效性和灵活性。其中算法的核心思想来源于物理中

的退火过程，即通过逐步降低温度，使系统最终稳定在最低能量状态[3]。其优化过程主要包括以下步骤：Step1 初始化：设定初始温度 T_0 和初始解 X_0，计算该解对应的目标函数值 $f(x_0)$。

Step2. 扰动生成：通过随机扰动 Δx 生成新解

$$x' = x + \Delta x$$

算新解的目标函数值 $f(x')$ 及其相对当前解的变化量

$$\Delta f = f(x') - f(x)$$

Step3. 解的接受策略：

若 $\Delta f \le 0$，即新解优于当前解，则直接接受新解，作为下一轮迭代的初始解。

若 $\Delta f \ge 0$，即新解较差，则根据 Metropolis 准则计算接受概率：

$$P(\Delta f) = \exp\left(-\frac{\Delta f}{KT}\right)$$

生成一个[0,1]之间的随机数 r，若

$$P(\Delta f) > r$$

则仍然接受新解；否则，保持当前解不变。

Step4. 温度更新：逐步降低温度 TTT，使算法从大范围搜索(高温阶段)逐步进入细粒度优化(低温阶段)。

Step5. 终止条件：当温度降至设定的最低值或达到最大迭代次数，算法终止，输出最终优化结果。

这一过程通过不断搜索、接受较优解，并在一定概率下接受较差解，使得算法能够跳出局部最优，寻找全局最优解。

Table 1. Correspondence of states in physical annealing and simulated annealing
表 1. 物理退火与模拟退火中的状态对应图

物理退火	模拟退火
粒子状态	每个状态对应一个(可行)解
能量最低态	最优解
溶解过程	设定初温(设定参数 T 的值)
等温过程	一个温度下，多次采样(Metropolis 采样过程)
冷却	控制参数下降
能量	目标函数

2.2. 模拟退火算法的核心思想

在搜索区间进行随机游走(通过生成随机数实现)，再利用 Metropolis 抽样准则，使随机游走逐渐收敛于局部最优解。在这里，温度是一个重要的控制参数，这个参数的大小控制了随机过程向局部或全局最优解移动的快慢[4]。具体的过程如下：

首先，定义一个 P：

$$p = e^{-\frac{E_2 - E_1}{T}}$$

这个公式表示，系统从 E_1 变化到 E_2，其概率为 p，就是上述公式左边的部分。

如果 $E_2 < E_1$，那么证明系统向更低的方向转移了，我们无条件接受此状态。

否则，以上述的概率，接受这个较坏的结果 E_2，注意，这里的原则，不同于贪心，每次选择最好的，而是有一定的概率接受坏一些的。(这个概率受到能量差和温度两个因素的影响)

$$p(1\to 2)=\begin{cases}1 & E_2<E_1\\ e^{-\frac{E_2-E_1}{T}} & E_2>E_1\end{cases}$$

这样通过一定的迭代次数，我们就会找到一个比较好的解了。

2.3. 模拟退火算法流程

算法实质分两层循环，在任一温度水平下，随机扰动产生新解，并计算目标函数值的变化，决定是否被接受。由于算法初始温度比较高，这样，使 E 增大的新解在初始时也可能被接受，因而能跳出局部极小值，然后通过缓慢地降低温度，算法就最终可能收敛到全局最优解，具体流程如图 1。

令 $T=T_0$，表示开始退火的初始温度，随机产生一个初始解 x_0，并计算对应的目标函数值 $E(x_0)$；

令 $T=kT$，其中 k 取值 0 到 1 之间，为温度下降速率；

对当前解 x_t 施加随机扰动，在其邻域内产生一个新解 x_{t+1}，并计算对应的目标函数值 $E(x_{t+1})$ 计算

$$\Delta E=E(x_{t+1})-E(x_t)$$

若 $\Delta E<0$，接受新解作为当前解，否则按照概率 $e^{\frac{-\Delta E}{T}}$ 判断是否接受新解；

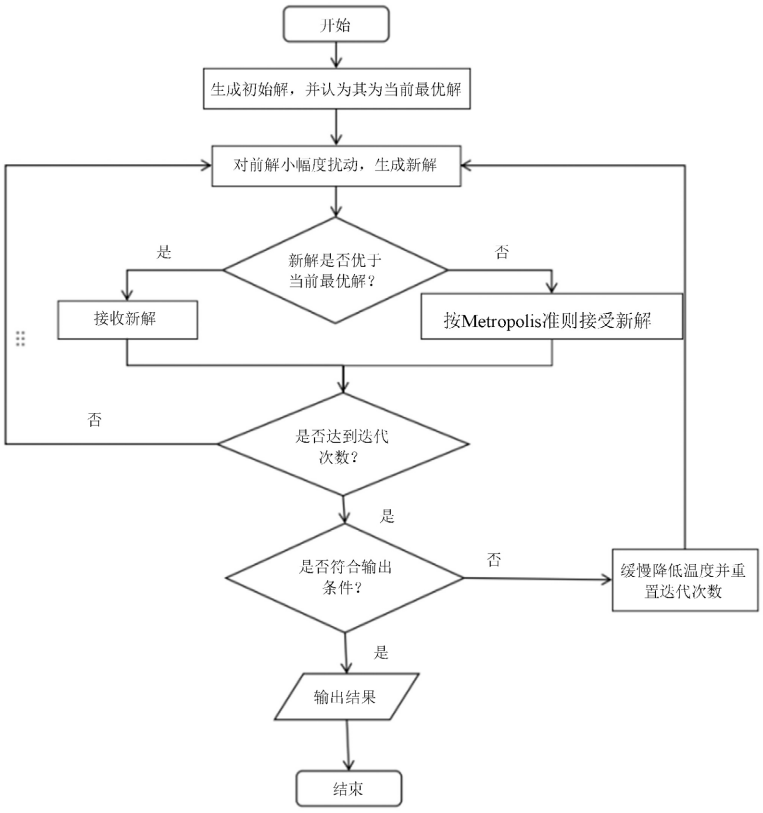

Figure 1. Flowchart of simulated annealing algorithm
图 1. 模拟退火算法流程图

2.4. 理论收敛性分析

渐近收敛性：在温度逐渐降低到 0 时，SA 算法能够以高概率找到全局最优解。这一结果是基于概率论和马尔可夫链理论得出的，具体来说，如果温度下降的速率满足某些条件，SA 算法能在无限时间内收敛到全局最优解。

实际收敛性：在实际应用中，虽然 SA 理论上能找到全局最优解，但由于计算时间和资源的限制，温度往往无法持续下降至极低值。因此，实践中常常设置一个合适的终止条件(如温度小于某一阈值，或达到最大迭代次数)来平衡求解质量与计算效率。

3. 模型建立

旅行商问题

1. 旅行商问题描述

有个旅行商，他要去中心城市 A，出发到另外 n 个城市(已给出城市的经纬度坐标)出售商品，最后再回到中心城市 A。如果旅行商想在路上花费的时间最短，请你帮他规划路线[5]。

2. 旅行商问题数字模型(DFJ 模型)

$$\min \sum_{i\in V}\sum_{j\in V} d_{ij}x_{ij}$$

$$subject\ to \sum_{j\in V} x_{ij} = 1,\ \forall i \in V,\ i \neq j$$

$$\sum_{i\in V} x_{ij} = 1,\ \forall j \in V,\ i \neq j$$

$$\sum_{i,j\in S} x_{ij} \le |S| - 1,\ 2 \le |S| \le N - 1,\ S \subset V$$

$$x_{ij} \in \{0,1\},\ \forall i, j \in V$$

3. 问题分析

模拟退火算法的实现主要可分为：解空间、新解的产生和目标函数三部分。

(1) 解空间

解空间 S 可表为 $\{1,2,3,\cdots,n,n+1,n+2\}$ 的所有固定起点和终点的循环排列集合，其中 1 和 n + 2 都表示中心城市，$\{2\sim n+1\}$ 表示旅行商经过的城市，如：[1 2 3 4 5 6 7 8 9 10]和[1 9 8 7 6 1 2 3 4 5 10]表示的就是通过 8 个城市旅行商的两条不同路径。另外为了计算过程更方便，我们还应提前构建表示两两城市之间的距离矩阵 d 如：*Dij* 就表示城市 *i* 到城市 *j* 的距离。

(2) 新解的产生

任选序号 u，v $(u<v)$交换 u 与 v 之间的顺序，此时的新路径为：

$$\{1,2,3\cdots x_{u-1}, x_v, x_{u+1}\cdots x_{v-1}, x_u, x_{v+1}\cdots n+2\}$$

产生新解的路径差为：

$$\left(d_{x_{u-1}x_v} + d_{x_u x_{v+1}}\right) - \left(d_{x_{u-1}x_u} + d_{x_v x_{v+1}}\right)$$

(3) 目标函数：即求路径的最小值 $\min f(1\cdots n+2)$

4. 算法设计

4.1. 初始化部分

Step1 初始化模块，该模块主要完成了算法运行前的准备工作，包括用 n 定义了需要访问的城市数量(这里设置为 20 个城市)；用 temperature 定义了模拟退火算法的初始温度(温度值通过城市数量 n 乘以 100 来计算，这通常是为了确保算法开始时接受新解的概率较高)；用 *iter* 定义了内部蒙特卡洛循环的迭代次数，即在每次温度更新之前，算法将尝试 *iter* 次不同的城市排列(扰动)，以寻找可能的更优解。用结构体数组 *city*，存储每个城市的坐标信息。使用 *for* 循环为每个城市随机生成一个坐标。$city(i).x$ 和 $city(i).y$ 分别表示第 i 个城市的 x 坐标和 y 坐标，这些坐标是在 1 到 100 之间的随机整数。

Step2 初始解计算模块，该模块调用 computer_tour 函数计算当前随机生成的城市坐标序列下的旅行路线总长度，并将这个长度存储在 *len* 数组的第一个元素中。这个函数应该遍历城市序列，计算相邻城市之间的欧氏距离，并将它们累加得到总长度。

Step3 初始解可视化模块，该模块通过调用 netplot 函数绘制初始的旅行路线，即在二维平面上用线段连接所有城市，形成一个闭环。这个函数通常使用 MATLAB 的绘图功能来可视化城市的位置和它们之间的连接关系。负责将初始城市序列及其旅行路线进行可视化展示，帮助用户直观地了解问题的初始状态。

4.2. 核心部分

Step1 主循环模块，该模块通过 while temperature > 0.001 循环控制整个模拟退火过程。在每次循环中，算法都会进行多次随机扰动并评估新解，根据概率接受准则更新当前解，并在每次外循环结束时降低温度。

Step2 扰动与评估模块，该模块通过 $for(i)=1:iter$ 循环，该模块实现了多次随机扰动。在每次迭代中，它调用 perturb_tour 函数生成当前城市序列的随机扰动版本，并计算新序列的总长度。然后，计算新旧序列长度的差值 delta_e。

Step3 解的接受模块，该模块根据 delta_e 的值和当前温度来决定是否接受新解。如果新解更好(即 delta_e < 0)，则无条件接受；否则，根据模拟退火算法的概率接受准则 exp(-delta_e/temperature) > rand() 来决定是否接受较差的新解。

Step4 结果记录模块，每次外循环结束后，该模块调用 computer_tour 函数计算当前城市序列的总长度，并将其记录在 len 数组中。这样，算法可以在迭代过程中跟踪并记录解的变化情况。

Step5 温度更新模块，在每次外循环的末尾，该模块负责更新温度值。通过乘以一个小于 1 的因子(这里是 0.99)，算法实现了温度的逐渐降低，模拟了物理退火过程中温度下降的过程。

Step6 辅助函数模块，computer_tour 和 perturb_tour 是两个辅助函数模块。computer_tour 函数用于计算给定城市序列的总长度，而 perturb_tour 函数用于生成当前城市序列的随机扰动版本。这两个函数为算法的主要模块提供了必要的计算支持。

Step7 绘制城市点和连线模块，绘制城市位置：使用一个循环从 $i=1$ 到 $n-1$，对于每个城市 $city(i)$，它使用 plot 函数将城市的位置绘制在图上，每个城市的位置由其 x 和 y 坐标表示，并使用红色星号(`'r*'`)作为标记。

Step8 绘制最后一个城市点和闭合连线模块，绘制城市之间的连线：在绘制城市位置的同时，该循环还使用 line 函数连接相邻的城市。对于每一对城市 $city(i)$ 和 $city(i+1)$，它绘制一条从 $city(i)$ 到 $city(i+1)$ 的线段，这样就在图上展示了旅行路线的形状。闭合循环：在循环结束后，代码单独处理最后一个城市 $city(n)$。它使用 plot 函数将最后一个城市的位置也绘制在图上，同样使用红色星号作为标记。然后，使用 line 函数从最后一个城市 $city(n)$ 绘制一条线段到第一个城市 $city(1)$，从而闭合旅行路线，形成一个闭环。

4.3. 算法分析

4.3.1. 初始解路径分析

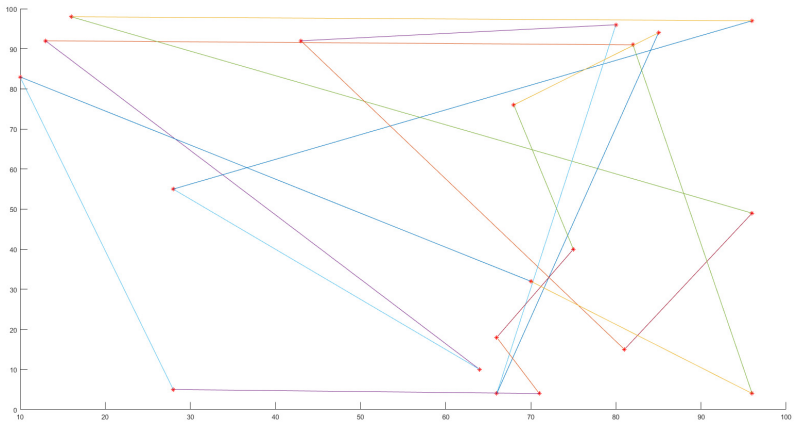

Figure 2. Initial traveler roadmap
图 2. 初始旅行商路线图

(1) 初始旅行商路线图像特点

图 2 显示了未经过模拟退火优化的旅行路径，即最初解。

1. 城市点：用 plot 函数绘制，标记为红色星号 `'r*'`，表示 TSP 问题中的所有城市。

2. 路径连线：使用 line 函数连接相邻城市，形成一条初始路径。

3. 随机性：由于 perturb_tour 初始时采用随机顺序排列城市，因此路径通常是无序的，可能存在大量交叉和冗余距离。

(2) 图 2 关键分析

路径长，存在大量交叉点：这是由于初始路径没有经过优化，TSP 问题的随机解通常是次优或最差情况。这是模拟退火优化的起点，后续算法需要逐步减少路径冗余，以获得更优的解。

如果 n 较大(城市数量多)，初始解的路径复杂度更高，优化的难度也更大。优化建议见表 2 图像分析图。

4.3.2. 最终解路径分析

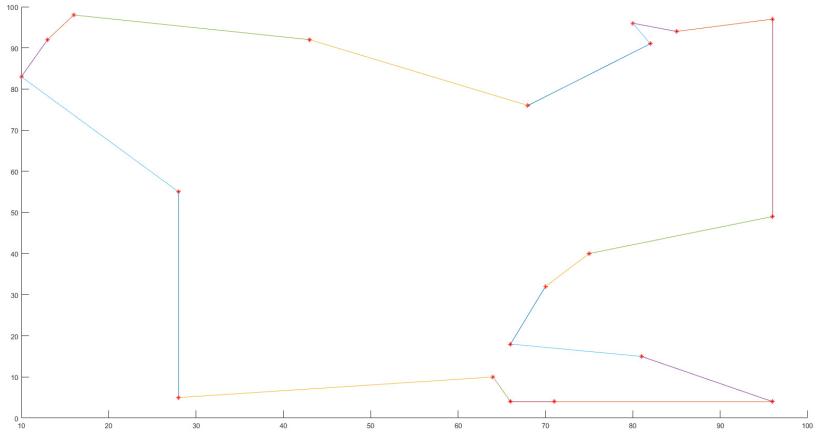

Figure 3. Finalized traveler routes
图 3. 最终求得的旅行商路线

(1) 最终求得的旅行商路线图像特点

图 3 显示了经过模拟退火优化后的旅行路径，即最优解。

路径更加平滑，交叉点减少，总距离缩短，说明算法有效减少了路径的冗余部分。

依然是闭合路径，保证了 TSP 的约束条件，即每个城市只访问一次，最后返回起点。

(2) 图 3 关键分析

1. 路径优化效果显著：

经过扰动与评估和解的接受模块的反复调整，最终路径比初始解更加合理。

交叉点减少，意味着旅行商走的冗余路线减少，提高了路径质量。

2. 收敛情况：

若最终路径仍然较长或仍有较多交叉，可能是：降温速率过快，导致未能充分优化(0.99 可适当调整为 0.95)。扰动方式不合理，可以考虑 2-opt 或 3-opt 交换方法，进一步减少路径冗余。

3. 局部最优问题：若路径仍未达到全局最优，可以增加 iter 以提高搜索能力，避免算法过早收敛。接受准则：在模拟退火(SA)算法中，解的接受准则是其核心机制之一，能够有效平衡局部搜索与全局搜索。当新解比当前解更优时，算法会无条件接受新解。当新解较差时，算法会根据一个概率来决定是否接受新解。这个概率由 exp(-delta_e/temperature)给出，其中 delta_e 是新解与当前解的差值。优化建议见表 2 图像分析图。

4.3.3. 迭代过程收敛曲线分析

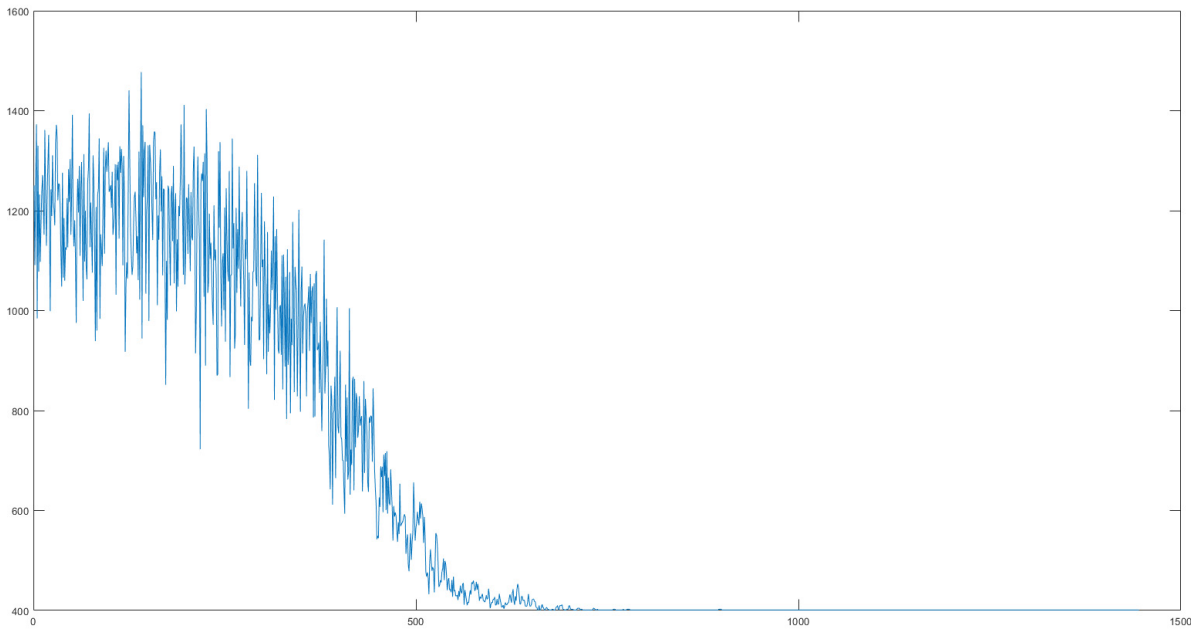

Figure 4. Traveling distance obtained at each iteration
图 4. 每次迭代求得的旅行距离

(1) 每次迭代求得的旅行距离图像特点

图 4 当中 X 轴代表迭代次数，Y 轴代表当前路径总长度。

曲线走势：初始阶段(温度较高)：曲线波动较大，路径总距离变化剧烈。中期(温度逐渐降低)：路径优化逐步稳定，总长度减少，波动幅度减小。后期(接近终止条件)：曲线趋于平稳，最终收敛到一个较优解。

(2) 关键分析

初始波动大：

由于 exp(-delta_e/temperature) > rand()允许接受较差解，因此初期路径可能有较大变动，探索解空间。

中期平稳下降：说明算法正在优化路径，较差的解被逐步淘汰，优质解被保留。

后期收敛：当温度 temperature < 0.001 时，路径不再发生显著变化，说明优化完成。如果曲线收敛太快：可能是降温速率过快，建议 0.99 改为 0.95，让算法有更多机会跳出局部最优解。如果曲线一直波动，未能收敛：可能是 *iter* 设定过小，或者初始温度 T_init 设定不合理。优化建议见表 2 图像分析图。

Table 2. Image analysis diagram
表 2. 图像分析图

图像类别	特点	分析与优化建议
初始解路径	路径随机，较长，交叉较多	说明需要优化，模拟退火算法的必要性
最终优化路径	交叉减少，路径更短	说明 SA 优化成功，但可进一步优化扰动方式
迭代过程收敛曲线	早期波动，中期平稳，后期收敛	说明 SA 算法有效，但可调整降温速率提升收敛效果

4.3.4. 时间复杂度分析

模拟退火(SA)算法的时间复杂度主要取决于以下因素：

(1) 温度衰减次数、每个温度下的迭代次数、以及每次状态转移的计算复杂度。

温度衰减次数：SA 逐步降低温度，通常采用指数衰减策略，整体降温过程的迭代次数与初始温度、终止温度和降温速率相关，通常呈对数增长。

(2) 每个温度下的迭代次数：通常与问题规模(城市数量)成正比或平方级增长，决定了在每个温度下执行多少次状态更新。

(3) 状态转移计算复杂度：TSP 采用 2-opt 或 3-opt 交换进行路径优化，每次计算的复杂度通常是线性或平方级。

综合来看，SA 算法在 SP 问题上的时间复杂度一般在 $O\left(n^2\right)$ 到 $O\left(n^3\right)$ 之间，具体取决于参数设置。相比于穷举法和动态规划，SA 能有效降低计算量，但在大规模 TSP 问题上仍可能面临计算开销较大的问题。优化方向包括改进温度衰减策略或结合其他优化算法以提升效率。

5. 总结与展望

模拟退火算法在解决旅行商问题(TSP)中表现出了其独特的优势。作为一种启发式搜索算法，它通过模拟物理退火过程，有效地在解空间中寻找近似最优路径[6]。这种方法不仅规避了传统优化方法可能面临的局部最优陷阱，还在处理大规模城市节点时展现了较高的效率。通过逐步降低系统的“温度”，模拟退火算法能够在全局范围内搜索，以概率的方式接受较差的解，从而有机会跳出局部最优，找到更接近全局最优的解。在实际应用中，模拟退火算法已经成为解决 TSP 问题的有力工具，为路径规划和优化提供了切实可行的方案。然而，模拟退火算法仍然存在一些局限性，如参数敏感性和收敛速度等问题[7]。

未来研究可能关注如何改进算法以提高求解质量和效率，例如通过引入其他优化策略或结合其他算法(如遗传算法、蚁群算法等)来形成混合算法；或探索模拟退火算法的并行化实现，以加快求解速度；又或者进行应用扩展，除了解决 TSP 问题外，模拟退火算法在解决车辆路径问题(VRP)、作业车间调度问题(JSP)等一系列问题中也有广泛的应用前景。

综上所述，模拟退火算法在解决 TSP 问题上已取得了显著成效，但仍具有广阔的发展空间和研究价

值。通过不断的改进和创新，这一算法有望在组合优化领域发挥更大的作用。

参考文献

[1] 王芸. 基于模拟退火算法的末端车载无人机物流配送路径规划研究[J]. 自动化与仪器仪表, 2025(2): 247-251.

[2] 徐刘伟, 张引弟, 鲁宇涛, 盛钲钦, 梁晓博, 宋珍珍. 基于模拟退火算法的盐穴储气库注气方案多目标优化[J]. 科学技术与工程, 2025, 25(5): 1887-1895.

[3] 于豪, 张伦铨, 林益晟, 许千千, 王鑫, 彭皓. 基于混合预测与多目标模拟退火算法的电商仓库品类分仓规划研究[J]. 中国设备工程, 2025(4): 247-250.

[4] 李从庆. 基于模拟退火算法的车辆取送货运输路径研究[J]. 郑州铁路职业技术学院学报, 2025, 37(1): 49-53.

[5] 史丰源, 欧阳丹彤, 张立明. 足够接近的旅行商问题研究综述[J]. 吉林大学学报(理学版), 2025, 63(1): 114-123.

[6] 余振, 董凡. 基于改进灰狼优化算法的多旅行商问题研究[J]. 郑州铁路职业技术学院学报, 2024, 36(4): 49-53.

[7] 古婵, 付燕, 叶圣丽. 基于着色旅行商问题的旅游路线规划[J]. 计算机科学, 2024, 51(S2): 165-172.