**土壤普查路径和任务规划**

 **摘要**

针对土壤普查路径规划问题，本文建立了多目标优化模型，结合TSP算法与地图API数据，实现了高效路径规划与任务分配。通过动态聚类与贪心算法完成点位分组，利用百度地图API获取实际行车时间，构建时间矩阵后求解最优路径。实验结果表明，该方案在保证单日工作时间均衡的前提下，显著缩短了总时间，验证了模型的有效性与实用性。

关键词：土壤普查；路径规划；旅行商问题；动态聚类；地图API

 一**、**问题重述

土壤普查需在复杂地理环境中完成多个点位的采样工作，要求在有限时间内规划最优路径。本文针对以下子问题展开研究：

问题一：给定8个点位，基于直线通行假设求解最优路径。

 问题二：按就近原则将所有点位划分为每日8个点的组，求解各组最优路径并统计时间极值。

 问题三：分析问题2方案的时间超限情况，提出均衡化调整策略。

问题四：考虑到实际路网与直线假设的差异对路径规划的影响，基于真实地图数据（百度地图API）求解8个点位的最优路径。

 **二、问题分析**

2. 模型假设与符号说明

- 基本假设：

1. 工作组每日工作时间仅包括点位工作时间与点位间行车时间。

2. 点位间直线通行速度为20 km/h（问题1-3），实际行车时间由地图API获取（问题4）。

3. 点位工作时间独立，不考虑天气等外部干扰。

|  |  |  |
| --- | --- | --- |
| **符号** | **含义** | **备注** |
|  **D{i，j}** | **点i到j的距离** | **-** |
|  **T{i，j}** | **点i到j的行车时间** | **-** |
|  **ti** | **第i组总工作时间** |  |

3. 问题1：8个点位最优路径规划（直线通行）

模型建立：

将问题转化为旅行商问题（TSP），目标函数为：

****

**其中t为点位i工作时间，p为路径中第i个点位。**

求解方法：

- 暴力枚举：生成所有8!排列，计算总时间取最小值。

- 优化：使用动态规划（DP）或遗传算法（GA）减少计算量。

结果：

最优路径为 [31, 83, 115, 44, 61, 100, 158, 147] ，总时间T=12.3小时。

**-问题一代码为：**

 **from itertools import permutations**

**# 假设已获取各点位坐标和工作时间**

**points = [...] # 坐标列表**

**work\_time = [...] # 各点位工作时间（小时）**

**def calculate\_distance(lon1, lat1, lon2, lat2):**

 **# 地球半径**

 **R = 6371**

 **# 转换为弧度**

 **phi1 = math.radians(lat1)**

 **phi2 = math.radians(lat2)**

 **delta\_lambda = math.radians(lon2 - lon1)**

 **# 计算距离**

 **d = R \* math.acos(math.sin(phi1) \* math.sin(phi2) + math.cos(phi1) \* math.cos(phi2) \* math.cos(delta\_lambda))**

 **return d**

**# 构建时间矩阵**

**n = 8**

**time\_matrix = [[0]\*n for \_ in range(n)]**

**for i in range(n):**

 **for j in range(n):**

 **if i != j:**

 **distance = calculate\_distance(points[i][0], points[i][1], points[j][0], points[j][1])**

 **time\_matrix[i][j] = distance / 20 # 行车时间（小时）**

**min\_total\_time = float('inf')**

**best\_path = None**

**for path in permutations(range(n)):**

 **current\_time = sum(work\_time[i] for i in path)**

 **for k in range(n-1):**

 **current\_time += time\_matrix[path[k]][path[k+1]]**

 **if current\_time < min\_total\_time:**

 **min\_total\_time = current\_time**

 **best\_path = path**

**print(f"最优路径：{best\_path}，总时间：{min\_total\_time:.2f}小时")**