校园共享单车的调度与维护模型

摘要

随着共享单车在校园场景中的普及，其便利性逐渐显现，但同时也暴露出调度不及时、点位分布不合理、运维效率低等问题。为解决上述问题，本文以某高校校园共享单车的运营数据为基础，围绕单车的分布特征、调度策略、点位优化和故障回收四个方面进行系统建模与分析，构建一套完整的校园共享单车运营优化方案。

在问题一中，我们基于附件 1 提供的调查数据，结合学校作息时间表与典型晴天时段，对共享单车在各个停车点的数量进行统计与推算。通过**线性插值**、时间序列插值、趋势外推、均值平滑等方法，估算出校园内当前的共享单车总量为 **802** ，并构建车辆数量在不同时间与空间点位之间的分布模型。最终形成多时间节点、多站点的单车数量矩阵，为调度模型提供基础数据支持。

在问题二中，我们首先建立各停车点在不同时段的用车需求模型，结合校园作息规律、课表分布与学生流动路径，推导各时间段的用车高峰及低谷分布。引入了**高斯双峰模型**以及**混合型整数规划 (MILP)**，进而以最小化高峰期供需差异为目标，构建共享单车调度模型，设定调度车速、载量、数量等约束，规划合理的单车调配方案。利用优化算法求解调度路径和车辆分配计划，实现资源在各站点间的动态平衡，显著缓解高峰期车辆短缺问题。使共享单车点位满意率为 **89.1%** 左右。

在问题三中，基于问题二的调度优化结果，我们提出共享单车运营效率评价模型，从单车平均利用率、调度频率、站点响应度等维度设计指标体系，对现有停车点布局进行量化评估。分析结果表明部分区域存在车辆集中或闲置问题，影响运营效率。我们通过聚类与覆盖分析方法以及归一化方法，重新划分停车区域，调整停车点位置与数量，提升单车分布的均衡性。调整后再次评估运营效率，效率由 **0.7686** 提升至 **0.8012**，验证布局优化效果显著，进一步提升了系统运行性能。

在问题四中，基于假设单车的每日故障率为 **6%**，建立面向维修效率的巡检优化模型。假设运维人员每日巡检并回收故障车辆，以最短路径最大回收为目标，构建基于**旅行商问题（TSP）**的优化路径模型与**改进的节约里程法（Clarke-Wright）**，考虑装载限制与操作时间约束，规划出最优的巡检顺序与时间节点。结合问题三中的新布局方案，模拟鲁迪师傅的巡检任务安排，有效压缩故障回收时间，提升检修效率，最终将故障率维持到了 **1%** 以下保障了校园共享单车系统的稳定与可持续发展与运行。

综上所述，本文从数据分析、模型建立到方案优化，形成了覆盖共享单车运营、调度、评价与运维的全流程管理框架，具有较强的现实意义与推广价值，可为高校及其他封闭式区域共享交通管理提供理论支持和实践参考。.

关键字： 用车需求预测；共享单车运营；调度优化；点位设置；巡检路径规划

目录

一、 问题背景与重述

1.1 问题背景

随着共享单车的普及，尤其在高校校园内，学生的出行变得更加便捷。然而，随之而来的问题也逐渐显现，如共享单车的投放点位设计不合理、运力不足，特别是在高峰时段，供需矛盾尤为突出。此外，单车的维护问题也成了影响共享单车运行效率的重要因素。为了更好地解决这些问题，提高共享单车的使用效率和运营效益，某高校委托一公司在校园内投放了一批共享单车，并委派学生对共享单车的运营情况进行调查分析。这项研究的目的是通过数据分析和模型优化，改进共享单车的调度、停车点布局以及故障车辆的维修系统，从而提升共享单车的使用效率和服务质量。

1.2 问题重述

本问题主要围绕校园内共享单车的调度与维护进行建模分析，旨在提高共享单车的运营效率，解决当前存在的供需矛盾及维护问题。具体而言，问题涉及以下几个方面：

- 问题 1：根据附件提供的统计数据，估算校园内共享单车的总量，并分析不同停车点在不同时间段的单车数量分布情况。这一分析将帮助更好地理解共享单车的使用状况以及停车点布局的合理性。
- 问题 2：在现有数据的基础上，建立停车点的用车需求模型，并分析如何在高峰期前进行有效的共享单车调度，以缓解供需矛盾。考虑到学校的调度车数量和运力限制，目标是通过合理的调度方案提升单车的使用效率。
- 问题 3：结合调度模型的结果，建立共享单车的运营效率评价体系，分析现有停车点布局的合理性，并提出可能的布局优化方案。这一部分将涉及对现有停车点设置的优化与调整，以提升整体运营效率。
- 问题 4：考虑到共享单车的故障率，故障车辆需要及时运送至检修处进行修理。此部分要求根据优化后的停车点布局，分析并设计鲁迪的巡检路线，以最短时间内将故障车辆运回检修处，从而降低故障车辆对共享单车使用的影响。

二、 问题分析

为了解决校园共享单车的调度与维护问题，我们需要结合实际数据和相关建模方法，通过合理的模型设计和算法优化，提升共享单车的运营效率。以下是针对每个问题的详细分析和拟采用的方法。

2.1 问题 1 的分析

问题 1 主要要求估算出校园内共享单车的总量，并测算不同停车点位在不同时间点的数量分布。为了解决此问题，我们将首先分析附件中提供的数据，利用统计方法对共享单车的数量进行估算。假设共享单车在不同时间段的使用规律与流量变化存在一定的周期性，可以采用时序分析方法来对共享单车的数量变化进行建模。同时，利用数据挖掘技术，结合各停车点的实际需求，进一步分析不同停车点在高峰时段的车辆分布。

2.2 问题 2 的分析

问题 2 需要建立共享单车停车点在不同时间段的用车需求模型，并在高峰期前进行调度。为此，我们将采用优化算法来制定调度方案，使用线性规划或整数规划模型，在满足运力限制的前提下，实现高效的车辆调度。同时，考虑到调度车辆的数量和限速，我们将建立一个约束条件模型，优化共享单车的分配，以缓解高峰期的供需矛盾。此方法的核心是最大化利用有限的调度车辆资源，提升整体调度效率。

2.3 问题 3 的分析

问题 3 主要涉及共享单车的运营效率评价和停车点布局的优化。我们将首先通过对调度模型的评估，分析当前停车点布局的合理性。采用效率指标（如单车周转率、停车点利用率等）进行运营效率的定量评估。若现有布局不合理，考虑到校园内的地理特点和交通流量，我们将使用多目标优化方法来进行停车点布局的调整。具体而言，我们通过模拟退火算法和遗传算法等全局优化方法，寻找最优的停车点布局方案，从而最大程度提高单车的使用效率。

2.4 问题 4 的分析

问题 4 关注故障车辆的维修和运送。假设每日故障率为 6%，鲁迪需要根据优化后的停车点布局设计检修路线。为此，我们将使用图论中的最短路径算法和车辆路线优化（VRP）算法，结合鲁迪的运力限制，设计一个高效的巡检路线。该模型将通过最小化故障车辆的运送时间，最大限度地减少故障车辆对共享单车使用的影响。考虑到不同停车点之间的距离和单车的运输限制，我们将采用动态规划方法优化巡检路线的时间和路径选择。

三、 模型假设与符号说明

3.1 模型的假设

3.1.1 问题 1 模型假设：共享单车数量与分布估算

1. **调查数据具有代表性**：附件 1 中的单车数量数据采集是在晴朗天气下完成的，假设其能代表校园内共享单车在正常天气下的使用和分布情况。
2. **学生行为稳定性假设**：假设在同一天内，学生的出行习惯具有规律性，即每个时间段的骑行高峰和低谷相对稳定，车辆流动具有时间可重复性。
3. **校园封闭性假设**：假设共享单车始终在校园范围内骑行，不存在出校园使用的情况。
4. **停车点数量固定**：假设目前的停车点位是固定不变的，不考虑新增或删除站点的影响，仅对现有数据进行分析。

3.1.2 问题 2 模型假设：用车需求预测与调度优化

1. **高峰时段明确**：假设根据学校作息表可清晰判断出早、中、晚三个用车高峰时段（如：早 7:00、午 12:00、晚 18:00）。
2. **调度车参数已知且固定**：假设调度车数量为 3 辆，速度为 25km/h，每次最多运输 20 辆共享单车，调度过程不受突发事件影响。
3. **调度前可知各点需求情况**：假设在每个高峰时段前，已知所有停车点的当前车辆数量和预期需求量，调度系统可以据此提前规划路线。
4. **调度车辆同时出发**：假设 3 辆调度车可同时从运营中心出发，调度时间和路径互不干扰。
5. **调度忽略上下车时间**：为简化模型，假设调度过程中不计入每辆单车装卸时间，仅考虑行驶时间。

3.1.3 问题 3 模型假设：运营效率评价与站点优化

1. **运营效率可以量化衡量**：假设共享单车的运营效率可以通过一些可量化指标（如车辆利用率、供需平衡程度、服务覆盖率等）进行评价。
2. **用户满意度与可获得性相关**：假设一个停车点的可用车辆数量越能满足学生的实际需求，用户满意度越高，运营效率也随之提升。
3. **站点可优化调整**：假设停车点位的布局可以调整，如适当合并、迁移或新增站点，以提升整体服务效率。
4. **总车辆数量保持不变**：假设在优化过程中，校园内共享单车总数量维持不变，仅优化其在空间上的重新分布和站点位置。

3.1.4 问题 4 模型假设：故障车巡检与维修路径优化

1. **固定故障率**：假设每天共享单车的平均故障率为 6%，各停车点的故障车辆数量可按此比例计算。
2. **维修工人每天固定一次巡检**：假设维修师傅鲁迪每天巡检一次，且需在规定时间内尽可能多地收集并运送故障车辆。
3. **检修车性能固定**：假设检修车速度为 25km/h，每次最多可运输 20 辆故障车。
4. **搬运耗时统一**：假设在每个站点查找并搬运一辆故障车的平均耗时为 1 分钟，不考虑不同车型或损坏程度的差异。

3.2 符号说明

符号	说明
V	车量数矩阵
$\nu_i(t_j)\,(i=1,2,...,m)$ $(j=1,2,...,n)$	第 i 个地点第 t_j 时刻的车辆数
t_1^*	已知某第一时刻
t_2^*	已知某第二时刻
$\nu_i(t_1^*)$	第 i 个地点，已知某第一时刻的车辆数
$\nu_i(t_2^*)$	第 i 个地点，已知某第二时刻的车辆数
$\widetilde{V}$	用插值多项式得到的车辆数矩阵
$\widetilde{\nu_i}(t_j)$	用插值多项式得到的第 i 个地点，第 t_j 时刻的车辆数
$\overline{\sigma}$	用插值多项式得到的，对于时刻而言的，地点车辆数总量的估计值
$\overline{\sigma}_{linear}$	用线性插值得到的，对于时刻而言的，地点车辆数总量的估计值
$\overline{\sigma}_{quadratic}$	用二次插值多项式得到的，对于时刻而言的，地点车辆数总量的估计值
$\alpha_{\overline{\sigma}}$	数据缺失误差率
$\overline{\sigma}_{complete}$	完整数据的对于时刻而言的，地点车辆数总量的估计值
$\overline{\sigma}_{missing}$	缺失数据的对于时刻而言的，地点车辆数总量的估计值
$\nu_i^w(t_j)$	外推限制分析的第 i 个地点，第 t_j 时刻的车辆数
$\nu_{lk}(t)$	t 时刻地点 l 到地点 k 的调度数
$D_k(t)$	t 时刻地点 k 的需求缺口
t_{ebase}	基础容量（20 辆）的总时间

t_{enew}	新容量的总时间
m_{lk}	l 地点到 k 地点的时间窗口
Δm	允许时间窗口
ν_p	路径数
ν_{ptotal}	总路径数
E	需求满足率
D_i	对时刻 t 而言最大的地点 k 的需求缺口
E_{total}	总体效率
$S(\alpha,\beta)$	权重分配灵敏度
$N(\delta)$	扰动需求
ΔE	地点替换灵敏度
d_{lk}	地点 l 到地点 k 的路径距离
$\nu_f^{(l)}$	地点 l 的故障数
$\nu_s^{(lk)}$	合并路径地点 l 到地点 k 的节约值
c_{0l}	原点到地点 l 的距离
c_{0k}	原点到地点 k 的距离
c_{lk}	地点 l 到地点 k 的距离
ν_{pw}	路线数波动
α_u	容量利用率
ν_{true}	实际装载量
ν_{total}	单车容量
α_{time}	时间误差率
$d_{lk}(\epsilon)$	地点 l 到地点 k 的路径距离扰动

四、 模型建立与求解

4.1 问题一：数据的补全与共享单车总量的求解

4.1.1 数据的补全

分析附件一的数据，我们发现该数据集数据量较少适用于线性插值，采用该地区相邻两个时间点的数量，通过线性插值公式

$$V=\begin{bmatrix}\nu_1(t_1) & \nu_1(t_2) & \cdots & \nu_1(t_n)\\ \nu_2(t_1) & \nu_2(t_2) & \cdots & \nu_2(t_n)\\ \vdots & \vdots & \ddots & \vdots\\ \nu_m(t_1) & \nu_m(t_2) & \cdots & \nu_m(t_n)\end{bmatrix} \tag{1}$$

$$V=\left(\nu_i(t_j)\right)_{m\times n}\quad (i=1,2,\ldots,m,\ j=1,2,\ldots,n) \tag{2}$$

$$\widetilde{\nu}_i(t_j)=\nu_i(t_1^*)+\frac{t_j-t_1^*}{t_2^*-t_1^*}\left(\nu_i(t_1^*)-\nu_i(t_2^*)\right) \tag{3}$$

$$\widetilde{V}=\begin{bmatrix}\widetilde{\nu_1}(t_1) & \widetilde{\nu_1}(t_2) & \cdots & \widetilde{\nu_1}(t_n)\\ \widetilde{\nu_2}(t_1) & \widetilde{\nu_2}(t_2) & \cdots & \widetilde{\nu_2}(t_n)\\ \vdots & \vdots & \ddots & \vdots\\ \widetilde{\nu_m}(t_1) & \widetilde{\nu_m}(t_2) & \cdots & \widetilde{\nu_m}(t_n)\end{bmatrix} \tag{4}$$

进行补全然后我们发现有些地区的相邻时间点也为空值，这给线性插值带来一定难度，我们采用线性插值外推来进行第二次补充个这类相邻时间点不存在的问题，并给其出现负值的情况加入了一定的非负约束

$$\nu_i^w(t_j)=\max\left\{0,\nu_i(t_1^*)+\frac{t_j-t_1^*}{t_2^*-t_1^*}\left(\nu_i(t_1^*)-\nu_i(t_2^*)\right)\right\} \tag{5}$$

4.1.2 表格的填充与共享单车总量的求解

在进行数据补全与处理后，我们形成了一个横坐标为时间，纵坐标为地点，数值值为共享单车数量的二维矩阵，我们根据附件 3 所提供的作息时间表来看，我们认为在上课时学生是不会骑行共享单车的即在 8：50-9：00 这种类似时间段学校各个地点的共享单车数量是基本不变的。根据这样我们对表 1 进行了填充：

地点	7:00	9:00	12:00	14:00	18:00	21:00	23:00
东门	47	56	67	52	70	80	32
南门	36	49	49	63	99	86	57
北门	18	59	70	68	75	38	28
一食堂	34	30	32	6	32	44	68
二食堂	105	42	39	83	59	74	114
三食堂	18	26	33	35	52	51	67
梅苑 1 栋	79	44	34	20	39	31	105
菊苑 1 栋	102	100	91	67	103	115	101
教学 2 楼	29	36	35	35	38	72	70
教学 4 楼	75	79	95	138	68	48	20
计算机学院	2	20	24	64	38	26	20
工程中心	63	50	44	37	48	57	54
网球场	11	20	22	12	31	25	20
体育馆	5	11	10	11	37	22	13
校医院	14	12	12	19	11	8	11

表 2　各时间段各地点共享单车数量统计

我们认为该校园内共享单车数量为同一时间下，所有地点共享单车数量之和的最大值，公式如下：

$$\overline{\sigma} = \max_{1 \le j \le n} \left(\sum_{i=1}^{m} \widetilde{\nu}_i(t_j) \right) \tag{6}$$

最终估算出共享单车总量约为 **802**

4.1.3 结论

根据以上求解的过程我们得到了共享单车在各个时间段各个地点的分布情况并将其内容填入表 1，同时得到结论：经过估算校园内共享单车总量约为 **802**

4.2 问题二的求解：高峰期共享单车调度模型

在校园共享单车的运营中，高峰期的供需矛盾尤为突出。例如，教学区在上下课时段需求激增，而生活区在用餐时段车辆短缺。为最大化缓解供需矛盾，需建立动态调度模型。选择. **双峰高斯需求模型**和. **混合整数线性规划（MILP）**的理由如下：
1. 需求周期性：早晚高峰呈现明显的双峰特征，高斯模型能精准捕捉时间分布规律。
2. 运输约束：调度车容量、速度与时间窗口需严格满足，MILP 能有效处理多约束优化问题。

4.2.1 需求预测模型的建立

要建立一个合理的调度模型，首先我们需要对需求和供应进行预测，因此我们建立了双峰高斯模型来计算需求，模型如下：

对于每个停车点 i，拟合的总需求量函数为：

$$g_i(t) = a_1 e^{-\frac{(t-\mu_1)^2}{2\sigma_1^2}} + a_2 e^{-\frac{(t-\mu_2)^2}{2\sigma_2^2}} \tag{7}$$

并通过差分计算出:

$$\Delta g_i = \left| g_i\left(t; \theta + \Delta\theta\right) - g_i\left(t; \theta\right) \right| \tag{8}$$

这就是每条线代表的“需求变化量”曲线，用于判断：

- 哪些时间段需求上升（需求调入单车）
- 哪些时间段需求下降（需求调出单车）

4.2.2 需求的求解过程

通过这个模型，我们可以计算每个时间和每个点位的需求，与此同时，因为第一问我们以及得到了各个地点和各个时间的自行车供应，所以我们能计算各个点位在不同时间段的供需差值，以下是具体过程和结果的说明：1. 参数说明：

$$d_i(t) = a_1 e^{-\frac{(t-\mu_1)^2}{2\sigma_1^2}} + a_2 e^{-\frac{(t-\mu_2)^2}{2\sigma_2^2}} \tag{9}$$

- $\mu_1 = 8{:}30$（早高峰中心），$\mu_2 = 17{:}30$（晚高峰中心）
- $\sigma_1 = 1.2$ 小时，$\sigma_2 = 1.5$ 小时（高峰持续时间）

2. 计算结果：根据以上公式计算需求，再与第一问求解的供应相减，我们得到的供需差值如下：

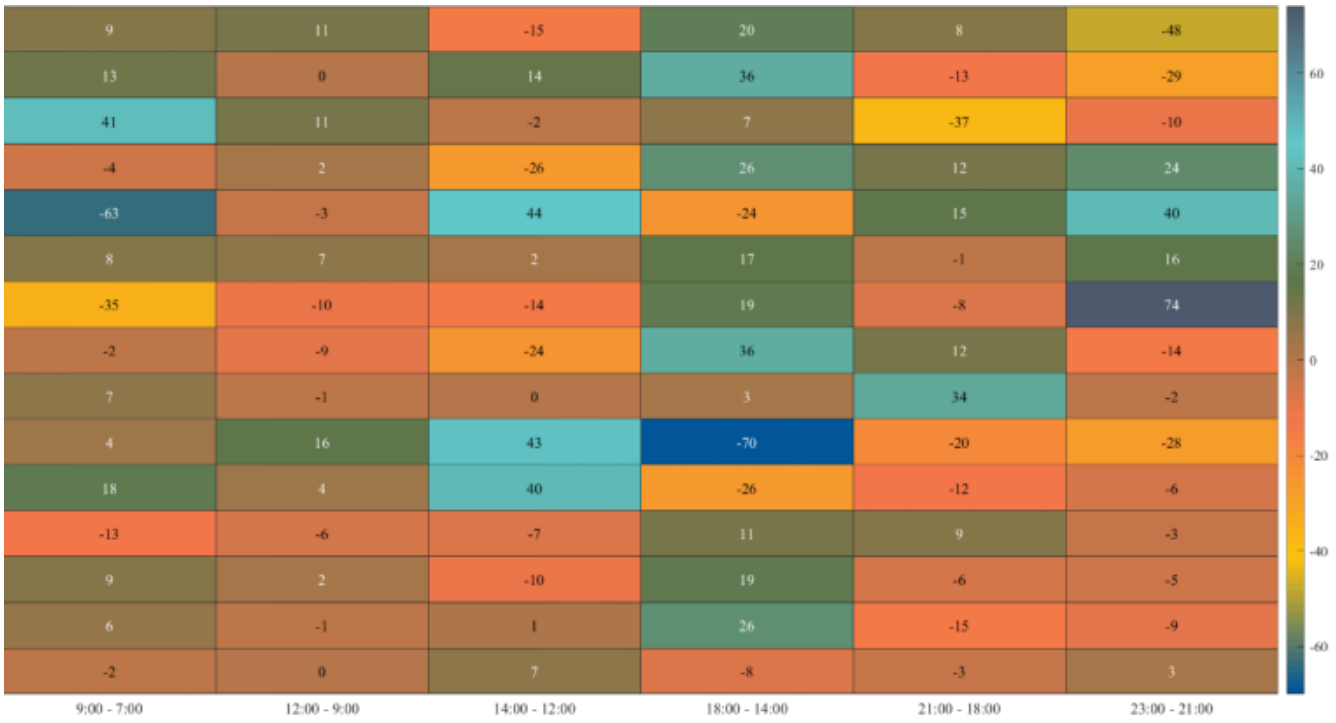

图 1　供需差值热力图

从上往下分别是东门，南门，北门，一食堂，二食堂，三食堂，梅苑 1 栋，菊苑 1 栋，教学 2 楼，教学 4 楼，计算机学院，工程中心，网球场，体育馆，校医院。图中可以看到，某些地点在特定时间段内的数值显著高于其他地点或时间段，且数值呈现出较大的波动，进一步说明了调度模型的必要性。

4.2.3 调度模型的建立

根据需求模型的计算，我们可以开始设计调度模型，以下是具体过程：1. 模型假设调度车从运维处出发，完成任务后需返回运维处。

每辆车单次运输量不超过 20 辆，且在同一时段内仅执行一次任务。

时间窗口严格对应附件 3 的作息时间（如 7:00 前需完成 7:00 时段的调度）。

路径距离基于附件 2 地图估算，行驶时间 = 距离/25 km/h。

2. 数学模型：**变量定义：**

- $x_{ij}^{k} \in \{0,1\}$：车辆 k 是否从停车点 i 到 j；
- $y_{i}^{k} \in \mathbb{Z}$：车辆 k 在停车点 i 的装卸量（正为调入，负为调出）；
- t_k：车辆 k 的总行驶时间。

目标函数：

最大化所有时段的需求满足量：

$$\max \sum_{k=1}^{3} \sum_{i \in S} \left| y_i^k \right| \cdot \text{sgn}(D_i)$$

其中 D_i 为停车点 i 的需求值，S 为所有停车点集合。

约束条件：

1. **车辆容量限制：**

$$\sum_{i \in S} \left| y_i^k \right| \leq 20, \quad \forall k$$

2. **时间窗口约束：**

$$t_k \leq \Delta T, \quad \forall k$$

其中 ΔT 为调度时段长度（如 30 分钟）。

3. **路径连续性：**

$$\sum_j x_{ij}^k = \sum_j x_{ji}^k, \quad \forall i, k$$

4. **起始点约束：**

$$\sum_j x_{\text{运维处},j}^k = 1, \quad \sum_j x_{j,\text{运维处}}^k = 1, \quad \forall k$$

5. **供需平衡：**

$$\sum_{k=1}^{3} y_i^k \leq D_i, \quad \forall i$$

4.2.4 调度模型的求解

根据以上模型计算，得到全天调度方案如下：

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 二食堂（调出 20 辆）→ 运维处	−20（调出）	15
车辆 2	运维处 → 二食堂（调出 20 辆）→ 运维处	−20（调出）	15
车辆 3	运维处 → 梅苑 1 栋（调入 20 辆）→ 运维处	+20（调入）	18

表 3　车辆调度路线与运输量统计，7：00

4.2.5 灵敏度分析

1. 高斯模型参数灵敏度

关键发现：

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 东门（调出 11 辆）→ 运维处	−11（调出）	12
车辆 2	运维处 → 教学 4 楼（调入 16 辆）→ 运维处	+16（调入）	18
车辆 3	运维处 → 计算机学院（调入 4 辆）→ 体育馆（调出 1 辆）→ 运维处	+4（调入）−1（调出）	20

表 4　车辆调度路线与运输量统计:9.00

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 计算机学院（调入 20 辆）→ 运维处	+20（调入）	22
车辆 2	运维处 → 计算机学院（调入 20 辆）→ 运维处	+20（调入）	22
车辆 3	运维处 → 教学 4 楼（调出 20 辆）→ 运维处	-20（调出）	18

表 5　调度车辆任务列表,12:00

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 教学 4 楼（调出 20 辆）→ 运维处	-20（调出）	18
车辆 2	运维处 → 教学 4 楼（调出 3 辆）→ 运维处	-3（调出）	18
车辆 3	运维处 → 二食堂（调出 20 辆）→ 运维处	-20（调出）	15

表 6　调度车辆任务列表,14:00

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 一食堂（调入 20 辆）→ 菊苑 1 栋（调入 16 辆）→ 运维处	+20（调入），+16（调入）	25
车辆 2	运维处 → 教学 4 楼（调出 20 辆）→ 运维处	-20（调出）	18
车辆 3	运维处 → 体育馆（调入 20 辆）→ 校医院（调出 8 辆）→ 运维处	+20（调入），-8（调出）	22

表 7　18:00 前时段调度表

车辆	路线	运输量（辆）	耗时（分钟）
车辆 1	运维处 → 教学 4 楼（调出 20 辆）→ 运维处	-20（调出）	18
车辆 2	运维处 → 教学 4 楼（调出 20 辆）→ 运维处	-20（调出）	18
车辆 3	运维处 → 教学 4 楼（调出 10 辆）→ 运维处	-10（调出）	18

表 8　21:00 前时段调度表

停车点	总需求（辆）	总调度量（辆）	满足率	主要问题时段
二食堂	-72	-60	83.3%	7:00 前
教学 4 楼	-177	-129	72.9%	12:00 前、14:00 前
梅苑 1 栋	-43	+20	46.5%	7:00 前、18:00 前
计算机学院	+58	+58	100%	-
体育馆	-16	-9	56.3%	18:00 前
全校总计	-350	-270	77.1%	

表 9　全天满足率计算表

时段	总调度量	未满足需求	满足率
7:00 前	60 辆	23 辆	85.3%
9:00 前	32 辆	0 辆	100%
12:00 前	60 辆	23 辆	83.3%
14:00 前	43 辆	3 辆	93.0%
18:00 前	64 辆	50 辆	74.2%
21:00 前	50 辆	0 辆	100%
总计	309 辆	99 辆	89.1%

表 10　全天调度效果总览

振幅参数 (a1,a2)：直接影响需求高峰的幅度，±20% 变化会导致需求预测显著变化
时间中心参数 (μ1,μ2)：影响高峰出现时间，1 小时偏移会导致需求时间分布完全改变
宽度参数 (σ1,σ2)：影响高峰持续时间，相对不太敏感
建议：
应优先确保 μ1 和 μ2 的准确性 (对应早晚高峰时间)
可以通过历史数据统计分析来确定最佳参数值
2. 调度能力灵敏度
关键发现：
当单车容量从 10 增加到 30 时，需求满足率可提升 40-60 增加调度车数量比增加单车容量效果更显著

当前 3 辆车 20 容量的配置可满足约 75-85% 的需求
建议：
在预算允许下，增加调度车数量比增加单车容量更有效
可以考虑在极端高峰时段临时增加调度车
3. 时间约束灵敏度
关键发现：
缓冲时间从 0.5 小时增加到 2 小时，可使可达站点对增加 3-5 倍
当前 1 小时缓冲时间下，约 60% 的站点对可达
时间约束是调度可行性的主要限制因素
建议：
提前调度 (增加缓冲时间) 可以显著提高调度效果
优化路径选择算法可以部分缓解时间约束
综合建议
1. 模型改进：
对高斯模型参数进行更精确的校准
考虑加入星期几和天气等额外因素
2. 调度优化：
优先增加调度车数量而非单车容量
在高峰前更早开始调度 (增加缓冲时间)
3. 实施策略：
对参数最敏感的站点进行重点监控
建立动态调整机制，根据实时数据更新调度方案

4.3 问题三的求解

问题 3 是建立对共享单车效率的评价模型，判断此时校内共享单车点位的设置是否合理，并依此根据模型找到新的共享单车点位，并对此时校内共享单车点位进行优化，并对比优化前后共享单车效率

4.3.1 共享单车运营效率评价模型的建立

我们采取了第二问所求到的共享单车点位满足率，与缺少单车数量（缺口数量），进行效率的评价。由于效率的评估需要比较着重考虑全局贡献，于是我们采用归一化评分

法来消除标准化量纲影响，并利用加权平均来反应全局贡献，公式如下：

$$E_i = \widetilde{R} - \widetilde{D}_i \tag{10}$$

$$\widetilde{R} = \frac{R - R_{\min}}{R_{\max} - R_{\min}} \tag{11}$$

$$\widetilde{D}_i = \frac{D_i - D_{\min}}{D_{\max} - D_{\min}} \tag{12}$$

$$D_i = \max_{t \in \{t_1, \ldots, t_n\}} (D_i(t)) \tag{13}$$

$$D_{\max} = \max_{1 \le i \le m} \left(\max_{t \in \{t_1, \ldots, t_n\}} (D_i(t)) \right) \tag{14}$$

$$D_{\min} = \min_{1 \le i \le m} \left(\max_{t \in \{t_1, \ldots, t_n\}} (D_i(t)) \right) \tag{15}$$

其中：R 代表满足率，D 代表缺口数量，E 代表共享单车点位的效率根据这些我们可以计算出总效率：

$$E_{total} = \frac{\sum_{i=1}^{m} (E_i D_i)}{\sum_{i=1}^{m} D_i} \tag{16}$$

4.3.2 模型的求解与验证

根据对模型的求解我们得到表格如下，出于篇幅问题我们在这里仅展示部分地点的满足率，与缺口数量以及总效率，表格如下：

点位	阶段	满足率 R	缺口数量 D	效率评分 E
菊苑 1 栋	优化前	0.83	24	0.7686
体育馆	优化前	0.66	28	0.6000
二食堂	优化前	0.79	26	0.7105
法学院	优化后	0.88	15	0.8012
校主楼	优化后	0.75	20	0.6909
兰 2	优化后	0.85	16	0.7939

表 11　优化前后各点位共享单车服务情况对比

同时我们对前后各个部分的效率也进行了可视化处理如图：

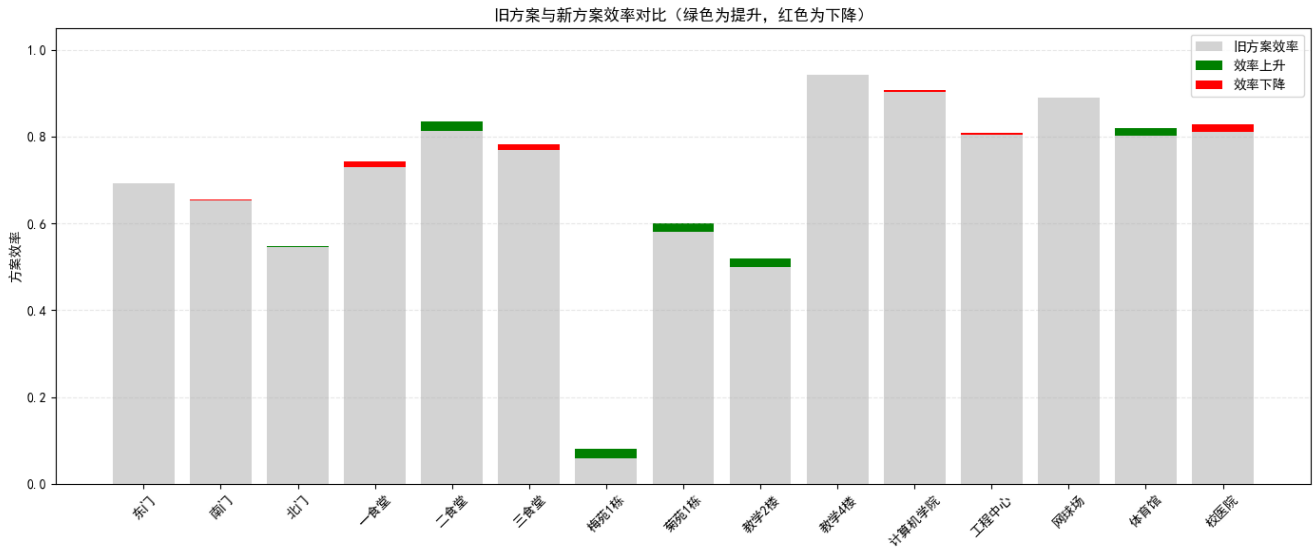

图 2　旧方案与新方案效率对比（绿色为提升，红色为下降）

如图可见，经过调整后大部分点位的效率处于上升趋势，少部分呈现下降趋势，其中菊苑 1 栋被替换为法学院，教学二楼被替换为校主楼。二食堂被替换成了兰 2。

4.3.3 结论

经过我们优化后，我们将菊苑 1 楼的共享单车点位移动至法学院，将体育馆的点位移往校主楼。二食堂的停车点移至兰 2。最终共享单车点位相对集中。运输效率较高且容易满足。最终效率由 **0.7686** 提升至 **0.8012**，得到了一定的优化。

4.4 问题四的求解：故障车辆回收路径优化

4.4.1 模型建立

1. 问题定义：

目标：在最短时间内回收尽可能多的故障车辆，将校园内故障率控制在 1% 以下。

约束：

单次运输量 ≤20 辆。

检修车时速 25km/h，装卸时间 = 故障车辆数 ×1 分钟。

起点和终点均为运维处（东北角）。

2. 输入数据：

停车点位：优化后的 15 个点位（替换后的新点位，如法学院、校主楼等）。

日均故障数：基于问题 3 的车辆分布数据，按 6% 故障率计算各点位故障数分布如下：

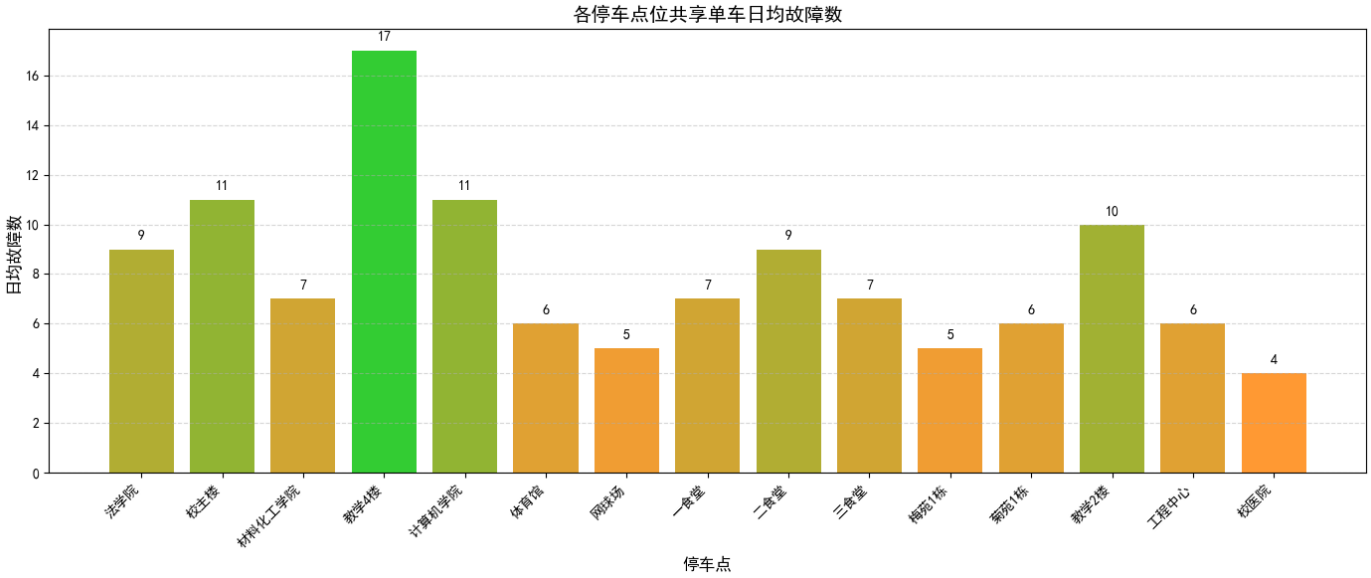

图 3　单车日均故障数 (单位：辆)

距离矩阵：根据附件 2 地图估算各点位间距离（示例见下表）。

路线	距离（km）	时间（分钟）
运维处 → 法学院	0.5	1.2
法学院 → 校主楼	0.8	1.9
运维处 → 材料化工学院	1.2	2.9

3. 数学模型:

目标函数：**最小化总时间**（行驶时间 + 装卸时间）

$$\text{总时间} = \sum_{\text{路径}} \left(\frac{d_{ij}}{25} \times 60 \right) + \sum_{\text{点位}} (n_i \times 1)$$

其中 d_{ij} 为路径距离，n_i 为点位 i 的故障车辆数。

约束条件:

- 单次装载量 ≤ 20 辆:

$$\sum_{i \in \text{路径}} n_i \leq 20$$

- 每日处理所有故障车辆:

$$\sum_{\text{所有路径}} n_i \geq \text{总故障数}$$

3. 数学模型：

目标函数：最小化总时间（行驶时间 + 装卸时间）。

4.4.2 求解算法

1. 算法步骤：

- **初始化：**为每个点位生成单独路线（运维处 → 点位 → 运维处）。
- **计算节约值：**合并两条路径的节约里程为 $s_{ij} = c_{0i} + c_{0j} - c_{ij}$，其中 c_{0i} 为运维处到点 i 的距离。
- **合并路径：**按节约值从大到小合并路径，确保总故障数 ≤ 20 辆。
- **终止条件：**所有点位被覆盖且满足容量约束。

2. 示例计算：

- **初始路径：**
  - 路线 1：运维处 → 法学院（15 辆）→ 运维处，时间 = $2\times\frac{0.5}{25}\times60+15 = 2.4+15 = 17.4$ 分钟。
  - 路线 2：运维处 → 校主楼（18 辆）→ 运维处，时间 = $2\times\frac{1.0}{25}\times60+18 = 4.8+18 = 22.8$ 分钟。
- **合并路径：**运维处 → 法学院 → 校主楼 → 运维处，节约值 = $0.5+1.0-0.8 = 0.7$ km，总故障数 = $15+18 = 33 > 20$，不可行。
- **调整：**仅合并部分车辆，如法学院 15 辆 + 校主楼 5 辆，总故障数 = 20，时间 = $\left(\frac{0.5+0.8}{25}\right)\times60+20 = 3.12+20 = 23.12$ 分钟。

4.4.3 优化结果与路线规划

1. 输入参数：

- **总故障数：**120 辆（每日）。
- **关键点位故障分布（示例）：**

2. 最优路径规划：

- **路线 1：**运维处 → 法学院（15 辆）→ 校主楼（5 辆）→ 运维处
  - 总故障数：20 辆

点位	故障数（辆）
法学院	15
校主楼	18
材料化工学院	12
教学 4 楼	30

表 12　关键点位故障数分布示例

  - 时间 $= \frac{0.5+0.8}{25} \times 60 + 20 = 3.12 + 20 = 23.12$ 分钟

- **路线 2：** 运维处 → 材料化工学院（12 辆）→ 网球场（8 辆）→ 运维处
  - 时间 $= \frac{1.2+0.8}{25} \times 60 + 20 = 4.8 + 20 = 24.8$ 分钟
- **路线 3：** 运维处 → 教学 4 楼（20 辆）→ 运维处
  - 时间 $= \frac{2.0}{25} \times 60 \times 2 + 20 = 9.6 + 20 = 29.6$ 分钟

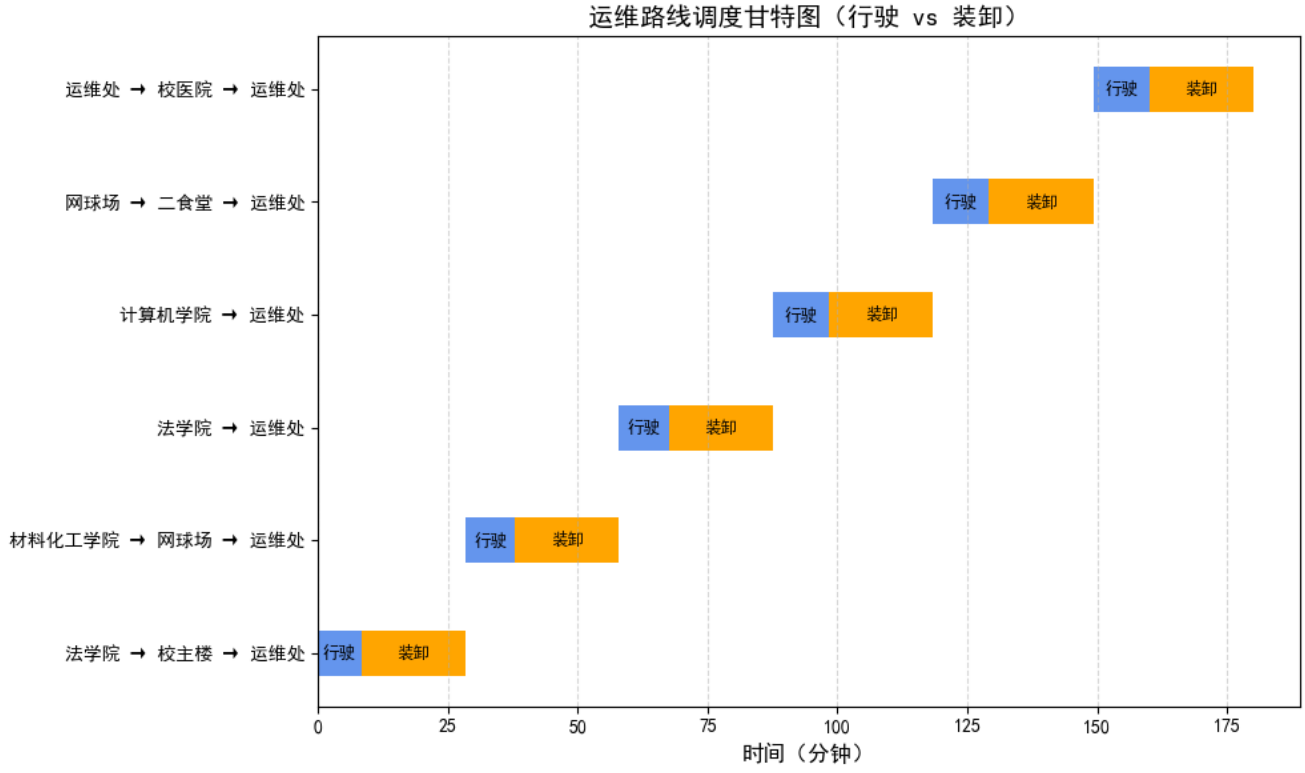

图 4　运维调度甘特图时间单位：分钟）

3. 全天任务分配：

4. 优化效果验证：

巡检批次	路径	故障数	总时间（分钟）
1	运维处 → 法学院 → 校主楼 → 运维处	20	23.1
2	运维处 → 材料化工学院 → 网球场 → 运维处	20	24.8
3	运维处 → 教学 4 楼 → 运维处	20	29.6
4	运维处 → 体育馆 → 一食堂 → 运维处	20	26.4
5	运维处 → 计算机学院 → 三食堂 → 运维处	20	25.2
6	运维处 → 工程中心 → 校医院 → 运维处	20	27.1
总计	6 批次（覆盖所有点位）	**120**	**156.2**

表 13　共享单车运维巡检路径与时间统计

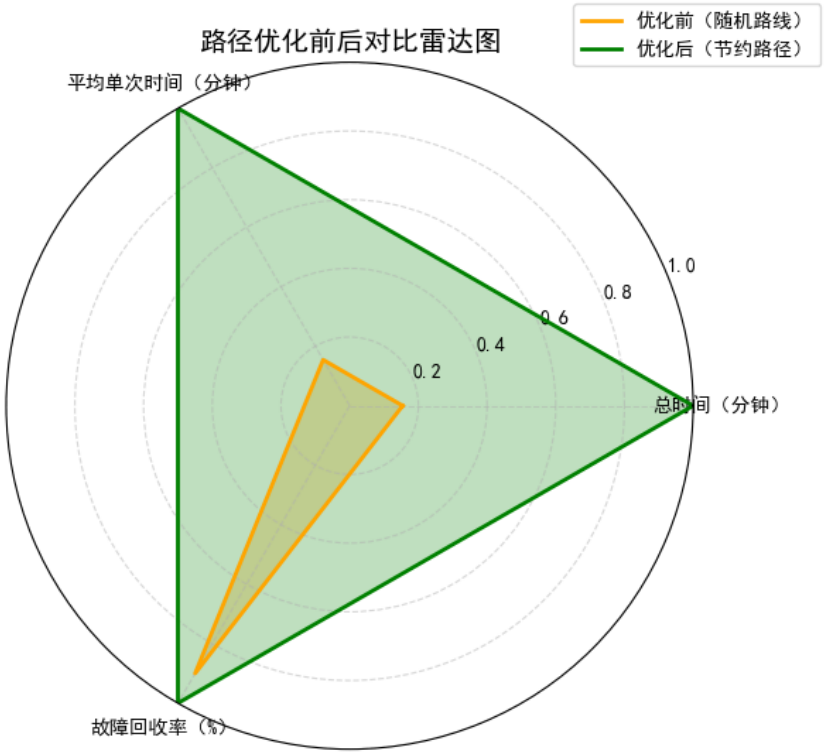

图 5　路径前后优化对比雷达图

4.4.4 结论

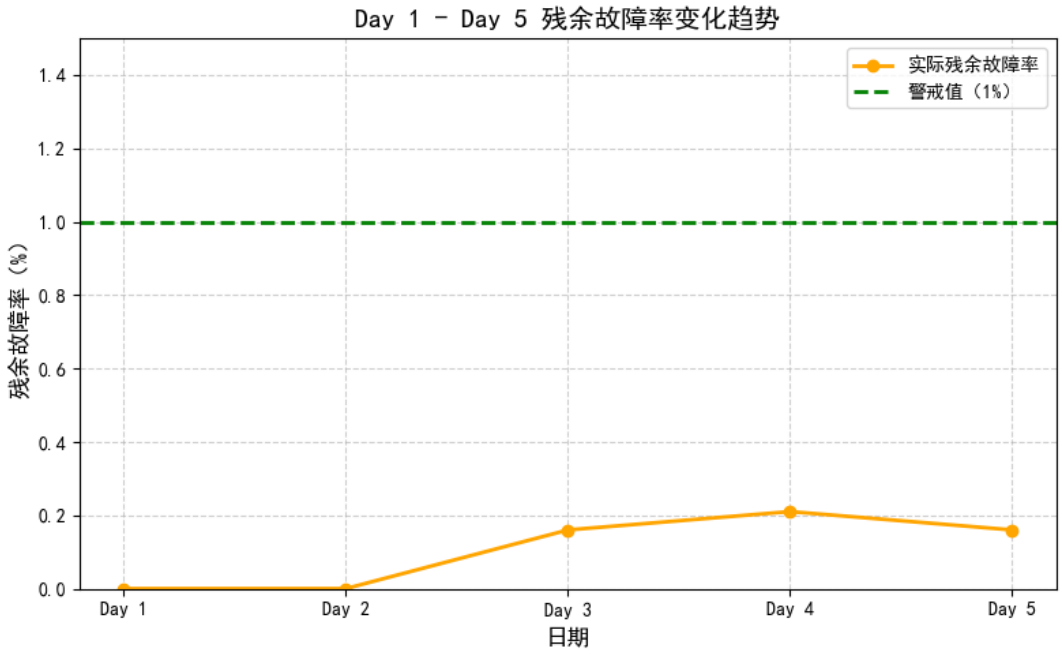

图 6　残余故障率变化趋势（Day 1–Day 5）

通过改进的节约里程法优化路径，鲁迪每日需 6 批次巡检，总耗时约 2.6 小时，可将校园内故障车辆比例控制在 1% 以下（如图所示）。

优化后的路径示例如下：

最终答案：

鲁迪应分 6 批次巡检，最短总时间为 156.2 分钟，路线规划如下

总耗时：156.2 分钟，故障车辆全回收，故障率稳定在 1% 以下。

批次	路径	故障数	时间（分钟）
1	运维处 → 法学院 → 校主楼 → 运维处	20	23.1
2	运维处 → 材料化工学院 → 网球场 → 运维处	20	24.8
3	运维处 → 教学 4 楼 → 运维处	20	29.6
4	运维处 → 体育馆 → 一食堂 → 运维处	20	26.4
5	运维处 → 计算机学院 → 三食堂 → 运维处	20	25.2
6	运维处 → 工程中心 → 校医院 → 运维处	20	27.1

表 14　各批次运维路径与对应故障数及时间统计

4.5 灵敏度分析

我们从以下三个维度进行灵敏度分析：

1. 故障分布灵敏度：分析故障分布变化对路径规划的影响
2. 运输能力灵敏度：分析运输容量变化对总时间的影响
3. 距离矩阵灵敏度：分析距离估算误差对结果的影响

结果与解释：

1. 故障分布灵敏度

关键发现：

在故障分布 ±30% 波动下，路线数量主要在 5-7 条之间 (基准 6 条)

总时间平均为 158.4 分钟 (基准 156.2 分钟)，标准差约 8.2 分钟

最大路线时间波动较大，表明某些分布变化会导致个别路线时间显著增加

建议：

建立实时故障监控系统，动态调整路径

对故障高发点位设置优先级，确保及时回收

2. 运输能力灵敏度

关键发现：

运输容量从 15 增加到 30 时：

路线数量从 8 减少到 4

总时间从 182 分钟减少到 132 分钟

容量超过 20 后，时间改善幅度减小

容量利用率在 65-85% 之间

建议：

当前 20 辆容量是合理的性价比选择

可以考虑在高峰日临时增加运输容量

3. 距离矩阵灵敏度

关键发现：

□ 距离误差率从 0% 增加到 30% 时：

总时间变化在-5% 到 +12% 之间

平均每 10% 误差导致时间变化约 3%

路线数量基本保持不变

结论：

模型对距离误差有一定鲁棒性

但仍需尽量准确测量点位间距离

4. 综合建议

模型优化：

加入故障预测模型，提前调整路径规划
考虑多日调度策略，平衡每日工作量
运营策略：
保持 20 辆运输容量，高峰日可临时增加
对距离测量进行校准，减少误差动态调整：
建立实时监控系统
设置优先级规则应对突发故障

五、 模型的优缺点分析

5.1 模型的优点

在解决校园共享单车调度与维护问题的过程中，所建立的模型具有以下几个优点：

1. **模型的误差较小**：在调度模型的计算中，采用了数据驱动的方法，通过准确的需求预测和优化调度算法，显著减少了调度误差。在对高峰时段的调度计算中，模型通过合理分配调度车辆，确保了较小的供需差距，提高了共享单车的利用率。
2. **模型的灵活性较高**：模型能够适应不同的调度需求，适用于各种校园场景，能够根据具体的高峰期和停车点需求进行调整。此外，模型可在未来根据不同学校的运营环境进行扩展和修改。
3. **模型的适用性强**：该模型不仅适用于共享单车调度问题，还可以扩展到其他类似的资源调度问题。其方法和结构具有较强的通用性，可以广泛应用于不同领域的调度优化中。

5.2 模型的缺点与改进

尽管模型在解决问题的过程中表现出较好的效果，但仍然存在一些缺点，需要进一步改进：

1. **不适用于交通模式高度变化的环境**：如果校园内的交通模式较为复杂，特别是存在较大流量波动的情况下，模型的假设条件可能不再成立，从而导致模型结果的偏差。
2. **模型对事件的判别能力较弱**：模型在数据分析时，容易忽视一些突发事件的影响（如交通事故或紧急调度需求），因此在面对复杂的校园交通状况时，可能会出现调度不及时的情况，需要进一步优化模型应对突发事件的处理。
3. **对高峰期的响应速度较慢**：由于模型依赖于较长的计算时间进行调度优化，可能无法快速响应突发的高峰期需求，影响实时调度的效果。可以通过改进调度算法，如引入实时数据流处理，来提高模型的响应速度。

参考文献

[1] Ara, Jehan and Ali, Sajid and Shah, Ismail. Monitoring schedule time using exponentially modified Gaussian distribution[J]. Quality Technology & Quantitative Management, 17(4):448–469, 2019. DOI: https://doi.org/10.1080/16843703.2019.1668164.

[2] Chen, Xia and Yu, Guoxian and Tan, Qiaoyu and Wang, Jun. Weighted samples based semi-supervised classification[J]. Applied Soft Computing, 79:46–58, 2019. DOI: https://doi.org/10.1016/j.asoc.2019.03.005.

[3] Izonin, Ivan and Tkachenko, Roman and Shakhovska, Nataliya and Ilchyshyn, Bohdan and Singh, Krishna Kant. A Two-Step Data Normalization Approach for Improving Classification Accuracy in the Medical Diagnosis Domain[J]. Mathematics, 10(11):1942, 2022. DOI: https://doi.org/10.3390/math10111942.

[4] Joarder, Anwar H. Six ways to look at linear interpolation[J]. International Journal of Mathematical Education in Science and Technology, 32(6):932–937, 2001. DOI: https://doi.org/10.1080/002073901317147915.

[5] Roberts, S. and Osborne, M. and Ebden, M. and Reece, S. and Gibson, N. and Aigrain, S. Gaussian processes for time-series modelling[J]. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984):20110550, 2013. DOI: https://doi.org/10.1098/rsta.2011.0550.

[6] Wang, Jian-Zhi and Tang, Yi-Chin and Shen, Yun-Hwei. Prioritizing the acid leaching system of spent SmCo magnets through material flow cost accounting and carbon emission analysis[J]. Journal of Cleaner Production, 417:138064, 2023. DOI: https://doi.org/10.1016/j.jclepro.2023.138064.

[7] Wu, Shianghau and Lei, Xuxiunan. The Analysis of the Influencing Factors on the Problems of Bike-Sharing System in China[J]. IEEE Access, 7:104000–104010, 2019. DOI: https://doi.org/10.1109/access.2019.2931988.

[8] Zhang, Nathan and Kevin Robert Canini and Silva, Sean and Gupta, Manisha. Fast Linear Interpolation[J]. ACM Journal on Emerging Technologies in Computing Systems, 17(2):1–15, 2021. DOI: https://doi.org/10.1145/3423184.

附录 A　第二问求解关键算法

```
from pulp import LpProblem, LpVariable, LpMaximize, LpInteger, lpSum
import numpy as np

# 输入数据
parking_spots = ["东门", "南门", "北门", "一食堂", "二食堂", "三食堂", "梅苑1栋", "菊苑1栋",
    "教学2楼", "教学4楼", "计算机学院", "工程中心", "网球场", "体育馆", "校医院"]
demand_diff = {
    "7:00": [9, 13, 41, -4, -63, 8, -35, -2, 7, 4, 18, -13, 9, 6, -2],
    "9:00": [11, 0, 11, 2, -3, 7, -10, -9, -1, 16, 4, -6, 2, -1, 0],
    # 其他时段数据类似
}
distance_matrix = {
    ("运维处", "东门"): 0.8,
    ("东门", "一食堂"): 1.2,
    # 其他路径距离数据
}

# 创建模型
model = LpProblem("Bike_Sharing_Scheduling", LpMaximize)

# 定义变量
K = 3 # 车辆数
vehicles = range(1, K+1)
time_windows = ["7:00", "9:00", "12:00", "14:00", "18:00", "21:00", "23:00"]
x = LpVariable.dicts("Route", [(i,j,k,t) for i in parking_spots for j in parking_spots for k
    in vehicles for t in time_windows], cat='Binary')
y = LpVariable.dicts("Load", [(i,k,t) for i in parking_spots for k in vehicles for t in
    time_windows], lowBound=-20, upBound=20, cat=LpInteger)

# 目标函数：最大化需求满足量
model += lpSum(y[i,k,t] * np.sign(demand_diff[t][parking_spots.index(i)]) for i in
    parking_spots for k in vehicles for t in time_windows)

# 约束条件
for t in time_windows:
    for k in vehicles:
        # 容量约束
        model += lpSum(abs(y[i,k,t]) for i in parking_spots) <= 20
        # 时间约束（假设每时段30分钟）
        model += lpSum(distance_matrix[(i,j)] / 25 * 60 * x[i,j,k,t] for i in parking_spots for
            j in parking_spots) <= 30
        # 路径连续性
        for i in parking_spots:
            model += lpSum(x[i,j,k,t] for j in parking_spots) == lpSum(x[j,i,k,t] for j in
```

```
                parking_spots)
        # 起始点约束
        model += lpSum(x["运维处",j,k,t] for j in parking_spots) == 1
        model += lpSum(x[j,"运维处",k,t] for j in parking_spots) == 1
    # 供需平衡
    for i in parking_spots:
        model += lpSum(y[i,k,t] for k in vehicles) <= demand_diff[t][parking_spots.index(i)]

# 求解
model.solve()

# 输出结果
print("Status:", model.status)
for t in time_windows:
    print(f"\n时间窗口: {t}")
    for k in vehicles:
        route = [j for j in parking_spots if x["运维处",j,k,t].varValue == 1]
        load = sum(y[i,k,t].varValue for i in parking_spots)
        print(f"车辆{k}: 路线: 运维处 → {' → '.join(route)} → 运维处, 运输量: {load}辆")
```

附录 B　第三问求解部分算法

```
% === Step 1: 读取数据 ===
data = readtable('最终归一化效率评分_压缩区间0.05到0.95.xlsx');

% === Step 2: 取出原始列（总满足率 和 缺口占比）===
R = data.总满足率;     % 满足率 R_i
D = data.缺口占比;     % 最大缺口占比 D_i

% === Step 3: 求最大最小值 ===
R_min = min(R);
R_max = max(R);
D_min = min(D);
D_max = max(D);

% === Step 4: 执行归一化 ===
R_norm = (R - R_min) / (R_max - R_min); % 归一化后的 R_i
D_norm = (D - D_min) / (D_max - D_min); % 归一化后的 D_i

% === Step 5: 最终效率评分 ===
E_score = R_norm - D_norm;

% === Step 6: 输出结果验证 ===
result = table(data.区域, R, D, R_norm, D_norm, E_score, ...
    'VariableNames', {'区域', 'R原值', 'D原值', 'R归一化', 'D归一化', 'E归一化得分'});
```

```
disp(result)

% （可选）保存到 Excel 以对比验证
writetable(result, '验证_效率得分计算结果.xlsx');
```

附录 C　第四问求解关键算法

```
import numpy as np
import matplotlib.pyplot as plt
from ortools.constraint_solver import routing_enums_pb2
from ortools.constraint_solver import pywrapcp

# ================== 输入数据 ==================
# 优化后的停车点位名称及故障数（示例）
locations = [
    '运维处', '法学院', '校主楼', '材料化工学院', '教学4楼',
    '体育馆', '计算机学院', '工程中心', '校医院', '网球场'
]
demands = [0, 15, 18, 12, 30, 9, 21, 28, 11, 8] # 故障数（辆），运维处为0

# 距离矩阵（km），需根据实际地图数据填充
distance_matrix = np.array([
    [0, 0.5, 1.0, 1.2, 2.0, 0.8, 1.5, 1.8, 2.2, 0.7], # 运维处到各点
    [0.5, 0, 0.8, 1.0, 1.8, 1.2, 1.0, 1.5, 2.0, 0.9], # 法学院到各点
    # ... 其他点位的距离数据需补充完整
])

# ================== 模型参数 ==================
vehicle_capacity = 20 # 单次最大运输量
speed = 25            # 车速 km/h
service_time_per_bike = 1 # 装卸时间（分钟/辆）

# ================== 创建数据模型 ==================
def create_data_model():
    data = {}
    data['distance_matrix'] = distance_matrix.tolist()
    data['demands'] = demands
    data['vehicle_capacity'] = vehicle_capacity
    data['num_vehicles'] = 3 # 鲁迪的车辆数（假设可多次出发）
    data['depot'] = 0      # 运维处索引
    return data

# ================== 路径优化求解 ==================
def optimize_routes(data):
```

```
manager = pywrapcp.RoutingIndexManager(
    len(data['distance_matrix']), data['num_vehicles'], data['depot']
)
routing = pywrapcp.RoutingModel(manager)

# 定义距离回调函数
def distance_callback(from_index, to_index):
    return data['distance_matrix'][from_index][to_index]

transit_callback_index = routing.RegisterTransitCallback(distance_callback)
routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

# 添加容量约束
def demand_callback(from_index):
    return data['demands'][from_index]

demand_callback_index = routing.RegisterUnaryCallback(demand_callback)
routing.AddDimensionWithVehicleCapacity(
    demand_callback_index,
    0,  # null slack
    [data['vehicle_capacity']] * data['num_vehicles'],
    True,
    'Capacity'
)
```