

基于双层自适应大邻域搜索的多救援中心 动态车辆路径规划方法

姜志豪, 戴欢[†], 徐本连

(苏州科技大学 电子与信息工程学院, 江苏 苏州 215000)

摘要: 针对城市核心区域 (以苏州市姑苏区为例) 应急救援场景下, 事故点时空分布高度集中、救援中心服务边界模糊以及需求实时爆发的现实挑战, 提出一种基于双层自适应大邻域搜索 (Two-Layer Adaptive Large Neighborhood Search, TL-ALNS) 的周期性多救援中心动态车辆路径规划方法。该方法采用周期性更新策略, 定时批量处理事故点以满足实时响应需求。TL-ALNS 通过双层结构学习到算子对之间的协同关系, 并设计数据驱动的预测型重归属破坏算子, 引导边界模糊区域事故点的重新分配, 快速生成高质量的救援路径。基于苏州市姑苏区应急场景数据集的应用验证表明, 该方法能够有效适应老城区特殊的空间布局与动态爆发特征, 在保证响应速度的同时显著优化了救援效率与系统鲁棒性, 为实际应急指挥调度提供了可行的决策支持。

关键词: 应急救援; 多救援中心; 动态车辆路径问题; 周期路径优化; 双层自适应大邻域搜索; 协同关系

中图分类号: TP18; U116 文献标志码: A

DOI: 10.13195/j.kzyjc.2025.1048

引用格式: 姜志豪, 戴欢, 徐本连. 基于双层自适应大邻域搜索的多救援中心动态车辆路径规划方法 [J]. 控制与决策.

Two-layer adaptive large neighborhood search for dynamic multi-depot vehicle routing in emergency rescue

JIANG Zhi-hao, DAI Huan[†], XU Ben-lian

(College of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou 215000, China)

Abstract: To address the real-world challenges in emergency rescue within dense urban core areas — exemplified by Gusu District, Suzhou — characterized by highly concentrated spatiotemporal distribution of incidents, ambiguous service boundaries between rescue centers, and real-time demand bursts, this paper proposes a periodic route optimisation method based on two-layer adaptive large neighborhood search (TL-ALNS). The method employs a periodic update strategy to batch-process accident locations at fixed intervals, thereby meeting real-time response requirements. The TL-ALNS utilizes a two-layer structure to learn the synergy between operator pairs and incorporates a designed data-driven predictive re-assignment destroy operator. This operator guides the re-allocation of accident demands in areas with fuzzy boundaries, facilitating the rapid generation of high-quality rescue routes. Application based on emergency incident data from Gusu District demonstrates that this method effectively adapts to the specific spatial layout and dynamic outbreak characteristics of the old city. It significantly optimizes rescue efficiency and system robustness while ensuring rapid response, providing feasible decision support for actual emergency command scheduling.

Keywords: emergency rescue; multi-depot; dynamic vehicle routing problem; periodic update mechanism; two-layer adaptive large neighborhood search; collaborative relationship

0 引言

在城市化进程不断加速的背景下, 人口与基础设施的高密度聚集导致城市系统在应对突发公共安

全事件时表现出显著的脆弱性。诸如危化品泄漏、城市内涝或大规模火灾等典型应急场景, 时间与空间维度上均呈现出高度的动态不确定性: 事故爆发的

收稿日期: 2025-10-10; 录用日期: 2026-04-25.

基金项目: 国家自然科学基金项目 (62576238, U24A20263, 62503348, 51805346); 苏州市科技计划项目 (SYG202351).

责任编委: 郭戈.

[†]通信作者. E-mail: daihuanjob@163.com.

时间随机、地点分散且难以预知^[1]. 与常规物流配送不同, 应急救援对时效性有着严苛要求, 任何对“黄金救援时间”的延误都可能导致不可逆转的生命财产损失. 这种动态的事故爆发特征与严格的时间约束, 将应急救援决策问题本质上刻画为动态车辆路径问题 (Dynamic Vehicle Routing Problem, DVRP)^[2].

鉴于应急需求在时间与空间维度上的高度不确定性, 现有的 DVRP 求解策略主要被划分为连续重优化与周期性重优化两类范式^[3]. 尽管连续优化策略能够对单一新增事件做出即时响应, 但在大规模并发事故场景下, 高频触发的重计算极易导致系统负荷过载. 相比之下, 周期性优化策略遵循滚动时域原则, 将时域离散化为若干决策周期, 通过批量处理累积的新增需求, 在系统响应速度与方案执行的稳定性之间取得了更优的平衡. Chen 与 Xu^[4] 早在动态列生成算法的研究中便验证了周期性策略在处理时间窗约束下的有效性; Wang 等^[5] 在最新的多救援中心研究中亦指出, 周期性重优化是解决大规模动态需求、规避计算维数灾难的主流范式.

将动态问题转化为多周期的静态子问题序列, 其本质仍属于 NP-Hard 组合优化难题. 尽管分支定价等精确算法能够保证解的最优性^[6], 但其计算耗时随问题规模呈指数级增长, 难以满足应急救援场景下的时效约束. 因此, 在处理大规模并发事故时, 具备高效寻优能力的元启发式算法成为了兼顾求解质量与计算速度的必然选择^[7].

多中心动态车辆路径问题 (MD-DVRP)^[8] 的求解面临着“事故点-救援中心分配”与“路径规划”的双重耦合难题. 特别是在动态救援场景下, 随着实时需求的随机爆发, 各救援中心的服务范围边界呈现出高度的模糊性与流动性. 事故点归属的微小变动往往会引发局部路径的剧烈重组, 导致解空间呈指数级扩张^[9,10].

针对上述问题, 现有研究主要分为两个方向: 一是“分解-求解”策略, 如文献 [11], 文献 [12] 通过聚类算法将问题降维为多个独立的单中心问题. 尽管降低了复杂度, 但割裂式处理忽视了多中心间的协作能力, 极易陷入局部最优. 二是全局协同优化策略, 如 Wen 等^[13] 在统一解空间内利用 ALNS 框架同时优化分配与路径. 此类变动通过引入特定的移除算子一定程度上缓解了耦合僵局, 但本质上缺乏对历史时空规律的学习能力. 面对边界模糊区域的复杂归属问题, 此类“盲目”的算子难以精准识别并修正潜在的错误分配并收敛至全局最优.

自适应大邻域搜索算法 (ALNS) 自 2006 年提出

以来, 凭借其强大的解空间探索能力与局部搜索灵活性, 在 VRP 及其变体求解中均展现出强大的性能优势^[14,15]. 其核心机制在于通过“破坏”与“修复”算子的迭代变换, 配合基于历史表现动态调整权重的自适应机制, 引导搜索过程在广度探索与深度开发之间取得平衡. 现有研究多是根据特定场景约束定制专属的破坏或修复算子以适配模型需求. 例如, Wang 等^[5] 针对多车场动态场景对实时性的极高要求, 设计了“成本降低”与“交换降低”两种新型移除算子, 并引入基于时间窗相容性的 O(1) 插入方法, 实现了快速重优化; Pereira 等^[16] 针对带时间窗的异构固定车队路径问题, 设计了整条路径移除算子以降低固定成本来解决复杂的车型选择难题.

另一方面, 也有学者致力于改进算法的自适应框架机制与算子选择逻辑, 以提升搜索效率. Kovacs 等^[17] 深入探究了破坏算子与修复算子间的依附关系, 指出特定的算子组合 (算子对) 往往优于独立选择, 强调了在权重更新中考虑算子协同效应的重要性; Voigt 与 Kuhn^[18] 在解决带中转的众包物流问题时, 对 ALNS 的自适应框架进行了微调, 通过引入组合算子策略及调整移除数量的采样分布, 有效增强了算法在复杂众包网络环境下的鲁棒性.

本文以苏州市姑苏区应急场景为例, 研究多救援中心动态车辆路径规划问题, 提出一种基于 TL-ALNS 算法的周期性优化方法. 主要贡献如下:

1) 建立了应急救援场景下的周期性多救援中心动态优化模型: 将复杂的实时救援响应转化为滚动时域下多阶段静态优化问题, 并设计了适应高度动态性的周期性更新策略, 确保方案的及时响应能力.

2) 提出 TL-ALNS 算法: 设计了双层自适应权重机制. 第一层筛选主导算子对, 第二层则基于历史反馈学习并匹配与之产生最大正向增益的协同算子对. 通过额外学习这种“算子对”层面的协同效应, 显著增强了算法的鲁棒性.

3) 设计数据驱动的预测型重归属算子: 针对多救援中心边界模糊区域的分配难题, 引入预测模型. 利用历史最优解的时空特征训练逻辑回归模型, 智能破坏具有“高重分配潜力”的事故点, 引导算法利用先验知识快速修正不合理的归属决策.

4) 多维度的实验验证与参数分析: 在公开基准数据集上, 相较于最先进的算法, 验证了 TL-ALNS 在鲁棒性及收敛效率上的显著优势. 基于苏州市姑苏区应急场景数据集的实验进一步验证了该方法在实际应急场景中的可行性与高效性.

1 建立模型

1.1 问题描述

本文将姑苏区应急场景描述为:确定的服务范围内,存在 N 个位置已知的救援中心和大量动态产生的事故点.各中心车辆规模及车辆上救援人员最大工作量已知,且路径规划遵循闭环约束.事故点地理位置、时间窗、处理工作量及处理时长,仅在其报警时刻才可获得.为方便建模,建立以下基本假设:

- 1) 救援车辆均从救援中心出发,完成任务后需返回原救援中心;
- 2) 所有救援车辆上救援人员的工作量是有限的,只能服务有限的事事故点,该救援路径上服务事故点的总工作量不得超过该路径车辆上救援人员最大工作量;
- 3) 事故点时间窗为软约束,允许违反但需在目标函数中施加惩罚.救援车辆可以提前到达,但必须等时间窗开始才能开启服务;
- 4) 所有事故点均需被服务,且每个事故点仅由一辆车服务一次;
- 5) 所有救援车辆型号统一且参数已知;
- 6) 模型忽略交通路况、驾驶员疲劳等外部因素对行驶的影响.

1.2 参数设置

模型中数学符号的含义如下所示:

$[0, H]$: 问题的总时间范围;

\vec{T} : 周期性更新策略中选择的更新时间点,

$\vec{T} = \{T_0 = 0, T_1, \dots, T_p = H\}$;

ε : 更新间隔的时长;

N : 救援中心集合, $N = \{1, 2, \dots, n\}$;

M_{last}^p : T_p 时刻以前处理过的事事故点集合, $M_{last}^p = \{1, 2, \dots, m\}$;

M_{new}^p : T_p 时刻还未处理的事事故点集合, $M_{new}^p = \{m + 1, m + 2, \dots, +\infty\}$;

M^p : T_p 时刻获取到的所有事故点集合, $M^p = \{M_{last}^p \cup M_{new}^p\}$;

K_n : 第 n 个救援中心的车辆集合, $K_n = \{1, 2, \dots, k_n\}$;

K_N : 所有车辆的集合, $K_N = \{K_1 \cup K_2 \cup \dots \cup K_n\}$;

Q : 救援车辆上救援人员的最大工作量;

$Q_{k_n}^p$: T_p 时刻第 k_n 个救援车辆上救援人员已经完成的工作量;

c_{ij} : 事故点 i 到事故点 j 的通勤代价, $i, j \in \{N \cup M\}$;

q_m : 第 m 个事故点的处理工作量的需求;

s_m : 第 m 个事故点的处理时间需求;

$t_{k_n m}$: 第 k_n 个车辆到达事故点 m 的时间;

$[e_m, l_m]$: 第 m 个事故点的时间窗要求;

模型中的决策变量如下所示.

$x_{k_n ij}$: 车辆 k_n 是否在路径规划中由 i 事故点行驶到 j 事故点. 1为是, 否则为0, $i, j \in \{N \cup M\}$;

$y_{k_n m}$: 事故点 m 是否被安排由车辆 K_n 处理. 1为是, 否则为0;

V_{start}^m : 第 m 个事故点的开始处理时间;

V_{end}^m : 第 m 个事故点的结束处理时间.

1.3 模型构建

在模型构建中,通勤成本仅考虑车辆行驶距离,且运动时间等值于事故点间距.能否降低通勤成本和满足事故点的时间窗要求是评价方案优劣的核心指标.时间窗延误成本乘以转换系数 $(\frac{\max c_{ij}}{\varepsilon})$,其中 $\max c_{ij}$ 表示事故点间距的最大值.这是为了消除量纲差异,使两者处于同一数量级.

1) 车辆通勤成本

$$z_1 = \sum_{i \in NUMP} \sum_{j \in NUMP} \sum_{k_n \in K_N} x_{k_n ij} \cdot c_{ij}. \quad (1)$$

2) 时间窗延误成本

$$cost_t = \sum_{m \in MP} \sum_{k_n \in K_N} y_{k_n m} \cdot \max\{t_{k_n m} - l_m, 0\}, \quad (2)$$

$$z_2 = cost_t \cdot (\frac{\max c_{ij}}{\varepsilon}). \quad (3)$$

本研究针对实时响应采取的是周期性优化策略,优化路径的搜索基于上一阶段的确切方案逐步推进,要求每一阶段确立唯一最优解.总救援成本构成如下:

$$\min Z = z_1 + \lambda \cdot z_2. \quad (4)$$

s.t.

$$\sum_{j \in M \setminus \{i\}} \sum_{k \in K_N} x_{k_n ij} = 1, \forall i \in M, \quad (5)$$

$$\sum_{i \in M \setminus \{j\}} \sum_{k \in K_N} x_{k_n ij} = 1, \forall j \in M, \quad (6)$$

$$\sum_{m \in MP} x_{k_n nm} - \sum_{m \in MP} x_{k_n mn} = 0, \forall k_n \in K_N, n \in N, \quad (7)$$

$$\sum_{j \in NUMP \setminus \{i\}} x_{k_n ij} - \sum_{j \in NUMP \setminus \{i\}} x_{k_n ji} = 0, \quad (8)$$

$$\sum_{i \in MP} q_i \sum_{j \in NUMP} x_{k_n ij} \leq Q, \forall k_n \in K_N, n \in N, \quad (9)$$

$$Q_{k_n} + \sum_{i \in M_{new}^p} q_i \sum_{j \in N \cup M_{new}^p} x_{k_n i j} \leq Q, \quad \forall k_n \in K_N, n \in N, \quad (10)$$

$$V_{start}^m + s_m = V_{end}^m, \quad \forall m \in M^p, \quad (11)$$

$$V_{start}^j = \max\{t_{k_n j}, e_j\}, \text{ if } \sum_{k_n \in K_N} x_{k_n i j} = 1, \quad \forall i, j \in M^p, \quad (12)$$

$$x_{k_n i j} \in \{0, 1\}, \quad \forall i, j \in N \cup M^p, k_n \in K_N, \quad (13)$$

$$y_{k_n m} \in \{0, 1\}, \quad \forall m \in M^p, k_n \in K_N, \quad (14)$$

$$Q > 0, q_i > 0, s_i > 0. \quad (15)$$

公式(4)将目标函数定义为车辆在每个重优化时刻规划路径后的时间窗延误成本和通勤成本的加权之和, λ 是设定的惩罚倍率, 通过赋予相对大权重表达应急场景中决策者对于时间紧迫性的重视^[19]. 约束(5)和(6)共同确保每个事故点被且仅被处理一次. 约束(7)要求每辆车必须从起始救援中心出发并最终返回该中心. 约束(8)保证了每个事故点的入度和出度一致. 约束(9)规定了救援车辆上救援人员工作量的限制, 约束(10)给出路径方案是否违反工作量限制的计算方法. 公式(11)表示每个事故点的开始处理时间, 要求处理时间和处理完成时间这三者之间的关系, 车辆在每个事故点的开始处理时间由约束(12)规定. 约束(13)和(14)定义了决策变量的取值. (15)表示一些非负变量的约束.

1.4 动态实时需求信息的周期性更新策略

针对动态增加的事故点信息, 本研究在时间范围 $[0, H]$ 内采用周期性再优化策略处理. 即批量将新到达的事故点融入现有救援方案, 以实现动态调整. $[0, H]$ 被分为 p 个等长区间, 每个区间结束时触发路径重优化任务. 设 $\vec{T} = \{T_0 = 0, T_1, \dots, T_p = H\}$ 表示每次优化路径的时间节点集合, $\varepsilon = T_p - T_{p-1}$ 代表时间间隔的长度, 通过调节 ε 值的大小可以适应不同响应程度的要求. 当 ε 值越小, 响应越频繁, 实时性越高. 基于该策略, 动态车辆规划问题转化为一系列顺序求解的静态问题.

图1展示了 T_{p-1} 时刻根据接收事故点信息生成的路径规划, 图2对应标注了各事故点的实际发生时刻. 每个更新节点均会执行一次路径重优化, 新路径包含该节点之前所有发生的事故点. 以 T_{p-1} 时刻为例, 该时刻前的6个事故点已被全部纳入路径规划.

每次重优化所针对的事故点集合不同, 因此每次重优化前需确定各车辆的实时位置. T_p 时刻, 救援

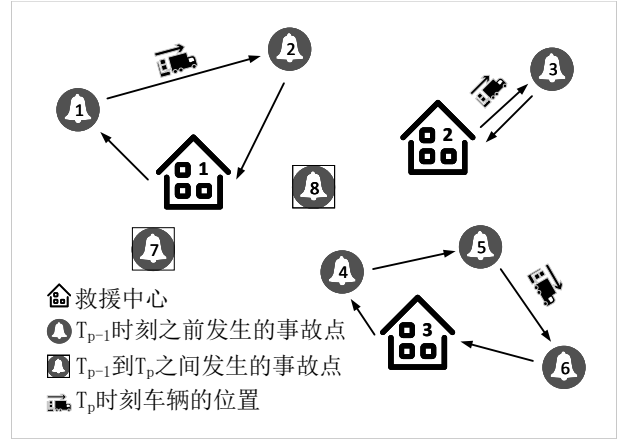


图1 T_{p-1} 时刻的路径规划

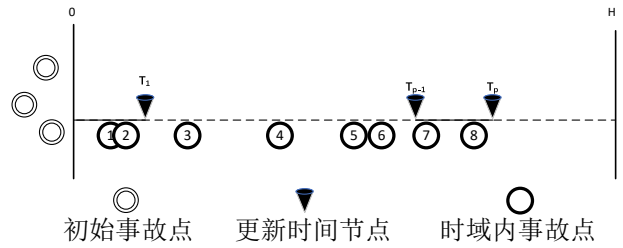


图2 发生事故点的时间序列

中心3的车辆处于自事故点5到驶向6的途中, 则事故点5及之前的路径段被固定. 原规划中尚未处理的事故点(如事故点6), 将与 T_{p-1} 到 T_p 时间周期内接收到的事故点7和8共同参与 T_p 时刻的路径重优化.

2 TL-ALNS 算法

针对苏州市姑苏区应急场景中救援中心服务边界模糊的特点, 以及多救援中心动态车辆路径规划问题本身时效性要求高的挑战, 本文提出一种双层自适应大邻域搜索(TL-ALNS)算法. 2.1节定义了用于评估解质量的适应度函数; 2.2节详细阐述了TL-ALNS的双层搜索框架及权重更新机制; 2.3节介绍了包括新型重归属破坏算子在内的算子设计.

2.1 解的适应度函数

适应度指的是解决方案在目标函数维度上的得分, 用来评价解的质量. 解的适应度值与其优劣程度成正比, 适应度值越大解越优. 设计适应度函数 f 为目标函数的倒数, 如式所示.

$$f_i = \frac{1}{Z_i}. \quad (16)$$

式中 f_i 和 Z_i 分别为方案 i 的适应度值和目标函数值.

2.2 双层搜索框架及更新机制

2.2.1 算法框架

Kovacs等^[17]通过将破坏算子与修复算子合并成算子对统一赋予选择权重, 证明了两者之间存在显著的协同效应. 本研究构建了双层邻域搜索框架,

进一步探索算子对层面的协同效应, 即两个算子对的组合可能会产生更好决策结果。

为刻画双层 TL-ALNS 与传统单层 ALNS 在搜索机制上的本质差异, 我们形式化其操作流程. 如图 3 所示, 考虑算子对集合 $P = \{N_1, N_2\}$. 对于搜索空间的当前解 x , 每次算子对的应用 (破坏与修复) 都会产生一个新的候选解位置. 由于同一算子对的破坏修复策略是固定的, 其对应的候选解在解空间中呈现局部聚集性, 可视为一种区域性映射. 传统单层 ALNS 在每轮迭代从 P 中独立采样一个算子对 p_a , 并基于其即时解质量收益决定是否接受新解 $x' = p_a(x)$, 该过程等价于在解空间中执行一次单步邻域映射:

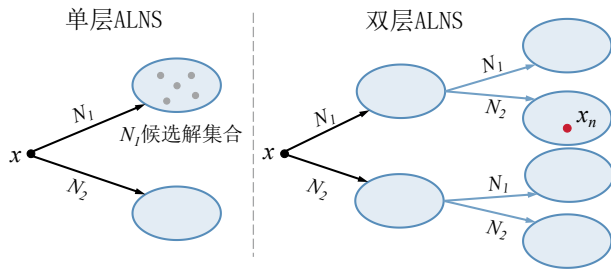


图3 单/双层 ALNS 搜索状态对比

$$E_{\text{one-level}}(x) = \{p_a(x) \mid p_a \in P\}. \quad (17)$$

其搜索具有“短视性”——仅评估单个算子对在当前状态下的即时收益, 忽略其与后续操作的协同潜力. 相比之下, TL-ALNS 引入两阶段协同操作机制. 第一层仍按权重向量采样主导算子对 p_a , 但第二层不再独立采样, 而是基于 p_a 的上下文条件, 从一个 $|P| * |P|$ 的条件转移矩阵 $W = [w_{ab}]$ 中采样从属算子对 p_b , 元素 w_{ab} 表示“在已应用 p_a 后, 选择 p_b 的相对偏好”. 一次迭代的完整操作可形式化为:

$$x_1 = p_a(x), x_2 = p_b(x_1), \text{ 其中 } p_b \sim W[a, \cdot]. \quad (18)$$

其组合邻域映射为:

$$E_{\text{two-level}}(x) = \{p_b(p_a(x)) \mid p_a, p_b \in P\}. \quad (19)$$

与单层的直接跳跃不同, 双层结构实现了两步的连续映射, 扩展了候选解的覆盖空间.

如图 3 右所示, 在 N_1 作用的基础上再应用 N_2 , 打破局部收敛, 引导搜索抵达全局最优解 x_n . 传统单层 ALNS 仅评估 p_a 的即时收益, 会因表现不佳降低其权重, 错过这条潜在高价值的路径; 而 TL-ALNS 在第二层保留 p_a 和 p_b 的联合权重 w_{ab} , 将组合策略作为学习单元.

为避免无效计算, 需控制每次搜索的层级深度. 若第一层 p_a 应用后解未发生变化, 学习不到序列关系, 算法跳过第二层, 退化为单层模式.

算法 1 概述了双层 ALNS 算法的框架流程, 该算法由四个部分组成: 初始化、算子应用、更新权重层和更新解.

Algorithm 1 双层自适应大邻域搜索算法

Input: 算子对的总数 n ; 初始温度 T ; 温度冷却速率 β

Output: 最终的解决方案 S^*

1. //初始化
2. 获得初始解 S
3. 设置当前解 $S' \leftarrow S$, 设置最优解 $S^* \leftarrow S$
4. 初始化两层算子对的权重 $\rho_1 = (1 \times n)$, $\rho_2 = (1 \times n^2)$
5. $iter$: 循环的次数
6. while ($iter > 0$) do
7. 根据权重选择两层的搜索算子对 P_a, P_b
8. $S_1 = P_a(S)$, $S_2 = P_b(S_1)$
9. //更新权重层
10. 对比 S_1 与 S 更新 ρ_1 , 对比 S_2 与 S 更新 ρ_2
11. 比较 S_1 与 S_2 , 得出综合最优的方案 S^+ , 作为最终结果
12. //更新解
13. if $f(S^+) > f(S^*)$ do
14. $S^* \leftarrow S^+$
15. if $f(S^+) > f(S')$ do
16. $S' \leftarrow S^+$
17. 随机生成一个 0 到 1 的数 z
18. if $f(S^+) < f(S')$ and $z < \exp(-\frac{f(S') - f(S^+)}{T})$ do
19. $S' \leftarrow S^+$
20. $iter \leftarrow iter - 1$, $T \leftarrow \beta T$

2.2.2 解的初始化

初始解的质量直接影响算法的收敛速度. 考虑到多救援中心场景下, 直接使用贪婪插入容易导致车辆在不同区域间长途奔袭, 产生交叉路径等劣解, 本文采用先聚类后路径的策略生成初始解.

在聚类阶段, 利用 K-means 算法将所有事故点划分为 N 个簇 (N 为救援中心数量), 每个簇对应一个救援中心的服务范围. 路径构建阶段, 在每个簇内部, 以对应的救援中心为起点, 按照事故点时间窗的紧急程度升序排列, 采用贪婪策略依次将事故点插入当前车辆路径中.

2.2.3 权重更新

如算法 1 的第 4 行所述, TL-ALNS 设计了双层权重结构. 第一层权重向量表示主导算子对的选择倾向, 其长度与算子对的数目相同. 在自适应机制中, 采用轮盘赌选择策略进行选择. 第 i 个算子对的选择概率 φ_i 公式如下所示:

$$\varphi_i = \frac{\rho_1[i]}{\sum_{k=1}^n \rho_1[k]}. \quad (20)$$

第二层权重向量描述了算子对之间的转移概率,其长度是算子对数目的指数增长(大小为算子对总数的二次方).若在第一轮中选择了第 a 个算子对,则在第二轮中计算接下来选择第 i 个算子对的概率 φ_i 公式如下所示:

$$\varphi_i = \frac{\rho_2[a \cdot n + i]}{\sum_{k=a \cdot n + 1}^{a \cdot (n+1)} \rho_2[k]}. \quad (21)$$

算法第10行记录了每次算子应用后的权重调整.第一层权重反映初始搜索方向,第二层的权重反映算子对之间的衔接适配程度.由于二者的职能不同,其权重更新独立进行.更新权重前,通过得分 σ 量化算子对在本次迭代中的表现:

$$\sigma = \max \begin{cases} \omega_1 & \text{如果新方案是全局最优,} \\ \omega_2 & \text{如果新方案比当前方案优秀,} \\ \omega_3 & \text{如果新方案比当前方案差但被接收,} \\ \omega_4 & \text{如果新方案被拒绝} \end{cases} \quad (22)$$

式中 $\omega_1, \omega_2, \omega_3$ 和 ω_4 分别代表了四种情况下的算子对得分.令 $\omega_1 \geq \omega_2 \geq \omega_3 \geq \omega_4 \geq 0$,表示效果越好的方法得到越高的分数,越容易受到鼓励.反之,持续表现不佳的算子对被视为对迭代资源的浪费,分配较低的得分来降低其在未来搜索中被选中的概率.

在将算子对得分融入权重动态更新的过程中,为了使权重更新既能体现对应算子对的过往综合表现又能避免历史得分记录无限累积所带来的计算负担和早期数据干扰,本文提出了一种基于固定长度队列的新权重更新策略,即在每个权重的位置上都维护一个长度为 l 的历史得分队列.第 a 个算子对更新公式为:

$$\rho_a = \begin{cases} \lambda \rho_a + (1 - \lambda) \frac{s_a}{u_a}, & \text{if } u_a < l \\ \lambda \rho_a + (1 - \lambda) \frac{s_a}{l}, & \text{else} \end{cases} \quad (23)$$

式中 $\lambda \in [0, 1]$ 是衰减参数,用来控制权重对于算子对近期表现做出反应的敏感程度. s_a 为该算子对历史得分列表的值总和, u_a 表示该算子对的使用次数.一方面,由于维护了一个有限的历史窗口,权重更新依然基于算子对的近期综合表现.另一方面,利用队列本身先进先出的特性,超出窗口范围的早期得分记录会被自动淘汰,不再干扰当前阶段的判断.其中,队列长度 l 决定对历史信息信任窗口的大小.

2.2.4 解的更新

算法1的11-19行描述了更新解的步骤,主要包括当前解和最优解的替换.拓展权重层数后,解的更新机制也需同步调整.单步迭代中两次算子对的应用产生了两个候选解,为满足解最优替换的原则,根据适应度函数将其融合成综合最优的解 s^+ .如果 s^+ 方案比全局最优解优秀则更新全局最优解,如果 s^+ 方案比当前解优秀则更新当前解.

算法第18行的判断语句是模拟退火算法思想的体现:为维持搜索多样性并避免陷入局部最优,设置动态衰减的接收阈值.即如果 s^+ 的劣于当前解,则接收概率为 $\exp(-(f(S') - f(S^+))/T)$.算法运行过程中,温度 T 的值随时间逐渐衰减,保证接收劣解的标准越来越严苛.具体初始温度以及衰减率的设置参照 KirkpatrickS 等^[20]的工作.

2.3 算子对设计

2.3.1 破坏算子

ALNS 算法通过集成多样性的破坏算子实现对解空间的广泛探索.本研究共采用了四种破坏算子从给定解中移除事故点.包括三种经典破坏算子以及一种专门为本文多救援中心场景设计的破坏算子,与两个经典修复算子依次组成算子对,以下是其详细介绍.

1) 随机删除:从解决方案中随机选择 a 个事故点进行删除,放入待修复事故点列表中.

2) 最差删除:该算子旨在移除当前解中导致成本增加最多的事故点,基于如下规则删除:

$$p(i, j, k) = \arg \max \{d_{ik} + d_{kj} - d_{ij}\}, \quad i, j \in N \cup M^p, k \in M^p. \quad (24)$$

式中 i 和 j 代表事故点 k 的前后事故点,旨在将成本最大的事故点依次加入待修复事故点列表中.

3) shaw 删除:该算子最初由 Shaw^[21]提出,基于“相关性”选择待移除的事故点.算法首先随机选择一个种子事故点 $i \in M^p$,随后通过计算相关性数值来识别与其最相似的事故点 i 进行移除.事故点 i 的选择准则为最小化加权归一化函数:

$$j = \arg \min \left\{ \alpha_1 \frac{c_{ij}}{\max c} + \alpha_2 \frac{|V_{\text{start}}^i - V_{\text{start}}^j|}{\varepsilon} + \alpha_3 \frac{|q_i - q_j|}{\max q} \right\}. \quad (25)$$

其中, $\max q$ 和 $\max c$ 分别表示最大需求量和任意两事故点间的最大行程距离,用于归一化; V_{start}^i 和 V_{start}^j 表示事故的开始服务开始时间.权重参数 $\alpha_1, \alpha_2, \alpha_3$ 的具体数值参考文献[5]的设置.

4) 新破坏算子——数据驱动的预测型重归属破

坏算子: 在多救援中心车辆路径问题中, 事故点与救援中心的归属关系直接决定了路径优化的下界. 本文提出一种新的重归属破坏算子, 结合预测模型识别出具有“高重分配潜力”的事故点并加以破坏, 迫使其在后续修复中归属到更优的救援中心.

本文将事故点的重分配潜力定义为“违背最近分配原则”的概率. 为提升归属决策的合理性, 将高质量解集作为训练集. 这些解集源自小规模经典数据集上的已知最优解, 由高性能求解器给出, 具备较高的可靠性. 预测模型采用逻辑回归算法学习事故点位置特征与违反最近分配原则概率之间的映射关系. 鉴于绝对距离缺乏泛化性, 且位于多个救援中心“边界区域”的点更容易发生归属变动, 模型基于相对距离构建特征向量. 对事故点到各救援中心的距离进行升序排列, 以最近距离为基准, 将其余距离表示为与最近距离的比值. 这种处理方式可以缩小模型在小规模数据集上训练结果与实际应用之间的差距, 让模型在不同场景下仍能有效识别边界模糊的点.

模型的评估阶段, 假设测试集样本集合为 $B = \{b_1, b_2, \dots, b_n\}$, 模型输出三组数据, 分别是符合最近分配原则的概率 $Fit = [fit_1, fit_2, \dots, fit_n]$, 违反概率 $Vio = [vio_1, vio_2, \dots, vio_n]$ 以及通过比较两者大小得到的预测标签 $Pre_label = [l_1, l_2, \dots, l_n]$. 与一般逻辑回归模型评估不同, 本场景中“违反最近分配原则”属于稀有事件, 违反概率整体数值偏小, 预测标签也普遍为 0, 呈现出极端类别不平衡现象. 传统指标如准确率、精确率等难以有效衡量模型性能. 本文提出一种基于排序截断的评估指标——违反准确率 vio_true , 其计算公式为:

$$vio_true = \frac{\text{预测违反中正确的样本数}}{\text{预测违反的总样本数}} \times 100\%. \quad (26)$$

假设测试集中真实违反样本数为 v . 我们将模型输出的违反概率向量降序排列, 取前 v 个样本预测为“违反”, 其余为“符合”. vio_true 定义为这前 v 个样本中预测正确的比例, 用于衡量模型对高风险样本的捕捉能力. 训练后的预测模型被集成到修改事故点到救援中心的归属这一破坏算子的设计中, 详细流程如算法 2 所示.

该算子从事故点集合 M^p 中选择固定数量的候选点进行归属调整, 将所有事故点输入至已训练好的预测模型中, 获取对应的符合最近分配原则概率 $Fit = [fit_1, fit_2, \dots, fit_m]$ 和违反概率 $Vio = [vio_1, vio_2, \dots, vio_m]$. 违反概率直接影响事故点被

重新分配的可能性, 对其归一化处理得到 Vio . 在持续的重归属尝试中, 对符合概率较低的事故点进行重分配通常被认为是合理的. 反之, 若一个符合概率较高的事故点在历史决策中被重新分配, 应提高其再次尝试的可能. 因此, 对符合概率归一化处理得到 Fit , 同时为各事故点引入一个是否被重分配的记录向量 $flag = [fl_1, fl_2, \dots, fl_m]$, 将历史决策中被重新分配的事故点标为 1, 否则标 0. 在该算子的混合选择策略中, 通过如下公式计算每个事故点 i 被选中进行归属调整的概率 P_i :

$$p_i = c_1 \cdot Vio + c_2 \cdot flag \cdot Fit. \quad (27)$$

其中 Vio 是模型输出的违反概率的归一化值, 反映了该点在空间结构上调整归属的理论潜力. $flag \cdot Fit$ 是历史决策偏离度的归一化值, 反映其应被重新考虑的概率. 即使模型因训练数据有限产生偏差, 如错误地将一个本该重新归属的事故点分配了极低的违反概率, 其他算子仍有机会在迭代中修正该状态. 一旦搜索历史显示该事故点经过重新分配后解更优, 后续迭代中第二项的介入就会纠正模型的初始误判, 迫使算法持续关注这个节点. 最后, 算法基于 P_i 值使用轮盘赌选择机制确定需改变归属的事故点.

Algorithm 2 修改事故点到救援中心归属的算子

Input: 目前总事故点集合 M^p ; 当前全局解 S ; 当前迭代次数 $epoch$; 最大迭代轮数 $Epoch$; 历史重分配标记向量 $flag$; 预测模型 Λ ; 移除节点数量 a

Output: 破坏后的解 S^-

1. if $epoch < Epoch$ do
 2. 初始化选择概率集合 P
 3. 对于 M^p 中的每一个事故点 i 执行
 4. \步骤1: 特征提取与模型预测
 5. 计算点 i 到各救援中心的距离并排序, 生成距离比值特征 r_i
 6. 调用模型预测违反概率: $vio_i \leftarrow \Lambda.predict(r_i)$
 7. // 步骤2: 结合历史信息
 8. 获取该点的历史重分配记录: $flag_i$
 9. // 步骤3: 计算综合选择概率
 10. $p_i = c_1 \cdot Vio_i + c_2 \cdot flag_i \cdot Fit_i$
 11. 将 p_i 加入集合 P
 12. 结束循环
 13. 基于集合 P 使用轮盘赌策略不重复的删除 a 个事故点
 14. 得到 S^-
-

2.3.2 修复算子

为了对破坏后的解进行修复, TL-ALNS 引入了经典的贪婪插入和遗憾插入机制. 通过遍历请求列

表, 评估所有潜在的可行插入位置及成本, 将被删除的事故点逐一插入. 两种算子的工作原理如下.

1) 贪婪插入修复: 该算子基于如下规则插入:

$$p(i, j, k) = \arg \min \{z_1(i, j, k)\},$$

$$i, j \in N \cup M^p, k \in \Theta. \quad (28)$$

式中 $z_1(i, j, k)$ 表示将事故点 k 插入到 i 和 j 中间后该车辆路径增加的成本, 旨在仅考虑当前收益情况下将删除的事故点插入到最合适的位置.

2) 后悔插入修复: 与贪婪算子仅关注当前的最小插入成本不同, 后悔插入算子通过计算“后悔值”来评估错失最佳插入位置的潜在代价. 对于每一个待插入的事故点 i , 计算其在最佳位置以及次佳位置的成本之差, 将其定义为后悔值, 算法在每次迭代中选择后悔值最大的事故点进行插入.

此外, 为了提高算法效率, 参考 Nagata^[22] 与 Wang 等^[5] 的方法, 在修复算子中引入了基于时间窗兼容性的快速校验机制. 通过预计算路径上各节点的松弛时间, 在 $O(1)$ 的时间复杂度内判断插入操作的可行性, 避免大量无效的路径遍历计算.

3 案例与实验

本文的算法用 Python 语言编写, 在 AMD 锐龙 5 PRO 4650G 处理器、3.70GHz、16.0GB RAM 和 Windows 10 操作系统的机器上运行.

本文在 3.1 节中分析新型破坏算子权重参数的合理取值, 在 3.2 节中, 将 TL-ALNS 与当前最先进的算法进行实验对比, 验证其性能. 在 3.3 节中, 通过苏州市姑苏区应急场景数据集, 验证算法的有效性和实用性. 最后, 在 3.4 节中, 通过消融实验验证了新破坏算子与双层结构的性能.

3.1 新型破坏算子权重参数的敏感性分析

本研究采用 MDVRPTW 基准集^[23] 进行测试. 该数据集由 Cordeau 等提出, 可从 <http://people.brunel.ac.uk/~mastjjb/jeb/info.html> 获取, 共包含 20 个算例, 50-360 个客户数、2-9 个场站. 其中, 每个实例均对车辆数量及载重量设置了约束. 算法涉及的主要参数取值如表 1 所示.

在上文提出的数据驱动的预测型重归属破坏算子中, 混合选择概率公式 (27) 引入了两个关键权重参数 c_1 和 c_2 . 其中, c_1 决定了算法对预测模型输出的“空间特征先验概率”的信任程度, 而 c_2 则控制算法对迭代过程中的“历史决策纠正能力”的重视程度.

为探究先验知识与决策纠正之间的最佳平衡点, 设计了参数取值实验. 实验设定约束 $c_1 + c_2 = 1$, 令 c_1 以 0.1 的步长在 $[0, 1]$ 区间内变化. 实验选取了具

表1 TL-ALNS 算法参数取值

符号	描述	值
a	删除的事故点数量	总数的15%
α_1	shaw算子中的距离权重	0.4
α_2	shaw算子中的时间权重	0.8
α_3	shaw算子中的需求权重	0.3
T	初始温度	100
β	温度冷却速率	0.995
$Epoch$	迭代次数	3000
$\omega_1, \omega_2, \omega_3, \omega_4$	算子对4种得分	1.5, 1.2, 0.8, 0.6
λ	时间成本的惩罚权重	10

有代表性的大规模算例 (pr05 和 pr10) 进行测试, 每个参数组合独立运行 10 次取平均值, 结果如下.

如图 4 所示, 随着 c_1 的增加, 求解成本呈现出明显的先降低后升高的“U 型”变化趋势. 当 c_1 取值较小时, 算子过度依赖历史决策信息, 未能充分利用预测模型提取的空间特征, 收敛效率受限; 反之, 当 c_1 取值过大, 破坏操作忽视了过往决策的重新审视, 极易陷入局部最优. 当 c_1 处于区间 $[0.5, 0.7]$ 时, 算法能够有效结合先验的空间分布特征与历史决策的纠正, 在保持解空间多样性的同时实现精准寻优. 本文在后续所有实验中统一设定 $c_1=0.6$, $c_2=0.4$, 以确保算法在不同场景下稳定输出的能力.

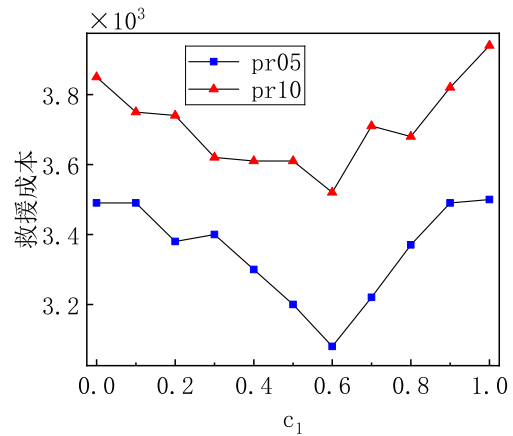


图4 参数的取值分析

3.2 算法对比

为了验证 TL-ALNS 算法的性能, 继续在该基准算例集上进行了测试. 我们与现有的先进算法进行了对比, 每个算例都独自运行 20 次. 对比基准包括: Sadati 等^[24] 整理的已知最优解 (BKS); Sadati 等提出的 VTNS 算法, 该算法代表了目前求解 MD-VRPTW 的先进水平; 以及 Wang 等^[5] 提出的自适应大邻域搜索算法 (简称 W-ALNS), 该算法专为动态环境设计, 在计算时效性上具有显著优势.

表 2 汇总了各算法在 20 个标准算例上的详细

结果, 其中对比算法的数值结果均从各自论文提取. 针对本文提出的 TL-ALNS 算法, 表内也展示了其运

行的最优目标值 (Best)、平均目标值 (Avg) 以及平均运行时间 (Time).

表2 对比算法结果

Inst	BKS	VTNS			W-ALNS			TL-ALNS		
		Best	Avg	Time(s)	Best	Avg	Time(s)	Best	Avg	Time(s)
pr01	1074.12	1074.12	1074.12	57.53	1074.12	1080.35	4.91	1074.12	1075.32	29.57
pr02	1762.21	1762.21	1763.54	313.20	1762.21	1787.29	24.66	1763.07	1783.97	73.23
pr03	2373.65	2373.65	2375.49	667.88	2373.65	2402.02	64.97	2408.42	2421.67	132.41
pr04	2814.34	2814.34	2819.16	1678.05	2836.59	2858.46	128.04	2898.04	2917.49	207.26
pr05	2964.65	2965.18	2979.87	1266.46	2968.81	3005.89	219.18	3042.62	3070.11	307.00
pr06	3588.78	3590.58	3601.05	2425.37	3620.71	3647.35	303.82	3659.17	3683.21	347.58
pr07	1418.22	1418.22	1418.28	135.10	1418.22	1425.82	11.23	1418.22	1423.10	44.26
pr08	2096.73	2096.73	2101.29	509.07	2096.73	2124.25	66.74	2150.22	2159.17	125.94
pr09	2712.56	2717.69	2722.96	917.21	2730.86	2765.01	170.92	2732.33	2744.96	253.92
pr10	3465.92	3469.29	3485.29	1805.63	3498.23	3514.41	322.81	3486.92	3512.72	350.99
pr11	1005.73	1005.73	1005.73	20.31	1005.73	1013.53	5.52	1005.73	1008.76	10.26
pr12	1464.50	1464.50	1470.89	249.05	1478.40	1502.50	26.41	1464.50	1478.03	70.03
pr13	2001.81	2001.81	2006.59	287.48	2004.59	2029.14	67.57	2020.58	2037.99	131.66
pr14	2195.33	2195.33	2204.29	837.60	2200.17	2249.05	111.65	2247.72	2269.24	193.66
pr15	2433.15	2434.94	2445.13	697.04	2467.85	2500.78	155.72	2509.75	2531.21	246.31
pr16	2836.67	2850.69	2855.69	1508.42	2872.45	2903.46	265.65	2943.90	2956.33	313.15
pr17	1236.24	1236.24	1239.81	82.51	1236.24	1243.64	11.76	1236.24	1244.27	44.91
pr18	1788.18	1788.18	1794.29	356.03	1802.34	1827.20	57.49	1809.35	1813.09	123.93
pr19	2261.08	2263.74	2270.59	533.27	2275.19	2315.03	162.22	2310.92	2335.46	270.97
pr20	2993.31	2995.08	3011.52	1027.08	3025.61	3027.25	258.16	3059.51	3072.12	323.67
Avg	2224.36	2225.91	2232.28	768.71	2237.44	2261.12	121.97	2262.07	7999.11	180.04

为了更直观地评估算法性能, 我们计算了最优解与平均解相对于 BKS 的百分比偏差 $\Delta\%$, 节省时间的百分比则是以 VTNS 为基准, 结果如表 3 所示. 从解的质量来看, VTNS 算法凭借其复杂的邻域搜索机制, 在寻优精度上表现优异, 其最优解与 BKS 的平均偏差仅为 0.05%, 但这也伴随着巨大的计算开销, 平均耗时高达 768.71 秒, 难以满足应急救援场景下对实时响应的严苛要求. 相比之下, 本文提出的 TL-ALNS 与 W-ALNS 均体现了面向动态场景的设计初衷, 即在解的质量与计算时间之间寻求平衡. 数据表明, 虽然 TL-ALNS 在最优目标值的平均偏差上 (1.44%) 高于 VTNS, 但其平均计算耗时仅为 180.04 秒, 与 VTNS 相比节省了约 68.73% 的时间成本, 与 W-ALNS 处于同一数量级, 均能满足实际应急指挥中的时效性约束.

此外, 在应急救援物资配送等高风险场景中, 决策的可靠性往往优于对极致最优解的追求. 一个虽然运行极快但结果波动较大的算法可能会在紧急时刻输出不可接受的劣解, 从而增加救援行动的不确定性风险. 因此, 在只有对比算法最优解和平均解数据的情况下, 我们将算法输出的稳定性指标 (Stability) 定义为平均解相对于自身最优解的偏差

率, 即 $(Avg - Best) / Best \times 100\%$, 该值越小表明算法在多次运行中输出结果越稳定, 抗随机扰动能

表3 百分比差异

Inst	$\Delta\%$							
	VTNS		W-ALNS			TL-ALNS		
	Best	Avg	Best	Avg	Time(%)	Best	Avg	Time(%)
pr01	0.00	0.00	0.00	0.58	91.47	0.00	0.11	48.60
pr02	0.00	0.08	0.00	1.42	92.13	0.05	1.23	76.62
pr03	0.00	0.08	0.00	1.20	90.27	1.46	2.02	80.17
pr04	0.00	0.17	0.79	1.57	92.37	2.97	3.67	87.65
pr05	0.02	0.51	0.14	1.39	82.69	2.63	3.56	75.76
pr06	0.05	0.34	0.89	1.63	87.47	1.96	2.63	85.67
pr07	0.00	0.00	0.00	0.54	91.69	0.00	0.34	67.24
pr08	0.00	0.22	0.00	1.31	86.89	2.55	2.98	75.26
pr09	0.19	0.38	0.67	1.93	81.37	0.73	1.19	72.32
pr10	0.10	0.56	0.93	1.40	82.12	0.61	1.35	80.56
pr11	0.00	0.00	0.00	0.78	72.82	0.00	0.30	49.49
pr12	0.00	0.44	0.95	2.59	89.40	0.00	0.92	71.88
pr13	0.00	0.24	0.14	1.37	76.50	0.94	1.81	54.20
pr14	0.00	0.41	0.22	2.45	86.67	2.39	3.37	76.88
pr15	0.07	0.49	1.43	2.78	77.66	3.15	4.03	64.66
pr16	0.49	0.67	1.26	2.35	82.39	3.78	4.22	79.24
pr17	0.00	0.29	0.00	0.60	85.75	0.00	0.65	45.57
pr18	0.00	0.34	0.79	2.18	83.85	1.18	1.39	65.19
pr19	0.12	0.42	0.62	2.39	69.58	2.20	3.29	49.19
pr20	0.06	0.61	1.08	1.13	74.86	2.21	2.63	68.49
Avg	0.05	0.31	0.50	1.58	83.90	1.44	2.09	68.73

力越强. 统计结果如表4所示, 尽管 TL-ALNS 在极限寻优能力上与 W-ALNS 算法存在微小差距, 但在算法的稳定性指标上表现出了优势. TL-ALNS 在所有算例上的平均稳定性偏差仅为 0.63%, 显著优于 W-ALNS 的 1.08%. 特别是在 pr12、pr14、pr19 等大规模或复杂算例中, TL-ALNS 的结果波动幅度远小于 W-ALNS, 这得益于双层权重机制对优质算子对组合的有效学习, 扩大了邻域搜索范围, 得以更稳定地收敛至高质量区域.

表4 稳定性差异

Inst	Stability		
	VTNS	W-ALNS	TL-ALNS
pr01	0.00	0.58	0.11
pr02	0.08	1.42	1.19
pr03	0.08	1.20	0.55
pr04	0.17	0.77	0.67
pr05	0.50	1.25	0.90
pr06	0.29	0.74	0.66
pr07	0.00	0.54	0.34
pr08	0.22	1.31	0.42
pr09	0.19	1.25	0.46
pr10	0.46	0.46	0.74
pr11	0.00	0.78	0.30
pr12	0.44	1.63	0.92
pr13	0.24	1.22	0.86
pr14	0.41	2.22	0.96
pr15	0.42	1.33	0.86
pr16	0.18	1.08	0.42
pr17	0.29	0.60	0.65
pr18	0.34	1.38	0.21
pr19	0.30	1.75	1.06
pr20	0.55	0.05	0.41
Avg	0.26	1.08	0.63

3.3 真实案例分析

针对苏州市姑苏区部分老城区因建筑老化而易发安全事故的问题, 在 86 栋建筑中部署了多类传感器, 包括烟雾探测器、电气火灾报警器、建筑物倾斜探测器、可燃气体探测器、磁门传感器、紧急呼叫按钮及智能水表等, 每个事故点与救援中心均具备真实的地理坐标(经纬度). 不同传感器所监测的风险类型各异, 其报警事件的紧急程度也相应不同, 反映到建模场景中就是长度不一、触发时间不同的实时需求时间窗. 随着实时报警信息的持续流入, 系统需动态生成从各救援中心出发、覆盖所有事故点的车辆路径规划方案, 模型构建如 1.3 节所示.

3.3.1 时间窗惩罚权重的敏感性分析

在应急救援场景中, 决策者需要在救援时效性与运力成本之间寻求平衡. 模型中的时间窗延误惩

罚权重 λ 是调节这一平衡的关键参数. 为确定 λ 的合理取值, 针对苏州市姑苏区应急场景数据集, 设计了参数敏感性分析实验.

实验设置的取值集合为 $\{1, 5, 10, 20, 50, 100\}$, 在保持其他参数不变的情况下, 分别运行算法 20 次取平均值. 记录不同权重下的车辆通勤成本 z_1 与时间窗延误成本 z_2 的变化趋势.

如图5所示, 随着 λ 从 1 逐步增大至 100, 两条线段呈现出此消彼长的关系. 在低权重区间, 算法对时间窗延误的容忍度较高. 以 $\lambda=1$ 为例, 虽然车辆通勤成本为 41,250, 但时间窗延误成本高达 3,253, 意味着存在大量严重超时的救援点. 随着 λ 增加至 10, 时间窗延误成本呈现断崖式下降, 表明绝大多数软时间窗约束已被满足. 此时, 通勤成本因车辆需绕行略有上升, 二者在该处形成较为理想的折中点, 总目标值达到较优水平. 当 λ 继续增大时, 时间窗延误成本进一步下降的幅度明显放缓, 通勤成本却快速攀升, 总成本随之迅速扩大. 当 λ 提高至 100 时, 时间窗延误成本几乎不再显著改善, 显示出典型的边际收益递减特征.

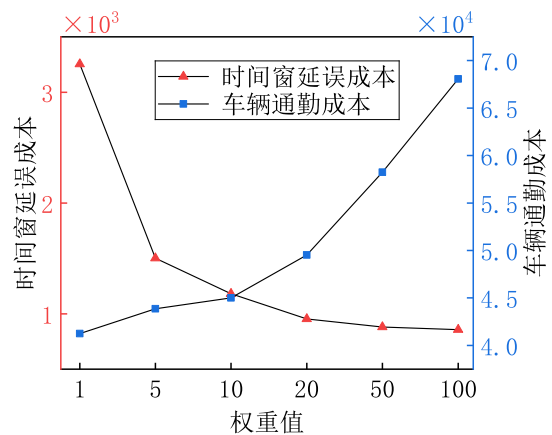


图5 时间窗延误成本的惩罚权重分析

综上所述, 针对姑苏区这一特定场景, 选取 $\lambda=10$ 作为惩罚权重是该数据集下的最佳“肘点”. 该数值并非普适最优, 而是为了适配当前城市路网密度与事故点分布特征, 在确保黄金救援时间的前提下实现系统总成本最优的配置.

3.3.2 实验结果

图6展示了姑苏区内 4 个救援中心与 86 个事故监测点的具体地理空间分布. 值得注意的是, 该数据集具有显著的“中心高密度聚集”特征: 绝大多数事故点并非均匀散布, 而是集中在老城核心区, 且 4 个救援中心同样部署在核心区域周边. 这种特殊的拓扑结构导致了大量的“边界模糊节点”, 即大量事

故点在地理距离上同时邻近多个救援中心, 且距离差异极小, 形成了复杂的多中心服务重叠区域, 对于确定事故点的最优归属带来很大挑战.



图6 姑苏区救援中心与检测点

基于现实应急需求与事故点规模, 每次信息更新的时间间隔设定为 30 分钟, 每次重优化迭代轮数设为 1000. 以 2024 年夏季某日的 12:00-14:00 为例, 整个时间段被分为 4 个时间间隔. 受高温影响, 监测点在此期间频繁报警.

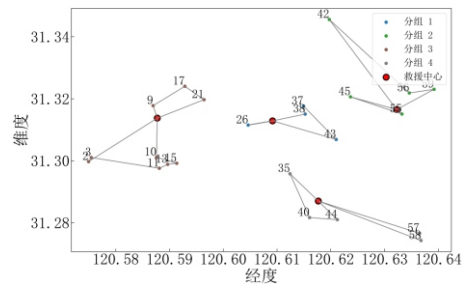
图 7 展示了该时段内的动态调度过程. 四个重优化时间节点均能够依据上一阶段的状态, 高效生成当前周期的救援车辆路径方案. 以更新节点二为例, 其在节点一已有路径方案的基础上, 动态接收并整合了新出现的 31、60 等事故点, 形成新的救援路径. 事故点 28 虽然距救援中心 1 更近, 但出于全局救援成本优化的考虑, 被合理分配至救援中心 3 的管辖范围内. 综上, 系统能够在综合考量多重约束与目标的基础上, 实现科学有效的调度决策.

图 8 展示了各阶段救援成本随优化迭代次数的收敛过程. 在每一个重优化阶段 (即每 1000 次迭代内), 救援成本均呈现出从初始高点迅速下降并最终趋于稳定的收敛趋势. 这证明了 TL-ALNS 算法在每个阶段都能够稳定且快速寻找到高质量解, 从而显著降低全局救援成本. 第四阶段的来回波动则验证了算法具备一定的自我纠错能力, 能够改正前期不合理的聚类结果.

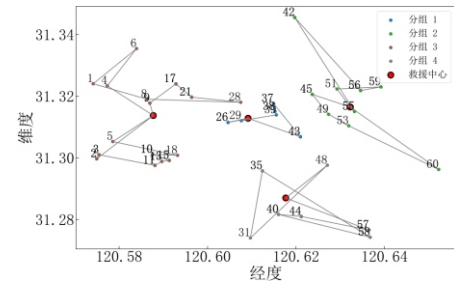
表 5 统计了每次重新优化所消耗的计算时间. 每个阶段的重优化耗时均保持在 1 分钟以内, 满足实际应急救援时的实时响应要求.

3.4 新算子与双层结构的有效性分析

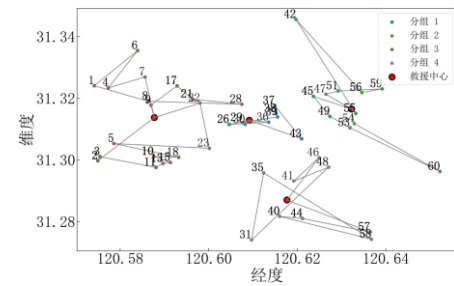
为了深入探究本文提出的 TL-ALNS 算法中两个核心创新模块——重归属破坏算子以及双层自适应权重机制在实际应急救援路径规划中的优化作用, 本节基于苏州市姑苏区应急场景数据集设计了递进



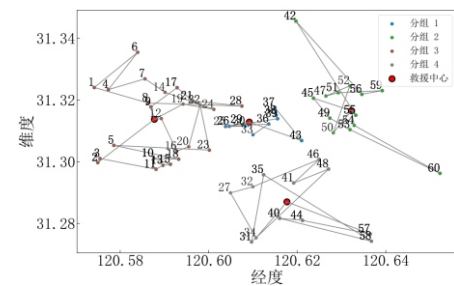
(a) 更新节点一



(b) 更新节点二



(c) 更新节点三



(d) 更新节点四

图7 救援路径阶段优化结果

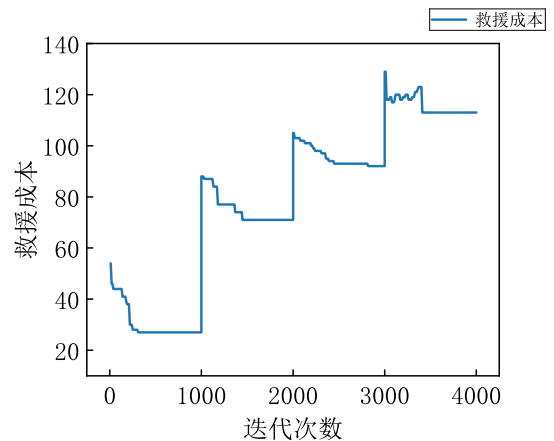


图8 救援成本收敛图

表5 阶段运行时间

	阶段一	阶段二	阶段三	阶段四	平均
Time(s)	28.34	40.19	38.53	42.24	37.33

式的消融实验. 实验构建了三种算法变体进行对比分析: 变体 I 为仅包含经典算子且采用传统单层权重更新机制的基准 ALNS 算法; 变体 II 在变体 I 的基础上引入了本文设计的重归属算子, 但仍保持单层权重机制, 旨在隔离验证新算子在解决多车场归属分配难题上的有效性; 变体 III(TL-ALNS) 则为引入双层自适应权重机制的完整算法, 重点考察该机制对算法稳定性及计算时间的影响. 各变体均在相同参数设置下独立运行 20 次, 统计结果如表 6 所示, 图 9 通过箱线图展示了所有解的分布特征.

表6 算法变体性能、稳定性与时间开销对比

算法	Avg	成本优化率	标准差	Time(s)
变体I	62110.5	—	1430.7	105.4
变体II	58163.5	6.35%	1718.8	108.7
TL-ALNS	56805.5	8.54%	471.5	125.6

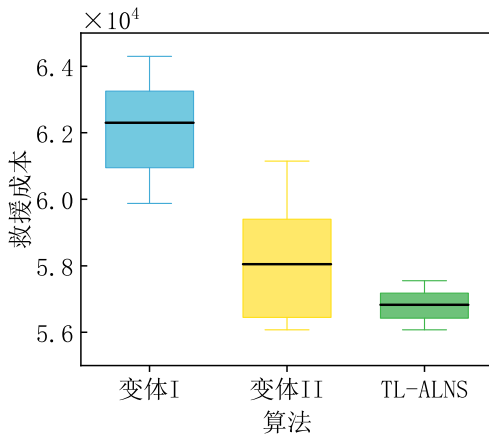


图9 消融实验对比

如表 6 所示, 相较于基准变体 I, 引入重归属算子的变体 II 将平均救援总成本从 62110.5 显著降低至 58163.5, 降幅达 6.35%, 有效应对了归属分配难题. 同时, 虽然变体 II 提升了平均求解质量, 但其标准差反而高于基准算法. 这一现象说明, 尽管新的破坏算子在寻找优质解方面表现优异, 但高频发的大规模破坏会导致搜索过程产生大幅度震荡, 有时会陷入局部收敛.

进一步对比变体 II 与 TL-ALN 可知, 双层权重机制的引入不仅使平均成本进一步降低至 56805.5, 其核心贡献更在于显著提升了算法的稳定性. TL-ALNS 的标准差降至 471.5, 图 9 也表示其解的分布呈现为高度收敛的扁平箱体. 这一现象表明, 双层机

制通过学习更复杂的高效算子组合, 有效抑制了由预测算子引入的波动, 确保算法在每次应急响应任务中均能稳定输出高质量的调度方案.

在计算效率方面, 双层机制引入了额外的计算开销. 如表 6 所示, TL-ALNS 的运行时间为 125.6 秒, 相较于变体 II 的 108.7 秒增加了约 15.5%. 这一时间增长主要源于双层机制需要维护更复杂的权重矩阵以及在每次迭代中进行额外的算子对选择计算. 但应急救援任务通常在分钟级的时间窗口内进行决策, 增加的十几秒计算时间仍在实际调度的可接受范围内.

4 结论

本文对苏州市姑苏区多救援中心应急救援问题进行建模, 采用周期性路径重优化方法进行流程处理. 在求解层面, 提出 TL-ALNS 算法通过双层机制学习算子对之间的协同关系, 设计预测型重归属破坏算子应对事故点高度聚集的特点, 提高了应急救援车辆调度效率.

参考文献 (References)

- [1] Ge S Y, Shan M, Zhai Z. Emergency preparedness in China's urban rail transit system: A systematic review[J]. *Sustainability*, 2025, 17(2): 524.
- [2] 侯莹, 乔聃, 韩红桂. 考虑动态配送时间需求的多策略协同车辆路径优化算法[J]. *控制与决策*, 2026, 41(4): 1143-1153.
(Hou Y, Qiao D, Han H G. Multi-strategy collaborative vehicle routing optimization algorithm considering dynamic delivery time requirements[J]. *Control and Decision*, 2026, 41(4): 1143-1153.)
- [3] Soeffker N, Ulmer M W, Mattfeld D C. Stochastic dynamic vehicle routing in the light of prescriptive analytics: A review[J]. *European Journal of Operational Research*, 2022, 298(3): 801-820.
- [4] Chen Z L, Xu H. Dynamic column generation for dynamic vehicle routing with time windows[J]. *Transportation Science*, 2006, 40(1): 74-88.
- [5] Wang S H, Sun W, Huang M. An adaptive large neighborhood search for the multi-depot dynamic vehicle routing problem with time windows[J]. *Computers & Industrial Engineering*, 2024, 191: 110122.
- [6] Ozbaygin G, Savelsbergh M. An iterative re-optimization framework for the dynamic vehicle routing problem with roaming delivery locations[J]. *Transportation Research Part B: Methodological*, 2019, 128: 207-235.
- [7] Ojeda R B H, Xavier E C, Miyazawa F K, et al. Recent dynamic vehicle routing problems: A survey[J]. *Computers & Industrial Engineering*, 2021, 160: 107604.

- [8] Wang Y, Gou M Y, Luo S Y, et al. The multi-depot pickup and delivery vehicle routing problem with time windows and dynamic demands[J]. *Engineering Applications of Artificial Intelligence*, 2025, 139: 109700.
- [9] 杨静, 石俊刚, 张玉清, 等. 考虑多车场出退勤的城轨交通乘务任务配对模型及算法[J]. *控制与决策*, 2024, 39(7): 2142-2150.
(Yang J, Shi J G, Zhang Y Q, et al. Optimization model and algorithm for urban rail crew pairing problem with multiple depots[J]. *Control and Decision*, 2024, 39(7): 2142-2150.)
- [10] Wang Y, Luo S Y, Fan J X, et al. The multidepot vehicle routing problem with intelligent recycling prices and transportation resource sharing[J]. *Transportation Research Part E: Logistics and Transportation Review*, 2024, 185: 103503.
- [11] Li X H, He B L, Zhao Y, et al. A two-stage optimization approach for multi-depot vehicle routing problem with time windows[C]. 2024 4th International Conference on Computer Communication and Artificial Intelligence. Xi'an, 2024: 540-545.
- [12] Li W T, Zhang Q H, Huang M, et al. Multi-depot vehicle routing problem considering customer satisfaction[C]. Proceedings of the 33rd Chinese Control and Decision Conference. Kunming, 2021: 4208-4213.
- [13] Wen M Y, Sun W, Yu Y, et al. An adaptive large neighborhood search for the larger-scale multi depot green vehicle routing problem with time windows[J]. *Journal of Cleaner Production*, 2022, 374: 133916.
- [14] Ropke S, Pisinger D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows[J]. *Transportation Science*, 2006, 40(4): 455-472.
- [15] 吴健, 姚锋, 杜永浩, 等. 基于数据驱动的自适应并行搜索算法求解多星协同调度问题[J]. *控制与决策*, 2024, 39(12): 4064-4072.
(Wu J, Yao F, Du Y H, et al. A data-driven adaptive parallel search algorithm for multiple agile satellites cooperative scheduling problem[J]. *Control and Decision*, 2024, 39(12): 4064-4072.)
- [16] Pereira V G, Alves-Junior O C, Baldo F. An approach to solve the heterogeneous fixed fleet vehicle routing problem with time window based on adaptive large neighborhood search meta-heuristic[J]. *IEEE Transactions on Intelligent Transportation Systems*, 2024, 25(7): 8148-8157.
- [17] Kovacs A A, Parragh S N, Doerner K F, et al. Adaptive large neighborhood search for service technician routing and scheduling problems[J]. *Journal of Scheduling*, 2012, 15(5): 579-600.
- [18] Voigt S, Kuhn H. Crowdsourced logistics: The pickup and delivery problem with transshipments and occasional drivers[J]. *Networks*, 2022, 79(3): 403-426.
- [19] Holguín-Veras J, Pérez N, Jaller M, et al. On the appropriate objective function for post-disaster humanitarian logistics models[J]. *Journal of Operations Management*, 2013, 31(5): 262-280.
- [20] Kirkpatrick S, Gelatt C D J, Vecchi M P. Optimization by simulated annealing[J]. *Science*, 1983, 220(4598): 671-680.
- [21] Shaw P. A new local search algorithm providing high quality solutions to vehicle routing problems[C]. CP98 Workshop on Large-Scale Combinatorial Optimization and Constraints. Pisa, 1998: 46-53.
- [22] Nagata Y. Efficient evolutionary algorithm for the vehicle routing problem with time windows: Edge assembly crossover for the VRPTW[C]. IEEE Congress on Evolutionary Computation. Singapore, 2007: 1175-1182.
- [23] Cordeau J F, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows[J]. *Journal of the Operational Research Society*, 2001, 52(8): 928-936.
- [24] Hesam S M E, Çatay B, Aksen D. An efficient variable neighborhood search with tabu shaking for a class of multi-depot vehicle routing problems[J]. *Computers & Operations Research*, 2021, 133: 105269.

作者简介

姜志豪 (1999-), 男, 硕士生, 主要研究方向为智能决策优化、生产调度, E-mail: 18361293672@163.com;

戴欢 (1983-), 男, 教授, 博士, 主要研究方向为网络与系统安全、区块链与数据安全, E-mail: daihuanjob@163.com;

徐本连 (1974-), 男, 教授, 博士, 主要研究方向为群智能、多目标跟踪、信息融合, E-mail: xu_benlian@usts.edu.cn.