

# 强化学习驱动的进化算法求解多车覆盖配送问题

潘立军<sup>1</sup>, 张伊帆<sup>2</sup>, 陈靖方<sup>3,4†</sup>, 刘喜梅<sup>1</sup>, 王兴杰<sup>2</sup>, 刘长石<sup>5</sup>

(1. 湖南工程学院 商学院, 湖南 湘潭 411104; 2. 湖南工程学院 纺织服装学院, 湖南 湘潭 411104;  
3. 北京工业大学 信息科学技术学院, 北京 100124; 4. 数字社区教育部工程研究中心, 北京 100124;  
5. 湖南工商大学 工商管理学院, 长沙 410205)

**摘要:** 包裹储物柜有效解决了末端配送客户与快递员交接时间不匹配的问题. 然而, 随着包裹储物柜的大量部署, 如何让客户更便利地使用储物柜进行交接, 已成为影响末端配送服务满意度的关键因素之一. 针对这一挑战, 提出考虑客户满意度的多车辆覆盖配送问题. 在问题层面, 运用问卷调查构建客户满意度函数, 进而建立该问题的数学模型; 在方法层面, 提出一种强化学习驱动的进化算法. 首先, 设计混合启发式方法, 用于生成高质量初始种群; 其次, 基于问题特性设计 9 种邻域算子与贪婪修复启发式方法, 用于高效搜索满意可行解; 进而, 提出一种强化学习驱动的搜索机制, 用于自适应选用合适算子; 最后, 设计一种基于单调下降基准函数的状态表征方法, 用于引导智能体学习算法的收敛进程. 仿真实验结果表明, 所提出算法在不同规模问题上均获得高质量解, 其求解性能优于对比算法和求解器; 消融实验结果表明, 状态函数引导的强化学习搜索机制平均提升算法性能达 5%.

**关键词:** 覆盖配送问题; 末端配送; 客户满意度; 强化学习; 进化算法

中图分类号: TP181; U491 文献标志码: A

DOI: 10.13195/j.kzyjc.2026.0149

引用格式: 潘立军, 张伊帆, 陈靖方, 等. 强化学习驱动的进化算法求解多车覆盖配送问题 [J]. 控制与决策.

## Reinforcement learning-driven evolutionary algorithm for solving multi-vehicle covering delivery problem

PAN Li-jun<sup>1</sup>, ZHANG Yi-fan<sup>2</sup>, CHEN Jing-fang<sup>3,4†</sup>, LIU Xi-mei<sup>1</sup>, WANG Xing-jie<sup>2</sup>, LIU Chang-shi<sup>5</sup>

(1. School of Business, Hunan Institute of Engineering, Xiangtan 411104, China; 2. School of Textiles and Garments, Hunan Institute of Engineering, Xiangtan 411104, China; 3. School of Information Science and Technology, Beijing University of Technology, Beijing 100124 China; 4. Engineering Research Center of Digital Community, Ministry of Education, Beijing 100124, China; 5. School of Management, Hunan University of Technology and Business, Changsha 410205, China)

**Abstract:** Parcel lockers effectively resolve the mismatched handover times between customers and couriers. However, with the widespread deployment of parcel lockers, selecting a locker convenient for customer handover has become one of the key factors influencing satisfaction in last-mile delivery services. To address this challenge, this paper proposes a multi-vehicle coverage delivery problem considering customer satisfaction. At the problem level, a questionnaire survey approach was employed to model the customer satisfaction, thereby establishing the mathematical model for this problem. At the methodological level, a reinforcement learning-driven evolutionary algorithm is proposed. Firstly, a hybrid heuristic method is designed to generate a high-quality initial population. Secondly, nine neighbourhood operators and a greedy repair heuristic are devised based on the problem's characteristics to efficiently search for satisfactory feasible solutions. Subsequently, a reinforcement learning-driven search mechanism is proposed to adaptively select suitable operators. Finally, a state representation method based on a monotonic descent benchmark function is designed to guide the convergence process of the agent learning algorithm. Simulation results demonstrate that the proposed algorithm achieves high-quality solutions across problems of varying scales, exhibiting superior computational performance compared to benchmark algorithms and solvers. Ablation experiments reveal that the state function-guided reinforcement learning search mechanism enhances algorithmic performance by an average of 5%.

收稿日期: 2026-02-13; 录用日期: 2026-04-24.

基金项目: 国家社会科学基金年度项目 (24GBL247); 国家自然科学基金项目 (62403272).

责任编辑: 王凌.

†通信作者. E-mail: chenjingfang@bjut.edu.cn.

**Keywords:** covering delivery problem; last-mile delivery; customer satisfaction; reinforcement learning; evolutionary algorithm

## 0 引言

我国电子商务交易额近几年持续增长, 2024 年实物商品网上零售额已达 13.08 万亿元, 增长 6.5%, 快递包裹量也随之激增, 大量网购包裹需由快递员配送至客户. 但快递员递送与客户接收包裹的时间通常不同步, 易产生二次递送, 导致最后一公里配送效率低下<sup>[1-2]</sup>. 为了克服这一配送瓶颈, 在最后一公里配送区域中配置包裹储物柜 (Parcel Locker, PL) 已成为有效解决措施<sup>[3-5]</sup>, 快递公司可将包裹临时寄存于 PL, 客户可依据自身时间安排从 PL 提取包裹. 目前, PL 在全球范围的部署规模正持续扩大, 据统计, 截至 2024 年, 欧洲的包裹柜数量已超过 40 万台, 中国主要城市的智能快递柜投放量也已超过 80 万组, 年均处理的包裹量高达数十亿件. 大量实证研究表明, 合理配置 PL 可显著提升末端配送效率<sup>[6-8]</sup>.

大量部署 PL 虽为交付提供了便利, 但也给末端配送环节带来了挑战. 快递员不仅需在服务客户前确认交付模式 (上门配送或存入 PL), 还需确定服务客户的路径顺序, 以期提升客户满意度. 这一配送场景可建模为覆盖配送问题 (CDP, Covering Delivery Problem)<sup>[9]</sup>: 包含仓库、PL 和客户三类顶点, 其中客户可分为"上门配送"、"存入 PL"、"两种交付模式均可"三类, 目标是规划一条具有最小长度和 PL 使用成本的哈密顿回路, 以满足各类客户的要求.

CDP 是覆盖旅行商问题 (Covering Traveling Salesman Problem, CSP) 及 (Covering Tour Problem, CTP) 的变体问题. CSP 由 Current 与 Schilling<sup>[10]</sup> 于 1989 年提出, 其目标是在一组节点的子集中寻找长度最短的环游线路, 且要求所有不在线路中的节点, 都被线路中至少一个节点覆盖. Maziero 等<sup>[11]</sup> 利用分支切割算法求解 CSP 并取得较好效果. Salari 等提出局部搜索算法与混合蚁群优化算法, 并通过仿真测试验证了算法求解 CSP 的有效性<sup>[12-13]</sup>. Gendreau 等将 CSP 中的节点划分为三类, 即必须访问节点、必须覆盖节点以及可覆盖或可访问节点, 并进一步提出了 CTP, 其目标是在必须访问节点的全部集合与可覆盖或可访问节点的子集之上, 构建一个长度最小的循环, 且未被访问的节点需被循环中至少一个节点覆盖<sup>[14]</sup>. Baldacci 等将 CTP 建模为双商品网络流问题, 开发了两种散点搜索元启发式算法<sup>[15]</sup>. Hachicha 等将经典 CTP 扩展为多车辆覆盖路径问题 (Multi-vehicle CTP, M-CTP), 其目标是在子集上

寻找总长度最小的车辆路线集合, 确保中每个节点至少被一条路线覆盖, 针对该问题建立了整数线性规划模型, 并开发了三种启发式算法, 通过随机实例及实际案例验证算法效果<sup>[16]</sup>. Glize 等基于列生成技术提出了一种 M-CTP 的精确解法<sup>[17]</sup>. Torabi 等基于海上监测这一实际场景提出了 CTP 的变体, 即可变覆盖半径的覆盖路径问题 (CTP with Varying Coverage, CTP-VC), 该问题中节点的覆盖半径会依据车辆在节点的停留时间进行动态变化<sup>[18]</sup>. Torabi 等还提出了一种基于深度强化学习超启发式算法, 通过大量实验与分析验证了该方法在求解 CTP-VC 的显著效果<sup>[19]</sup>.

在 CTP 的基础上, Jiang 等考虑部署 PL 的末端物流配送场景并提出了 CDP, 该问题中选择存入 PL 服务的客户需要支付相应费用, 目标是最小化 PL 使用费用与路径费用<sup>[9]</sup>. Li 等采用双层规划框架将问题分解为领导者问题与跟随者问题, 开发了混合生物地理优化算法, 通过标准测试集验证了所提方法的显著优势<sup>[20]</sup>. Vukićević 等考虑电动汽车配送场景, 旨在设计一次环游来满足客户的需求并最大限度减少旅行总时间, 作者建立了该问题的混合整数规划模型, 并开发了有效的可变邻域搜索启发式算法<sup>[21]</sup>. Tao 等在双层规划模型的基础上, 提出了一种基于局部搜索的异构教学-学习优化算法求解 CDP, 并基于标准测试集验证了所提算法的显著优越性<sup>[22]</sup>.

但相比实际场景, 现有 CDP 建模存在两个不足: 一是大多考虑单车场景, 当配送任务量大, 需要多车服务时, 难以满足需求; 二是当有多个 PL 可供客户选择时, 未考虑客户满意度来合理选择 PL. 在当前车辆路径问题的研究成果中, 对客户满意度的考量主要集中在快递员送达时间与客户期望收货时间的偏离度. 王利娟等在车辆路径模型引入软时间窗来量化满意度指标<sup>[23]</sup>. 侯莹等研究了考虑动态配送时间需求的多策略协同车辆路径优化问题, 将最小等待时间纳入优化目标, 构建差分进化算法进化策略库, 设计多策略协同车辆路径优化算法求解该问题<sup>[24]</sup>. Rajak 等将满意度纳入多车场模型, 建立了适配求解的集成模型. 而在本文研究的问题场景下, PL 的大规模部署, 已有效解决了因"时间不匹配"引发的客户不满, 如何选择具体的快递柜点位, 以最大化客户取件便利性, 已成为影响服务满意度的新的关键因素<sup>[25]</sup>. 因此, 构建以取件便利性为核心的 CDP 客户

满意度模型,是值得进一步探索的研究方向。

随着实际配送问题复杂度的不断提升,单一方法已难以满足问题高效求解需求。目前,大量研究<sup>[26-29]</sup>将强化学习等先进技术融入各种调度优化问题的求解框架,有效提升了算法性能。针对异构车辆路径问题, Qin 等开发了一种基于强化学习的超启发式算法,将具有不同特性的元启发式算法作为底层策略,并采用基于策略的强化学习实现高层的策略选择,验证了算法在大规模问题求解中的显著优势<sup>[30]</sup>。赵仕存等利用强化学习驱动进化的模因算法求解准时制分布式柔性作业车间调度问题,强化学习将交配池中的父本视为状态和动作,并以子代的质量评估环境奖励,仿真显示较好的效果<sup>[31]</sup>。Tang 等则使用双 Q 学习自适应调整遗传算法求解该类问题,取得了较好的求解效果<sup>[32]</sup>。鉴于此,如何引入强化学习赋能复杂 CDP 高效求解,值得深入研究。

基于上述现状分析,本文聚焦部署多个 PL 的末端配送场景,以客户取件便利性和配送成本为优化目标,研究考虑客户满意度的多车覆盖配送问题 (Multi-Vehicle CDP Considering Customer Satisfaction, M-CDPCS),旨在弥补现有文献主要集中在单车 CDP 以及未充分建模客户满意度的不足。针对该复杂问题,本文提出一种 Q 学习驱动的进化算法 (Q-learning-driven evolutionary algorithm, QEA),通过启发式初始化、考虑问题特征的局部搜索算子与贪婪修复策略、状态引导 Q 学习的局部搜索等策略,实现高效求解。

## 1 问题描述与数学建模

### 1.1 基于问卷调查的客户满意度度量

在部署 PL 的末端配送场景,交接服务的便利性是影响客户满意度的关键因素。客户取件距离长短往往是客户选择 PL 的关键指标,直接决定了满意度水平。因此,本文利用客户取件出行距离来衡量客户满意度。但实践中客户存在个体差异,不同客户对相同取件距离的满意度不一。为此,本文进一步采用问卷调查的方式,旨在准确刻画客户满意度与取件距离间的关系,具体过程如下:

首先,制定包括三个关键问题的问卷:1)"到离家或公司多少距离内的 PL 取件,您感觉满意?" 2)"当取件超过多远时,您感觉不满意?" 3)"您日常单次步行的平均距离是多少?". 进而,选取湖南长沙、湘潭、株洲三市 350 名经常利用 PL 取件的小区居民、公司员工填写问卷,通过问卷星平台进行问卷发放、回收与整理,共获得 332 份有效问卷。统计分析有以下发现:1)用户日常单次步行的平均距离因年龄、性

别、身体状况等原因有所差异;2)用户满意的取件距离与单次步行平均距离的比值相对固定;3)超过用户日常单次步行平均距离去取件会导致不满意。因此,本文用式 (1) 来量化用户满意度:

$$S_b = \begin{cases} 1, & l_b \leq \theta \cdot r_b \\ e^{-\lambda(l_b - \theta \cdot r_b)}, & \theta \cdot r_b < l_b < r_b \\ 0, & l_b \geq r_b \end{cases} \quad (1)$$

其中,  $l_b$  和  $r_b$  分别为客户的取件距离和日常单次步行的平均距离,  $\lambda > 0$  和  $\theta \in [0.4, 0.6]$  为函数参数,  $\lambda$  表征满意度衰减速率,  $\theta$  刻画客户满意的取件距离占日常步行平均距离的比例,  $S_b$  的取值范围为  $[0, 1]$ 。

### 1.2 问题描述

在部署 PL 的末端配送场景下,配送员可将包裹直送客户或存入客户周边的 PL。物流公司有一组同质快递车辆,每辆车从中心仓库出发,完成任务后返回。配送线路需确保每个客户要么被访问,要么其取件范围内的 PL 被车辆访问。问题目标是在满足车辆容量约束的前提下,最优化车辆总行驶距离与式 (1) 所示的客户满意度。图 1 展示了该问题的一个示例,实线为车辆的配送路径,虚线圈代表圈中心客户的出行半径覆盖区域。

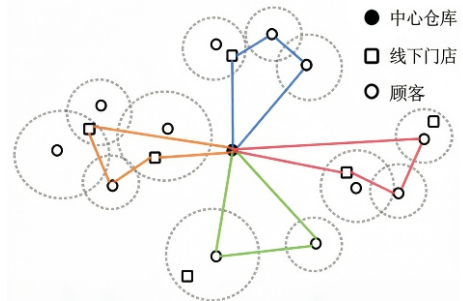


图1 考虑满意度的多车覆盖配送路径问题示意图

### 1.3 数学模型

模型中使用的参数与决策变量如表 1 所示。根据以上问题描述,建立整数规划模型如下。

$$\min \sum_{k \in K} \sum_{i, j \in V} d_{ij} x_{ijk} + f \sum_{k \in K} \sum_{i \in V_a \cup V_b} \sum_{j \in V_d} x_{ijk} + h \sum_{b \in V_b} (1 - S_b), \quad (2)$$

$$\text{s.t.} \sum_{a \in V_a} y_{ba} I_{ba} + \sum_{k \in K} W_{bk} = 1, \forall b \in V_b, \quad (3)$$

$$\sum_{i \in V} x_{ijk} = \sum_{i \in V} x_{jik} = W_{jk}, \forall j \in V, \forall k \in K, \quad (4)$$

$$\sum_{j \in V_a \cup V_b} x_{djk} = \sum_{j \in V_a \cup V_b} x_{jdk} = W_{dk}, \forall d \in V_d, \forall k \in K, \quad (5)$$

$$\sum_{i \in V} q_i w_{ik} \leq Q, \forall k \in K, \quad (6)$$

$$\sum_{i, j \in V} u_{ijk} - \sum_{i, j \in V} u_{jik} = \sum_{i, j \in V} x_{jik}, \forall k \in K, \quad (7)$$

$$u_{ijk} \leq n x_{ijk}, \forall i, j \in V, \forall k \in K, \quad (8)$$

$$\sum_{k \in K} \sum_{i \in V_a \cup V_b} \sum_{j \in V_d} x_{ijk} < |K|, \quad (9)$$

$$\sum_{k \in K} W_{ak} \leq \sum_{b \in V_b} y_{ba}, \forall a \in V_a, \quad (10)$$

$$y_{ba} \leq I_{ba}, \forall a \in V_a, \forall b \in V_b. \quad (11)$$

式(2)为目标函数,最小化总运输成本和客户不满意处罚成本;式(3)确保任意客户点要么被访问,要么由至少一个PL服务;式(4)确保节点出入平衡;式(5)确保任意车辆需从配送中心出发再回到配送中心;式(6)为车辆容量约束;式(7)和式(8)为消除子回路约束;式(9)为车辆数量约束;式(10)为PL访问约束,如果有客户选择PL取件,则该PL必须被至少一辆车访问;式(11)确保客户只能由其覆盖范围内的PL服务。

表1 问题符号表

类型	符号	描述
参数	$V_a$	PL集, $V_a = \{1, 2, \dots, N\}$ , 共 $N$ 个
	$V_b$	客户集, $V_b = \{N+1, N+2, \dots, N+M\}$ , 共 $M$ 个
	$V_d$	中心仓库, $V_d = \{N+M+1\}$
	$V$	节点集合, $V = V_a \cup V_b \cup V_d$
	$E$	边集合, $E = \{(i, j) \mid i, j \in V, i \neq j\}$
	$d_{ij}$	边 $(i, j)$ 的长度
	$f$	单车辆固定使用成本
	$h$	最大不满意惩罚成本
	$K$	车辆集合, $K = \{1, 2, \dots, k\}$
	$q_b$	客户 $b$ 的需求量
	$Q$	车辆容量
	$r_b$	客户日常单次步行平均距离
	$I_{ba}$	覆盖关系矩阵, 若以 $r_b$ 为半径的区域覆盖 $PL_a$ , 则为1, 否则为0
$S_b$	客户满意度	
决策变量	$x_{ijk}$	0-1变量, 车辆 $k$ 经过边 $(i, j)$ 为1, 否则为0
	$W_{ik}$	0-1变量, 车辆 $k$ 经过节点 $i$ 为1, 否则为0
	$y_{ba}$	0-1变量, 客户 $b$ 由 $PL_a$ 服务为1, 否则为0

## 2 强化学习驱动的进化算法

### 2.1 编码与种群初始化

算法采用整数编码,以包含中心仓库、PL与客户点编号的序列作为个体染色体,用于表征车辆依次访问的顺序。图2给出了一个编码示例,其中0表

示中心仓库,1、2为PL,其余为客户点,该染色体表示2台车的配送路线安排,行驶路线分别为0240,05610。

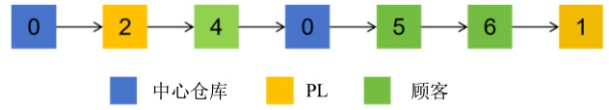


图2 编码示例

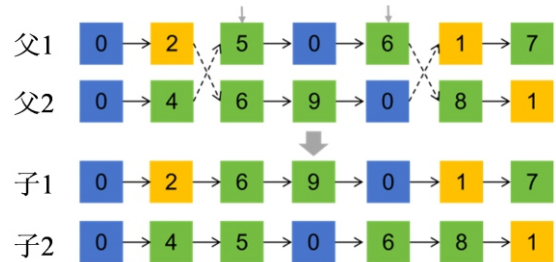
初始种群质量对进化算法求解性能影响较大。为了生成优质初始种群,本文设计了一种混合初始化方法,包含三种生成策略。一是随机策略,从PL和客户集合中随机选择节点插入到当前路径,直至所有客户节点均被服务。二是最近邻居插入策略,从中心仓库节点出发,在待访问节点集合中选择距离当前节点最近且满足容量约束的节点插入到路径中,直至所有客户节点均被服务。三是节约算法,利用节约值引导插入点的选择,直至所有客户节点均被服务。以上策略中,若被选择节点为PL,则运用覆盖关系矩阵将能被该PL服务的客户点加入到当前路径。

### 2.2 交叉算子

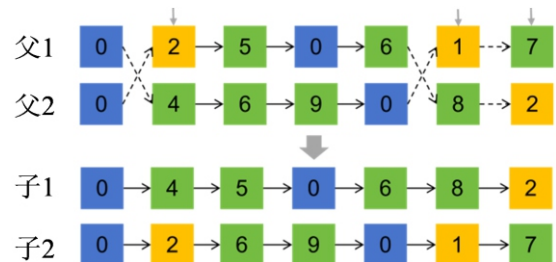
算法采用轮盘赌策略选择两个个体进行交叉,交叉时先不考虑客户与PL的覆盖关系,依据两个个体的基因信息开展两点交叉、多点交叉,具体如下:

(1) 两点交叉。在配对的两个父染色体中随机选择2个交叉位,交换两位置区间内的基因。如图3(a)所示,随机选择父染色体中的第3到5个位置进行交叉,进行片段交换后生成两个子染色体。

(2) 多点交叉。在配对的两个父染色体中随机选



(a) 两点交叉



(b) 多点交叉

图3 交叉算子示意图

取多个交叉位,交换交叉位上的基因,产生新的子代个体.如图3(b)所示,随机选择父染色体位置2、6、7的基因进行交换,生成两个子代染色体.

由于交叉时未考虑客户与PL的覆盖关系,子代个体可能会出现重复基因、冗余覆盖(多个PL可以服务同一个客户)、节点缺失(部分客户或其覆盖的PL未访问),以及线路超载等问题,为此本文设计了贪婪修复启发式方法对交叉后的染色体进行修复,如算法1所示.

---

**Algorithm 1** 贪婪修复启发式方法
 

---

```

1: 输入: 交叉后的染色体 $R$ , PL集 $V_a$ , 客户集 $V_b$ , 覆盖关系矩阵 $I_{ba}$ 
2: 输出: 修复后的染色体 $R'$ 
3: 初始化已服务的客户集合 $C_r = \{\}$ ,  $R' = \{\}$ 
4: 设置当前车辆载重量 $Load = 0$ 
5: 将配送中心加入 $R'$ 
6: for  $r$  in  $R$  do                                ▷遍历 $R$ 中的节点
7:   if  $r$  in  $C_r$  then                                ▷重复节点
8:     continue
9:   else if  $r$  in  $V_a$  then                            ▷当前点是PL
10:    获取可覆盖 $r$ 的客户点集 $PL_a\_Cus$ 
11:    计算 $PL_a\_Cus$ 集合的客户总需求量 $Q_{pta}$ 
12:    if  $Q_{pta} + Load \leq Q$  then
13:       $R' = R' \cup \{r\}$                                 ▷将PL加入到 $R'$ 中
14:       $C_r = C_r \cup PL_a\_Cus$                             ▷更新已服务客户集合
15:    else
16:      将 $PL_a\_Cus$ 中的点按需求量升序排列
17:       $S = Q - Load$                                     ▷计算当前车辆的剩余载重量
18:      选择 $PL_a\_Cus$ 中前 $n$ 个需求量之和不超过 $S$ 的客户子集 $Cus_a$ 
19:       $R' = R' \cup \{r\}$                                 ▷将PL加入到 $R'$ 中
20:       $C_r = C_r \cup Cus_a$                             ▷更新已服务客户集合
21:    end if
22:    else                                ▷当前点是Cus
23:      if  $Load + q_r > Q$  then
24:         $R' = R' \cup \{0\}$                             ▷新开一条路径
25:         $Load = q_r$ 
26:      else
27:         $R' = R' \cup \{r\}$ 

```

```

28:         $C_r = C_r \cup \{r\}$ 
29:         $Load = Load + q_r$ 
30:      end if
31:    end if
32: end for

```

---

### 2.3 变异算子

变异算子用于进化算法的局部搜索,基于问题结构特点设计了以下9种变异算子.

1) 2-交换:选择单条路径,交换两个访问节点.如在某条路径 $[0,1,4,5,2,3]$ 中,交换位置3和位置5的节点,得到新路径 $[0,1,2,5,4,3]$ .

2) 片段反转:选择单条路径,反转路径中的子片段.如在某条路径 $[0,1,4,5,2,3]$ 中,反转位置3到位置5的节点片段,得到新路径 $[0,1,2,5,4,3]$ .

3) 3-交换:选择单条路径,选择三个位置点重组节点片段.如在某条路径 $[0,1,4,5,2,3]$ 中,选择位置3、4、5,将路径分为三个片段,重新连接片段得到新路径 $[0,1,4,2,3,5]$ .

4) 重插入:选择单条路径,将节点移到同路径内新位置.如在某条路径 $[0,1,4,5,2,3]$ 中,将位置6的节点移到位置3节点后,得到新路径 $[0,1,4,3,5,2]$ .

5) 路径间2-交换:交换不同路径间的两个节点.如交换路径 $[0,1,4,5]$ 的节点1与路径 $[0,2,3,6]$ 的节点6,得到新的路径 $[0,6,4,5]$ 与 $[0,2,3,1,7]$ ,如图4(a)所示.

6) 路径间重插入:选择一条路径中任意一个位置的节点,将其插入到另一条路径.如将路径 $[0,2,3,7]$ 中节点7从路径中移除,插入路径 $[0,1,4,5]$ 的节点4之后,获得新路径 $[0,2,3]$ 和 $[0,1,4,7,5]$ ,如图4(b)所示.

7) 服务模式切换:任选一个客户节点,将其删除,并将其对应的PL节点插入到原位置.如客户5可由PL8服务,将路径 $[0,1,4,5]$ 中节点5删除并插入节点8,得到新路径 $[0,1,4,8]$ ,如图4(c)所示.

8) PL变换:任选一个PL节点,将其替换为另一个未被访问的PL节点.如在路径 $[0,1,4,5]$ 中,将PL1替换为PL8,得到新路径 $[0,8,4,5]$ ,如图4(d)所示.

9) 破坏-修复:在个体中任意移除 $k$ 个节点后再重新插入到最优位置.如路径 $[0,1,4,5,2,3]$ 中,移除节点1和3,重新插入当前路径的最优位置,得到新路径 $[0,4,1,5,3,2]$ .

上述变异算子中,2-交换、片段反转、3-交换、重插入、破坏-修复5种算子只作用于单条路径,不影响路径间的节点分配,而路径间2-交换、路径间重插

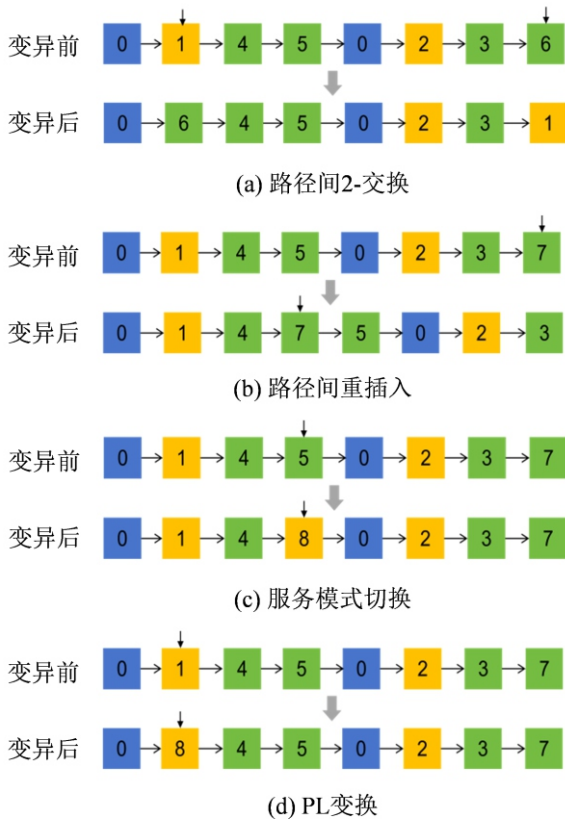


图4 部分变异算子示意图

入、服务模式切换、PL变换4种算子涉及路径间的节点变换,会产生冗余覆盖、节点缺失以及线路超载等问题,需运用2.2节中的贪婪修复启发式方法实施修复.部分变异算子的示意图如图4所示.

## 2.4 状态引导的强化学习搜索机制

利用变异算子增强个体适应度是提升进化算法性能的关键.为此,本文提出了一种状态引导的强化学习搜索机制,运用理想的状态变化参数引导Q学习状态更新,智能体则依据状态自适应调整每种变异算子的选择概率,以提升搜索效率.

### 2.4.1 状态表征与引导

为刻画种群在进化过程中的动态特性,本文提出基于种群多样性变化( $S_1$ )与最优解变化( $S_2$ )的2维状态表征方法,用种群个体适应度方差变化表征 $S_1$ ,用种群最优解适应值变化表征 $S_2$ ,每个状态均有改进(记为"+")、未改进(记为"-")两种取值,共4个组态,如表2所示, $S_1$ 和 $S_2$ 的取值方式如下.

表2 状态表征

组态	种群多样性变化 $S_1$	最优解改进 $S_2$	种群状态
$S_1+, S_2+$	增加	变优	探索开发均衡
$S_1+, S_2-$	增加	未变优	开发不足
$S_1-, S_2+$	未增加	变优	定向收敛
$S_1-, S_2-$	未增加	未变优	局部最优

若当前种群最优个体适应度优于上一代最优个体适应度, $S_2$ 取"+",否则取"-".通常而言,种群个体适应度方差随进化算法的收敛而不断降低,因此,对种群多样性的评价基准需适配算法的收敛进程.为此,本文设计了一个单调下降的参照函数 $B_{\text{var}}(t)$ ,作为进化过程中种群个体适应度方差的理想值,并将其与种群个体适应度的实际方差 $G_{\text{var}}(t)$ 进行比较,从而实现对种群多样性变化的合理评价. $B_{\text{var}}(t)$ 的具体形式如下:

$$B_{\text{var}}(t) = \begin{cases} I_{\text{var}}, & t = 0 \\ B_{\text{var}}(t) \cdot \beta, & \text{mod}(t, N) = 0 \\ B_{\text{var}}(t-1), & \text{mod}(t, N) \neq 0 \end{cases} \quad (12)$$

其中, $t$ 为种群进化代数; $I_{\text{var}}$ 为初始种群个体适应度值方差; $\beta$ 为下降率,取值范围为 $(1, 0.8]$ ,用于每隔 $N$ 代使 $B_{\text{var}}(t)$ 衰减.若 $G_{\text{var}}(t) > B_{\text{var}}(t)$ ,则 $S_1$ 取"+",否则取"-". $B_{\text{var}}(t)$ 随种群进化而单调下降的特点,使得状态 $S_1$ 隐含了算法的收敛信息,从而能够引导智能体学会依据算法的收敛过程合理选择算子.

### 2.4.2 动作设计

QEA采用上述9个变异算子作为各状态下智能体可采取的动作,基于当前状态和动作对应的Q值计算动作选择概率,并依据概率执行相应动作,Q值越大,选择该动作的可能性越大,反之则越小.Q值更新采用以下公式:

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha_L [r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q_t(s_t, a_t)]. \quad (13)$$

其中, $Q_t(s_t, a_t)$ 是当前状态-动作对的Q值; $\alpha_L$ 为学习率,控制新信息对旧Q值的更新幅度; $r_t$ 为当前奖励; $\gamma$ 为折扣因子,表示对未来奖励的重视程度; $\max_{a \in A} Q(s_t, a)$ 表示下一状态 $s_{t+1}$ 下所有可能动作的最大Q值.

### 2.4.3 奖励设计

奖励 $r_t$ 用于评估所选算子在当前状态下的寻优效果,本文计算方式如下:

$$r_t(a) = (e_t(a) - e_{t-1}(a)) \times \frac{T_t^{\text{sel}}(a) + T_t^{\text{eff}}(a)}{2}, \quad (14)$$

$$e_t(a) = \frac{T_t^{\text{eff}}(a)}{T_t^{\text{sel}}(a)}. \quad (15)$$

其中, $e_t(a)$ 用来衡量动作 $a$ 在第 $t$ 代的改进效果, $T_t^{\text{sel}}(a)$ 和 $T_t^{\text{eff}}(a)$ 分别为第 $t$ 代中选择动作 $a$ 的次数以及选择动作 $a$ 后改进的次数.每轮迭代中,基于动作 $a$ 前后两代的改进效果之差以及本轮选中与有效的次数,计算其奖励值.

## 2.5 算法流程

由以上算法环节可见, QEA 的核心在于, 基于问题特征的启发式初始化策略、交叉和变异算子, 以及强化学习驱动的算子自适应选择策略. 算法的具体步骤如下:

Step1: 采用混合初始化方法生成  $PS$  个个体并计算个体适应度值;

Step2: 采用轮盘赌策略选两个父代个体交叉生成新个体, 对交叉后的个体执行贪婪修复策略, 将修复后的个体放入个体池  $C1$ ;

Step3: 从个体池  $C1$  中随机选择个体, 计算状态 ( $S1, S2$ ), 采用  $Q$  学习机制选取合适的交叉算子并执行, 将新个体放入新个体池  $C2$ ;

Step4: 统计各算子的选用次数、有效次数, 计算各算子的效率值, 依据式 (14)、(15) 计算各算子奖励值, 并更新  $Q$  表;

Step5: 将个体池  $C1$  与  $C2$  合并, 淘汰重复个体, 保留适应度值最优的前  $PS$  个个体, 更新最优解;

Step6: 若最优解连续  $N$  代未更新, 采用初始化策略随机生成个体替换当前种群后 20% 的个体;

Step7: 若满足算法终止条件, 则输出当前最优解; 否则, 转至 step2.

## 3 实验结果

### 3.1 实验设置

#### 3.1.1 测试环境

所有算法均在相同计算环境运行, 以确保结果的可比性和可重复性. 测试平台硬件配置为 Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz 处理器. 算法均采用 Python 3.8 语言开发, 并在 PyCharm 2024.3.5 开发环境中编译运行. 每个算法在每个测试实例上独立运行 20 次, 记录其性能指标, 并采用显著性水平为 0.05 的 Mann-Whitney U 检验来判断算法性能之间是否存在显著差异, 性能明显劣于 QEA 的结果用 "(\*)" 标记.

#### 3.1.2 算例构造

由于所研究问题尚无标准测试算例, 本文基于 TSPLIB 标准算例集来构造问题算例, 具体方法如下. 原始算例中的第一个点为配送中心, 其余点按照 4:6 的比例分别设定为 PL 点与客户点, 每个客户的单次步行的平均距离  $r_b$  由算例中各点间的平均距离与 0-1 间随机小数相乘产生, 对于相同点集取不同的随机数种子生成不同算例. 基于以上方法, 本文生成了三个测试算例集, 分别为小规模算例集  $S(|V| \leq 100)$ 、中规模算例集  $M(100 < |V| \leq 200)$  和大规模

算例集  $L(200 < |V| \leq 600)$ , 算例命名规则为  $Tv_s$ , 其中  $T, v, s$  分别代表算例集规模 ( $S/M/L$ )、节点总数, 以及使用的随机数种子.

#### 3.1.3 参数设置

鉴于现有文献中暂无求解本文所提新问题 M-CDPCS 的针对性算法, 故采用通用优化方法与 QEA 开展对比测试, 包括 GUROBI 求解器、遗传算法 (Genetic Algorithm, GA)、模因算法 (Memetic Algorithm, MA). 满意度函数参数依据调研结果拟合设定, 其他算法参数及问题参数通过预实验合理设置, 具体如表 3 所示.

表3 参数设置

参数	GA	MA	QEA
$\lambda$	1	1	1
$\theta$	0.4	0.4	0.4
种群规模	50	50	50
最大迭代次数	100	100	100
交叉概率	0.8	0.6	0.6
变异概率	0.1	/	/
学习率 $\alpha_L$	/	/	0.01
折扣因子 $\gamma$	/	/	0.9
个体适应度方差基准值下降率 $\beta$	/	/	0.9
个体适应度方差基准值下降间隔代数 $N$	/	/	10
最大不满意度惩罚成本 $h$	1000	1000	1000
单车辆固定使用成本 $f$	300	300	300

表 4、表 5、表 6 分别展示了各算法在小、中、大规模算例集上的表现, 表中计算时间以秒 (s) 为单位. 评价指标为偏差  $Gap$ , 计算方法如式 (16) 所示.

$$Gap_i = \frac{Metric_{QEA} - Metric_i}{Metric_i} \times 100\%. \quad (16)$$

其中,  $Gap_i$  表示 QEA 与对比算法  $i$  之间的指标偏差,  $Metric_{QEA}$  和  $Metric_i$  分别表示 QEA 和对比算法  $i$  的指标值, 如目标函数值或计算时间.

### 3.2 对比实验

#### 3.2.1 小规模算例对比结果

表 4 为算法在小规模算例集上的测试结果. 1 小时计算时间内, GUROBI 仅在节点数量小于 22 的算例下获得了最优解, 且随着算例规模增加, GUROBI 的计算时间呈指数级增长; 对于节点数量大于 22 的算例, GUROBI 在 1 小时内只能获得可行解. 平均上看, QEA 的最小目标值与 GUROBI 仅相差 3%. 并且, QEA 获得了 4 个算例的最优解 (表中加粗表示). 相较于 GA、MA, QEA 在平均目标值上分别改善了 7.8%、2.4%. 而在计算时间上, 三种智能优化算法均能在 24 秒内获得满意解.

表4 小规模算例集测试结果

算例	GUROBI			GA		MA		QEA			偏差(Gap)				
	上界	下界	时间	平均值	时间	平均值	时间	最小值	平均值	时间	最小值 vs Gurobi	平均值 vs GA	时间 vs GA	平均值 vs MA	时间 vs MA
S11_3	<b>3201.2</b>	/	5.3	3331.8(*)	1.2	3328.6(*)	1.6	<b>3201.2</b>	3220.5	1.6	0.0%	-3.3%	33.3%	-3.2%	0.0%
S14_3	<b>4474.6</b>	/	5.7	4695.2(*)	1.1	4567.4	2.5	<b>4474.6</b>	4572.4	2.5	0.0%	-2.6%	127.3%	0.1%	0.0%
S16_3	<b>2916.6</b>	/	151.5	3128.8(*)	1.4	3098.4(*)	2.3	<b>2916.6</b>	3070.4	2.9	0.0%	-1.9%	107.1%	-0.9%	0.0%
S16_4	<b>6248.6</b>	/	163.3	6549.7(*)	2.6	6518.6(*)	2.9	<b>6248.6</b>	6486.1	2.9	0.0%	-1.0%	11.5%	-0.5%	0.0%
S22_3	2947.2	/	1342.0	3262.1(*)	2.9	3237.2(*)	3.1	2978.1	3169.6	3.1	1.0%	-2.8%	6.9%	-2.1%	0.0%
S41_3	10234.7	2094.7	3600.0	12144.4(*)	6.8	12048.1(*)	10.6	10629.6	10992.2	11.5	3.9%	-9.5%	69.1%	-8.8%	8.5%
S51_3	8953.9	1434.7	3600.0	9748.5(*)	11.6	9645.9(*)	12.3	9200.4	9571.5	13.1	2.8%	-1.8%	12.9%	-0.8%	6.5%
S75_3	15701.4	3685.9	3600.0	18691.3(*)	16.0	16519.4	18.7	16285.6	16549.9	19.1	3.7%	-11.5%	19.4%	0.2%	2.1%
S75_4	13623.4	3500.5	3600.0	18630.0(*)	18.4	16101.3(*)	22.1	14299.2	15552.1	20.3	5.0%	-16.5%	10.3%	-3.4%	-8.1%
S85_3	15015.8	1758.7	3600.0	16410.5(*)	21.2	16198.5(*)	22.7	15634.1	15899.5	23.4	4.1%	-3.1%	10.4%	-1.8%	3.1%
平均	8331.7	/	1966.8	9649.0	8.3	9136.6	9.9	8586.8	8908.4	10.0	<b>3.1%</b>	<b>-7.8%</b>	<b>20.7%</b>	<b>-2.4%</b>	<b>1.0%</b>

表5 中规模算例集测试结果

算例	GA			MA			QEA			偏差(Gap)					
	最小值	平均值	时间	最小值	平均值	时间	最小值	平均值	时间	最小值 vs GA	最小值 vs MA	平均值 vs GA	平均值 vs MA	时间 vs GA	时间 vs MA
M100_3	20086.3(*)	21062.3(*)	11.5	21718.0(*)	22627.0(*)	25.8	19934.3	20320.8	25.4	-0.8%	-8.2%	-3.5%	-10.2%	120.9%	-1.6%
M100_4	18685.7(*)	19694.3(*)	12.3	18546.1(*)	19158.1(*)	25.8	17359.6	17631.6	25.8	-7.1%	-6.4%	-10.5%	-8.0%	109.8%	0.0%
M105_2	25206.9(*)	26130.3(*)	13.6	26808.6(*)	27724.4(*)	35.3	23165.8	24526.3	30.6	-8.1%	-13.6%	-6.1%	-11.5%	125.0%	-13.3%
M105_3	29344.7(*)	29945.4(*)	15.1	30698.6(*)	31294.1(*)	36.4	27634.2	28192.2	30.4	-5.8%	-10.0%	-5.9%	-9.9%	101.3%	-16.5%
M120_2	30056.4(*)	31208.9(*)	17.7	30752.3(*)	31294.2(*)	53.6	27535.2	27972.3	52.8	-8.4%	-10.5%	-10.4%	-10.6%	198.3%	-1.5%
M120_3	27483.6(*)	28641.3(*)	20.8	27813.3(*)	28773.7(*)	54.3	25661.1	26376.7	51.4	-6.6%	-7.7%	-7.9%	-8.3%	147.1%	-5.3%
M130_2	24326.9(*)	26017.5(*)	24.0	23136.2(*)	24657.7(*)	67.5	21418.9	21998.8	59.9	-12.0%	-7.4%	-15.4%	-10.8%	149.6%	-11.3%
M130_3	24588.9(*)	25927.4(*)	24.5	23931.8(*)	24782.3(*)	66.3	22312.3	22638.9	58.9	-9.3%	-6.8%	-12.7%	-8.6%	140.4%	-11.2%
M150_2	29661.7(*)	31404.7(*)	36.7	31393.3(*)	32266.2(*)	87.8	27683.3	28258.4	79.6	-6.7%	-11.8%	-10.0%	-12.4%	116.9%	-9.3%
M150_3	32802.3(*)	33912.1(*)	31.5	31597.4(*)	32634.9(*)	88.8	27873.0	28586.7	76.3	-15.0%	-11.8%	-15.7%	-12.4%	142.2%	-14.1%
M160_2	54012.1(*)	56364.9(*)	42.4	56463.8(*)	57756.0(*)	98.8	50535.3	50946.1	88.3	-6.4%	-10.5%	-9.6%	-11.8%	108.3%	-10.6%
M160_3	62082.6(*)	63872.8(*)	41.2	65233.2(*)	66314.7(*)	90.3	58467.1	58997.0	86.7	-5.8%	-10.4%	-7.6%	-11.0%	110.4%	-4.0%
M180_2	46936.3(*)	48334.4(*)	50.4	45216.8(*)	47942.7(*)	117.9	41023.5	41523.9	114.9	-12.6%	-9.3%	-14.1%	-13.4%	128.0%	-2.5%
M180_3	33588.5(*)	34600.3(*)	53.3	33262.2(*)	34515.1(*)	121.7	30163.8	30705.0	119.6	-10.2%	-9.3%	-11.3%	-11.0%	124.4%	-1.7%
M200_2	47392.3(*)	49048.3(*)	54.5	48304.1(*)	49274.0(*)	136.5	40370.8	40902.0	124.4	-14.8%	-16.4%	-16.6%	-17.0%	128.3%	-8.9%
M200_3	42574.2(*)	44021.0(*)	59.9	41711.0(*)	42637.4(*)	138.2	37907.1	38668.6	125.5	-11.0%	-9.1%	-12.2%	-9.3%	109.5%	-9.2%
平均	34301.8	35636.6	31.8	34786.7	35853.3	77.8	31190.3	31765.3	71.9	<b>-9.1%</b>	<b>-10.3%</b>	<b>-10.9%</b>	<b>-11.4%</b>	<b>125.9%</b>	<b>-7.6%</b>

表6 大规模算例集测试结果

算例	GA			MA			QEA			偏差(Gap)					
	最小值	平均值	时间	最小值	平均值	时间	最小值	平均值	时间	最小值 vs GA	最小值 vs MA	平均值 vs GA	平均值 vs MA	时间 vs GA	时间 vs MA
L220_2	57217.7(*)	59190.2(*)	62.9	54171.4(*)	55663.1(*)	110.4	44799.4	46427.9	100.1	-21.7%	-17.3%	-21.6%	-16.6%	59.1%	-9.3%
L220_3	57738.7(*)	59431.2(*)	79.4	51210.3(*)	54057.3(*)	116.3	45821.4	46425.9	105.7	-20.6%	-10.5%	-21.9%	-14.1%	33.1%	-9.1%
L235_2	84568.1(*)	87167.4(*)	100.4	83121.7(*)	85212.3(*)	120.1	77808.0	78532.1	110.4	-8.0%	-6.4%	-9.9%	-7.8%	10.0%	-8.1%
L235_3	72710.8(*)	74421.4(*)	76.1	67667.9(*)	69826.5(*)	110.1	55983.9	56590.6	101.3	-23.0%	-17.3%	-24.0%	-19.0%	33.1%	-8.0%
L285_2	101141.9(*)	104980.8(*)	123.7	101037.0(*)	102883.7(*)	142.4	91418.8	92266.4	127.5	-9.6%	-9.5%	-12.1%	-10.3%	3.1%	-10.5%
L285_3	103082.6(*)	104999.3(*)	128.9	102120.7(*)	104496.1(*)	150.3	92891.0	93466.2	136.3	-9.9%	-9.0%	-11.0%	-10.6%	5.7%	-9.3%
L320_2	108375.6(*)	110471.7(*)	161.6	103662.5(*)	105328.1(*)	200.2	93298.1	93805.3	172.4	-13.9%	-10.0%	-15.1%	-10.9%	6.7%	-13.9%
L320_3	102430.0(*)	104806.7(*)	159.7	97131.3(*)	99750.3(*)	209.6	87772.1	88222.0	180.3	-14.3%	-9.6%	-15.8%	-11.6%	12.9%	-14.0%
L435_2	142903.6(*)	145317.3(*)	341.6	135750.5(*)	137395.1(*)	391.6	119611.1	120474.4	361.7	-16.3%	-11.9%	-17.1%	-12.3%	5.9%	-7.6%
L435_3	139840.8(*)	142584.8(*)	330.9	134396.9(*)	136648.4(*)	400.8	117306.6	118694.9	371.7	-16.1%	-12.7%	-16.8%	-13.1%	12.3%	-7.3%
L530_2	159756.5(*)	163085.0(*)	522.5	144792.5(*)	146590.1(*)	622.5	129833.3	131045.5	571.8	-18.7%	-10.3%	-19.6%	-10.6%	9.4%	-8.1%
平均	102706.0	105132.3	189.8	97733.0	99804.6	234.0	86958.5	87813.7	212.7	<b>-15.3%</b>	<b>-11.0%</b>	<b>-16.5%</b>	<b>-12.0%</b>	<b>12.0%</b>	<b>-9.1%</b>

图5为GA、MA和QEA在小规模算例上平均目标函数值的对比结果,可见QEA的平均求解质量优于GA、MA,且随着问题规模扩大,QEA的优势更加明显.此外,根据表4中的非参数检验实验结果,QEA在所有小规模算例上的表现均显著优于GA和MA.因此,以上实验结果验证了本文所提QEA在求解小规模算例上的有效性.

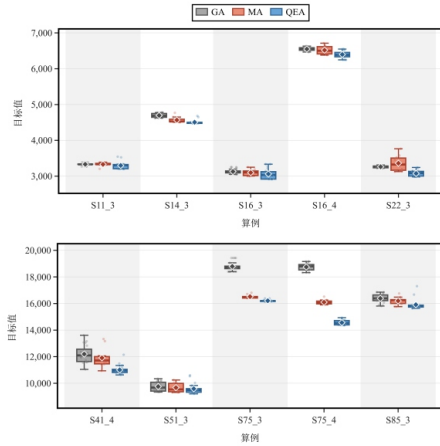


图5 小规模算例测试结果箱线图

### 3.2.2 中规模算例对比结果

表5为3类智能优化算法在中规模算例集的测试结果,由于GUROBI无法在可接受时间内获得可行解,因此其结果未展示于表中.相较于GA、MA,QEA在目标函数最小值上分别改善了9.1%、10.3%,在目标函数平均值上分别改善了10.9%、11.4%,并且在所有中规模算例上的表现均显著优于前两者.在相同最大迭代次数下,GA、MA、QEA在该测试集上的平均运行时间为31.8秒、77.8秒、71.9秒,QEA比MA降低了7.6%.

图6为GA、MA和QEA在中规模算例上平均目标函数值的对比结果,可见QEA的平均求解质量仍优于GA和MA,并且在大部分算例上具有更好的稳定性.上述实验结果验证了本文所提QEA在求解中规模算例上的有效性.

### 3.2.3 大规模算例对比结果

表6为大规模算例集的测试结果,相较于GA、MA,QEA在目标函数最小值上分别改善了15.3%、11.0%,在目标函数平均值上分别改善了16.5%、12.0%,并且在所有大规模算例上的表现均显著优于前两者.在相同最大迭代次数下,GA、MA与QEA的平均运行时间分别为189.8秒、234.0秒、212.7秒,QEA较MA降低了9.1%.图7展示了GA、MA和QEA在大规模算例上的对比结果,可见QEA在节点数量较多的场景中依然展现了较好的性能.

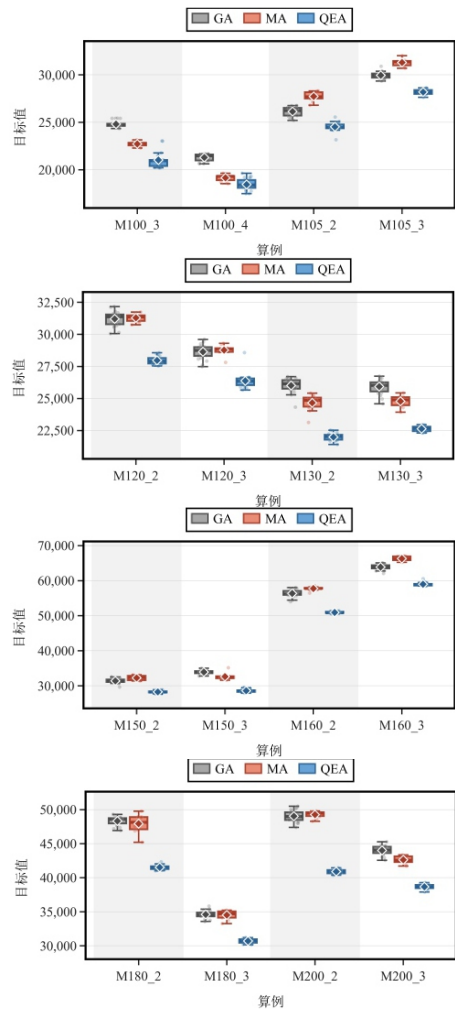


图6 中规模算例测试结果箱线图

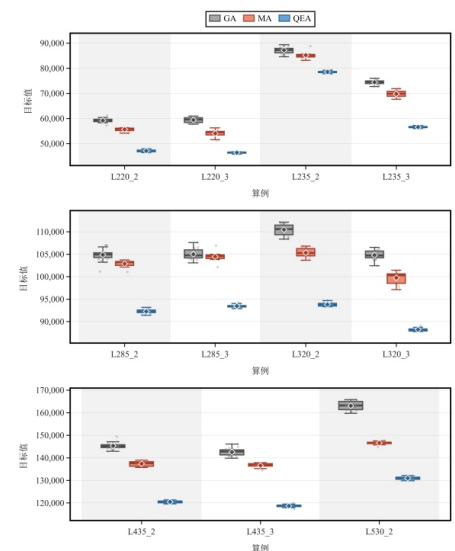


图7 大规模算例测试结果箱线图

综合三类规模的算例测试结果可以发现,GA、MA、QEA三类群智能优化算法在求解节点数量超过100的算例时,性能优于求解器,而MA由于具有局部搜索机制且交叉概率低于GA,其综合性能优于GA,而QEA在主要参数设置与MA相同的情况

下,引入Q学习机制来引导变异算子的选用,使其综合性能上略优于MA.

### 3.3 消融实验

为进一步测试状态引导机制对Q学习驱动进化算法求解效果的影响,开展如下消融实验:移除S1的状态引导机制,改为计算进化过程前后两代种群个体适应度方差的变化值,若方差增加,S1取"+",否则取 "-",同时保持算法的其它设置与参数不变,将以上算法记为QEA1,并在小、中、大三种规模算例集上进行测试.

表7、8、9中的结果表明:在小、中、大三种规模的算例集上,QEA的目标函数最小值较QEA1分别改善2.6%、5.1%、5.6%,目标函数平均值分别改善1.7%、5.6%、7.7%.此外,根据非参数检验实验结果,在大部分小规模算例以及所有中大规模算例上,QEA的表现显著优于QEA1,验证了引导机制的有效性,尤其是在应对中大规模问题上.可见,QEA利用单调下降的种群方差参照函数作为基准,引导智能体依据算法收敛进程合理寻优,从而能更有效发挥强化学习的探索能力,且随着问题规模增大,引导机制带来的性能提升更为显著.

表7 小规模算例集消融实验结果

算例	QEA1		QEA		偏差(Gap)	
	最小值	平均值	最小值	平均值	最小值	平均值
S11_3	3201.2	3410.3(*)	3201.2	3220.5	0.0%	-5.6%
S14_3	4474.6	4528.6	4474.6	4572.4	0.0%	1.0%
S16_3	2927.4(*)	3095.8(*)	2916.6	3070.4	-0.4%	-0.8%
S16_4	6435.9(*)	6500.7(*)	6248.6	6486.1	-2.9%	-0.2%
S22_3	3102.6(*)	3185.5(*)	2978.1	3169.6	-4.0%	-0.5%
S41_3	11073.8(*)	11945.1(*)	10629.6	10992.2	-4.0%	-8.0%
S51_3	9418.9(*)	9600.9(*)	9200.4	9571.5	-2.3%	-0.3%
S75_3	16464.5(*)	16582.9(*)	16285.6	16549.9	-1.1%	-0.2%
S75_4	15124.4(*)	15667.4(*)	14299.2	15552.1	-5.5%	-0.7%
S85_3	15924.4(*)	16067.4(*)	15634.1	15899.5	-1.8%	-1.0%
平均	8814.77	9058.46	8586.8	8908.4	-2.6%	-1.7%

为进一步分析状态引导机制对算子选择的影响,记录算法迭代过程中各变异算子在不同状态下的选择比例.探究实验以算例M105\_2为例,结果如图8所示,图中A1-A9依次表示各变异算子.可见,在算法迭代的早、中、晚期,不同状态下各算子的选择比例动态变化,说明算法能够依据不同状态下算子改进效果及种群适应度方差,动态选择合适算子.

具体而言,在(S1+,S2-)组态,A1、A5、A7、A8选择概率较高,其在改进个体质量方面效果显著,而A2、A4在(S1-,S2+)组态的选择概率较高,其对增强种群多样性有一定效果,A3、A9、A6在(S1+,

表8 中规模算例集消融实验结果

算例	QEA1		QEA		偏差(Gap)	
	最小值	平均值	最小值	平均值	最小值	平均值
M100_3	21285.8(*)	21772.4(*)	19934.3	20320.8	-6.3%	-6.7%
M100_4	18395.9(*)	18824.8(*)	17359.6	17631.6	-5.6%	-6.3%
M105_2	26996.3(*)	27472.7(*)	23165.8	24526.3	-14.2%	-10.7%
M105_3	29696.4(*)	30389.3(*)	27634.2	28192.2	-6.9%	-7.2%
M120_2	29382.5(*)	30255.1(*)	27535.2	27972.3	-6.3%	-7.5%
M120_3	26393.7(*)	27273.3(*)	25661.1	26376.7	-2.8%	-3.3%
M130_2	22286.2(*)	23305.9(*)	21418.9	21998.8	-3.9%	-5.6%
M130_3	22936.2(*)	23832.1(*)	22312.3	22638.9	-2.7%	-5.0%
M150_2	29249.2(*)	29964.6(*)	27683.3	28258.4	-5.4%	-5.7%
M150_3	28283.1(*)	29309.9(*)	27873.0	28586.7	-1.4%	-2.5%
M160_2	52889.6(*)	53123.9(*)	50535.3	50946.1	-4.5%	-4.1%
M160_3	60168.3(*)	61155.5(*)	58467.1	58997.0	-2.8%	-3.5%
M180_2	42621.8(*)	43687.3(*)	41023.5	41523.9	-3.7%	-5.0%
M180_3	31980.4(*)	32928.9(*)	30163.8	30705.0	-5.7%	-6.8%
M200_2	43562.1(*)	44317.3(*)	40370.8	40902.0	-7.3%	-7.7%
M200_3	39917.9(*)	40784.8(*)	37907.1	38668.6	-5.0%	-5.2%
平均	32877.8	33649.9	31190.3	31765.3	-5.1%	-5.6%

表9 大规模算例集消融实验结果

算例	QEA1		QEA		偏差(Gap)	
	最小值	平均值	最小值	平均值	最小值	平均值
L220_2	49837.2(*)	50590.2(*)	44799.4	46427.9	-10.1%	-11.4%
L220_3	48609.2(*)	51314.6(*)	45821.4	46425.9	-5.7%	-9.5%
L235_2	81259.4(*)	83411.6(*)	77808.0	78532.1	-4.2%	-5.8%
L235_3	60184.9(*)	61497.2(*)	55983.9	56590.6	-7.0%	-8.0%
L285_2	96489.2(*)	98669.9(*)	91418.8	92266.4	-5.3%	-6.5%
L285_3	97843.5(*)	102319.9(*)	92891.0	93466.2	-5.1%	-8.7%
L320_2	97874.3(*)	99307.8(*)	93298.1	93805.3	-4.7%	-5.5%
L320_3	93410.3(*)	96435.2(*)	87772.1	88222.0	-6.0%	-8.5%
L435_2	122690.5(*)	129362.8(*)	119611.1	120474.4	-2.5%	-6.9%
L435_3	126899.4(*)	130319.4(*)	117306.6	118694.9	-7.6%	-8.9%
L530_2	138498.4(*)	141290.2(*)	129833.3	131045.5	-6.3%	-7.3%
平均	92145.1	94956.3	86958.5	87665.7	-5.6%	-7.7%

S2+)组态的选择概率较高,表明其兼具改进个体质量与种群多样性的能力.总体上看,虽然各算子选择比例呈现差异性,但未出现极端选择比例,说明各算子在迭代过程均能发挥作用,不存在冗余算子,因此该实验也验证了变异算子设计的有效性.

## 4 结论

在末端配送部署大量包裹储物柜场景下,物流配送需同时优化配送车辆路径与PL选择.相较于即时配送场景下客户满意度对时间窗口敏感,PL的取件便利性是该场景下影响客户满意度的关键因素.本文基于该现实需求提出了一类新的覆盖交货问题,即考虑客户满意度的多车辆覆盖配送问题,构建了同时优化配送线路与PL选择的整数规划模型,并提出一种强化学习驱动的进化算法求解该问题.

本文的核心创新点如下:首先,基于问卷调查构

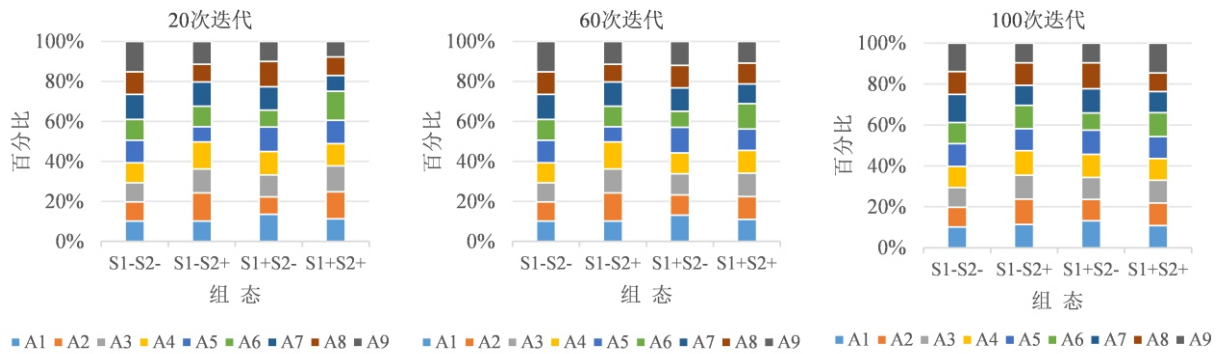


图8 变异算子选择比例变化

建了考虑客户个体差异的满意度函数;其次,设计了考虑种群多样性变化和最优解改进变化的状态表征方法,使智能体全面感知种群状态;再次,设计了随进化代数单调下降的种群方差基准函数,引导智能体学习算法的收敛过程;最后,基于问题结构设计了9种邻域算子,将其作为强化学习动作集,并依据算子的即时效用与使用频率动态计算奖励,实现了邻域算子的自适应选择.基于大量算例的实验结果表明,本文所提强化学习驱动的进化算法在解决小、中、大规模问题时,相较于GUROBI可以在更短的时间内获得较好的结果,相较于GA、MA可在所有算例上取得更好的性能.

未来研究可以将状态引导强化学习驱动的进化算法应用于其他复杂组合优化问题,例如带时间窗或者不确定性的多车覆盖配送问题,以验证算法在广泛场景下的适用性,促进物流配送行业高质量发展.

#### 参考文献 (References)

- [1] Atkins R, Gianiodis P. An investigation at the intersection of the sharing economy and supply chain management: A strategic perspective[J]. *International Journal of Logistics Research and Applications*, 2022, 25(11): 1425-1443.
- [2] Pahwa A, Jaller M. A cost-based comparative analysis of different last-mile strategies for e-commerce delivery[J]. *Transportation Research — Part E: Logistics and Transportation Review*, 2022, 164: 102783.
- [3] Che Z H, Chiang T A, Luo Y J. Multiobjective optimization for planning the service areas of smart parcel locker facilities in logistics last mile delivery[J]. *Mathematics*, 2022, 10(3): 422.
- [4] Seghezzi A, Siragusa C, Mangiaracina R. Parcel lockers vs. home delivery: A model to compare last-mile delivery cost in urban and rural areas[J]. *International Journal of Physical Distribution & Logistics Management*, 2022, 52(3): 213-237.
- [5] Yu Y Q, Lian F, Yang Z Z. Pricing of parcel locker service in urban logistics by a TSP model of last-mile delivery[J]. *Transport Policy*, 2021, 114: 206-214.
- [6] 周林, 康燕, 宋寒, 等. 送提一体与终端共享下的最后一公里配送选址 — 路径问题[J]. *计算机集成制造系统*, 2019, 25(7): 1855-1864.
- [7] Schnieder M, Hinde C, West A. Land efficient mobility: Evaluation of autonomous last mile delivery concepts in London[J]. *International Journal of Environmental Research and Public Health*, 2022, 19(16): 10290.
- [8] Janjevic M, Winkenbach M, Merchán D. Integrating collection-and-delivery points in the strategic design of urban last-mile e-commerce distribution networks[J]. *Transportation Research — Part E: Logistics and Transportation Review*, 2019, 131: 37-67.
- [9] Jiang L, Zang X N, Alghoul I I Y, et al. Scheduling the covering delivery problem in last mile delivery[J]. *Expert Systems with Applications*, 2022, 187: 115894.
- [10] Current J R, Schilling D A. The covering salesman problem[J]. *Transportation Science*, 1989, 23(3): 208-213.
- [11] Maziero L P, Usberti F L, Cavellucci C. Branch-and-cut algorithms for the covering salesman problem[J]. *RAIRO — Operations Research*, 2023, 57(3): 1149-1166.
- [12] Salari M, Naji-Azimi Z. An integer programming-based local search for the covering salesman problem[J]. *Computers & Operations Research*, 2012, 39(11): 2594-2602.
- [13] Salari M, Reihaneh M, Sabbagh M S. Combining ant colony optimization algorithm and dynamic programming technique for solving the covering salesman problem[J]. *Computers & Industrial Engineering*, 2015, 83: 244-251.
- [14] Gendreau M, Laporte G, Semet F. The covering tour problem[J]. *Operations Research*, 1997, 45(4): 568-576.
- [15] Sharda R, Voß S, Rego C, et al. Metaheuristic optimization via memory and evolution[M]. Boston: Kluwer Academic Publishers, 2005.
- [16] Hachicha M, Hodgson M J, Laporte G, et al. Heuristics for the multi-vehicle covering tour problem[J]. *Computers & Operations Research*, 2000, 27(1): 29-42.
- [17] Glize E, Roberti R, Jozefowicz N, et al. Exact methods for mono-objective and bi-objective multi-vehicle covering tour problems[J]. *European Journal of*

- Operational Research*, 2020, 283(3): 812-824.
- [18] Torabi P, Oleynik A, Hemmati A, et al. Covering tour problem with varying coverage: Application to marine environmental monitoring[J]. *Applied Mathematical Modelling*, 2023, 124: 279-299.
- [19] Torabi P, Hemmati A, Oleynik A, et al. A deep reinforcement learning hyperheuristic for the covering tour problem with varying coverage[J]. *Computers & Operations Research*, 2025, 174: 106881.
- [20] Li J, Konuş U, Langerak F, et al. Customer channel migration and firm choice: The effects of cross-channel competition[J]. *International Journal of Electronic Commerce*, 2017, 21(1): 8-42.
- [21] Vukićević M, Ratli M, Rivenq A, et al. Covering delivery problem with electric vehicle and parcel lockers: Variable neighborhood search approach[J]. *Computers & Operations Research*, 2023, 157: 106263.
- [22] Tao X M, Wang Y W, Sun Y Q, et al. Heterogeneous teaching-learning based optimization with local search for the covering delivering problem in last mile delivery[J]. *Expert Systems with Applications*, 2024, 252: 124176.
- [23] 王利娟, 崔利刚, 徐东洋. 基于软时间窗的多商品供需未匹配两阶段车辆路径问题研究[J]. *管理学报*, 2026, 23(3): 573-582.  
(Wang L J, Cui L G, Xu D Y. Research of multi-commodity two-echelon vehicle routing problem with unpaired supply-demand and soft time windows[J]. *Chinese Journal of Management*, 2026, 23(3): 573-582.)
- [24] 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.)
- [25] Rajak S, Parthiban P, Dhanalakshmi R. Multi-depot vehicle routing problem based on customer satisfaction[J]. *International Journal of Services Technology and Management*, 2020, 26(2/3): 252.
- [26] Ji J J, Guo Y N, Gao X Z, et al. Q-learning-based hyperheuristic evolutionary algorithm for dynamic task allocation of crowdsensing[J]. *IEEE Transactions on Cybernetics*, 2023, 53(4): 2211-2224.
- [27] 张广辉, 魏晨轩, 冯彦翔, 等. 基于分布式元强化学习的多敏捷卫星多目标调度算法[J]. *控制与决策*, 2026, 41(4): 1065-1076.  
(Zhang G H, Wei C X, Feng Y X, et al. Multiple agile satellites multi-objective scheduling algorithm based on distributed meta-reinforcement learning[J]. *Control and Decision*, 2026, 41(4): 1065-1076.)
- [28] 郭恒伟, 桑红燕, 潘全科. 学习驱动的迭代局部搜索算法求解分布式流水车间鲁棒调度问题[J]. *控制与决策*, 2026, 41(4): 977-986.  
(Guo H W, Sang H Y, Pan Q K. A learning-driven iterated local search algorithm for solving distributed flowshop robust scheduling problems[J]. *Control and Decision*, 2026, 41(4): 977-986.)
- [29] 周梦, 王境琦, 吴楚格, 等. 带有二维装箱约束车辆路径问题的知识驱动强化学习求解[J]. *控制与决策*, 2026, 41(4): 931-943.  
(Zhou M, Wang J Q, Wu C G, et al. Knowledge-driven reinforcement learning method for solving capacitated vehicle routing problem with two-dimensional loading constraints[J]. *Control and Decision*, 2026, 41(4): 931-943.)
- [30] Qin W, Zhuang Z L, Huang Z Z, et al. A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem[J]. *Computers & Industrial Engineering*, 2021, 156: 107252.
- [31] 赵仕存, 周泓. 强化学习驱动进化的模因算法求解准时制分布式柔性作业车间调度问题[J]. *控制与决策*, 2026, 41(4): 905-918.  
(Zhao S C, Zhou H. Reinforcement learning-driven evolutionary memetic algorithm for solving just-in-time distributed flexible job shop scheduling problem[J]. *Control and Decision*, 2026, 41(4): 905-918.)
- [32] Tang H T, Xiao Y, Zhang W, et al. A DQL-NSGA-III algorithm for solving the flexible job shop dynamic scheduling problem[J]. *Expert Systems with Applications*, 2024, 237: 121723.

## 作者简介

潘立军 (1977-), 男, 教授, 博士, 主要研究方向为物流工程, E-mail: [pansoftware@126.com](mailto:pansoftware@126.com);

张伊帆 (2000-), 女, 硕士, 主要研究方向为物流工程, E-mail: [576613895@qq.com](mailto:576613895@qq.com);

陈靖方 (1995-), 男, 研究员, 博士, 主要研究方向为智能优化调度, E-mail: [chenjingfang@bjut.edu.cn](mailto:chenjingfang@bjut.edu.cn);

刘喜梅 (1978-), 女, 副教授, 硕士, 主要研究方向为企业管管理, E-mail: [liuxm@163.com](mailto:liuxm@163.com);

王兴杰 (2001-), 男, 硕士, 主要研究方向为物流工程, E-mail: [1581870084@qq.com](mailto:1581870084@qq.com);

刘长石 (1975-), 男, 教授, 博士, 主要研究方向为物流管理与多目标决策, E-mail: [liuchangshi964@126.com](mailto:liuchangshi964@126.com).