优先配送绿色VRP的混合启发式求解算法

崔焕焕，官礼和*
(重庆交通大学 数学与统计学院，重庆 400074)

摘要：考虑物流配送中部分客户货物存在不可混装的问题，在传统同时取送绿色车辆路径问题基础上，根据客户需求将客户划分为优先配送、非优先只取不送、非优先有取有送3种类型，并建立最小化总成本的单目标非线性优化模型。基于模拟退火和自适应大邻域搜索算法设计了一种混合启发式求解方法，采用改进的节约算法构造初始解，在模拟退火算法框架中利用5种破坏算子和2种修复算子进行自适应大邻域搜索，直至稳定。仿真结果表明：所提算法能有效降低总成本和减少车辆数，且具有较快的收敛速度和较强的稳定性。

关键词：绿色车辆路径问题；同时取送货；节约算法；模拟退火；自适应大邻域搜索
中图分类号：TP391.9 文献标志码：A 文章编号：1004-731X(2025)02-0413-11
DOI: 10.16182/j.issn1004731x.joss.23-1125
引用格式: 崔焕焕, 官礼和. 优先配送绿色VRP的混合启发式求解算法[J]. 系统仿真学报, 2025, 37(2): 413-423.
Reference format: Cui Huanhuan, Guan Lihe. A Hybrid Heuristic Algorithm for Solving the Green VRP with Priority Delivery[J]. Journal of System Simulation, 2025, 37(2): 413-423.

A Hybrid Heuristic Algorithm for Solving the Green VRP with Priority Delivery

Cui Huanhuan, *Guan Lihe**
(School of Mathematics and Statistics, Chongqing Jiaotong University, Chongqing 400074, China)

Abstract: This paper considers the problem that some customers' goods cannot be mixed in logistics distribution. *Based on the traditional green vehicle routing problem with simultaneous pickup and delivery, customers are divided into three types: priority delivery, non-priority only pickup without delivery, and non-priority pickup with delivery. A single objective nonlinear optimization model is established to minimize the total cost.* A hybrid heuristic method based on simulated annealing and adaptive large neighborhood search algorithm is designed. *An improved saving algorithm is used to construct the initial solution. And 5 kinds of destruction operators and 2 kinds of repair operators are used in the simulated annealing algorithm framework for adaptive large neighborhood search, until stability is achieved*. The simulation results indicate that the proposed method can effectively reduce total costs and the number of vehicles, and has fast convergence speed and strong stability.

Keywords: green vehicle routing problem; simultaneous pickup and delivery; savings algorithm; simulated annealing; adaptive large neighborhood search

收稿日期：2023-09-12 修回日期：2023-10-22
基金项目：国家自然科学基金(12271067)；重庆市高校创新研究群体项目(CXQT21021)；重庆市研究生联合培养基地建设项目(JDLHPYJD2021016)；2023年重庆市研究生教育“课程思政”示范项目(YKCSZ23136)
第一作者：崔焕焕(1997-)，女，硕士生，研究方向为智能优化计算。
通讯作者：官礼和(1975-)，男，副教授，博士，研究方向为智能信息处理、机器学习。

0 引言

随着物联网技术的崛起，物流行业得到快速发展。在倡导绿色发展的背景下，推进物流配送领域的节能减排已刻不容缓[1]。文献[2]提出绿色车辆路径问题(green vehicle routing problem, GVRP)以来，得到了国内外学者的广泛关注。GVRP在满足客户要求、车载容量等约束条件下，以降低运营成本、减少油耗和碳排放量、提高客户满意度等为目标，科学规划发车数量、发车时间和车辆行驶路线，实现环境、经济和社会等多方效益的协调优化。由于油耗和碳排放受车辆载重、车速、车辆特征参数、道路特点和交通拥堵状况等多种确定和不确定因素的影响，导致GVRP的建模与求解要比传统的车辆路径问题(VRP)复杂得多。因此，GVRP的建模与求解是一个极具挑战性的工作，具有重要的理论和现实意义。

目前，对GVRP的优化目标主要有最小化车辆总行驶距离、最小化车辆数、最小化油耗或碳排放量，以及最小化经济和环境的综合成本等。求解算法主要分为精确算法[3]和启发式算法，其中，精确算法仅适用于小规模问题求解。因此，针对较大规模的GVRP如何设计高效的启发式算法求解高质量的近似解是重要的研究内容。已有的启发式求解算法主要基于智能优化算法进行设计，如蚁群算法[4]、粒子群优化算法[5]、基于禁忌搜索的混合模拟退火算法[6]和自适应大邻域搜索算法[7]等。此外，随着研究的深入，依据不同的实际需求，GVRP已出现了多车场GVRP[4]、多车型GVRP[8]、时间依赖GVRP[9]、开放式GVRP[10]等多种变体，同时取送的GVRP(GVRP with simultaneous pickup and delivery, GVRP-SPD)[11]就是其中一个重要变体。Majidi等[11]针对取货和送货需求均为不确定情况下带时间窗的GVRP-SPD建立了数学模型，应用可信度测度理论来处理取货和送货需求的不确定性，提出了一种启发式的求解算法。Olgun等[12]针对GVRP-SPD，建立了以总燃油消耗成本最小为目标的数学模型，并基于迭代局部搜索和可变邻域下降启发式提出了一种超启发式迭代局部搜索求解算法(hyper heuristic-iterative local search, HH-ILS)，实验验证了目标函数的合理性。

在物品配送过程中，时常会因为货物体积形状、车辆装载箱限制、货物间不可接触和客户特殊要求等原因，导致客户货物存在不可混装的问题，这给物流配送带来了困难。Polat等[13]对生牛奶收集过程中不同品质原料牛奶不可混装的问题建立了最小化牛奶储罐量、车辆总行驶距离和总网络成本的混合整数线性规划模型，提出了一种可变邻域搜索元启发式的求解方法。Wang等[14]对带时间窗和不相容负载约束的异构多车型VRP建立了最小化总成本为目标的数学模型，提出了基于破坏和再造算子的启发式求解方法和一种元启发式禁忌搜索算法。本文考虑一种带优先配送要求的GVRP-SPD，即不仅考虑同时取送和车载容量等约束，而且依据客户需求分为优先配送客户、非优先只取不送客户和非优先有取有送客户。对该问题建立了最小化总成本的单目标非线性优化模型，基于SA[15]和ALNS[7]设计了一种混合启发式求解算法SA-ALNS，并在Solomon和CMTX算例上验证了算法的有效性。

1 数学模型

在具有优先配送的GVRP-SPD中，根据客户需求划分为优先配送客户、非优先只取不送客户和非优先有取有送客户3种类型。所谓优先配送客户是指客户配送货物由于体积形状、车辆装载箱限制、货物间不可接触和客户特殊要求等原因，不可与其他物品混装，因此需要优先配送的客户。带优先配送的GVRP-SPD可描述为配送中心安排一定车辆为客户进行取送货服务，每个客户取送货量及位置均已知，且存在一定量特殊物品需要优先配送，车辆在完成配送任务后返回配送中心，在满足客户取送货需求及车辆载重量限制的条件下合理规划配送路线，使总成本最小。表1为符号说明。

表1　符号说明
Table 1　Explanation of notations

符号	含义
N_1	需要优先配送的客户集合
N_2	非优先只取不送的客户集合
N_3	非优先有取有送的客户集合
N	客户集合，包括配送中心0，且客户数为n
K_1	用于优先配送客户的车辆集合
K_2	用于非优先配送客户的车辆集合
K	配送车辆集合，即$K=K_1\cup K_2$
r_k	第$k\in K$辆车的行驶路线
$d_{i,j}$	客户$i\in N$和$j\in N$之间的距离
Q	车辆最大载重量
q_i	客户$i\in N/\{0\}$的配送需求量
p_i	客户$i\in N/\{0\}$的取货需求量
$u_{i,j}^k$	车辆$k\in K$从客户$i\in N$到客户$j\in N(i\neq j)$时车载剩余送货量
$v_{i,j}^k$	车辆$k\in K$从客户$i\in N$到客户$j\in N(i\neq j)$时车载已取货量
$x_{i,j}^k$	车辆$k\in K$从客户$i\in N$到客户$j\in N(i\neq j)$时$x_{i,j}^k=1$，反之为0
Q_{D}	配送中心需配送货物总量，且$Q_{\mathrm{D}}=\sum_{i\in N/\{0\}}q_i$
Q_{P}	配送中心需取回货物总量，且$Q_{\mathrm{P}}=\sum_{i\in N/\{0\}}p_i$
$\rho_{i,j}^k$	车辆$k\in K$从客户$i\in N$到客户$j\in N(i\neq j)$的油耗率
$\rho_{\min}$	车辆空载时的油耗率
$\rho_{\max}$	车辆满载时的油耗率
c_{p}	司机薪资的每公里单价
c_{f}	燃油成本单价
c_{v}	车辆成本单价

目标函数：

$$\min Z=c_{\mathrm{p}}\sum_{k\in K}\sum_{i,j\in N}d_{i,j}x_{i,j}^k+c_{\mathrm{f}}\sum_{k\in K}\sum_{i,j\in N}\rho_{i,j}^k d_{i,j}x_{i,j}^k+c_{\mathrm{v}}\sum_{k\in K}\sum_{j\in N/\{0\}}x_{0,j}^k \tag{1}$$

约束条件：

$$\rho_{i,j}^k=\rho_{\min}+(u_{i,j}^k+v_{i,j}^k)/Q(\rho_{\max}-\rho_{\min}),\ \forall i,j\in N,\forall k\in K \tag{2}$$

$$\sum_{k\in K}\sum_{j\in N/\{i\}}x_{i,j}^k=1,\forall i\in N/\{0\} \tag{3}$$

$$\sum_{k\in K}\sum_{j\in N/\{i\}}x_{j,i}^k=1,\forall i\in N/\{0\} \tag{4}$$

$$\sum_{i\in N/\{0\}}x_{0,i}^k=\sum_{i\in N/\{0\}}x_{i,0}^k=1,\forall k\in K \tag{5}$$

$$\sum_{k\in K}\sum_{j\in N/\{0\}}u_{0,j}^k=Q_{\mathrm{D}} \tag{6}$$

$$\sum_{i\in N/\{0\}}u_{i,0}^k=0,\forall k\in K \tag{7}$$

$$\sum_{k\in K}\sum_{i\in N/\{0\}}v_{i,0}^k=Q_{\mathrm{P}} \tag{8}$$

$$\sum_{j\in N/\{0\}}v_{0,j}^k=0,\forall k\in K \tag{9}$$

$$u_{i,j}^k\leqslant\sum_{h\in N/\{0\}}(q_h\cdot\sum_{l\in N}x_{h,l}^k),\forall k\in K,\forall i,j\in N \tag{10}$$

$$v_{i,j}^k\leqslant\sum_{h\in N/\{0\}}(p_h\cdot\sum_{l\in N}x_{h,l}^k),\forall k\in K,\forall i,j\in N \tag{11}$$

$$u_{i,j}^k\geqslant q_j x_{i,j}^k,\ \forall k\in K,\forall i\in N,\forall j\in N/\{0\} \tag{12}$$

$$v_{i,j}^k\geqslant p_i x_{i,j}^k,\ \forall k\in K,\forall i\in N/\{0\},\forall j\in N \tag{13}$$

$$\sum_{k\in K}\left(\sum_{l\in N}u_{l,i}^k\cdot x_{l,i}^k-\sum_{j\in N}u_{i,j}^k\cdot x_{i,j}^k\right)=q_i,\forall i\in N/\{0\} \tag{14}$$

$$\sum_{k\in K}\left(\sum_{j\in N}v_{i,j}^k\cdot x_{i,j}^k-\sum_{l\in N}v_{l,i}^k\cdot x_{l,i}^k\right)=p_i,\forall i\in N/\{0\} \tag{15}$$

$$(u_{i,j}^k+v_{i,j}^k)x_{i,j}^k\leqslant Q\cdot x_{i,j}^k,\forall k\in K,\forall i,j\in N \tag{16}$$

$$x_{0,i}^k\geqslant x_{i,j}^k,\forall k\in K,\forall i\in N_1,\forall j\in N_2 \tag{17}$$

$$x_{i,j}^k\in\{0,1\},\forall k\in K,\forall i,j\in N \tag{18}$$

式中：$x_{i,j}^k$为决策变量；$\rho_{i,j}^k$、$u_{i,j}^k$、$v_{i,j}^k$为中间变量；c_{p}、c_{f}、c_{v}、$d_{i,j}$、Q_{D}、Q_{P}、Q、q_i、p_i为常参数；K、N、N_1、N_2、N_3为常集合。

上述模型中，Z表示总成本，包括司机薪资、油耗成本、车辆使用成本，其中，司机薪资为单价c_{p}乘总行驶距离，车辆油耗成本为单价c_{f}乘总油耗，车辆使用成本为固定成本c_{v}乘车辆数。式(2)表示依据车辆$k\in K$从客户i到客户j的装载量按比例计算其燃料消耗率，车辆满载时燃料消耗率达到最大$\rho_{\max}$，车辆空载时燃料消耗率达到最小$\rho_{\min}$；式(3)和(4)表示每个客户有且仅有一辆车对其服务一次；式(5)表示每辆车均从配送中心出发最后回到配送中心；式(6)表示所有车辆离开配送中心时车载送货量之和为总送货量；式(7)和(8)分别表示每辆车返回仓库时已完成其送货任务和取货任务；式(9)表示所有车辆离开配送中心时车载取货量均为0；式(10)~(13)确定辅助决策变量的上界和下界；式(14)和(15)分别表示车辆经过客户$i\in N/\{0\}$前后的送、取货量的平衡关系；式(16)表

示车辆在行驶途中不超载；式(17)表示客户配送的优先性约束；式(18)表示决策变量的取值范围。

2 算法设计

ALNS 算法对给定解进行破坏和修复得到其邻域解，自适应机制能增加找到高质量解的概率。而 SA 算法在搜索过程中概率性地接受劣质解，能够有效避免算法陷入局部最优，增强算法全局寻优能力。针对单目标非线性优化模型，基于 SA 和 ALNS 设计一种混合启发式求解算法(SA-ALNS)，算法过程中所涉及符号说明如表 2 所示。

表 2 算法过程中符号说明
Table 2 Explanation of notations in algorithm

符号	含义
R_1	优先配送客户路径集合
R_2	非优先只取不送客户路径集合
R_3	非优先有取有送客户路径集合
w	$w=\|r_k\|$为路线r_k服务的客户数
L	记路线r_k的客户k_l与客户k_{l+1}之间的位置为第L个候选插入位置，$0\leqslant L=l\leqslant w$
$\mathcal{L}(k,j)$	路线r_j的客户插入路线r_k的可行候选插入位置集合
H	破坏算子集合$H=\{H_1,H_2,H_3,H_4,H_5\}$
I	修复算子集合$I=\{I_1,I_2\}$
ℓ	删除列表
s	删除列表长度
T_0	初始温度
T_{end}	终止温度
m	降温次数
T_m	第m次降温后的温度
$P_h^{T_m}$	算子$h\in H\cup I$在温度T_m时被选择的概率
η	轮盘赌参数
t	每个温度下的迭代次数
M	最大迭代次数
R_{best}	当前最优解
R_{cur}	当前解
α	温度下降率
$\pi_h^{T_m}$	算子$h\in H\cup I$在温度T_m下的得分
$\omega_h^{T_m}$	算子$h\in H\cup I$在温度T_m下已被使用次数
δ_1	新解优于当前最优解时算子得分
δ_2	新解劣于当前最优解但优于当前解时算子得分
δ_3	新解比当前解差但被接受时算子得分

2.1 初始解构造

节约算法是依次将 2 个子回路合并为 1 条路径，使合并后总行驶距离减少量最大。贺琪等[16]考虑了单个待插入客户的最优可行插入位置，使初始解总行驶距离较短。本文在此基础上，根据带优先配送的 GVRP-SPD 约束定义可行路径，利用车辆行驶距离和油耗来计算成本，进一步定义客户的最优可行插入位置。

设配送客户数$|N/\{0\}|=n$。首先，构造n条仅包含配送中心和单个客户的初始路径集$R^*=R_1\cup R_2\cup R_3$。然后，依次选择R_2和R_3中单客户路径与其余路径进行最优路径成本可行合并，直至R_2和R_3中已无单客户路径进行可行插入时便得到初始解。注意，R_2中单客户路径可与R_1、R_2、R_3中的路径进行合并，且合并时非优先配送客户只能插入到优先配送客户之后；R_3中单客户路径只能与R_2、R_3中的路径进行合并。

定义 1 可行路径：设路径$r_k=\{k_0,k_1,\cdots,k_w,k_{w+1}\}$，且$k_0=k_{w+1}=0$，$k_l\in N/\{0\}(1\leqslant l\leqslant w)$。若满足车载容量约束，即$u_{k_l,k_{l+1}}^k+v_{k_l,k_{l+1}}^k\leqslant Q(0\leqslant l\leqslant w)$，以及优先配送要求，即若$k_1\in N_1$，则$k_l\in N_2(2\leqslant l\leqslant w)$，否则$k_l\in N_2\cup N_3(1\leqslant l\leqslant w)$，则称$r_k$为可行路径，且路径成本为

$$C(r_k)=c_{\text{p}}\sum_{l=0}^{w}d_{k_l,k_{l+1}}+c_{\text{f}}\sum_{l=0}^{w}\rho_{k_l,k_{l+1}}^k d_{k_l,k_{l+1}}+c_{\text{v}} \tag{19}$$

定义 2 最优可行插入：若将单客户路径$r_j=\{j_0,j_1,j_2\}\in R_2\cup R_3, j_0=j_2=0, j_1\in N/\{0\}$，合并到路径$r_k=\{k_0,k_1,\cdots,k_w,k_{w+1}\}$后，得新路径：

$$r_k^*=\begin{cases}\{k_0,k_1,\cdots,k_l,j_1,k_{l+1},\cdots\},\\ r_k\in R_1,r_j\in R_2,1\leqslant l\leqslant w\\ \{k_0,k_1,\cdots,k_l,j_1,k_{l+1},\cdots\},\\ r_k,r_j\in R_2\cup R_3,0\leqslant l\leqslant w\end{cases} \tag{20}$$

若在第L个候选插入位置插入后r_k^*为可行路径，则称L为候选可行插入位置，将L加入候选可行插入位置列表$\mathcal{L}(k,j)$，且合并前后的总成本节约值为

$$\text{save}(r_k,r_j,L)=C(r_k)+C(r_j)-C(r_k^*) \tag{21}$$

进而，路径r_j中客户j_1在r_k的最优可行插入位置为

$$\hat{L}=\arg\max_{L\in\mathcal{L}(k,j)}\text{save}(r_k,r_j,L) \tag{22}$$

初始解构造算法如算法1所示，图1为路线构造示例图。

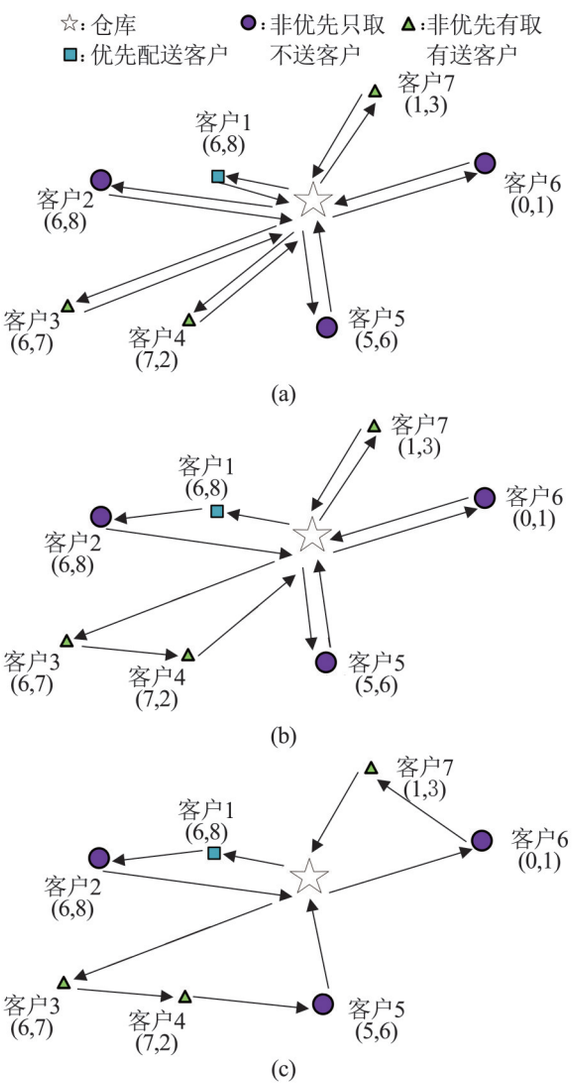


图1　路线构造示例图

Fig. 1　Route construction example diagram

算法1：构造初始解

输入：数据集，参数$\rho_{\min}$、$\rho_{\max}$

输出：初始解R_0

step 1：构造单客户初始路径集$R^*=\bigcup_{i=1}^{3}R_i$，其中，$R_i=\{r_j|r_j=\{0,j,0\},j\in N_i\}$，$i=1,2,3$；

step 2：对$\forall r_k\in R^*$和$\forall r_j\in R_2\cup R_3$，$k\neq j$且$r_j$为单客户路径，计算$r_j$上客户在路线$r_k$上的候选可行插入位置列表$\mathcal{L}(k,j)$，并找到最优可行插入位置$\hat{L}$，计算插入位置$\hat{L}$的成本节约值$\text{save}(r_k,r_j,\hat{L})$；

step 3：比较step 2中所有路径对的最优可行插入位置的成本节约值，选出成本节约值最大的两条路径进行合并，并更新R^*；

step 4：若$R_2\cup R_3$中存在单客户路径r_j，且存在路径$r_k\in R^*$对r_j有可行插入位置，则返回step 2，否则输出初始解$R_0=R^*$，结束。

图1中(q_i,p_i)为客户i的送货量和取货量。首先构造单客户路径集如图1(a)所示，此时$R_1=\{r_1\}$，$R_2=\{r_2,r_5,r_6\}$，$R_3=\{r_3,r_4,r_7\}$，其中，$r_j=\{0,j,0\}(j=1,2,\cdots,7)$。计算所有路径对的最佳可行插入位置成本节约值，若$r_3$、$r_4$合并时所节约成本最大，将$r_4$合并到$r_3$中得到$r_3=\{0,3,4,0\}$，合并后路线如图1(b)所示，此时$R_1=\{r_1\}$，$R_2=\{r_2,r_5,r_6\}$，$R_3=\{r_3,r_7\}$。重复进行单客户路径的可行插入位置及其对应成本节约值的计算，从而选择成本节约值最大的路径合并，可得初始解如图1(c)所示，此时$R_1=\{r_1\}$，$R_2=\{r_6\}$，$R_3=\{r_3\}$，其中，$r_1=\{0,1,2,0\}$，$r_3=\{0,3,4,5,0\}$，$r_6=\{0,6,7,0\}$。

2.2 基于SA的自适应大邻域搜索算法

为了增加解的多样性，避免算法陷入局部最优，设计了5种破坏算子和2种修复算子对当前解进行邻域搜索，根据搜索算子的效果自适应更新算子权重，并由SA算法确定解的接受准则。破坏算子是将非优先配送客户从当前路径中删除，并加入删除列表ℓ；修复算子是将列表ℓ中的客户逐一进行最优可行插入。

随机删除H_1：随机选择s个客户，将其中非优先配送客户从当前解R_{cur}中的相应车辆路径上移除，并入ℓ。

最差距离删除H_2：计算每个客户与其所在车

辆路径上前后客户的距离和，选择距离和最大的前s个客户，将其中非优先配送客户从对应路径移除，并入ℓ。

邻域客户删除H_3：随机选择一个客户，然后选择距该客户较近的前$s-1$个客户，将所选s个客户中非优先配送客户从对应路径移除，并入ℓ。

最差成本删除H_4：对$\forall r_k=\{k_0,k_1,\cdots,k_w,k_{w+1}\}\in R$，计算移除客户$k_l(1\leqslant l\leqslant w)$后该路径成本降低量$\Delta(k_l,r_k)=C(r_k)-C(r_k/\{k_l\})$，选择降低量较多的前$s$个客户，将其中非优先配送客户从其所在路线移除，并入ℓ。

路径删除H_5：在未包含优先配送客户的路径中，随机选择客户数最少或最大装载量最少或成本最高的路径删除，并将该路径上的客户并入ℓ。

最佳距离可行插入I_1：逐一为ℓ中客户在每条路径寻找可行插入位置，在所有可行插入位置中选择路径距离增量最小的位置进行插入。

最佳成本可行插入I_2：逐一为ℓ中客户在每条路径寻找可行插入位置，在所有可行插入位置中选择路径成本增量最小的位置进行插入。

在SA算法中，基于轮盘赌机制选择破坏算子和插入算子进行邻域搜索。初始时，5种破坏算子和2种修复算子的选择概率分别为0.2和0.5，每一温度下初始得分均为0。在迭代计算过程中，算子的选择概率更新公式为

$$P_h^{T_{m+1}}=(1-\eta)P_h^{T_m}+\eta\pi_h^{T_m}/\omega_h^{T_m},\quad h\in\{H_1,H_2,H_3,H_4,H_5,I_1,I_2\}\tag{23}$$

各算子的得分按如下规则计算：使用所选破坏算子和修复算子对当前解依次进行破坏和修复操作，并得到新解R^*，若新解R^*优于当前最优解R_{best}，则所选算子得分增加δ_1；若新解R^*优于当前解R_{cur}，则所选算子得分增加δ_2；若新解R^*比当前解R_{cur}差但被接受，则所选算子得分增加δ_3。一般地，3个参数满足$\delta_1>\delta_2>\delta_3$[17]。于是，基于模拟退火的自适应大邻域搜索(SA-ALNS)算法的具体描述如算法2所示。

算法2：SA-ALNS

输入：数据集、初始温度T_0、终止温度T_{end}、最大迭代次数M、温度下降率α，破坏算子集$H=\{H_1,H_2,H_3,H_4,H_5\}$、插入算子集$I=\{I_1,I_2\}$

输出：近似最优解R_{best}

step 1：调用算法1，构造初始解R_0；

step 2：令$R_{\text{cur}}=R_{\text{best}}=R_0$，计算解的总成本$Z(R_{\text{cur}})=Z(R_{\text{best}})=Z(R_0)$，并置降温次数$m=0$，算子选择概率$P_{H_i}^{T_m}=0.2(i=1,2,\cdots,5)$，$P_{I_j}^{T_m}=0.5(j=1,2)$；

step 3：算子初始得分$\pi_h^{T_m}=0$和被选择次数$\omega_h^{T_m}=0(h\in H\cup I)$，并置$t=1$；

step 4：依据选择概率$P_h^{T_m}(h\in H)$选择破坏算子$H_i\in H$，对R_{cur}进行破坏，得到ℓ；

step 5：依据$P_h^{T_m}(h\in I)$选择修复算子$I_j\in I$，利用ℓ对R_{cur}进行修复，得到新解R^*，并计算R^*的总成本$Z(R^*)$；

step 6：若$Z(R^*)\leqslant Z(R_{\text{best}})$，则$R_{\text{best}}=R^*$，$Z(R_{\text{best}})=Z(R^*)$，更新算子得分$\pi_h^{T_m}=\pi_h^{T_m}+\delta_1(h\in\{H_i,I_j\})$，否则转step 7；

step 7：若$Z(R^*)\leqslant Z(R_{\text{cur}})$，则$R_{\text{cur}}=R^*$，$Z(R_{\text{cur}})=Z(R^*)$，更新算子得分$\pi_h^{T_m}=\pi_h^{T_m}+\delta_2(h\in\{H_i,I_j\})$，转step 9，否则转step 8；

step 8：依概率接受R^*，即生成随机数$\varepsilon\in[0,1]$，若$e^{-(Z(R^*)-Z(R_{\text{cur}}))/T_m}\geqslant\varepsilon$，则置$R_{\text{cur}}=R^*$，$Z(R_{\text{cur}})=Z(R^*)$，更新算子得分$\pi_h^{T_m}=\pi_h^{T_m}+\delta_3(h\in\{H_i,I_j\})$；

step 9：更新算子被选用次数$\omega_h^{T_m}=\omega_h^{T_m}+1(h\in\{H_i,I_j\})$，置迭代次数$t=t+1$；

step 10：若迭代次数$t\leqslant M$，转step 4；否则令温度$T_{m+1}=\alpha T_m$，更新算子的选择概率$P_h^{T_{m+1}}=(1-\eta)P_h^{T_m}+\eta\pi_h^{T_m}/\omega_h^{T_m}(h\in\{H_i,I_j\})$并置$m=m+1$，转step 11；

step 11：若$T_m<T_{\text{end}}$，则输出近似最优解R_{best}，算法结束，否则转step 3。

3 实验分析

为了验证本文方法的有效性，首先，对优先配送客户分别采用传统的单独配送和优先配送2种策略进行实验分析，其次，将SA-ALNS算法与HH-ILS算法[12]进行对比实验。算法采用MATLAB R2021a编程，在Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz RAM8G Win10环境下运行。实验数据采用文献[18]中修改的Solomon标准算例(增加了客户取货量)和文献[12]中CMTX算例。修改的Solomon算例共有R1、C1、RC1、R2、C2和RC2六类58个算例，每个算例均有1个配送中心和100个客户，车载容量均为200，且R类、C类和RC类客户地理位置分别为随机分布、聚类分布以及随机与聚类的混合分布，R1、C1、RC1相较于R2、C2、RC2有更紧的时间窗由于本文未考虑时间窗，故选取R1、C1、RC1共29个Solomon算例。CMTX包含14个算例，客户数n从50到199。此外，修改的Solomon算例和CMTX算例均未考虑优先配送，故在每个算例中采用随机方式按比例1:4:5将客户分为优先配送、非优先只取不送和非优先有取有送三类客户。

3.1 实验对比结果

经测试得到较好的算法参数为$T_0=100$，$T_{\text{end}}=10$，$M=90$，$\eta\in[0,1]$，$\delta_1=30$，$\delta_2=10$，$\delta_3=6$。一般地，删除客户数$s\in[4,16]$，为了保证解的多样性取$s=16$；温度下降率$\alpha\in[0.85,0.999\,99]$，下降率较小时迭代次数较少，难以保证收敛到近似最优解，下降率过大时，算法需要更长的迭代时间，为了加快收敛速度取$\alpha=0.99$。此外，$c_{\text{f}}=7.21$元/L，$c_{\text{p}}=10$元/km，$c_{\text{v}}=200$元/辆。

对优先配送客户分别采用传统单独配送(TSD)和优先配送(SA-ALNS)两种策略在修改的Solomon算例上进行对比。表3是算法独立运行10次的实验结果，从表3可以看出：

表3　两种配送策略在Solomon算例上的结果
Table 3　Results of two distribution strategy on Solomon instances

算例	TSD(单独配送)			SA-ALNS(优先配送)			相对偏差/%		
	(C_1, Z_1)	(C_2, σ_1)	(Z_2, σ_2)	(C_1, Z_1)	(C_2, σ_1)	(Z_2, σ_2)	RE_1	RE_2	RE_3
Rdp101	(16, 18 862)	(16.1, 0.32)	(19 158, 241)	(14, 17 725)	(14.2, 0.42)	(17 949, 148)	14.29	6.41	6.74
Rdp102	(16, 18 990)	(16.3, 0.48)	(19 193, 164)	(14, 17 945)	(14.4, 0.52)	(18 239, 110)	14.29	5.82	5.23
Rdp103	(16, 18 830)	(16.5, 0.53)	(19 171, 261)	(14, 17 563)	(14, 0)	(17 670, 96)	14.29	7.21	8.50
Rdp104	(16, 18 866)	(17.1, 0.57)	(19 659, 349)	(14, 18 150)	(14.7, 0.48)	(18 312, 119)	14.29	3.95	7.35
Rdp105	(16, 18 771)	(16.6, 0.52)	(19 367, 342)	(15, 17 965)	(15, 0)	(18 069, 72)	6.67	4.49	7.19
Rdp106	(17, 18 877)	(17.3, 0.48)	(19 489, 385)	(14, 18 172)	(14.5, 0.53)	(18 330, 125)	21.43	3.88	6.32
Rdp107	(17, 19 338)	(17.1, 0.32)	(19 807, 246)	(15, 18 493)	(15, 0)	(18 669, 102)	13.33	4.57	6.09
Rdp108	(16, 18 783)	(16.3, 0.48)	(19 137, 211)	(14, 17 882)	(14.5, 0.53)	(18 220, 273)	14.29	5.03	5.03
Rdp109	(17, 19 271)	(17, 0)	(19 666, 204)	(15, 18 333)	(15, 0)	(18 374, 44)	13.33	5.12	7.03
Rdp110	(17, 19 778)	(17.1, 0.32)	(20 038, 182)	(15, 18 217)	(15, 0)	(18 491, 106)	13.33	8.57	8.37
Rdp111	(17, 19 539)	(18, 0.47)	(20 082, 259)	(15, 18 413)	(15, 0)	(18 540, 94)	13.33	6.11	8.32
Rdp112	(17, 19 317)	(17.4, 0.52)	(19 948, 298)	(15, 18 050)	(15, 0)	(18 290, 202)	13.33	7.02	9.06
平均值	(16.5, 19 102)	(16.9, 0.42)	(19 560, 262)	(14.5, 18 076)	(14.7, 0.21)	(18 263, 124)	13.85	5.68	7.10
Cdp101	(19, 20 810)	(19.6, 0.52)	(21 043, 147)	(15, 19 503)	(15.9, 0.32)	(19 781, 162)	26.67	6.70	6.38
Cdp102	(19, 22 149)	(19.5, 0.53)	(22 339, 195)	(14, 20 710)	(14, 0)	(20 865, 147)	35.71	6.95	7.07
Cdp103	(19, 22 763)	(19, 0)	(23 042, 217)	(16, 20 859)	(16, 0)	(20 954, 67)	18.75	9.13	9.97
Cdp104	(20, 21 247)	(20, 0)	(21 301, 24)	(15, 20 532)	(15.9, 0.32)	(20 904, 380)	33.33	3.48	1.90

续表

算例	TSD(单独配送)			SA-ALNS(优先配送)			相对偏差/%		
	(C_1, Z_1)	(C_2, σ_1)	(Z_2, σ_2)	(C_1, Z_1)	(C_2, σ_1)	(Z_2, σ_2)	RE_1	RE_2	RE_3
Cdp105	(19, 21 696)	(19.1, 0.32)	(21 762, 96)	(14, 21 157)	(14, 0)	(21 367, 113)	35.71	2.54	1.85
Cdp106	(20, 22 439)	(20, 0)	(22 439, 0)	(14, 21 108)	(14, 0)	(21 163, 64)	42.86	6.30	6.03
Cdp107	(20, 21 976)	(20, 0)	(22 757, 275)	(15, 21 092)	(15.6, 0.52)	(21 198, 84)	33.33	4.19	7.36
Cdp108	(19, 21 967)	(19.3, 0.48)	(22 060, 65)	(14, 20 508)	(14.4, 0.52)	(21 153, 259)	35.71	7.12	4.29
Cdp109	(19, 21 668)	(19.7, 0.48)	(21 819, 92)	(16, 20 552)	(16, 0)	(20 653, 73)	18.75	5.43	5.64
平均值	(19.3, 21 857)	(19.6, 0.26)	(22 063, 123)	(14.8, 20 669)	(15.1, 0.19)	(20 893, 150)	31.20	5.76	5.61
RCdp101	(18, 23 336)	(18, 0)	(23 394, 69)	(14, 22 186)	(14, 0)	(22 385, 264)	28.57	5.18	4.51
RCdp102	(18, 23 678)	(18, 0)	(23 740, 37)	(14, 20 901)	(14, 0)	(21 236, 170)	28.57	13.29	11.79
RCdp103	(18, 25 139)	(18, 0)	(25 282, 86)	(14, 22 019)	(14, 0)	(22 174, 159)	28.57	14.17	14.02
RCdp104	(18, 23 887)	(18, 0)	(24 060, 121)	(14, 21 069)	(14.5, 0.53)	(21 420, 288)	28.57	13.37	12.32
RCdp105	(18, 23 635)	(18, 0)	(24 061, 279)	(14, 20 715)	(14, 0)	(20 913, 86)	28.57	14.09	15.06
RCdp106	(18, 22 866)	(18.1, 0.32)	(23 438, 408)	(14, 20 864)	(14.6, 0.52)	(21 195, 309)	28.57	9.59	10.58
RCdp107	(18, 24 685)	(18.2, 0.42)	(25 243, 360)	(14, 20 406)	(14, 0)	(20 525, 87)	28.57	20.97	22.99
RCdp108	(17, 23 488)	(17.9, 0.32)	(23 922, 295)	(14, 21 819)	(14, 0)	(21 932, 76)	21.43	7.65	9.07
平均值	(17.9, 23 839)	(18, 0.13)	(24 142, 207)	(14, 21 247)	(14.1, 0.13)	(21 472, 180)	27.68	12.29	12.54
总均值	(17.8, 21 264)	(18, 0.29)	(21 601, 204)	(14.4, 19 756)	(14.7, 0.18)	(19 964, 148)	23.05	7.53	8.14

注：C_1、Z_1、C_2和Z_2分别表示10次求解的最优车辆数、最优总成本、平均车辆数和平均总成本；σ_1和σ_2分别表示车辆数和总成本的标准差；RE_1、RE_2、RE_3分别表示两种配送策略下的最优车辆数、最优总成本、平均总成本的相对偏差。

(1) SA-ALNS算法在所有算例上的车辆数均有显著减少，其中，R类算例减少1~3辆，平均减少13.85%；C类算例减少3~6辆，平均减少31.2%；RC类算例减少3~4辆，平均减少27.68%。可见，SA-ALNS算法对客户地理位置分布较集中的C类算例具有较好的车辆数优化效果，符合实际。

(2) SA-ALNS算法在所有算例上的总成本均有减少，其中，R类、C类和RC类算例平均分别减少5.68%、5.76%、12.29%。对于客户地理位置混合分布的RC类算例，SA-ALNS算法具有较好的总成本优化效果，而R和C类算例总成本优化效果相近。

(3) 从10次求解结果的标准差看，在R类、C类和RC类算例上，单独配送策略的车辆数标准差平均分别为0.42、0.26、0.13，总成本标准差平均分别为262、123、207；SA-ALNS算法的车辆数标准差平均分别为0.21、0.19、0.13，总成本标准差平均分别为124、150、180。这表明算法SA-ALNS均有较强的稳定性。

综上，采用优先配送策略的算法SA-ALNS在求解带优先配送客户的GVRP-SPD时，在车辆数和总成本两方面均优于传统单独配送策略，且具有较强的稳定性。

为了进一步验证SA-ALNS算法的有效性，在CMTX算例上与文献[12]中相似算法HH-ILS进行对比实验。由于HH-ILS算法未考虑优先配送需求，而本文方法需要考虑一定数量的优先配送客户，为此，实验时将优先配送客户的送货量均设置为车辆最大载重量，使其在配送时优先配送这一类客户，而在计算可行解成本时以客户实际送货量进行计算。两种算法对CMTX中每个算例独立求解10次，结果如表4所示。

从表4可见，对不同规模的CMTX算例，SA-ALNS算法所求得的车辆数、总成本均优于HH-ILS算法，且标准差较小。表明SA-ALNS算法是有效的，且具有较高的稳定性。

3.2 算子性能检验

为检验SA-ALNS算法中不同算子的有效性，记SA-ALNS去除算子$h(h\in\{H_1,H_2,H_3,H_4,H_5,I_1,I_2\})$后的算法为SA-ALNS-$h$。表5和表6给出了部分Solomon算例的实验结果。从表5和表6可见，从算法SA-ALNS中任意去除5种破坏算子和2种修复算子中1个时，在9个算例上的总成本均有显著增加，表明7种算子对优化总成本是有效的。

3.3 算法收敛性分析

对于SA-ALNS算法的收敛性，图2给出了部分算例收敛变化曲线。

表4　两种算法在CMTX算例上的结果
Table 4　Results of two algorithm on CMTX instances

算例	n	HH-ILS[12]			SA-ALNS			相对偏差(%)		
		$(C_1, Z_{1)}$	(C_2, σ_1)	(Z_2, σ_2)	(C_1, Z_1)	(C_2, σ_1)	(Z_2, σ_2)	RE_1	RE_2	RE_3
CMTX1	50	(8, 11 501)	(8.5, 0.7)	(11 966, 402)	(7, 9 332)	(7, 0)	(9 462, 108)	14.29	23.24	26.47
CMTX2	75	(13, 14 030)	(14.1, 0.7)	(15 606, 623)	(11, 12 420)	(11.3, 0.5)	(12 596, 148)	18.18	12.96	23.90
CMTX3	100	(16, 21 872)	(17.8, 1.6)	(23 290, 994)	(12, 17 672)	(12.1, 0.3)	(17 710, 43)	33.33	23.77	31.51
CMTX4	150	(22, 28 793)	(23.3, 1.2)	(29 769, 1 115)	(19, 24 089)	(19, 0)	(24 236, 87)	15.79	19.53	22.83
CMTX5	199	(32, 38 297)	(32.8, 0.9)	(41 115, 1 488)	(25, 32 737)	(25, 0)	(33 010, 154)	28.00	16.98	24.56
CMTX6	50	(9, 12 133)	(10, 0.7)	(12 878, 547)	(7, 9 301)	(7, 0)	(9 301, 0)	28.57	30.44	38.46
CMTX7	75	(15, 15 100)	(16.4, 1.1)	(15 939, 604)	(11, 12 562)	(11.1, 0.3)	(12 769, 84)	36.36	20.21	24.83
CMTX8	100	(18, 22 980)	(19.1, 0.7)	(24 120, 970)	(13, 17 270)	(13, 0)	(17 408, 118)	38.46	33.07	38.56
CMTX9	150	(22, 29 050)	(23.9, 0.9)	(30 139, 687)	(19, 23 932)	(19, 0)	(24 062, 100)	15.79	21.38	25.25
CMTX10	199	(33, 40 029)	(33.6, 0.7)	(41 740, 1 519)	(25, 32 683)	(25.4, 0.5)	(33 252, 263)	32.00	22.48	25.53
CMTX11	120	(16, 27 745)	(16.9, 0.9)	(28 421, 516)	(14, 25 949)	(14.3, 0.5)	(26 409, 442)	14.29	6.92	7.62
CMTX12	100	(15, 20 963)	(16.3, 0.9)	(22 429, 1 349)	(13, 19 143)	(13.3, 0.5)	(20 037, 529)	15.38	9.51	11.94
CMTX13	120	(16, 26 295)	(17.4, 0.7)	(28 856, 1 124)	(14, 25 850)	(14.4, 0.5)	(26 293, 383)	14.29	1.72	9.75
CMTX14	100	(15, 20 456)	(15.7, 0.7)	(21 818, 1 414)	(13, 18 593)	(13, 0)	(19 215, 412)	15.38	10.02	13.54
平均值		(17.9, 23 517)	(19, 0.9)	(24 863, 954)	(14.5, 20 110)	(14.6, 0.2)	(20 411, 205)	22.87	18.02	23.20

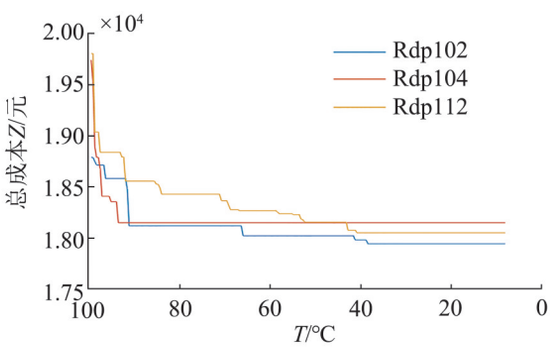
表5　算法SA-ALNS中破坏算子性能测试结果
Table 5　Results of destruction operators in SA-ALNS

算例	SA-ALNS	SA-ALNS-H_1	SA-ALNS-H_2	SA-ALNS-H_3	SA-ALNS-H_4	SA-ALNS-H_5
	Z_2	ΔZ_2-H_1	ΔZ_2-H_2	ΔZ_2-H_3	ΔZ_2-H_4	ΔZ_2-H_5
Rdp101	17 949	+22	+9	+161	+21	+93
Rdp103	17 670	+71	+73	+90	+56	+17
Rdp105	18 069	+145	+16	+230	+48	+2
Cdp102	20 865	+673	+334	+400	+273	+302
Cdp103	20 954	+276	+67	+150	+119	+104
Cdp105	21 367	+336	+117	+145	+146	+98
RCdp103	22 174	+16	+45	+399	+9	+13
RCdp104	21 420	+1 071	+1 040	+1 419	+879	+676
RCdp105	20 913	+577	+459	+596	+491	+496
平均值		+354	+240	+399	+227	+200

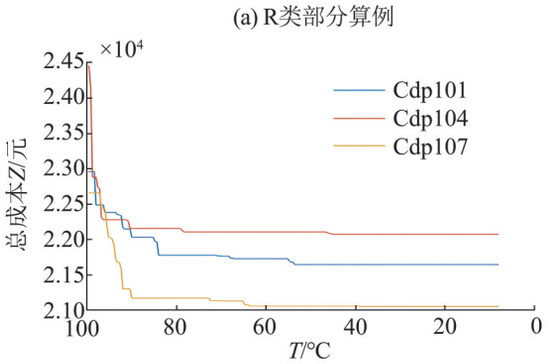
注：ΔZ_2-h表示SA-ALNS-h相对于SA-ALNS的最优总成本均值增量。

表6　算法SA-ALNS中修复算子性能测试结果
Table 6　Results of insertion operators in SA-ALNS

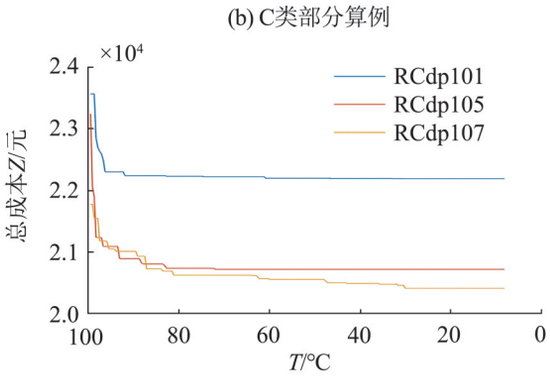
算例	SA-ALNS	SA-ALNS-I_1	SA-ALNS-I_2
	Z_2	ΔZ_2-I_1	ΔZ_2-I_2
Rdp101	17 949	+80	+23
Rdp103	17 670	+74	+60
Rdp105	18 069	+112	+58
Cdp102	20 865	+349	+214
Cdp103	20 954	+105	+67
Cdp105	21 367	+168	+181
RCdp103	22 174	+55	+96
RCdp104	21 420	+1 040	+1 098
RCdp105	20 913	+542	+461
平均值		+281	+251



(a) R类部分算例



(b) C类部分算例



(c) RC类部分算例

图2　SA-ALNS收敛曲线图
Fig. 2　Convergence curve of SA-ALNS

从图2可以看出，在SA-ALNS算法的寻优过程中，R类、RC类算例均存在降温初期就达到收敛和降温迭代后期才达到收敛的情况；C类算例均在降温至40~60℃范围达到收敛。说明SA-ALNS算法求解不同类型算例均能快速收敛到最优解，且客户地理位置分布情况对SA-ALNS算法的收敛性影响不明显。

4　结论

本文在GVRP-SPD基础上，进一步考虑带有优先配送客户这一实际情况，建立了以总成本最小为目标的非线性优化模型。针对该模型，采用改进的节约算法构造初始解，并以SA算法为框架，设计了具有5种破坏算子和2种修复算子的SA-ALNS算法进行邻域搜索求解。在Solomon算例及CMTX算例上的对比实验表明：SA-ALNS能有效降低总成本和减少车辆数，且具有较快的收敛速度和较强的稳定性。在实际物流配送中，客户的位置和需求往往是在不断变化的，不确定性会使车辆路径规划变得更复杂、更困难。因此，在今后研究中可以进一步考虑供给和需求的不确定性。

参考文献：

[1] 周鲜成, 周开军, 王莉, 等. 物流配送中的绿色车辆路径模型与求解算法研究综述[J]. 系统工程理论与实践, 2021, 41(1): 213-230.
Zhou Xiancheng, Zhou Kaijun, Wang Li, et al. Review of Green Vehicle Routing Model and Its Algorithm in Logistics Distribution[J]. Systems Engineering-Theory & Practice, 2021, 41(1): 213-230.

[2] Sevgi Erdoğan, Miller-Hooks E. A Green Vehicle Routing Problem[J]. Transportation Research Part E: Logistics and Transportation Review, 2012, 48(1): 100-114.

[3] Ferreira K M, Queiroz T A, Toledo F M B. An Exact Approach for the Green Vehicle Routing Problem with Two-dimensional Loading Constraints and Split Delivery [J]. Computers & Operations Research, 2021, 136: 105452.

[4] 胡蓉, 陈文博, 钱斌, 等. 学习型蚁群算法求解绿色多车

场车辆路径问题[J]. 系统仿真学报, 2021, 33(9): 2095-2108.
Hu Rong, Chen Wenbo, Qian Bin, et al. Learning Ant Colony Algorithm for Green Multi-depot Vehicle Routing Problem[J]. Journal of System Simulation, 2021, 33(9): 2095-2108.

[5] Poonthalir G, Nadarajan R. A Fuel Efficient Green Vehicle Routing Problem with Varying Speed Constraint (F-GVRP) [J]. Expert Systems with Applications, 2018, 100: 131-144.

[6] 高飞. 不确定因素下配送路径优化问题研究[D]. 北京: 北京交通大学, 2019.
Gao Fei. Research on Distribution Routing Optimization Problems with Uncertain Factor[D]. Beijing: Beijing Jiaotong University, 2019.

[7] Majidi S, Hosseini-Motlagh S M, Ignatius J. Adaptive Large Neighborhood Search Heuristic for Pollution-routing Problem with Simultaneous Pickup and Delivery [J]. Soft Computing, 2018, 22(9): 2851-2865.

[8] Foroutan R A, Rezaeian J, Mahdavi I. Green Vehicle Routing and Scheduling Problem with Heterogeneous Fleet Including Reverse Logistics in the Form of Collecting Returned Goods[J]. Applied Soft Computing, 2020, 94: 106462.

[9] Qi Rui, Li Junqing, Wang Juan, et al. QMOEA: A Q-learning-based Multiobjective Evolutionary Algorithm for Solving Time-dependent Green Vehicle Routing Problems with Time Windows[J]. Information Sciences, 2022, 608: 178-201.

[10] Niu Yunyun, Yang Zehua, Wen Rong, et al. Solving the Green Open Vehicle Routing Problem Using a Membrane-inspired Hybrid Algorithm[J]. Sustainability, 2022, 14(14): 8661.

[11] Majidi S, Hosseini-Motlagh S M, Yaghoubi S, et al. Fuzzy Green Vehicle Routing Problem with Simultaneous Pickup-delivery and Time Windows[J]. Rairo-Operations Research, 2017, 51(4): 1151-1176.

[12] Olgun B, Koç C, Altıparmak F. A Hyper Heuristic for the Green Vehicle Routing Problem with Simultaneous Pickup and Delivery[J]. Computers & Industrial Engineering, 2021, 153: 107010.

[13] Polat O, Kalayci C B, Topaloğlu D. Modelling and Solving the Milk Collection Problem with Realistic Constraints[J]. Computers & Operations Research, 2022, 142: 105759.

[14] Wang Zheng, Li Ying, Hu Xiangpei. A Heuristic Approach and a Tabu Search for the Heterogeneous Multi-type Fleet Vehicle Routing Problem with Time Windows and an Incompatible Loading Constraint[J]. Computers & Industrial Engineering, 2015, 89: 162-176.

[15] Kirkpatrick S, Gelatt C D, Vecchi M P. Optimization by Simulated Annealing[J]. Science, 1983, 220(4598): 671-680.

[16] 贺琪, 官礼和, 崔焕焕. 硬时间窗VRP的混合变邻域禁忌搜索算法[J]. 计算机工程与应用, 2023, 59(13): 82-91.
He Qi, Guan Lihe, Cui Huanhuan. Hybrid Variable Neighborhood Tabu Search Algorithm for Vehicle Routing Problem with Hard Time Window[J]. Computer Engineering and Applications, 2023, 59(13): 82-91.

[17] 李珺, 段钰蓉, 郝丽艳, 等. 混合优化算法求解同时送取货车辆路径问题[J]. 计算机科学与探索, 2022, 16(7): 1623-1632.
Li Jun, Duan Yurong, Hao Liyan, et al. Hybrid Optimization Algorithm for Vehicle Routing Problem with Simultaneous Delivery-pickup[J]. Journal of Frontiers of Computer Science & Technology, 2022, 16 (7): 1623-1632.

[18] Wang Hsiaofan, Chen Yingyen. A Genetic Algorithm for the Simultaneous Delivery and Pickup Problems with Time Window[J]. Computers and Industrial Engineering, 2012, 62(1): 84-95.