

# 多车场带时间窗车辆路径问题的改良自适应大邻域搜索算法

李焱<sup>1,2</sup>, 潘大志<sup>1,2\*</sup>, 郑思情<sup>1</sup>

(1. 西华师范大学 数学与信息学院, 四川 南充 637009;

2. 最优化理论与应用四川省高校重点实验室(西华师范大学), 四川 南充 637009)

(\* 通信作者电子邮箱 pdzzj@126.com)

**摘要:**针对多车场带时间窗车辆路径问题(MDVRPTW),提出一种改良自适应大邻域搜索算法(IALNS)。首先,在构造初始解阶段改进一种路径分割算法;其次,在优化阶段利用设计的移除和修复启发式算子相互竞争择优选取算子,为各算子引入评分机制,采用轮盘赌方式选取启发式算子;同时,将迭代周期分段,动态调整各周期内的算子权重信息,有效避免算法陷入局部最优;最后,采取模拟退火机制作为解的接受准则。在Cordeau规范算例上进行实验,确定IALNS的相关参数,将所提算法求解结果与该领域其他代表性研究成果对比。实验结果表明,所提算法与变邻域搜索(VNS)算法的求解误差不超过0.8%,在某些算例上甚至更优;与多相位改进的蛙跳算法相比,算法的平均耗时减少12.8%,所提算法在绝大多数算例上运行时间更短。因此,验证了所提算法是求解MDVRPTW的有效算法。

**关键词:**多车场带时间窗车辆路径问题;自适应大邻域搜索;序列分割;自适应权重;模拟退火

**中图分类号:**TP301.6 **文献标志码:**A

## Improved adaptive large neighborhood search algorithm for multi-depot vehicle routing problem with time window

LI Yan<sup>1,2</sup>, PAN Dazhi<sup>1,2\*</sup>, ZHENG Siqing<sup>1</sup>

(1. School of Mathematics & Information, China West Normal University, Nanchong Sichuan 637009, China;

2. Sichuan Colleges and Universities Key Laboratory of Optimization Theory and Applications

(China West Normal University), Nanchong Sichuan 637009, China)

**Abstract:** Aiming at the Multi-Depot Vehicle Routing Problem with Time Window (MDVRPTW), an Improved Adaptive Large Neighborhood Search algorithm (IALNS) was proposed. Firstly, a path segmentation algorithm was improved in the stage of constructing the initial solution. Then, in the optimization stage, the designed removal and repair heuristic operators were used to compete with each other to select the optimal operator, a scoring mechanism was introduced for the operators, and the heuristic operator was selected by roulette. At the same time, the iteration cycle was segmented and the operator weight information was dynamically adjusted in each cycle, effectively to prevent the algorithm from falling into local optimum. Finally, simulated annealing mechanism was adopted as the acceptance criterion of the solution. The relevant parameters of the IALNS were determined by experiments on the Cordeau normative instances, and the solution results of the proposed algorithm were compared with other representative research results in this field. The experimental results show that the solution error between IALNS and Variable Neighborhood Search (VNS) algorithm does not exceed 0.8%, even better in some cases; compared with the multi-phase improved shuffled frog leaping algorithm, the average time-consuming of the proposed algorithm is reduced by 12.8%, and the runtime is shorter for most instances. So the above results verify IALNS is an effective algorithm for solving MDVRPTW.

**Key words:** Multi-Depot Vehicle Routing Problem with Time Window (MDVRPTW); Adaptive Large Neighborhood Search (ALNS); sequence segmentation; adaptive weight; simulated annealing

## 0 引言

自Dantzig等<sup>[1]</sup>1959年首次提出车辆路径问题(Vehicle Routing Problem, VRP),学者们对它进行了广泛、深入的研究,衍生出该问题的许多模型,并取得了丰硕的研究成果。随着科技和社会的不断进步,运输行业迎来蓬勃发展,为了降低企业运输成本,响应国家可持续发展政策,降低碳排放,

同时为客户提供优质、便捷、满意的服务,多车场带时间窗VRP(Multi-Depot VRP with Time Window, MDVRPTW)近年来得到了学者们的广泛关注,且已被证明是NP(Non-deterministic Polynomial)难问题。过去的几十年中,学者们针对VRP提出了许多算法,包括精确算法与启发式算法,但精确算法针对大规模的问题进行求解十分耗时,适用

收稿日期:2023-06-14;修回日期:2023-09-16;录用日期:2023-09-20。

基金项目:国家自然科学基金资助项目(11871059);西华师范大学英才科研基金资助项目(17YC385)。

作者简介:李焱(1997—),男(苗族),四川泸州人,硕士研究生,主要研究方向:智能计算、组合优化;潘大志(1974—),男,四川三台人,教授,博士,主要研究方向:智能计算、算法设计;郑思情(2000—),女,四川南充人,硕士研究生,主要研究方向:均匀凸优化。

性不强,目前的研究主要集中于启发式算法。

由于时间和空间的约束,使问题的复杂度大幅增加,现有的研究主要集中在多车场 VRP (Multi-Depot VRP, MDVRP)上,但目前对 MDVRPTW 的研究有限。Cordeau 等<sup>[2]</sup>首次使用禁忌搜索(Tabu Search, TS)算法求解 MDVRPTW,采用带禁忌的邻域搜索,搜索过程允许产生不可行解,算法在设计的 Cordeau 规范算例上取得显著的效果。Cordeau 等<sup>[3]</sup>又提出了车辆行驶时间的优化计算方法,更新了上述算法的最优解。Polacek 等<sup>[4]</sup>提出了变邻域搜索(Variable Neighborhood Search, VNS)算法求解 MDVRPTW,设计了 12 种不同的邻域结构振荡初始路径,应用向前松弛时间(Forward Time Slack)概念改善生成的解,实验结果表明该算法求解结果优于 Cordeau 等<sup>[3]</sup>提出的算法。Dono 等<sup>[5]</sup>提出一种三阶段启发式算法并在 Solomon 算例上验算,该算法基于聚类的思想,结果较理想。洪联系等<sup>[6]</sup>提出了一个聚类 and 迭代混合的遗传算法,该算法也是一个三阶段过程,先进行客户聚类分配,再规划路径,最后改进路径,使用了竞争-插入策略改进路径,在 Cordeau 数据集上进行验证,表现良好。王征等<sup>[7]</sup>提出了一种改进型 VNS 算法求解 MDVRPTW,采用聚类的方法完成客户的分配,并产生初始解,运用多种混合算子进行局部搜索,采用 Gendreau 等<sup>[8]</sup>提出的 US (Unstringing and Stringing)算法进行路径的后优化过程,实验验证了该算法的求解质量较未改进前有所提高。于滨等<sup>[9]</sup>提出了两阶段启发式算法求解 MDVRPTW,第 1 阶段先将客户分配给各车场,第 2 阶段采用改进的蚁群(Ant Colony Optimization, ACO)算法求解,设计了交叉算子和自适应权重信息素更新策略,在 Cordeau 数据集上验证了该算法是一种有竞争力的算法。Luo 等<sup>[10]</sup>提出了一种多阶段改进混合蛙跳算法求解 MDVRPTW,在第 1 阶段对每个青蛙集群执行 MSFLA (Modified Shuffled Frog Leaping Algorithm)进行本地搜索,第 2 阶段利用二元锦标赛选择好的个体构建新种群,第 3 阶段对种群实施聚类调整生成新的聚类,实验结果表明该算法能在较短时间获得高质量的解,适用于求解大规模问题。Tamashiro 等<sup>[11]</sup>提出扩展节约里程算法(Clark-Wright saving algorithm, C-W)结合 TS 算法求解 MDVRPTW,TS 算法在客户分配阶段有效实现了邻域的缩减,扩展的 C-W 算法用于处理时间窗等约束,实验结果表明该算法能取得较好的结果。Sadati 等<sup>[12]</sup>提出了一种变量邻域 TS 算法求解 MDVRPTW,在强化阶段采用颗粒状局部搜索机制,在 VNS 的多样化阶段采用禁忌振荡机制,实验结果表明该算法经过适当的改变可用于求解多种类型的 VRP。

综合 MDVRPTW 的相关研究,解决此问题的思想大致分为客户划分和路径优化这 2 个阶段,运用的启发式算法或单独、或混合作用于这 2 个阶段。由于时间和空间的双重约束,使各启发式算法之间未能得到较好的调和,自适应经验的提取未得到较好的解决,某些启发式算法还受限于问题的规模,使求解质量和效率未能满足实用性。邻域搜索算法是目前解决该问题的算法之一,为此,本文提出一种改良自适应大邻域搜索算法(Improved Adaptive Large Neighborhood Search algorithm, IALNS)求解 MDVRPTW,较好解决启发式算法之间的融合及自适应学习经验的提取,在 Cordeau 规范算例上的实验结果表明,所提算法对问题的规模适应性强,并能取得较满意的求解结果。

## 1 MDVRPTW 数学描述及模型

在基本的 VRP 中,已知一组客户及每个客户的需求、一队车辆及各车辆的容量,以及地理空间数据(其中包括道路网络每条边上的旅行成本),本文试图使总路线具有最低的旅行成本并满足客户的需求。除传统的 VRP 约束外,MDVRPTW 还有多个仓库和时间窗的约束,规定了每个客户的可用服务时间,因此车辆只能在时间窗内为客户提供服务。该问题的数学模型<sup>[11]</sup>描述如下:

$$\text{Minimize } \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{i,j}^k x_{i,j}^k \quad (1)$$

$$\text{s. t. } \sum_{k \in K} \sum_{j \in N} x_{i,j}^k = 1; \forall i \in C \quad (2)$$

$$\sum_{i \in N} x_{i,h}^k - \sum_{j \in N} x_{h,j}^k = 0; \forall h \in C, \forall k \in K \quad (3)$$

$$\sum_{i \in C} x_{i,h}^k - \sum_{j \in C} x_{h,j}^k = 0; \forall h \in D, \forall k \in K \quad (4)$$

$$\sum_{i \in C} m_i \sum_{j \in N} x_{i,j}^k \leq Q_k; \forall k \in K \quad (5)$$

$$\sum_{i \in N} \sum_{j \in N} d_{i,j} x_{i,j}^k \leq R_k; \forall k \in K \quad (6)$$

$$x_{i,j}^k (at_i + st_i + wt_i + t_{i,j}) \leq at_j; \forall i \in N, j \in N, k \in K \quad (7)$$

$$et_i \leq at_i \leq lt_i; \forall i \in N, \forall k \in K \quad (8)$$

$$x_{i,j}^k \in \{0, 1\}; \forall i \in N, \forall j \in N, \forall k \in K \quad (9)$$

其中:车场集  $D = \{1, 2, \dots, m\}$ , 车辆集  $K = \{1, 2, \dots, k\}$ , 客户集  $C = \{1, 2, \dots, n\}$ , 节点集  $N = \{n_i | n_i \in (D \cup C)\}$ 。对于任意车辆  $K_k$ ,  $c_{i,j}^k$  表示从  $n_i$  到  $n_j$  的旅行成本,  $Q_k$  表示该车辆的最大容量,  $R_k$  表示该车辆的最长旅行距离。对于任意客户  $C_i$ ,  $m_i$  表示该客户的需求,  $at_i$  表示该客户的到达时间,  $wt_i$  表示该客户的等待时间。  $et_i$ ,  $lt_i$  分别表示  $n_i$  的最早和最迟开始服务时间,  $d_{i,j}$  表示  $n_i$  到  $n_j$  的距离。  $st_i$  表示  $C_i$  的服务时间,  $t_{i,j}$  表示  $C_i$  到  $C_j$  的旅行时间。  $x_{i,j}^k$  是 0-1 决策变量,如果在车辆  $k$  的路线上客户  $j$  紧邻在客户  $i$  后,则  $x_{i,j}^k=1$ ; 否则  $x_{i,j}^k=0$ 。

式(2)表示每个客户仅由一辆车服务一次;式(3)表示到达客户  $h$  的车辆数和离开客户  $h$  的车辆数相同;式(4)表示从每个车场出发的车辆数等于回到车场的车辆数;式(5)表示车辆的容量限制;式(6)表示车辆的最长旅行距离限制;式(7)表示连续的两个客户时间顺序的一致性;式(8)表示客户的服务时间窗口限制。MDVRPTW 的示意图如图 1 所示。

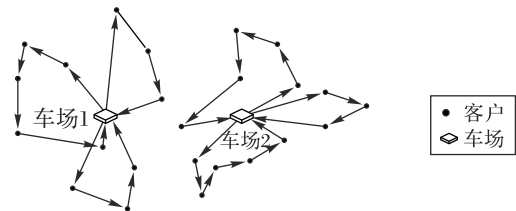


图1 WDVPTW 的示意图

Fig. 1 Schematic diagram of WDVPTW

## 2 MDVRPTW 的改良 ALNS

大邻域搜索算法(Large Neighborhood Search algorithm, LNS)求解带时间窗的 VRP (VRP with Time Window, VRPTW)效果较好,移除较差位置的客户和将客户快速插入更好的位置是影响 LNS 搜索效果的两个重要因素。刘小兰等<sup>[13]</sup>在 LNS 思想基础上,在初始解的构造方面添加“开始服

务时间最早优先法”,再基于该优先法提出“短路径优先策略”,以减少服务客户的车辆数。邻域搜索方面,事先估计邻域搜索算子改进的一个较好目标值,将它作为改进目标值的一个下界,将问题转化为k-生成树问题,大幅缩小搜索的范围,提高搜索效率;但LNS在移除算子和插入算子种类的设置方面较单一,对各算子历史表现也没有设置一个好的策略提取自适应经验。

自适应大邻域搜索算法(Adaptive Large Neighborhood Search algorithm, ALNS)是Ropke等<sup>[14]</sup>在解决带时间窗的提货交付问题(Pickup and Delivery Problem with Time Windows, PDPTW)时所提出的一种启发式算法。它的基本思想是根据对邻域搜索算子的历史表现,自适应地选择好的算子破坏和修复当前解,利用算子间的互相竞争产生解的邻域结构,从而更大几率地获得更好的解。

本文在ALNS的基础上,相应修改算法的主要过程,提出用于求解MDVRPTW的IALNS。图2为IALNS的流程。IALNS的核心组成部分包括初始解的构造、算子的设计、算子选择策略和自适应权重调整机制,以及解的接受准则和算法的停止条件。

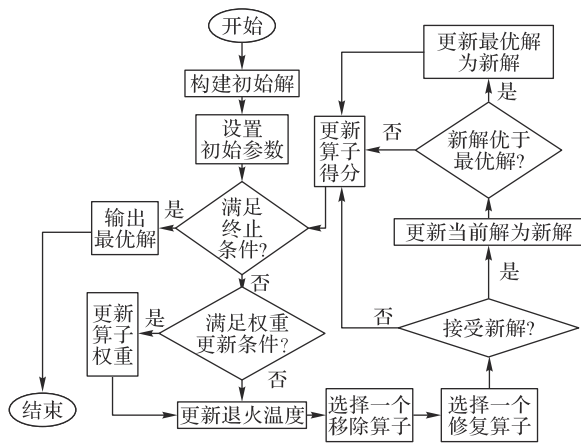


图2 IALNS的流程

Fig. 2 Flow of IALNS

## 2.1 初始解的构造

传统的ALNS在解决VRPTW时,大多采用简单的构造启发式算法如节约里程法<sup>[11]</sup>产生初始解,由于MDVRPTW在空间上更复杂,较合理地划分客户给各车场,对解的质量至关重要,因此,本文改进路径分割算法<sup>[15]</sup>,以产生初始解。

Cheng等<sup>[15]</sup>提出的路径分割算法分为序列分割和路径提取这2部分。序列分割将所有的客户组成一组序列,该序列信息包含客户的需求,在满足车辆的装载容量和客户时间窗的前提下,附带考虑旅行成本,将每个客户分配给相应的车辆。该算法能有效解决单车场问题,但MDVRPTW涉及多个车场,因此在计算旅行成本时,选取哪个车场作为出发点成为亟须考虑的问题。为了尽可能使客户序列段分配给各车场都满足客户时间窗和载重等约束,本文先计算各车场位置的重心,生成虚拟的车场中心,将它作为计算旅行成本的车场,如图3所示。

按照Cheng等<sup>[15]</sup>提出的算法,将车场1作为旅行成本计算的车场,所产生的某客户路径序列为a-b-c-d-e-f,因为头客户a和尾客户f到车场3的距离之和更小,将把此客户序列分配给车场3;然而,车场3到客户a的旅行距离d3a比车场1

到客户a的距离d1a远,它的旅行时间更久,因此,它更适合车场1的客户路径序列,将它分配到车场3可能会违反时间窗约束,导致产生不可行解。为了解决这个问题,产生虚拟车场中心,并由此虚拟车场计算旅行成本重新构建客户序列。算法具体实现步骤为:

1)计算各车场位置的重心,产生的虚拟车场中心作为用于计算客户旅行成本的车场。

2)所有的客户组成一组序列,并为每个客户设置初始的所属车辆标签(路径标签)和车辆服务到此客户产生的旅行成本(示意图如图4所示,其中customer和demand表示客户和需求)。

3)在满足车辆载重容量和客户时间窗的前提下,以旅行成本更小化为依据,通过算法迭代不断更新客户所属车辆(车辆标签),具体参见算法1。

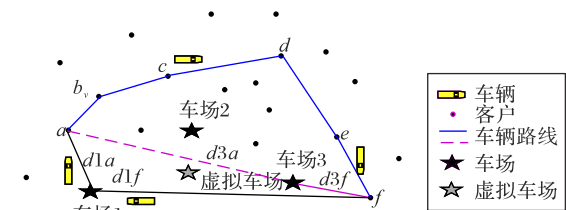
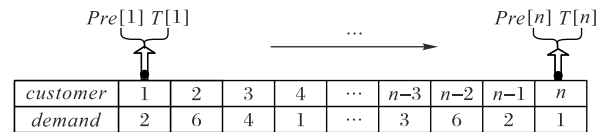


图3 路径分割示意图

Fig. 3 Schematic diagram of path segmentation



注: Pre和T均为列表,分别存放客户的对应车辆标签和车辆服务到此客户产生的旅行成本。

图4 客户序列结构示意图

Fig. 4 Schematic diagram of customer sequence structure

路径提取是指检索上述每个客户对应的车辆标签,将具有相同车辆标签的客户组合在一起,生成客户序列段。

上述两部分操作已使客户序列段满足容量和时间窗限制,接下来考虑将这些客户序列段分配给车场,最终完成完整车辆路径的构建;客户序列段分配给车场的规则是:计算每个客户序列段的头部客户与尾部客户到各车场的旅行成本,选取旅行成本最小的车场,将该客户序列段分配给该车场。若某客户序列段到多个车场的最小旅行成本相同,则按随机方式分配车场。路径分割算法的伪代码如算法1所示。

算法1 路径分割算法。

```

1) Begin
2) //node_id:客户节点集
3) //T:各客户节点到车场的旅行成本;初始设置正无穷大
4) //Pre:客户的路径标签,初始设置为用于计算旅行成本的
   //车场的序号
5) for i←node_id[0] to node_id[-1] do
6)   ni=node_id[i]; cost=0; j=i
7)   while ture do
8)     n2=node_id[j];
9)     if n1=n2 then
10)      cost=2*dist(d, n2)
11)      //dist(d, n2):车场到节点n2的旅行成本
12)     else

```



```

13)    $n_3 = \text{node\_id}[j-1]$ 
14)    $\text{cost} = \text{cost} - \text{dist}(d, n_3) + \text{dist}(n_3, n_2) + \text{dist}(n_2, d)$ 
15)   if 当前所选客户满足时间窗和车辆容量约束 then
16)        $n_4 = \text{node\_id}[i-1]$ 
17)       if  $T[n_4] + \text{cost} \leq T[n_2]$  then
18)            $T[n_2] \leftarrow T[n_4] + \text{cost}$ 
19)            $\text{Pre}[n_2] \leftarrow i-1$ 
20)        $j++$ 
21)   else break
22)   if  $j = \text{len}(\text{node\_id})$  then
23)       break
24)    $\text{route\_list} = \text{path extraction}(\text{node\_id}, \text{Pre})$ 
25)       //path extraction()为路径提取函数
26)   return  $\text{route\_list}$ 
27) End while
28) End for
29) End

```

## 2.2 算子设计

完成初始解的构造后,通过本文算法得到的初始解的优势为满足时间窗约束和车辆容量约束,且得到的客户序列段总旅行成本较低;然而这样做的代价是客户的序列段数可能超过车场拥有的车辆数,造成无法完成车辆派遣任务。由于路径分割算法未考虑车场拥有的车辆数的限制,所以设计启发式算子,通过算法迭代,把路径数调整为车场拥有的车辆数,并使旅行成本尽可能低。启发式算子分为移除算子和修复算子。

### 2.2.1 移除算子

#### 1) 随机移除算子。

随机从当前解中的路径上选取  $q$  个客户节点进行移除,将  $q$  个客户节点放入一个虚拟的请求库,在之后的修复操作中,又将请求库中的客户插入路径。对于随机移除算子,一般得到的移除点集合较差,对提高解的质量帮助可能不大,但它能有助于解的搜索多样性,避免路径陷入局部最优。

#### 2) 最差移除算子。

最差移除算子指移除当前路径的客户节点中对路径旅行成本增加影响最大的客户。移除的客户依旧放入虚拟的请求库,在修复阶段把请求库中的客户插入解的其他位置,以降低当前解的旅行成本。

对于解  $X$  上的客户  $i$ ,定义  $\Delta S_{-i} = C(X) - C_{-i}(X)$ ,其中  $C(X)$  表示当前解  $X$  的旅行成本,  $C_{-i}(X)$  表示假设解  $X$  移除客户  $i$  后的旅行成本,  $\Delta S_{-i}$  表示旅行成本之间的差值。  $\Delta S_{-i}$  按降序排列选择前  $q$  个需要移除的客户放入请求库。

#### 3) 相关移除算子。

如果节点间的相似度不高,将它们移除放入请求库;在修复阶段,这些节点大概率只能插入原先的位置或更差位置,对解的质量提升不大;相似的节点对解的改进可能更大。

定义  $R(i, j)$  表示客户  $i$  和客户  $j$  的相关度:

$$R(i, j) = \alpha \frac{1}{|e_i - e_j| + |l_i - l_j|} + \beta \frac{1}{|q_i - q_j|} + \gamma \frac{1}{|d_i - d_j|} \quad (10)$$

其中:  $[e_i, l_i]$  为客户  $i$  的时间窗;  $q_i$  为客户  $i$  的需求;  $d_i$  为客户  $i$  到车场的平均距离;  $\alpha, \beta, \gamma$  为相应项的权值。  $R(i, j)$  值越高,代表相似性越大。按  $R(i, j)$  的降序排列,选取与要移除的客户  $i$  相似性靠前的前  $q-1$  个客户,连同客户  $i$  一起放入请求库中。

### 2.2.2 修复算子

#### 1) 随机修复算子。

随机修复算子与随机移除算子一样,都是为了保证解的搜索多样性,避免陷入局部最优,具体实现方式为:从请求库中选择一个客户,将它随机插入满足约束的路径,直至把请求库中的客户全部插入。

#### 2) 贪婪修复算子。

贪婪修复算子指将请求库中的客户插入使路径成本增加最小的位置;定义  $\Delta F_{i,k}$  为将客户  $i$  插入路径  $k$  引起的路径旅行成本的变化,若客户  $i$  不能插入路径  $k$ ,则设定  $\Delta F_{i,k} = \infty$ ;定义  $c_i = \min_{k \in K} \{\Delta F_{i,k}\}$ ,  $\mu$  表示请求库中的客户集,则  $\min_{i \in \mu} c_i$  指把请求库中按增加的插入旅行成本最低的客户插入它对应的最佳位置,该过程持续进行,直至把请求库中的客户全部插入或不再有客户能被插入。

#### 3) 遗憾修复算子。

遗憾修复算子旨在弥补贪婪修复算子的劣势。贪婪修复算子每次插入的客户是当前请求库中插入成本增加最少的客户,这会使插入成本高的客户只能靠后插入,然而请求库中的客户越是靠后插入,路径中的客户越是趋于饱和;它的插入路径的可能位置越少,因此,设计遗憾修复算子改进它。

定义  $\Delta F_{i,xik}$  表示将客户  $i$  插入使路径旅行成本增加第  $k$  小的路径,引起路径旅行成本变化;定义遗憾值  $c_i^* = \Delta F_{i,xi2} - \Delta F_{i,xi1}$ ,其中  $\Delta F_{i,xi1}$  即表示贪婪修复算子中的  $c_i$ 。

遗憾修复算子具体实现为找出请求库中的客户  $i$  使得  $c_i^*$  最大(即  $\max_{i \in \mu} c_i^*$ ),然后将该客户  $i$  插入使旅行成本增加最低的位置。

通过适当地扩展启发式,得到遗憾- $k$  修复算子,具体实现为:在请求库中找到客户  $i$  使  $\sum_{j=1}^k (\Delta F_{i,xij} - \Delta F_{i,xi1})$  最大化

$$\left( \text{即 } \max_{i \in \mu} \left\{ \sum_{j=1}^k (\Delta F_{i,xij} - \Delta F_{i,xi1}) \right\} \right), \text{再将它插入路径最佳位置,上}$$

述过程持续进行,直至将请求库中的客户全部插入或不再有客户能被插入。可以这样理解遗憾修复算子,对于遗憾值大的客户如果现在不插入,将来插入会使路径旅行成本增大得更多,从而感到后悔和遗憾。

## 2.3 算子的选择策略和自适应权重调整机制

### 2.3.1 算子的选择策略

2.2 节提出了 3 种移除算子和 3 种修复算子。这些算子对不同的问题实例,甚至对同一问题实例具有不同的适应性,因此,并不单独采用同一种算子,而是交替使用这些移除算子和修复算子,使解的质量更好。为这些算子分配权重,并使用轮盘赌选择原则;设  $Y = \{y|1, 2, \dots, s\}$  表示  $s$  个移除启发式算子的集合,  $F = \{f|1, 2, \dots, s\}$  表示修复启发式算子的集合,各个算子的初始权重相同,用  $w(y)$  和  $w(f)$  表示移除算子和修复算子的权重,它们的概率计算公式为:

$$\begin{cases} p(y) = w(y) / \sum_{j=1}^s w(y) \\ p(f) = w(f) / \sum_{j=1}^s w(f) \end{cases} \quad (11)$$

### 2.3.2 自适应权重调整机制

在算法的迭代过程中,某些算子在对目标值改进得更好的那些迭代阶段的某一阶段可能适合,但在后续阶段可能并不适合,因此,把总迭代次数分段,在每一段中设定 $pu$ 次迭代,在 $pu$ 次迭代后,通过对算子的历史表现动态调整算子权重,指导下一段迭代算子的选取;定义 $w_{i,j}$ 表示算子 $i$ 在第 $j$ 段的权重, $\xi$ 表示反应系数, $s_i$ 表示算子 $i$ 的得分, $t_i$ 表示算子 $i$ 成功运用的次数; $w_{i,j+1} = (1 - \xi)w_{i,j} + \xi(s_i/t_i)$ ;  $s_i$ 的初始值设置为零, $s_i$ 每次增加 $\delta_k$ 个单位; $\delta_k$ 的情况分为3种情形:

- 1) $\delta_1$ 。通过算子找到的解是当前的全局最优解。
- 2) $\delta_2$ 。通过算子找到的解优于当前解。
- 3) $\delta_3$ 。通过算子找到的解劣于当前解,但是被接受准则接受。

上述的第3种情形针对算法的搜索多样性而提出,虽然算子搜索的当前解比已有解更差,但它通过接受准则被接受,且此前该解从未被搜索到过。 $\xi \in [0, 1]$ 为反应系数,它控制着最近成功运用的算子对它的权重的影响; $\xi=0$ 时表示算子权重始终保持不变; $\xi=1$ 表示以往迭代周期的权重与当前权重的更新无关,它仅与当前的迭代周期算子的成功运用有关。为了更好地突出自适应学习的特性,一般设定 $0 < \xi < 1$ 。

### 2.4 新解的接受准则和算法的停止条件

算子在迭代寻找最优解的过程中,若得到的新解优于当前解,则用新解更新当前解,若得到的新解劣于当前解,则以一定的概率选择是否接受得到的新解;采用模拟退火算法的思想生成解的接受准则。Ropke等<sup>[14]</sup>提出的新解的接受概率为:

$$P = e^{-(f(s') - f(s))/T} \quad (12)$$

其中: $f(s')$ 表示新解 $s'$ 的目标函数值, $f(s)$ 表示当前解的目标函数值, $T$ 为模拟退火算法的目前温度值。

本文提出的接受概率为:

$$P = \cos\left(\frac{\pi}{3} \times \frac{f(s') - f(s)}{T}\right) \quad (13)$$

$T$ 的初始值设置为 $\omega f(s_0)$ , $\omega$ 为人为设定的比率; $f(s_0)$ 表示初始解 $s_0$ 的目标函数值,设定 $(f(s') - f(s))/T \in [0, 3/2]$ 。式(13)表示在初始时,新的可行解比初始解差 $\omega$ (比率),也有50%的概率被接受,算法不断迭代,将温度 $T$ 随迭代不断乘以一个恒定的退火速率 $\rho$ , $\rho \in (0, 1)$ ,则温度以 $T = \rho T$ 逐步下降,随着算法的迭代,接受劣解的可能性不断降低,到最后几乎不再接受劣解,使得解的收敛性得到保证。本文采用的算法终止条件是:

- 1)达到迭代次数上限;
- 2)达到运行时间上限。

## 3 实验与结果分析

### 3.1 测试算例

本文选取Cordeau等<sup>[2]</sup>标准算例进行测试,并将测试结果与解决MDVRPTW的几种代表性算法(TS算法<sup>[3]</sup>、改进的蚁群(Ant Colony, ACO)算法<sup>[16]</sup>、遗传算法(Genetic Algorithm, GA)<sup>[17]</sup>、差分进化(Differential Evolution, DE)算法<sup>[18]</sup>和离散粒子群(Discrete Particle Swarm Optimization, DPSO)算法<sup>[19]</sup>)进行对比,验证IALNS的寻优性和鲁棒性。

Cordeau算例可以从网站 <http://www.iescm.com/vrp/instances.asp> 中获取,共有20个MDVRPTW算例,其中算例pr01到pr10的客户时间窗口较窄,车场用于服务的车辆数较多;pr11到pr20的客户时间窗口较宽,车场用于服务的车辆数较少;每个算例中各车场的车辆是同质的,即具有相同的最大载重和最长运行时间。本文实验均在同一实验环境下完成,英特尔i5处理器,2.50 GHz主频,16 GB的RAM,编程环境为Python 3.10.8。

### 3.2 参数设置

为了获取参数的最佳组合情况,在调优阶段选取部分算例测试,各参数的取值利用控制变量法进行测试。经调优后各参数具体取值如下:

相关度函数中 $\alpha$ 、 $\beta$ 、 $\gamma$ 的取值分别为9、3、3;权重反应系数 $\xi=0.4$ ;算子的奖励分数 $\delta_1$ 、 $\delta_2$ 、 $\delta_3$ 经实验发现:当 $\delta_1 \in [28, 35]$ 、 $\delta_2 \in [16, 23]$ 、 $\delta_3 \in [7, 13]$ 时,算法能取得较好的效果,本文设定 $\delta_1=30$ 、 $\delta_2=20$ 、 $\delta_3=10$ ;在解的接受准则中,设定模拟退火初始温度 $T_0 = \omega f(s_0)$ ,其中 $\omega = 0.2$ ;恒定的退火速率 $\rho = 0.9$ ;总迭代次数分为10段即(epoch=10),设置每段中迭代次数 $pu=6$ ;在遗憾- $k$ 修复算子中,本文选取遗憾-5修复算子;实验确定了各参数的最佳取值后,给出IALNS与其他算法求解MDVRPTW的对比实验结果。

### 3.3 实验结果

关于遗憾- $k$ 修复算子中 $k$ 的选取,本文将 $k$ 的不同取值应用于基准算例,评价指标是:每个算子在20次求解该算例得到的结果与该算例最优解之间的平均差距(Avg. gap)、各算子在修复过程产生不可行解的次数,即算子应用于算例失败的次数(Fails),以及算子每次作用于算例的运行时间(Times),实验结果如表1所示。从遗憾-1到遗憾-5,Avg. gap减小,而遗憾-5之后的遗憾- $m(m \geq 6)$ ,Avg. gap增大,综合各项指标,选取遗憾-5修复算子作为本文的遗憾修复算子。

表1 遗憾- $k$ 修复算子性能

Tab. 1 Performance of regret- $k$  repair operator

算子类型	Avg. gap/%	Fails	Times/s
遗憾-1	36.2	4	0.02
遗憾-2	29.3	3	0.02
遗憾-3	27.6	3	0.02
遗憾-4	26.4	3	0.02
遗憾-5	25.7	2	0.02
遗憾- $m$	27.8	0	0.03

权重反应系数 $\xi$ 决定自适应学习的方向,控制着以往段的权重信息对当前段的权重信息的影响。它的取值对算法的效率起着重要作用,设计实验进行探讨结果如表2所示。它的评价指标为求解结果与最优解的平均差距Avg. gap、算法在迭代过程中产生未改进解(陷入局部最优)的次数(Local Optimal Times, LOT)。

从表2可见, $\xi$ 取值太小或太大都会增大算法陷入局部最优的可能性,求解结果与最优解之间的平均差距Avg. gap也会随之增大,因为把以往迭代段的算子权重的重要性程度考虑得太大或太小,使当前迭代段与以往迭代段无法实现好的平衡,当前迭代段适应度较好的算子的选择概率受到影响,求解效率和质量较差。综合各项指标,选取 $\xi=0.4$ 作为本文的权重反应系数。

表2 权重反应系数 $\xi$ 的测试结果  
Tab. 2 Test results of weight response coefficient  $\xi$

$\xi$	Avg. gap/%	LOT	$\xi$	Avg. gap/%	LOT
0.1	22.4	14	0.6	17.2	10
0.2	16.8	11	0.7	19.3	11
0.3	12.6	7	0.8	18.0	10
0.4	5.2	4	0.9	24.3	16
0.5	13.5	5			

模拟退火速率 $\rho$ 从一定程度上决定着算法的收敛快慢,它的设定不能太小,否则算法收敛太快,接受劣解的可能性降低,算子搜索多样性减弱,从而增大落入局部最优的可能性。本文探讨 $\rho$ 的较大取值,实验在pr01规范算例上测定,确定 $\rho$ 的较好取值,相关实验结果如图5所示。由图5可见,算法在迭代周期的前期对解的改进较大,解的质量在短时间内获得较大的提升,从一定程度上说明了设计的算子求解该问题的高效性;另一方面,当 $\rho = 0.9$ 时,算法在迭代周期的中后期收敛,且整体收敛态势与其他取值相比较平缓,说明它充分考虑了算子搜索多样性与算法收敛速率的平衡。因此本文选取 $\rho = 0.9$ 作为模拟退火的速率。

算法参数调试完成,将IALNS与该领域代表性研究成果进行对比,自Cordeau规范算例后,某些算例的最优解被后来的研究者更新,Polacek等<sup>[4]</sup>在之后设计的VNS算法更新了

部分算例的最优解,具有一定的参考价值,Luo等<sup>[10]</sup>提出的多相位改进的蛙跳算法(Multi-Phase Modified Shuffled Frog Leaping Algorithm, MPMSFLA)求解此问题也取得较好的效果,因此对比IALNS与VNS、MPMSFLA算法求解的结果。

确定各实验参数后,用本文算法求解规范Cordeau算例,并与该领域的代表性算法求解结果对比,实验结果如表3所示。

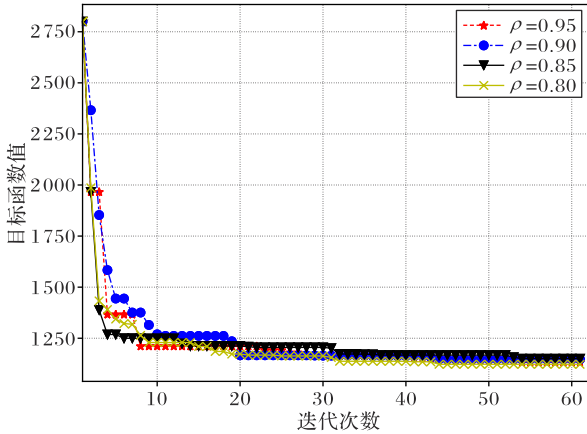


图5 模拟退火速率 $\rho$ 对算法收敛速率的影响

Fig. 5 Influence of simulated annealing rate  $\rho$  on algorithm convergence rate

表3 IALNS与其他算法实验结果对比

Tab. 3 Comparison of experimental results between IALNS and other algorithms

Instance			TS		VNS		MPMSFLA		IALNS		
name	$N$	$D$	$KBS$	$TIME/min$	$AVE$	$TIME/min$	$AVE$	$TIME/min$	$AVE$	$TIME/min$	$GAP/\%$
pr01	48	4	1 074.12	28	1 074.12	9.49	1 074.34	<b>0.45</b>	1 074.28	0.39	0.01
pr02	96	4	1 762.48	79	1 763.66	27.63	1 768.23	1.42	1 766.41	0.68	0.16
pr03	144	4	2 397.06	115	2 388.73	75.88	2 378.23	2.96	2 399.52	3.15	0.45
pr04	192	4	2 865.71	144	2 847.56	93.53	2 858.35	4.21	2 852.41	3.26	0.17
pr05	240	4	3 050.80	181	3 015.27	89.36	3 021.56	4.89	3 027.97	3.95	0.42
pr06	288	4	3 670.13	221	3 674.60	96.63	3 685.28	<b>6.33</b>	3 686.14	<b>6.85</b>	0.31
pr07	72	6	1 418.22	53	1 418.22	7.66	1 418.78	1.89	1 421.21	1.72	0.21
pr08	144	6	2 118.50	102	2 103.21	54.92	2 111.47	3.07	2 110.36	2.87	0.34
pr09	216	6	2 760.46	160	2 753.61	69.11	2 770.61	3.85	2 760.23	2.96	0.24
pr10	288	6	3 507.26	227	<b>3 541.01</b>	65.70	3 555.86	6.97	<b>3 534.27</b>	6.38	<b>-0.19</b>
pr11	48	4	1 016.59	32	1 011.65	18.30	1 009.70	0.48	1 019.35	0.35	0.76
pr12	96	4	1 486.26	81	<b>1 488.32</b>	89.40	1 489.11	1.90	<b>1 487.15</b>	1.36	<b>-0.08</b>
pr13	144	4	2 028.85	143	2 012.37	82.30	2 026.47	3.25	2 025.65	2.59	0.66
pr14	192	4	2 228.64	188	2 239.02	106.51	2 242.89	4.16	2 251.23	3.29	0.55
pr15	240	4	2 527.60	227	2 498.85	89.39	2 508.28	4.88	2 503.14	3.15	0.17
pr16	288	4	2 960.93	261	2 909.45	99.69	2 930.14	<b>6.86</b>	2 918.35	<b>6.92</b>	0.31
pr17	72	6	1 241.25	61	1 247.51	62.15	1 240.19	1.60	1 251.31	1.36	0.30
pr18	144	6	1 823.24	146	1 809.25	99.24	1 815.74	3.07	1 814.35	2.56	0.28
pr19	216	6	2 288.38	262	2 294.19	90.84	2 300.65	4.29	2 310.12	3.88	0.69
pr20	288	6	3 120.32	263	3 093.51	86.38	3 109.12	6.64	3 107.25	6.12	0.44

表3中 $N$ 代表客户数, $D$ 代表车场数, $KBS$ 表示Cordeau算例设计时已知最优解, $AVE$ 表示相应算法求解该问题20次的平均解, $TIME$ 表示算法的平均求解时间, $GAP$ 表示将本文求解结果与VNS算法比对后的相对误差率(Polacek等<sup>[4]</sup>提出的VNS算法更新了Cordeau算例的部分最优解)。通过将IALNS与TS算法、VNS和Luo等<sup>[10]</sup>提出的MPMSFLA的求解结果对比发现所提算法能较好地求解MDVRPTW,算法在大多数算例上求解结果和已知最优解十分接近,相对误差率不

超过0.8%,在部分算例(pr10、pr12)上甚至更优。

另一方面,MPMSFLA也能较好地求解MDVRPTW且算法的耗时均比TS算法和VNS算法少。在某些算例上本文算法求解结果劣于MPMSFLA,但与之相对误差也非常小,IALNS的优势是绝大多数算例(除pr06、pr16)上求解时间比MPMSFLA算法短。

为了凸显本文算法求解速率的优越性,将2.1节的路径分割算法构建初始解后,分别应用于其他智能算法(TS<sup>[3]</sup>、



ACO<sup>[16]</sup>、GA<sup>[17]</sup>、DE<sup>[18]</sup>和DPSO<sup>[19]</sup>) 在Cordeau规范算例pr01上进行测试,各算法的实验结果如图6所示。

由图6可见,IALNS收敛速度<sup>[20]</sup>最高,收敛结果最好。在设定的迭代周期中,DE算法甚至几乎未改进解的质量,虽然它的单次迭代耗时比IALNS短,在相同的迭代周期<sup>[21]</sup>内稍有失偏颇,但后续的实验发现DE算法至少需要2000次迭代才能达到IALNS相似的结果,算法总耗时远超本文算法;其他对比算法单次迭代耗时与IALNS单次迭代耗时大致持平,因此,在相同的迭代周期中比较它们的收敛速率<sup>[22]</sup>是合理的,本文算法可以高效求解此问题。

为了测试算法的稳定性<sup>[23]</sup>,将IALNS应用于不同规模的问题算例研究,宽时间窗口的算例:pr01、pr03、pr05;窄时间窗口算例:pr12、pr18、pr20。实验结果如图7所示。从图7中可见,IALNS无论是求解时间窗宽窄亦或是客户数多少的算例,都能

取得较好的结果,求解效率和质量并未受问题规模影响,仿真实验结果说明本文算法具有较强的稳定性。

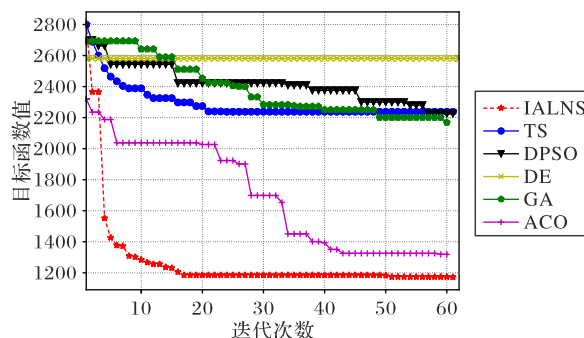


图6 IALNS与其他算法收敛速率对比

Fig. 6 Comparison of convergence rate between IALNS and other algorithms

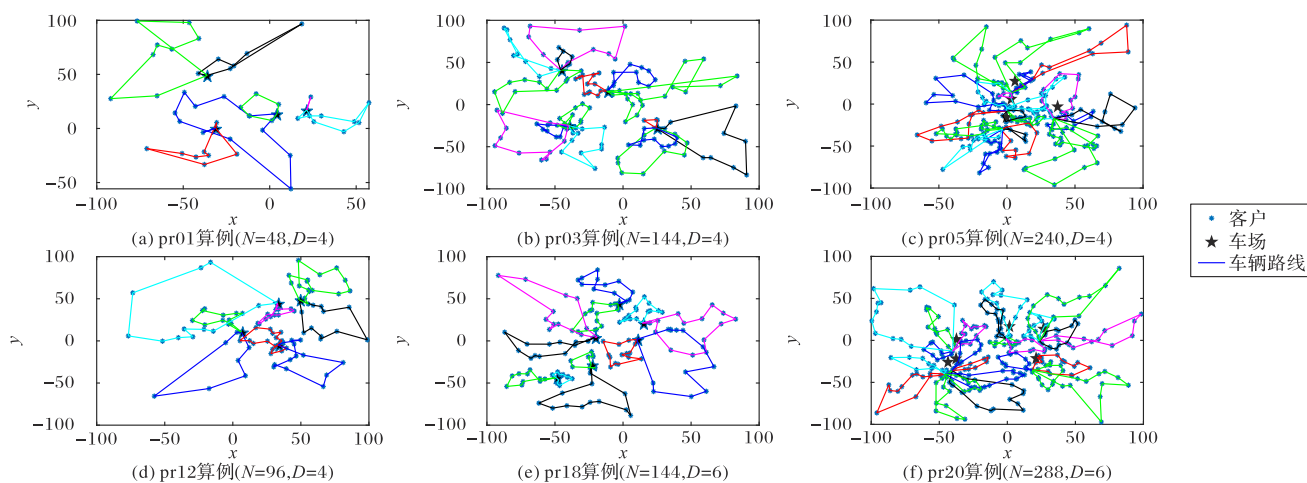


图7 IALNS求解部分Cordeau算例结果

Fig. 7 Solution results of part Cordeau instances by IALNS

## 4 结语

MDVRPTW在生活中应用广泛,研究它有助于提高资源利用率、增大企业利润和客户满意度,同时降低碳排放等。求解此问题较难,本文针对此问题提出一种IALNS,将该算法在Cordeau规范算例<sup>[2]</sup>上实验,并将求解结果与该领域的代表性研究成果进行比较。实验结果表明,本文算法能高效快速求解MDVRPTW,具备较强的鲁棒性<sup>[24]</sup>,是一种求解MDVRPTW的有效算法。

本文在生成初始解阶段,提出一种路径分割算法,产生的客户序列片段可能会超出车场拥有的车辆数,程序的具体实现是预先将车场的车辆数设置较大以满足客户序列段的分配,最后再调试参数并改进解<sup>[25]</sup>,使车辆路径数与实际所拥有车辆数匹配;因此,未来如何构造更有效的初始解是值得重视的;另外,为提高算法的运行时间和算子的搜索效率,如何构造更高效的邻域搜索算子,并设计更高效的算法框架也是未来研究工作的一个重点。

## 参考文献 (References)

- [1] DANTZIG G B, RAMSER J H. The truck dispatching problem[J]. Management Science, 1959, 6(1):80-91.
- [2] CORDEAU J-F, LAPORTE G, MERCIER A, et al. A unified tabu search heuristic for vehicle routing problems with time windows [J]. Journal of the Operational Research Society, 2001, 52:

928-936.

- [3] CORDEAU J-F, LAPORTE G, MERCIER A. Improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows [J]. Journal of the Operational Research Society, 2004, 55: 542-546.
- [4] POLACEK M, HARTL R F, DOERNER K, et al. A variable neighborhood search for the multi depot vehicle routing problem with time windows [J]. Journal of Heuristics, 2004, 10: 613-627.
- [5] DONDO R, CERDÁ J. A cluster-based optimization approach for the multi- depot heterogeneous fleet vehicle routing problem with time windows [J]. European Journal of Operational Research, 2007, 176: 1478-1507.
- [6] 洪联系,董绍华. MDVRPTW问题多阶段迭代启发式算法[J]. 计算机工程与应用, 2007, 43(26): 217-222. (HONG L X, DONG S H. Multi-phase iterative heuristic approach for MDVRPTW [J]. Computer Engineering and Applications, 2007, 43 (26) : 217-222. )
- [7] 王征,张俊,王旭坪. 多车场带时间窗车辆路径问题的变邻域搜索算法[J]. 中国管理科学, 2011, 19(2): 99-109. (WANG Z, ZHANG J, WANG X P. A modified variable neighborhood search algorithm for the multi depot vehicle routing problem with time windows [J]. Chinese Journal of Management Science, 2011, 19 (2):99-109. )
- [8] GENDREAU M, HERTZ A, LAPORTE G, et al. A generalized insertion heuristic for the traveling salesman problem with time

- windows [J]. *Operations Research*, 1998, 46(3): 330-335.
- [9] 于滨, 靳鹏欢, 杨忠振. 两阶段启发式算法求解带时间窗的多中心车辆路径问题[J]. *系统工程理论与实践*, 2012, 32(8): 1793-1800. (YU B, JIN P H, YANG Z Z. Two-stage heuristic algorithm for multi-depot vehicle routing problem with time window [J]. *System Engineering — Theory & Practice*, 2012, 32(8): 1793-1800.)
- [10] LUO J, CHEN M. Multi-phase modified shuffled frog leaping algorithm with extremal optimization for the MDVRP and the MDVRPTW[J]. *Computers & Industrial Engineering*, 2014, 72: 84-97.
- [11] TAMASHIRO H, NAKAMURA M, OKAZAKI T, et al. A tabu search approach combined with an extended saving method for multi-depot vehicle routing problems with time windows (TOTAL OPERATIONS MENAGEMENT) [J]. *International Journal of Biomedical Soft Computing and Human Sciences: the Official Journal of the Biomedical Fuzzy Systems Association*, 2010, 15(1): 29-37.
- [12] SADATI M E H, ATAY B, AKSEN D. An efficient variable neighborhood search with tabu shaking for a class of multi-depot vehicle routing problems [J]. *Computers and Operations Research*, 2021, 133: 105269.
- [13] 刘小兰, 郝志峰, 汪国强, 等. 有时间窗的车辆路径问题的近似算法研究[J]. *计算机集成制造系统*, 2004, 10(7): 825-831. (LIU X L, HAO Z F, WANG G Q, et al. Improved large neighborhood search algorithm for vehicle routing problem with time windows [J]. *Computer Integrated Manufacturing Systems*, 2004, 10(7): 825-831.)
- [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] CHENG L. A genetic algorithm for the vehicle routing problem with time windows[D]. *Wilmington, North Carolina: University of North Carolina Wilmington*, 2009: 105-121.
- [16] 陈小玲. 蚁群算法及其在车辆路径问题中的应用研究[D]. 成都: 电子科技大学, 2009: 16-34. (CHEN X L. Ant colony algorithm and its application research in vehicle routing problems [D]. *Chengdu: University of Electronic Science and Technology of China*, 2009: 16-34.)
- [17] CHEN C-M, LV S, NING J, et al. A genetic algorithm for the waitable time-varying multi-depot green vehicle routing problem [J]. *Symmetry*, 2023, 15(1): 124.
- [18] 吕焱远. 混合差分进化算法在多目标车辆路径问题中的应用[D]. 太原: 太原科技大学, 2021: 23-42. (LU Y Y. Application of hybrid differential evolution algorithm in multi-objective vehicle routing problem [D]. *Taiyuan: Taiyuan University of Science and Technology*, 2021: 23-42.)
- [19] 周蓉, 沈维蕾. 软硬时间窗共存装卸一体化车辆路径问题的混合离散粒子群优化算法[J]. *合肥工业大学学报(自然科学版)*, 2016, 39(8): 1022-1026. (ZHOU R, SHEN W L. Hybrid discrete particle swarm optimization algorithm for vehicle routing problem with simultaneous delivery and pickup under coexistence of soft and hard time windows [J]. *Journal of Hefei University of Technology (Natural Science)*, 2016, 39(8): 1022-1026.)
- [20] WANG Y, WEI Y, WANG X, et al. A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints [J]. *Applied Soft Computing*, 2023, 133: 109922.
- [21] FAIED M, MOSTAFA A, GIRARD A. Dynamic optimal control of multiple depot vehicle routing problem with metric temporal logic [C]// *Proceedings of the 2009 American Control Conference*. Piscataway: IEEE, 2009: 3268-3273.
- [22] RAJMOHAN M, SHAHABUDEEN P. Metaheuristic for solving routing problem in logistics management [J]. *International Journal of Operational Research*, 2009, 6(2): 223-246.
- [23] 向婷, 潘大志. 求解需求可拆分车辆路径问题的聚类算法[J]. *计算机应用*, 2016, 36(11): 3141-3145. (XIANG T, PAN D Z. Clustering algorithm for split delivery vehicle routing problem [J]. *Journal of Computer Applications*, 2016, 36(11): 3141-3145.)
- [24] MUNARI P, SAVELSBERGH M. A column generation-based heuristic for the split delivery vehicle routing problem with time windows [J]. *SN Operations Research Forum*, 2020, 1: 26.
- [25] FONTAINE P. The vehicle routing problem with load-dependent travel times for cargo bicycles [J]. *European Journal of Operational Research*, 2022, 300: 1005-1016.

This work is partially supported by National Natural Science Foundation of China (11871059); Talent Research Fund of China West Normal University (17YC385).

**LI Yan**, born in 1997, M. S. candidate. His research interests include intelligent computing, combinatorial optimization.

**PAN Dazhi**, born in 1974, Ph. D., professor. His research interests include intelligent computing, algorithm design.

**ZHENG Siqing**, born in 2000, M. S. candidate. Her research interests include uniform convex optimization.