DOI:10.13196/j.cims.2020.07.019

时间依赖型同时取送货 VRP 及超启发式算法

张景玲,刘金龙,赵燕伟,王宏伟,冷龙龙,冯勤炳
(浙江工业大学 特种装备制造与先进加工技术教育部重点实验室,浙江 杭州 310014)

摘 要:为有效地协调正逆向物流,更好地体现城市物流配送速度时变的特点,降低物流配送成本,以时间依赖型同时取送货车辆路径问题为对象,建立其数学规划模型;设计了基于禁忌搜索的超启发式算法对其进行求解。在算法高层,设计了基于禁忌搜索评分制的选择策略及模拟退火的接收准则,实时监控底层启发式算子的性能并选择最优算子。通过基准实例测试及实验对比分析,表明了该算法能快速地找到满意解,所设计高层策略能保证算法跳出局部最优并快速收敛,从而证明了所提算法求解该问题的有效性。

关键词:车辆路径问题;同时取送货;时间依赖网络;超启发式算法;禁忌搜索
中图分类号:TP301.6;U116.2 **文献标识码**:A

Hyper-heuristic for time-dependent VRP with simultaneous delivery and pickup

ZHANG Jingling, LIU Jinlong, ZHAO Yanwei, WANG Hongwei, LENG Longlong, FENG Qinbing
(Key Laboratory of special equipment manufacturing and advanced processing technology, Ministry of Education, Zhejiang University of Technology, Hangzhou 310014, China)

Abstract: To harmonize the relationship between logistics and reverse logistics and to describe the characteristics of vehicle with variable speed more specifically in urban logistics, a model of Time Dependent Vehicle Routing Problem with Simultaneous Delivery and Pickup (TDVRPSDP) was built. Aiming at reducing the costs of logistics activity, a Hyper-Heuristic (HH) algorithm was designed to solve TDVRPSDP. The proposed algorithm utilized Tabu Search (TS) as select strategy and Simulated Annealing (SA) as acceptance criteria (AC) in high-level heuristic to improve the performance of hyper-heuristic framework by timely and rapidly monitoring the performance information of Low-Level Heuristics (LLH) and choosing the best one according to its performance before. Simulation results and comparisons showed that the proposed algorithm was effective to solve TDVRPSDP within reasonable computing time. In addition, the ability to jump out local optimum and speed up optimizing of strategies designed had been proved by the test of benchmark instance and comparison of result from predecessor.

Keywords: vehicle routing problem; simultaneous delivery and pickup; time dependent network; hyper-heuristics algorithm; tabu search

0 引言

在当今激烈的市场竞争中,越来越多的企业趋向为客户提供全生命周期的产品及服务,如家电、食品、汽车行业,这就要求企业在保证配送时效的同时,对客户点进行配送与回收服务。传统车辆路径问题(Vehicle Routing Problem, VRP)的研究不能描述城市物流配送过程中同时取送货的特点,越来越多的学者转向对其分支问题——同时取送货的车辆路径问题(Vehicle Routing Problem with Simultaneous Delivery and Pickup, VRPSDP)进行研究。

Min[1]于 1989 年首次提出 VRPSDP 用于解决

收稿日期:2018-08-29;修订日期:2018-12-06。Received 29 Aug. 2018; accepted 06 Dec. 2018.
基金项目:国家自然科学基金资助项目(61402409,51875524);浙江省自然科学基金资助项目(LY19F030017)。**Foundation items**: Project supported by the National Natural Science Foundation, China(No. 61402409, 51875524), and the Natural Science Foundation of Zhejiang Province, China(No. LY19F030017).

公共图书馆问题；Poonthalir 等[2]从路径最短的角度对混合回程的车辆路径问题进行建模，并考虑车辆巡航以及空载时燃油消耗及碳排放；Ninikas 等[3]研究了一种动态 VRPSDP，在执行预制定的配送方案过程中实时接收动态的取货请求；Lin 等[4]提出一个集成了混合邻域搜索算法的决策支持系统(Decision Support System，DSS)原型来解决离线和在线需求的动态车辆路径问题；Hu 等[5]研究了货物不兼容情况下，具有不确定性送货和确定性取货的动态闭环 VRP；Wang 等[6]等以降低人力成本、运输成本以及提高客户满意度为目标，采用邻域搜索算法对带软时间窗的 VRPSDP 进行多目标优化；王超等[7]提出一种离散布谷鸟算法(Discrete Cuckoo Search，DCS)算法，以最小化分配成本与行驶成本之和为目标对硬时间窗的 VRPSDP 进行建模并求解。上述研究很少涉及时间依赖型的 VRPSDP，现有时间依赖型车辆路径问题的研究多集中在无取货的 VRP。Malandraki 等[8]对时间依赖型车辆路径问题(Time Dependent Vehicle Routing Problem，TDVRP)做出了具体描述，首次提出了三段的行程速度分段函数，将 VRP 扩展为 TDVRP，揭开了对时间依赖型问题的研究的序幕；Figliozzi[9]分析了市区的早晚交通高峰，引入了 1～2.5 的旅行速度，使得 TDVRP 更具有实际的应用价值；穆东等[10]在主从式并行模拟退火算法框架下，使用 4 种邻域搜索法求解 TDVRP，优化目标为最小化配送车辆数量与总行驶距离；Zhang 等[11]将行程速度分段函数引入 VRPSDP，并设计了集成蚁群与禁忌搜索的新型算法(Ant Colony System and Tabu Search algorithms，ACS-TS)求解该问题。现有 VRPSDP 研究假设车辆匀速行驶，忽略了车速对配送活动的影响，导致 VRPSDP 模型不能很好地描述城市物流配送速度时变的特点，因此本文研究时变车速对配送成本的影响，建立了时间依赖型同时取送货车辆路径问题(Time Dependent Vehicle Routing Problem with Simultaneous Delivery and Pickup，TDVRPSDP)模型。

超启发式(Hyper-Heuristics，HH)算法因其具有一定的通用性可用来求解不同领域的组合优化问题[12]，获得了越来越多研究者的关注。HH 算法框架包含高层启发式策略(High-Level Heuristic，HLH)和底层启发式算子(Low-Level Heuristics，LLH)两个层次。HLH 包含 LLH 的选择策略(Selection)以及解的接受准则(acceptance criterion)。通过 HLH 管理或操纵一系列 LLH 以选择和产生新的启发式算法对解空间进行搜索。Chen 等[13]提出了基于蚁群的 HH 算法用于解决锦标赛问题，采用蚁群算法来管理和操纵 LLH 以获得新的启发式算法；Burke 等[14]针对教育时间表问题提出了基于禁忌搜索的 HH 算法，采用禁忌搜索(Tabu Search，TS)方法来获得新的 LLH 的组合而成的新的启发式算法；Dokeroglu 等[15]提出了一种集成模拟退火、禁忌搜索和蚁群优化算法寻找最优解的 HH 算法去解决二次分配的问题(Quadratic Assignment Problem，QAP)；Sabar 等[16]使用一种基因表达编程算法，根据解的评价自动选择高层启发算法控制 LLH 搜索最优解；Zamli 等[17]提出一种集成的 HH 算法，采用禁忌搜索作为高级元启发式和 4 种低级元启发式的强度，包括基于教学优化、全局邻域算法、粒子群算法和布谷鸟搜索算法选择最适合 LLH；Asta 等[18]提出多级选择的超启发算法，且引入了学习机制；Özcan 等[19]使用群体决策策略作为超启发算法的移动接受准则，且每次迭代的接受准则不限于一种。HH 算法不但可求解问题种类广泛，框架设计灵活多样，而且求解组合优化问题时，可以在较短的时间获得可接受解。冷龙龙等[20]以量子进化策略作为超启发式算法的高层学习策略，并结合滑动窗口机制实现底层算子的准确搜索，提高算法性能。

鉴于超启发式算法在求解组合优化问题上的优良性能，本文提出了基于禁忌搜索的超启发式算法，将禁忌搜索机制作为高层选择策略用于搜索 LLH 空间，根据 LLH 历史表现进行评分并逐代更新禁忌表，每次迭代选取最佳算子对解空间进行搜索，从而实现高层策略对底层算子的有效控制。

1 时间依赖型同时取送货的车辆路径问题

1.1 问题描述

TDVRPSDP 描述如下：给定一个配送中心，一个车辆集合，一个客户点集合。车辆从配送中心出发为同一城市不同位置的客户点进行配送与回收服务。配送过程中，车辆按顺序依次访问各客户点，车辆到达客户点进行配送的同时完成客户点的回收取货需求。车辆应在客户允许的时间范围内提供服务，完成客户点的访问任务后最终返回配送中心。所有车辆具有相同的容量，且配送车辆在不同的配

送时间段有不同的旅行速度。当车辆早于客户点的时间窗要求到达该客户点时,配送车辆进行等待,产生惩罚成本计入总成本。

构建模型要求满足以下条件,模型参数如表 1 所示。

(1)装载量限制。在任何时刻,车辆的载重量不得超过车辆的最大载重。

(2)车辆路线约束。车辆从配送中心出发,完成配送后回到配送中心。

(3)节点约束。车辆进入某个客户节点,也必须从该节点离开。

(4)配送车辆服务约束。一辆车可以为多个客户服务,但一个客户点只能被一辆车服务。

(5)客户点时间窗约束。配送车辆必须在客户点的时间窗要求内访问该客户点,提前到达客户点进行等待产生惩罚成本。

表 1　参数符号定义

变量	定义
p_i	客户 i 的配货量
q_i	客户 i 的集货量
U	$U=\{0,1,2,\cdots,n\}$配送网络中节点集合
D	$D=\{0\}0$:配送中心
S	$S=\{1,2,\cdots,n\}1,2,\cdots,n$:客户
$[e_0,l_0]$	配送中心开放时间
$[e_i,l_i]$	客户 i 的时间窗要求
d_{ij}	客户点 i 到客户点 j 之间的距离
Q	配送车辆最大容量
V	车辆集合 $V=\{k\},k=1,2,\cdots,K$
R	配送中心开放时间分段数量
W	时间间隔 $W=\{r\},r=1,2,\cdots,R$
C_F	每辆车的租赁成本
C_T	车辆单位时间使用成本
C_W	车辆的惩罚成本
v_{ij}^r	第 r 时间段客户点 i 到 j 的车辆速度
T_{ij}^r	第 r 时间段客户点 i 到 j 的行驶时间
L_{ij}^k	车辆 k 由客户点 i 到 j 的载重
t_{ik}	车辆 k 到达客户点 i 的时刻
S_r	第 r 时间段的开始时刻
E_r	第 r 时间段的结束时刻
x_{ijk}^r	车辆 k 在第 r 时间间隔经过弧(i,j)时为 1,否则为 0

1.2　时间依赖型路网

TDVRPSDP 是研究车辆旅行速度随时间变化的一类车辆路径问题,同样属于 NP-Hard 问题,其求解更为复杂。本文采用了 Zhang 等[11]的行程速度分段函数,将总配送时段分为早高峰(morning Rush hour)、中午平峰(non-rush hour)、晚高峰(evening rush hour)三段,且三时段时长相同,每个时段车速不变,行程距离随不同的旅行速度线性变化,如图 1 所示。

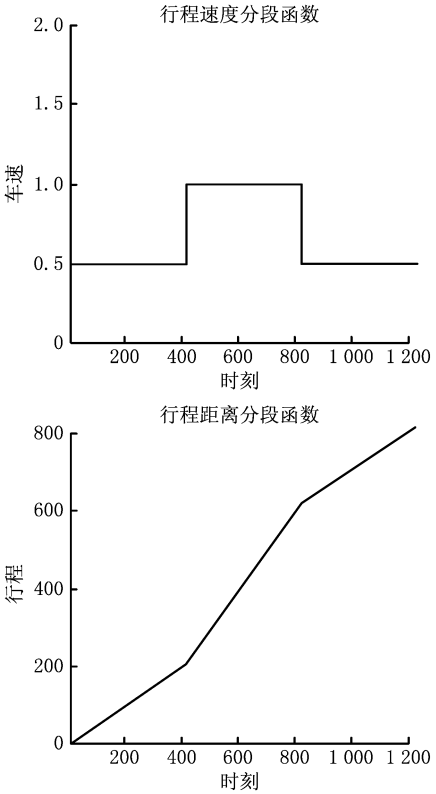

图1　行程速度、行程距离分段函数

本文对 TDVRPSDP 的目标函数值提出如下计算方法,具体步骤为:

步骤 1　将配送时间段$[e_0,l_0]$以时间一定的间隔等分为 R 个时段,每个时段上对应于车辆的行驶速度,即分段函数形式的速度时间函数。

步骤 2　判断配送车辆离开客户点 i 后向客户点 j 出发时刻 t_{ik} 所处的时间段$[t_{r-1},t_r]$,该时间段的速度 $v_{ij}^1=v_r,(r\in R)$。在时间间隔$[t_{r-1},t_r]$的运行距离为 $d_{ij}^1=(t_r-t_{ik})\cdot v_{ij}^1$。

步骤 3　根据 d_{ij},计算车辆从客户点 i 离开到达客户 j 时跨越的 w 个时间段。若 $\Delta T\cdot v_{r+1}+\cdots+\Delta T\cdot v_{r+m+1}\leqslant d_{ij}-d_{ij}^1\leqslant\Delta T\cdot v_{r+1}+\cdots+\Delta T\cdot v_{r+m+2}$,则 $w=m+2$。

因此,得到车辆从客户点 i 离开到达客户 j 时跨越的 w 个时间段的速度、行驶距离以及配送时间为:

$$v_{ij}=(v_{ij}^1,v_{ij}^2,\cdots,v_{ij}^w)=(v_r,v_{r+1},\cdots,v_{r+w-1}),\tag{1}$$

$$(d_{ij}^1,d_{ij}^2,\cdots,d_{ij}^w)=((t_r-t_{ik})\cdot v_r,$$

$$\Delta T\cdot v_{r+1},\cdots,\Delta T\cdot v_{r+w-2},d_{ij}-\sum_{x=1}^{w-1}d_{ij}^x),\tag{2}$$

$$T_{ij}=\sum_{x=1}^{w}(d_{ij}^r/v_{ij}^x),\tag{3}$$

$$d_{ij}=\sum_{x=1}^{w}d_{ij}^x。\tag{4}$$

步骤 4　分段带入计算车辆旅行成本以及车辆等待的惩罚成本。车辆离开客户点 i 到达客户点 j 的旅行成本以及等待时间惩罚成本为

$$C_T\sum_{r\in W}(T_{ij}^r\cdot x_{ijk}^r)+C_W\sum_{r\in W}(x_{ijk}^r\cdot\max\{e_j-t_{ik}-T_{ij}^r,0\})。\tag{5}$$

1.3　数学模型

本文采用文献[11]所建立的 TDVRPSDP 模型，其目标函数是总成本最小，包括车辆租赁成本、旅行成本以及惩罚成本。数学模型如下：

$$\min C=C_F\sum_{k\in V}\sum_{i\in D}\sum_{j\in S}\sum_{r\in W}x_{ijk}^r+C_T\sum_{k\in V}\sum_{i\in U}\sum_{j\in U}\sum_{r\in W}(T_{ij}^r\cdot x_{ijk}^r)+C_W\sum_{k\in V}\sum_{i\in U}\sum_{j\in U}\sum_{r\in W}(x_{ijk}^r\cdot\max\{e_j-t_{ik}-T_{ij}^r,0\})。\tag{6}$$

$$\text{s.t.}\quad S_r\cdot x_{ijk}^r\leqslant x_{ijk}^r\cdot t_{ik}\leqslant E_r\cdot x_{ijk}^r,\forall i,j\in U,\forall k\in V,\forall r\in W;\tag{7}$$

$$0\leqslant L_{ijk}\leqslant Q,\forall i,j\in U,\forall k\in V;\tag{8}$$

$$\sum_{i\in D}\sum_{j\in S}L_{ijk}=\sum_{i\in U}\sum_{j\in U}\sum_{r\in W}x_{ijk}^r\cdot q_j,\forall k\in V;\tag{9}$$

$$\sum_{i\in D}\sum_{j\in S}L_{ijk}=\sum_{i\in U}\sum_{j\in U}\sum_{r\in W}x_{ijk}^r\cdot p_j,\forall k\in V;\tag{10}$$

$$e_i\leqslant t_{ik}\leqslant l_i,\forall i\in U,\forall k\in V;\tag{11}$$

$$\sum_{i\in U}\sum_{k\in V}\sum_{r\in W}x_{ijk}^r=1,\forall j\in D;\tag{12}$$

$$\sum_{i\in U}\sum_{r\in W}x_{ijk}^r-\sum_{i\in U}\sum_{r\in W}x_{jik}^r=0,\forall j\in D,\forall k\in V;\tag{13}$$

$$\sum_{j\in S}\sum_{r\in W}x_{ijk}^r\leqslant 1,\forall i\in U,\forall k\in V;\tag{14}$$

$$t_{jk}=\sum_{r\in W}x_{ijk}^r\cdot(t_{ik}+T_{ij}^r+\max\{e_j-t_{ik}-T_{ij}^r,0\}),\forall i,j\in U,\forall k\in V;\tag{15}$$

$$\sum_{r\in W}\sum_{i\in S}x_{ijk}^r\cdot(L_{ij}^k-q_j)=\sum_{r\in W}\sum_{i\in S}x_{jik}^r\cdot(L_{ij}^k-p_j),\forall j\in U,\forall k\in V。\tag{16}$$

其中：式(6)为 TDVRPSDP 的目标函数，表示总成本最小化，包括车辆租赁成本、车辆旅行成本以及车辆等待的惩罚成本；约束(7)保证了客户在特定的时间间隔内被访问；约束(8)配送车辆在任意的时刻都满足容量约束；约束(9)车辆出发时的载重等于所有要访问的客户点的需求量总和；约束(10)车辆回到配送中心的时的载重量等于所有已访问的客户点的回收量的总和；约束(11)车辆访问客户点的时间窗约束；约束(12)保证每个客户点必须且仅仅被访问一次；约束(13)保证每条路径，从配送中心出发回到配送中心的是同一辆车；约束(14)保证每辆车仅被使用一次；约束(15)保证了车辆从客户点 i 出发到到达客户点 j 的时间等于车辆旅行时间加上等待时间；约束(16)保证配送车辆访问客户点 i 前后载重的变化等于需求量减去回收量。

2　基于禁忌搜索的超启发式算法设计

超启发式算法通过管理或操纵一系列 LLH，以选择和产生新的启发式算法对解空间进行搜索。因此，算法的关键是如何设计高层策略和底层算子。本文提出的基于禁忌搜索的超启发式算法将从以下 5 个方面开展研究：①初始解的构成；②底层启发式算子设计；③算法高层接收准则及选择策略设计；④超启发式算法框架设计；⑤算法复杂性的计算。

2.1　初始解的构成

一个完整的解表示全部路径的集合，它包含所有客户点，每个客户点只出现一次，并划分为 k 条路径，由 k 辆车同时配送，每条路径包含一定数量的客户点，路径的起始点都是配送中心。可行解要求在每条路径的任一时刻，配送车辆都要满足容量约束以及客户时间窗约束。

以配送中心为起点构造初始路径。判断最近的客户点是否符合时间窗以及车辆容量约束，若不满足则隔离该客户点并选择下一个最近的客户点，否则将其纳入当前路径，并将所有隔离客户点解除隔离，重复此步骤直至所有客户点都不满足约束，关闭该路径并开启新的路径。当所有客户点都被安排到路径内时，关闭最后一条路径。最后对产生的可行解执行若干次变异，得到丰富多样的种群，再选择较优的解作为初始解组。

2.2　底层启发式算子设计

LLH 要根据具体的问题进行设计,一般分为 3 类:局部优化算子(Local Research, LLH-L)、变异算子(Mutation, LLH-M)和破坏与重构算子(Location-based Radial Ruin, LLH-LR)。VRP 问题中底层启发式算子设计如表 2 所示。

表 2　底层启发式算子设计

算子类型	线路内	线路间
LLH-L	1)Adjacent Swap 2)General Swap 3)Single Insertion 4)Block Insertion 5)Inside-2opt	1)Shift(1,0) 2)Shift(2,0) 3)Swap(1,1) 4)Swap(2,1) 5)Swap(2,2)
LLH-M	1)Single Insertion 2)Block Insertion 3)2-opt 4)Or-opt	1)Shift(1,0) 2)Shift(2,0) 3)Swap(1,1) 4)Swap(2,1) 5)Swap(2,2)
LLH-LR	选择一个客户节点作为基准,按照其他客户节点与该基准的欧式距离重新构造一个新的可行解。	

表 2 中:Adjacent(General)Swap 表示相邻(不相邻)节点交换位置;Single(Block)Insertion 表示一个(两个相邻)节点移动到两相邻节点之间;Shift(m,0)表示当前路径中 m 个相邻节点插入另一条路径中;Swap(m,n)表示当前路径中的 m 个相邻节点和另一条路径中的 n 个相邻节点互换位置;Inside-2opt 表示连接两客户节点的路线反向后替换原来的路线。

2.3　解的接受准则及选择策略设计

2.3.1　接受准则

接受准则(acceptance criterion)用于判断是否接收子代解取代种群中的个体。接收准则设计的合理与否直接影响超启发算法收敛速度与优化精度,若执行算子后得到改进解,则总是接收;若得到非改进解,则以一定的概率接收。本文高层策略选择模拟退火(Simulated Annealing, SA)作为接收准则,非改进解以概率 P 接收,

$$P = \exp(\Delta E / T_k), \tag{17}$$

$$T_{k+1} = T_k \cdot \beta。 \tag{18}$$

其中:ΔE 为领域操作前后解的质量差;β 为降温系数,k 为温度计数器。

2.3.2　选择策略(*Selection*)

本文设计了基于禁忌搜索的上层选择策略。禁忌对象是底层启发式算子,禁忌表的禁忌长度 L 与局部搜索算子的个数紧密相关。根据算子对当前解的优化效果,对 3 类算子进行评分,同类算子具有相同的初始分数:S^L、S^M、S^{LR}。当算子改进当前解时,该算子加分,且加分分值随该算子的改进效果增加而增加;若无改进,则对算子进行扣分。3 类算子得分有上限和下限 $[S^L_{lower}, S^L_{upper}]$, $[S^M_{lower}, S^M_{upper}]$, $[S^{LR}_{lower}, S^{LR}_{upper}]$,且 3 类算子上限与下限不相等。每次迭代从不在禁忌表内的算子中依据其得分以轮盘赌方式选取,迭代完成后根据算子表现对该算子进行加分或扣分,并按照评价得分从低到高将算子放入固定长度的禁忌列表中。为保证 3 种算子的评价分数处于同一基线,设置参数 α 作为变异算子与破坏重构算子的得分系数,3 种算子的评价分数为 S^L、$\alpha \cdot S^M$、$\alpha \cdot S^{LR}$。另外,禁忌表的长度 L 影响了可选取算子的更新速度。同时,必须要有变异算子存在禁忌表之外,以保证一定的变异概率,因此 L 根据算子的数量进行选取。S_{lower} 和 S_{upper} 与算子被选择的概率密切相关,当算子得分达到下限或下限时,算子仍然具有恰当的被选择机率。例如,当 LLH-L 无法优化当前解时,对 LLH-L 进行扣分。若 S^L_{upper} 过大,需要经过多次迭代,对该算子的分数 S^L 进行多次扣分,使 S^L 减少到合适的量级,LLH-L 被选择概率减小,同时 LLH-M 和 LLH-LR 被选择的概率增大。在下一次 LLH-M 和 LLH-LR 被选择之前,选择并执行 LLH-L 做了过多无效的搜索,降低了算法效率。同样地,若 S^M_{lower} 和 S^M_{lower} 设置的过小,LLH-M 和 LLH-LR 被选择的概率太低,算法仅具有较弱的跳出局部最优的能力。要提高算法效率,须减少无效算子被执行的概率,选择最大可能改进当前解的算子。

同时,本文使用以下两种选择策略作为对比:①简单随机(Simple Random, SR),每次迭代随机选取算子;②随机下降(Random Descent, RD),随机选择一个算子,一直重复使用该算子直到解没有改进,然后再选择其他算子进行搜索直到满足停止条件。

2.4　算法框架设计

算法采用种群机制,种群规模 NI,为保证每次迭代以合理的概率选择 LLH、接受非改进解,设计出如图 2 所示算法流程,具体步骤如下:

步骤 1　初始化。生成初始可行解组 $x^G = (x^G_1, x^G_2, \cdots, x^G_{NI})$, $G=0$,并计算种群适应度 f^G、$f^G = (f^G_1, f^G_2, \cdots, f^G_{NI})$,以及整体最优目标函数值

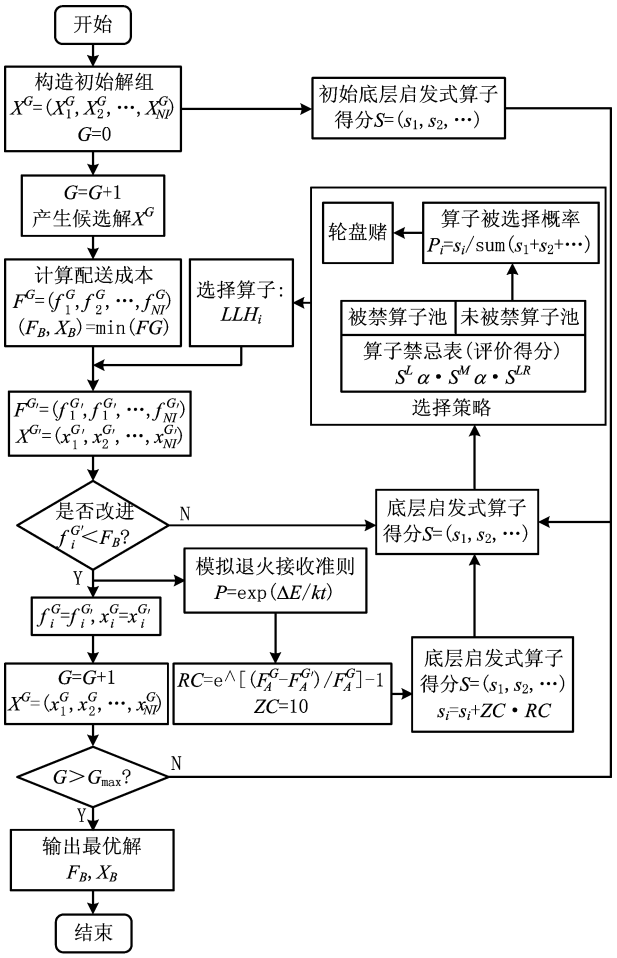

图2 算法流程

(F_B^G)、整体最优个体(X_B^G)、当代最优解目标函数值(F_C^G)、当代最优个体(X_C^G)、当代目标函数值的平均值(F_A^G)。算子初始得分为 $S^0=[S_1, S_2, \cdots, S_{20}]$,不同种类算子的初始得分不同,利用轮盘赌规则产生初始的 h^0。

步骤 2 产生并选择变异解。利用底层算子序列 h^G 构造新的路径 $X^{G'}=(x_1^{G'}, x_2^{G'}, \cdots, x_{NI}^{G'})$,$G=1$,计算 $f^{G'}=(f_1^{G'}, f_2^{G'}, \cdots, f_{NI}^{G'})$、$F_C^{G'}$、$F_A^{G'}$。

步骤 3 计算算子得分。根据 $F_A^{G'}$ 与 F_A^G 计算调整参数 RC。RC 是一个小于 1 的小数,根据第 G 代子代解的改进率计算各个底层启发式算子的得分 S^G,将得分低的算子放入禁忌表,调整在生成 h^G 时算子被选择的概率。在 h^G 算子 i 的解取得的进步较大时,当代 i 算子的 RC 值较大;当 h^G 算子 i 的解取得的进步较小时,当代算子 i 的 RC 值较小,调整每个对解进行有效改进的算子加分分值。在第 G 代中,算子 i 的加分为 S_i^G,$S_i^G=S_i^{G-1}+ZC \cdot RC_i^G$,其中:$ZC$ 为常数,$RC_i^G=e\wedge[(F_A^G-F_A^{G'})/F_A^G-1]$。若算子 i 扣分,则 $s_i^G=s_i-ZC \cdot RC_i^G$,$RC_i^G=e\wedge[(F_A^{G'}-F_A^G)/F_A^G-1]$。

步骤 4 保留候选解。若 $f_i^{G'}<f_i^G$,则保留 $f_i^G=f_i^{G'}$,$x_1^G=x_1^{G'}$;若 $f_i^{G'}>f_i^G$,按概率 $P=\exp(\Delta E/T_k)$(其中 $\Delta E=F_C^{G'}-F_C^G$)接收 $f_i^G=f_i^{G'}$,$x_1^G=x_1^{G'}$,$k=k+1$,$T_{k+1}=T_k \cdot \beta$。

步骤 5 保留最优解。若 $F_C^G<F_B^G$,则 $X_B^G=X_C^G$,$F_B^G=F_C^G$,$G=G+1$。

步骤 6 更新禁忌表。根据算子得分 $S^G=[S_1, S_2, \cdots, S_{20}]$,计算评价分数 S^L、$\alpha \cdot S^M$、$\alpha \cdot S^{LR}$,更新算子禁忌表,并根据不在禁忌表内的算子的得分,使用轮盘赌策略选择算子 h^G。

步骤 7 退出迭代。若 $G>G_{\max}$,算法结束,输出最优解,否则转步骤 2。

2.4 基于禁忌搜索的超启发式算法复杂度分析

根据上述流程,算法的复杂度包括:底层算子执行复杂度、解的接收执行复杂度、算子评分与禁忌表更新执行复杂度、选择策略执行复杂度、适应度值计算执行复杂度、最优解更新执行复杂度和初始化操作执行复杂度。本文基于禁忌搜索的超启发式算法参数如下:种群规模为 NI、迭代次数 $G_{\max}$、客户数量为 N、车辆数量为 K、底层算子个数为 NL 以及配送时间分段数量 R。

(1)底层算子执行复杂度 LLH-L 和 LLH-M 算子完成客户点的插入操作,并计算适应度值,因此复杂度为 $O(O(1)+O(R \cdot N^2))$;LLH-LR 算子重构一个可行解,则复杂度为 $O(O(N^2)+O(R \cdot N^2))$。算法采用种群机制,因此对于 LLH-L 和 LLH-M 的执行复杂度为 $O(NI \cdot (O(1)+O(R \cdot N^2)))$,对于 LLH-LR 执行复杂度为 $O(NI \cdot (O(N^2)+O(R \cdot N^2)))$。

(2)解的接收执行复杂度 与最优解进行比较,判断是否接收,复杂度为 $O(NI \cdot O(1))$。

(3)算子评分 对每个算子进行加分或扣分,复杂度为 $O(NI)$。

(4)禁忌表更新执行复杂度 按评价分数将算子排序,直接插入排序的复杂度为 $O(NL)$。

(5)选择策略执行复杂度 轮盘赌选择算子,复杂度为 $O(NL)$。

(6)适应度计算复杂度 对于时间依赖型车辆路径问题,需要计算车辆由客户点 i 出发到达下一个客户点 j 所跨越的时间段 R,分时段计算适应度值再求和,因此适应度计算的复杂度为 $O(R \cdot N^2)$。

(7)最优解更新执行复杂度　将种群的解进行比较,取最优,复杂度为 $O(NI)$。

(8)初始化操作　初始化解与算子得分,复杂度为 $O(N^2)$。

算法一次迭代的复杂度为:

若选择 LLH-L 或 LLH-M 算子,复杂度为:

$$O(HH)=O(O(NI\cdot(O(1)+O(R\cdot N^2)))+O(NI\cdot O(1))+O(NI)+O(NL)+O(NL))=O(NI\cdot R\cdot N^2);\tag{19}$$

若选择 LLH-LR 算子,复杂度为:

$$O(HH)=O(O(NI\cdot(O(N^2)+O(R\cdot N^2)))+O(NI\cdot O(1))+O(NI)+O(NL)+O(NL))=O(NI\cdot R\cdot N^2)。\tag{20}$$

则算法的总复杂度为:

$$O(HH)=O(N^2)+G_{\max}\cdot O(O(NI\cdot(O(N^2)+O(R\cdot N^2)))+O(NI\cdot O(1))+O(NI)+O(NL)+O(NL))=O(G_{\max}\cdot NI\cdot R\cdot N^2)。\tag{21}$$

由上述分析可知,算法的整体计算复杂度约为 $O(G_{\max}\cdot R\cdot NI\cdot N^2)$,即算法上层策略包括选择函数以及接受准则的复杂度可以忽略不计。影响算法复杂度因素主要包括:算法的迭代次数 $G_{\max}$、种群规模 NI、时间分段 R 以及客户点规模 N。对于一般的用于解决旅行商问题的基于种群的算法,复杂度为 $O(G_{\max}\cdot NI\cdot N^2)$,如刘欣欣[21]提出的基于片段插入的类粒群算法,其复杂度为 $O(G_{\max}\cdot N^3)$(其种群规模 $NI=2N$);冷龙龙等[20]提出量子超启发式算法用于解决低碳选址—路径问题,其算法复杂度约为 $G_{\max}\cdot NI\cdot O(N^2)$。根据算法的复杂度判断,本文所提出的基于禁忌搜索的超启发式算法属于多项式级的算法,其复杂度处于计算机可承受范围之内。

3　数值实验

实验环境 Intel(R)core-i5-8250U,8 GB RAM,程序独立运行 10 次,并计算最优解目标函数值(Min),最优解目标函数值方差(s^2),最优解目标函数值平均值(Avg)、算法运行时间(CT)和最优解改进(g_M)和算法运行时间改进(g_C)。

3.1　算法性能测试

为测试算法的实际性能,设计了第一个对比实验。对比数据来自于文献[11]中对 TDVRPSDP 算例所得的测试结果。本实验中,将基于禁忌搜索的超启发式算法(Hype-Heuristics—Tabu Search,HH-TS)和简单随机作为选择策略的超启发式算法(Hyper-Heuristics—Simple Random, HH-SR)的求解结果进行比较以评估算法高层选择策略对算法性能的影响。实验设置参数 $C_F=100$,$C_T=1$,$C_W=1$,迭代次数 $G_{\max}=20\ 000$。本文采用试取法选择参数,如表 3 所示。实验中的速度时间分段函数如表 4 所示。同时,将 HH-TS 与文献[11]的求解结果做比较以评估算法获得最优解的能力,如表 5 所示。表 5 中左侧给出文献[11]设计的集成蚁群与禁忌搜索的新型算法(ACS-TS)所得 56 个测试算例最优解,其中蚂蚁数量为 20,TS 迭代次数为 60,ACS 迭代次数为 300,计算机配置:Window XP. Intel(R) Core(TM)Celeron M. 1. 67 GHz。

表 3　参数选取

参数	L	α	β	NI	S^M_{lower}	S^M_{upper}	S^{LR}_{lower}	S^{LR}_{upper}	S^L_{lower}	S^L_{upper}	ZC
取值	9	60	0.95	10	10	20	10	20	550	1 200	10

表 4　配送车辆速度

算例类型	平峰车辆速度	高峰车辆速度
$C1$	1	0.5
$C2$	0.2	0.1
$R1$	20	10
$R2$	1	0.5
$CR1$	5	2.5
$CR2$	1	0.5

续表 4

表 5　基于禁忌搜索的超启发式算法性能测试

算例	ACS-TS			HH-SR			HH-TS			g_M		g_C
	Avg	Max	Min	Min	Avg	CT	Min	Avg	CT	$g_{M\text{-}T/A}$/%	$g_{M\text{-}T/S}$/%	$g_{C\text{-}T/S}$/%
C101	7 901	8 021	7 786	5 653	6 076	11	4 699	5 159	8	27.4	16.9	27.3
C102	7 341	7 398	7 257	4 291	4 596	29	4 207	4 571	8	40.9	2.0	72.4

续表 5

*C*103	7 368	7 449	7 274	3 597	3 888	33	3 425	3 896	12	50.5	4.8	63.6
*C*104	6 856	6 934	6 760	3 205	3 292	30	3 033	3 346	14	52.6	5.4	53.3
*C*105	7 822	7 896	7 723	4 111	4 450	22	3 704	4 093	6	46.8	9.9	72.7
*C*106	7 825	7 885	7 733	4 331	4 636	27	3 848	1 250	7	44.0	11.2	74.1
*C*107	7 742	7 816	7 609	3 679	3 792	28	3 280	3 729	7	51.6	10.8	75.0
*C*108	7 589	7 677	7 454	3 479	3 579	30	3 271	3 575	9	53.3	6.0	70.0
*C*109	7 511	7 656	7 420	3 113	3 254	35	3 026	3 249	22	58.0	2.8	37.1
*C*201	11 699	12 123	10 745	13 741	14 274	38	12 356	15 408	11	−28	10.1	71.1
*C*202	16 834	17 099	16 107	12 496	14 096	32	11 413	12 719	12	22.4	8.7	62.5
*C*203	20 514	20 788	19 280	11 383	12 124	55	10 579	11 699	12	41.0	7.1	78.2
*C*204	21 124	21 930	20 973	8 739	10 130	50	8 505	9 010	18	58.3	2.7	64.0
*C*205	11 685	11 858	11 328	10 918	11 231	42	10 581	11 355	9	3.6	3.1	78.6
*C*206	10 949	11 149	10 845	9 771	10 237	44	9 514	10 944	12	9.9	2.6	72.7
*C*207	12 185	12 428	11 752	10 362	10 901	41	8 755	9 968	14	11.8	15.5	65.9
*C*208	12 028	12 333	11 325	10 035	10 302	46	8406	8 469	18	11.4	16.2	60.9
*R*101	1 781	1 798	1 760	1 600	1 694	42	1 573	1 609	11	9.1	1.7	73.8
*R*102	1 783	1 798	1 775	1 283	1 336	40	1 283	1 297	12	27.7	0.0	70.0
*R*103	1 717	1 771	1 685	1 007	1 025	53	1 003	1 012	15	40.2	0.4	71.7
*R*104	1 546	1 617	1 525	928	940	76	910	932	21	39.1	1.9	72.4
*R*105	1 711	1 726	1 697	1 140	1 230	60	1 137	1 199	11	32.8	0.3	81.7
*R*106	1 656	1 674	1 634	989	981	62	973	1 054	19	39.5	1.6	69.4
*R*107	1 571	1 595	1 550	929	936	58	921	939	12	40.1	0.9	79.3
*R*108	1 536	1 574	1 519	891	901	70	890	900	23	41.3	0.1	67.1
*R*109	1 646	1 657	1 637	1 043	1 048	65	1 042	1 064	13	36.3	0.1	80.0
*R*110	1 648	1 655	1 629	1 013	1 097	62	1 002	1 039	16	37.8	1.1	74.2
*R*111	1 680	1 693	1 668	999	1 006	61	1 001	1 048	13	40.1	−0.2	78.7
*R*112	1 489	1 506	1 478	1 091	1 094	81	976	982	42	26.2	10.5	48.1
*R*201	4 161	5 226	3 860	3 831	4 032	60	3 714	4 084	23	0.8	3.1	61.7
*R*202	4 190	5 097	4 059	3 510	3 612	110	3 319	3 554	28	13.5	5.4	74.5
*R*203	3 628	4 963	3 449	2 818	3 032	130	2 637	2 956	19	18.3	6.4	85.4
*R*204	2 849	4 759	2 624	2 137	2 383	124	1 782	2 165	28	18.6	16.6	77.4
*R*205	3 622	5 203	3 321	2 865	3 262	53	2 783	3 008	18	13.7	2.9	66.0
*R*206	3 488	4 626	3 247	2 390	2 871	42	2 371	2 664	19	26.4	0.8	54.8
*R*207	2 998	4 793	2 743	2 181	2 512	105	2 069	2 263	33	20.5	5.1	68.6
*R*208	2 762	4 736	2 191	1 654	2 034	155	1 630	1 785	56	24.5	1.5	63.9
*R*209	3 718	4 518	3 508	2 251	2 798	88	2 209	2 463	19	35.8	1.9	78.4
*R*210	4 083	5 233	3 662	2 451	2 877	91	2 442	2 792	21	33.1	0.4	76.9
*R*211	3 398	4 284	2 860	1 901	2 313	251	1 876	2 151	100	33.5	1.3	60.2
*CR*101	2 570	2 586	2 538	2 196	2 201	60	2 038	2 240	15	13.5	7.2	75.0
*CR*102	2 524	2 544	2 459	1 709	1 759	72	1 662	1 728	8	30.5	2.8	88.9
*CR*103	2 417	2 463	2 325	1 588	1 602	83	1 553	1 633	18	31.7	2.2	78.3

续表 5

*CR*104	2 231	2 245	2 204	1 445	1 466	103	1 343	1 441	26	34.4	7.1	74.8
*CR*105	2 578	2 589	2 543	1 958	2 074	66	1 959	2 190	5	23.0	−0.1	92.4
*CR*106	2 397	2 431	2 355	1 837	1 950	80	1 701	1 830	19	22.0	7.4	76.3
*CR*107	2 330	2 345	2 296	1 758	1 795	92	1 599	1 757	22	23.4	9.0	76.1
*CR*108	2 177	2 247	2 124	1 532	1 633	114	1 509	1 609	29	27.9	1.5	65.1
*CR*201	6 104	6 331	5 759	4 275	4 820	42	4 296	4 886	6	25.8	−0.5	85.7
*CR*202	5 920	6 117	5 805	3 937	4 306	52	3 778	4 406	21	32.2	4.0	59.6
*CR*203	6 180	6 274	6 153	3 314	3 480	239	3 299	3 476	35	46.1	0.5	42.6
*CR*204	5 712	5 779	5 542	2 359	2 753	96	2 141	2 414	28	57.4	9.2	70.8
*CR*205	6 257	6 382	6 134	3 969	4 561	46	3 905	4 418	12	35.3	1.6	73.9
*CR*206	5 649	5 954	5 334	3 210	3 829	57	2 962	3 415	19	39.8	7.7	66.7
*CR*207	5 274	5 402	5 032	2 745	3 027	66	2 662	2 933	22	45.4	3.0	66.7
*CR*208	5 004	5 189	4 788	2 284	3 013	82	1 998	2 408	74	52.3	12.5	9.8

表 5 中,g_M 项下展示了 HH-TS 与 ACS-TS 算法所得最优解相比较所取得的改进 $g_{M\text{-}T/A}$。在 56 个算例中,相较于 ACS-TS,使用 HH-TS 有 55 个算例取得了大幅度改进。在 55 个取得改进的算例中,最大改进达到了 58.0%,最小改进为 0.8%,平均取得改进 31.2%。另外,文献[11]给出了多次求解的平均值,如表 5 第 2 列。平均值只能在一定程度上反映算法的求解能力,但不能体现数据的离散程度。为保持完整性,本实验也给出了多次运行算法求解的平均值。HH-SR、HH-TS 相较于 ACS-TS,平均值分别取得了平均 28.81%、32.02%的改进。

表 5 中,g_M 项下展示了 HH-TS 较 HH-SR 所取得的最优解改进 $g_{M\text{-}T/S}$,g_C 项下展示了 HH-TS 较 HH-SR 所取得的算法运行时间的改进 $g_{C\text{-}T/S}$。在 56 个算例中,相较于 HH-SR,使用 HH-TS 52 个算例取得了目标函数值最大 16.9%,平均 4.9%的改进,此外的 4 个算例也取得了最差 −0.5%的相对较劣的解。由此可见,超启发式算法在标准算例的求解上具有较好的优化精度。另外,HH-SR 采用随机的选择函数与模拟退火接受准则作为上层策略,对 3 个类型底层算子进行随机的调用。HH-SR 相较于 HH-TS,执行无效的算子耗费了大量的时间。因此,对于同样的迭代次数,HH-TS 耗时相对 HH-SR 大幅减少,最大 92%,平均 68.5%,如表 5 第 13 列所示。因此,超启发式算法较优的上层策略设计能在一定程度上,调用底层算子向函数值下降最快的方向高效地进行最优解搜索,可以保证一定的收敛速度和跳出局部最优的能力。在求解 TDVRSDP 时,往往超启发算法拥有更好的优势,能在短时间内求得可接受的解。

3.2　超启发式算法求解 TDVRPSDP 标准算例

实验增加了基于随机下降选择策略的超启发式算法(Hyper-Heuristics—Random Descent, HH-RD)实验数据,将 HH-TS 求解 TDVRPSDP 的结果与 HH-SR、HH-RD 对比。实验的测试实例来自于文献[22],并构造了高峰、平峰车速不同于文献[11]的速度时间分段函数,以一定的比率提高车辆速度,比率为 0.5～5(如表 6),得到 TD-VRPSDP 的测试实例,如表 7 所示。

表 6　配送车辆速度

算例类型	平峰车辆速度	高峰车辆速度	速度比率
*LC*1	1.50	0.750	1.0
*LC*2	0.75	0.385	0.5
*LR*1	3.00	1.500	2.0
*LR*2	6.00	3.000	4.0
*LCR*1	7.50	3.750	5.0
*LCR*2	7.50	3.750	5.0

注:速度比率=平峰车速×(1/3)+高峰车速×(1/3)×2。

文献[22]在 Solomon 的 56 个 VRPTW 标准测试实例的基础上,构造了 6 类 VRPSPDPTW 标准测试实例(*LR*1,*LR*2,*LC*1,*LC*2,*LRC*1,*LRC*2)并提供最优解:车辆数量(*Veh*)和行驶距离(*Dis*),如表 7 左侧第 1 和第 2 列所示。

表 7 TDVRPSDP 实例测试

算例	Best-Known		HH-SR		HH-RD		HH-TS				g_M	
	Veh	*Dis*	*Min*	s^2	*Min*	s^2	*Min*	s^2	*Dis*	*Veh*	$g_{M\text{-}T/S}$/%	$g_{M\text{-}T/R}$/%
*LC*101	10	828.94	4 145	349	4 010	449	3 020	336	953	12	27.1	24.7
*LC*102	10	828.94	3 327	317	3 557	319	3 047	148	985	13	8.4	14.4
*LC*103	9	1 035.35	3 017	247	2 970	250	2 535	1 29	1 026	11	16.0	14.6
*LC*104	9	860.01	2 478	123	2 432	146	2 140	149	958	10	13.7	12.0
*LC*105	10	828.94	3 117	236	3 027	320	2 793	521	983	12	10.4	7.8
*LC*106	10	828.94	3 108	225	3 356	226	2 835	221	968	12	8.8	15.5
*LC*107	10	828.94	2 610	222	2 684	232	2 609	168	971	12	0	2.8
*LC*108	10	826.44	2 632	243	2 768	169	2 356	247	945	12	10.5	14.9
*LC*109	9	1 000.6	2 444	158	2 636	69	2 222	86	1 082	11	9.1	15.7
*LC*201	3	591.56	5 594	839	4 495	1 498	3 495	735	787	4	37.5	22.3
*LC*202	3	591.56	5 178	919	4 027	1 068	2 932	804	777	5	43.4	27.2
*LC*203	3	591.17	4 210	602	3 659	1 162	2 768	755	715	5	34.2	24.3
*LC*204	3	590.6	3 068	375	3 083	506	2 489	391	750	4	18.9	19.3
*LC*205	3	588.88	3 227	1 187	3 431	755	2 727	291	788	4	15.5	20.5
*LC*206	3	588.49	2 947	815	3 142	1 751	2 542	1 025	742	4	13.8	19.1
*LC*207	3	588.29	3 092	597	3 329	615	2 461	509	706	4	20.4	26.1
*LC*208	3	588.32	2 411	264	2 656	651	2 341	549	746	4	2.9	11.9
*LR*101	19	1 650.8	3 414	127	3 232	143	3 015	105	1 838	16	11.7	6.7
*LR*102	17	1 487.57	2 717	115	2 525	196	2 266	176	1 606	12	16.6	10.3
*LR*103	13	1 292.68	2 002	149	2 093	134	1 840	155	1 423	11	8.1	12.1
*LR*104	9	1 013.39	1 614	121	1 588	104	1 514	83	1 065	9	6.2	4.7
*LR*105	14	1 377.11	2 425	90	2 290	179	2 227	126	1 429	13	8.2	2.8
*LR*106	12	1 252.62	1 970	108	1 939	141	1 875	89	1 320	11	4.8	3.3
*LR*107	10	1 111.31	1 670	126	1 849	75	1 670	56	1 303	10	0.0	9.7
*LR*108	9	968.97	1 447	84	1 449	73	1 365	33	1 021	9	5.7	5.8
*LR*109	11	1 208.96	2 055	116	1 889	159	1 857	132	1 315	11	9.6	1.7
*LR*110	10	1 159.35	1 970	89	1 818	125	1 775	80	1 217	11	9.9	2.3
*LR*111	10	1 108.9	1 873	84	1 845	160	1 661	98	1 211	10	11.3	9.9
*LR*112	9	1 003.77	1 704	76	1 741	70	1 555	68	1 037	10	8.7	10.7
*LR*201	4	1 253.23	1 282	44	978	162	1 256	45	1 406	3	2.1	−28.3
*LR*202	3	1 197.67	1 122	105	611	147	741	230	1 409	3	33.9	−21.2
*LR*203	3	949.4	634	183	629	121	609	128	1 058	2	4.1	3.3
*LR*204	2	849.05	511	129	478	102	467	42	998	2	8.6	2.4
*LR*205	3	1 054.02	772	363	731	323	685	395	1 249	2	11.3	6.3
*LR*206	3	931.63	753	132	619	273	547	117	1 140	2	27.3	11.6
*LR*207	2	903.06	525	118	525	43	525	63	1 103	2	0	0
*LR*208	2	734.85	444	119	450	116	434	81	837	2	2.4	3.6
*LR*209	3	930.59	592	234	567	150	574	10	1 088	2	3.0	−1.2
*LR*210	3	964.22	618	129	637	86	617	102	1 113	2	0.3	3.2

续表 7

*LR*211	2	911.52	912	224	482	395	479	207	1 085	2	47.5	0.8
*LRC*101	14	1 708.8	1 888	97	1 885	17	1 695	71	2 027	10	10.2	10.1
*LRC*102	12	1 558.07	1 402	79	1 366	96	1 252	41	2 033	9	10.7	8.3
*LRC*103	11	1 258.74	1 198	59	1 163	111	1 129	55	1 580	8	5.7	2.9
*LRC*104	10	1 128.4	1 094	42	1 088	49	936	49	1 145	7	14.4	14.0
*LRC*105	13	1 637.62	1 864	93	1 681	110	1 492	111	1 886	9	20.0	11.2
*LRC*106	11	1 424.73	1 575	120	1 568	87	1 401	51	1 582	9	11.0	10.6
*LRC*107	11	1 230.14	1 511	136	1 488	121	1 320	73	1 446	9	12.6	11.3
*LRC*108	10	1 147.43	1 279	88	1 302	81	1 268	42	1 219	9	0.9	2.6
*LRC*201	4	1 406.94	1 269	107	1 190	165	1 033	103	1 708	3	18.5	13.2
*LRC*202	3	1 374.27	1 139	74	736	169	710	211	1 656	2	37.7	3.6
*LRC*203	3	1 089.07	574	192	717	180	522	157	1 320	2	9.0	27.1
*LRC*204	3	818.66	459	121	462	136	451	18	1 050	2	1.8	2.3
*LRC*205	4	1 302.2	942	104	976	126	981	85	1 644	2	0.3	3.8
*LRC*206	3	1 159.03	686	309	653	250	638	451	1 296	2	6.9	2.3
*LRC*7	3	1 062.05	540	207	550	164	536	145	1 303	2	0.7	2.6
*LRC*8	3	852.76	617	282	595	286	432	169	1013	2	30.0	27.4

下面从模型的角度分析数据结果。表 8 列举了配送车辆速度变化对车辆数量及车辆总行驶距离的影响。相较于最优解,当车速增加时,配送车辆数量随车速增加而减少,但车辆的行驶距离并没有随之减少,而是微量增加。当车速取 0.5 和 1 的倍率时,较慢的车速使得配送车辆满足客户点时间窗的能力降低,车辆最大容量在一定程度上已不再是限制单车配送多个客户点的关键因素。因此,单车可访问客户点的数量下降,需要派发更多车辆才能保证每个客户点的配送时效。例如,在 0.5 倍速时,对于 LC1 算例,车辆数量增加 41.67%,车辆行驶距离增加 21.37%,如表 8 第 2 行。因此,车辆的出发成本与旅行成本较高,总成本较高。反之,车速取 2、4 和 5 倍速率时,配送车辆满足客户点时效要求的能力升高,单车可访问客户点数量上升,但是受限于车辆的最大容量,单车可访问客户点数量少量增加,使用车辆数量不会大量减少。同时,随着车速提高,单位时间车辆使用成本不变($C_T=1$),单位时间车辆可行驶距离变为原来的 0.5、1、2、4、5 倍。车辆的行驶时间大幅减少,因此车辆的旅行成本大幅降低,总成本降低。总结如下:①当车速提高时,使用车辆数量逐渐减少,受限于车辆最大容量,车辆数量减少数量终将到达一个固定的极限;②随车速提高,车辆行驶距离少量增加,车辆的旅行成本大幅降低,物流配送成本大幅降低,如表 7 第 8 列。因此,由实验结果可知:当车速提高时,增大车辆容量将有效减少车辆的使用数量,减少所有车辆总的行驶时间,有效地降低成本。

表 8　配送车辆速度对车辆数量及总行驶距离的影响

算例类型	速度比率	车辆数量增加	车辆数量变化/%	总行驶距离变化/%
*LC*1	1	2.000	20.69	11.44
*LC*2	0.5	1.250	41.67	21.37
*LR*1	2	−0.833	−6.99	7.05
*LR*2	4	−0.545	−19.98	14.46
*LCR*1	5	−2.750	−23.91	13.10
*LCR*2	5	−1.000	−30.77	17.48

从算法角度分析,HH-TS 取得最优的计算结果。表 7 *Min* 项下分别展示了 HH-TS 与 HH-SR 算法所的最优解相比较所取得的改进 $g_{M\text{-}T/S}$,HH-TS 与 HH-RD 比较所取得的改进 $g_{M\text{-}T/R}$。在 56 个算例中,HH-TS 相对于 HH-SR,全部算例均取得了更优的解,最大改进 43.4%,平均改进 12.9%。同时,相较于 HH-RD,53 个算例取得了更优的解,最大改进 27.2%,平均改进 9.1%。HH-SR、HH-RD 和 HH-TS 在全部算例的计算结果得到的平均

方差分别为 230、282 和 205。可以从方差的角度分析 3 种策略算法对求解 TDVRPSDP 最优解的离散程度。SR 策略因其随机地选择 LLH 对最优解进行搜索，在有限的迭代之后通常得不到最优解，而 RD 策略具有一定的避免选择无效 LLH 的能力，但无法选择最有效的那些算子，所以仅具有微弱的跳出局部最优的能力，求解结果波动较大。TS 策略根据评分选择禁忌表外的 LLH，且以轮盘赌方式分配 L、M 和 LR 被选择的概率。因此，HH-TS 每次求得最优解的波动也最小，性能也最好。

由此可见，上层策略的设计对解空间的搜索效率起着至关重要的作用，较好的上层策略不但能取得更好的解，而且所得结果波动较小，可以获得更好地收敛曲线并保证跳出局部最优的能力。因此，计算结果说明了禁忌搜索为选择策略的超启发式算法在求解 TDVRPSDP 上的有效性。

4 结束语

本文研究了时间依赖型同时取送货车辆路径问题，将配送中心的开放时间等分为 3 段，每段对应不同的车辆旅行速度，以车辆的租赁成本、车辆旅行成本以及车辆等待的惩罚成本之和为目标建模。设计了基于禁忌搜索的超启发式算法，高层选择策略采用禁忌搜索机制，依据算子的性能选择最有希望改进当代解的算子进行搜索，算子性能的评分由其历史表现决定，并逐代更新；高层接收准则采用模拟退火机制，保证了算法前期的快速收敛，算法中后期以一定的概率接受非改进解，使算法具备了跳出局部最优的能力。最后，通过标准算例及实验对比表明，算法在求解该问题上能较快速地取得可接受的解。对于时间依赖型车辆路径问题，车速与配送成本的关系较为复杂，后续将围绕更加复杂的车辆旅行速度以及可变车速下客户满意度、配送成本、能源消耗与行车路径之间的关系对问题模型展开研究。此外，尽管超启发式算法在求解 TDVRPSDP 上表现出良好的性能，但仍然具有一定的改进空间，未来考虑将超启发式算法与其他策略结合，开发出更加高效的上层策略。

参考文献：

[1] MIN H. The multiple vehicle routing problem with simultaneous delivery and pick-up points[J]. Transportation Research Part A:General,1989,23(5):377-386.

[2] POONTHALIR G, NADARAJAN R, GEETHA S. CO_2 and idling emission estimation for vehicle routing problem with mid way halts[C]//Proceedings of the International Joint Conference on SOCO'16-CISIS'16-ICEUTE'16. Berlin, Germany: Springer-Verlag, 2016:167-176.

[3] NINIKAS G, MINIS I. Re-optimization strategies for a dynamic vehicle routing problem with mixed backhauls[J]. Networks, 2015, 64(3):214-231.

[4] LIN C, CHOY K L, HO G T S, et al. A decision support system for optimizing dynamic courier routing operations[J]. Expert Systems with Applications, 2014, 41(15):6917-6933.

[5] HU Zhihua, SHEU J B, ZHAO Lei, et al. A dynamic closed-loop vehicle routing problem with uncertainty and incompatible goods[J]. Transportation Research Part C:Emerging Technologies, 2015, 55:273-297.

[6] WANG Jiahai, ZHOU Ying, WANG Yong, et al. Multi-objective vehicle routing problems with simultaneous delivery and pickup and time windows: Formulation, instances, and algorithms[J]. IEEE Transactions on Cybernetics, 2017, 46(3): 582-594.

[7] WANG Chao, LIU Chao, MU Dong, et al. VRPSDPTW problem solving by discrete cuckoo search[J]. Computer Integrated Manufacturing Systems, 2018, 24(3):4-16(in Chinese). [王 超，刘 超，穆 东，等. 基于离散布谷鸟算法求解带时间窗和同时取送货的车辆路径问题[J]. 计算机集成制造系统，2018，24(3)：4-16.]

[8] MALANDRAKI C, DASKIN M S. Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms[J]. Transportation Science, 1992, 26(3):185-200.

[9] FIGLIOZZI M A. The time dependent vehicle routing problem with time windows: Benchmark problems, an efficient solution algorithm, and solution characteristics[J]. Transportation Research Part E: Logistics and Transportation Review, 2012, 48(3):616-636.

[10] MU Dong, WANG Chao, WANG Shengchun, et al. Solving TDVRP based in parallel-simulated annealing algorithm[J]. Computer Integrated Manufacturing Systems, 2015, 21(6): 1626-1636(in Chinese). [穆 东，王 超，王胜春，等. 基于并行模拟退火算法求解时间依赖型车辆路径问题[J]. 计算机集成制造系统，2015，21(6)：1626-1636.]

[11] ZHANG T, CHAOVALITWONGSE W A, ZHANG Y. Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery[J]. Journal of Combinatorial Optimization, 2014, 28(1):288-309.

[12] OCHOA G, HYDE M, CURTOIS T, et al. HyFlex: A benchmark framework for cross-domain heuristic search [C]//Proceedings of European Conference on Evolutionary Computation in Combinatorial Optimization. Berlin, Germany: Springer-Verlag, 2012, 136-147.

[13] CHEN P C, KENDALL G, BERGHE G V. An ant based

hyper-heuristic for the travelling tournament problem[C]// Proceedings of IEEE Symposium on Computational Intelligence in Scheduling. Washington, D. C., USA: IEEE, 2007: 19-26.

[14] BURKE E K, MCCOLLUM B, MEISELS A, et al. A graph-based hyper-heuristic for educational timetabling problems[J]. European Journal of Operational Research, 2007, 176(1): 177-192.

[15] DOKEROGLU T, COSAR A. A novel multi-start hyper-heuristic algorithm on the grid for the quadratic assignment problem[J]. Engineering Applications of Artificial Intelligence, 2016, 52: 10-25.

[16] SABAR N R, AYOB M, KENDALL G, et al. Automatic design of a hyper-heuristic framework with gene expression programming for combinatorial optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(3): 309-325.

[17] ZAMLI K Z, ALKAZEMI B Y, KENDALL G. A Tabu search hyper-heuristic strategy for *t*-way, test suite generation[J]. Applied Soft Computing, 2016, 44: 57-74.

[18] ASTA S, ÖZCAN E. A tensor-based selection hyper-heuristic for cross-domain heuristic search[J]. Information Sciences, 2015, 299: 412-432.

[19] OZCAN E, MISIR M, KHEIRI A. Group decision making hyper-heuristics for function optimization[C]//Proceedings of 13th UK Workshop on Computational Intelligence (UKCI). Washington, D. C., USA: IEEE, 2013: 327-333.

[20] LENG Longlong, ZHAO Yanwei, ZHANG Chunmiao, et al. Quantum-inspired hyper-heuristics for low-carbon location-routing problem with simultaneous pickup and delivery[J]. Computer Integrated Manufacturing Systems, 2018, 22(1): 1-22(in Chinese). [**冷龙龙, 赵燕伟, 张春苗, 等. 求解物流配送同时取送货低碳选址—路径问题的量子超启发式算法[J]. 计算机集成制造系统**, 2018, 22(1): 1-22.]

[21] LIU Xinxin. Gene fragment inserting algorithms for traveling salesman problem[D]. Zhangzhou: Minnan Normal University, 2015(in Chinese). [**刘欣欣. 旅行商问题的基因片段插入算法研究[D]. 漳州: 闽南师范大学**, 2015].

[22] LI Haibing, LIM A. Ameta-heuristic for the pickup and delivery problem with time windows[J]. International Journal of Artificial Intelligence Tools, 2001, 12(2): 160-167.

作者简介:

张景玲(1980—),**女**,**湖北黄冈人**,**副教授**,**博士**,**硕士生导师**,**研究方向**:**人工智能、物流优化调度**,E-mail: jlzhang@zjut.edu.cn;

刘金龙(1993—),**男**,**安徽阜阳人**,**硕士研究生**,**研究方向**:**物流优化调度**;

赵燕伟(1959—),**女**,**河南郑州人**,**教授**,**博士生导师**,**研究方向**:**物流配送与优化调度等**;

王宏伟(1981—),**男**,**黑龙江哈尔滨人**,**副教授**,**博士**,**博士生导师**,**研究方向**:**智能优化调度**;

冷龙龙(1991—),**男**,**江西宜春人**,**博士后**,**研究方向**:**物流配送与优化调度**;

冯勤炳(1995—),**男**,**浙江嘉兴人**,**硕士研究生**,**研究方向**:**物流配送与优化调度**。