

多目标同时取送货选址-路径问题的多起点变邻域搜索算法

陈希琼, 胡大伟[†], 王 宁

(长安大学 运输工程学院, 陕西 西安 710064)

摘要: 为使同时取送货的选址-路径问题(LRPSPD)的总成本和各路径间最大长度差最小化, 建立同时考虑车辆容量和行驶里程约束的LRPSPD双目标模型. 采用多蚁群算法构造多个以信息素为关联的初始解, 作为多目标变邻域搜索算法搜索的多个起点, 构造四类邻域结构进行变邻域搜索, 并根据最新获得的最优邻域解更新蚂蚁信息素, 从而使蚁群算法产生的多个初始解间、以及初始解与变邻域搜索产生的解之间均存在正向影响关系. 用该算法求得文献中4组共128个算例的近似Pareto解集, 结果证明了最小化路径间最大长度差目标对于节点及需求分布不集中算例的重要意义. 以绝对偏向最小化总成本的解与文献中仅最小化总成本的几种算法的算例结果进行比较, 结果表明算法可在极短的运行时间里求得权衡各目标的Pareto解, 并使最小总成本目标值具有竞争性.

关键词: 综合交通运输; 多起点变邻域搜索; 多蚁群算法; 同时取送货选址路径; 多目标局部搜索

引用格式: 陈希琼, 胡大伟, 王宁. 多目标同时取送货选址-路径问题的多起点变邻域搜索算法. 控制理论与应用, 2022, 39(7): 1229 – 1241

DOI: 10.7641/CTA.2022.10265

A multi-start variable neighborhood search for multi-objective location routing problem with simultaneous pickup and delivery

CHEN Xi-qiong, HU Da-wei[†], WANG Ning

(School of Transportation Engineering, Chang'an University, Xi'an Shaanxi 710064, China)

Abstract: In order to minimize the total cost and maximum gap among each tour in a location-routing problem with simultaneous pickup and delivery (LRPSPD), a LRPSPD bi-objective model, which considers the vehicle capacity and travel distance constraints simultaneously, is established. Multiple ant colonies are used to construct the initial solutions. The initial solutions are associated with each other by multiple pheromones and are taken as multi-starts of bi-objective variable neighborhood search algorithm. Four types of neighborhood structures are established for the variable neighborhood search. The pheromones are updated according to the latest optimal neighborhood solutions, so that there are positive influence relationships not only among multiple initial solutions generated by ants, but also between initial solutions and solutions generated by variable neighborhood search. The proposed multi-start variable neighborhood search (MSVNS) approach is then used to solve 128 instances, which are classified in 4 sets in the literature and obtain the approximate Pareto solutions. The results demonstrate the significance of considering the maximum gap among each tour as a minimized objective in some instances, which have scattered nodes and demands distribution. Compare the solutions with lowest cost (absolute preference to minimize the costs) from Pareto solutions with previous results from literature which minimize the total costs only. The comparison results indicate that the procedure can be operated in extremely short running time and the Pareto solutions acquired by the algorithm provide good tradeoff between two objectives and yield a competitive solution of the cost-minimizing objective.

Keywords: comprehensive transportation; multi-start variable neighborhood search; multiple ant colony algorithm; location routing problem with simultaneous pickup and delivery; multi-objective local search

Citation: CHEN Xiqiong, HU Dawei, WANG Ning. A multi-start variable neighborhood search for multi-objective location routing problem with simultaneous pickup and delivery. *Control Theory & Applications*, 2022, 39(7): 1229 – 1241

收稿日期: 2021-03-30; 录用日期: 2022-01-23.

[†]通信作者. E-mail: dwu@chd.edu.cn; Tel.: +86 29-82334647.

本文责任编辑: 吴敏.

国家自然科学基金项目(71971030), 陕西省自然科学基金重点项目(2021JZ-20), 中央高校基本科研业务费专项(300102220102, 300102229304), 陕西省教育厅专项科研基金项目(17JK0284)资助.

Supported by the National Natural Science Foundation of China (71971030), the Key Natural Science Project of Shaanxi Province (2021JZ-20), the Fundamental Research Funds for the Central Universities (300102220102, 300102229304) and the Special Scientific Research Funding of Education Department of Shaanxi Province (17JK0284).

1 引言

选址-路径问题(location-routing problem, LRP)集成处理设施选址和车辆路径两个关键物流决策问题. 传统LRP研究的车辆路径问题(vehicle routing problem, VRP)中, 客户仅存在取货或送货单一需求, 而现实物流系统中, 通常存在需要同时提供送货和取货服务的客户, 且要求在访问中同时完成取送货, 如: 需要回收空瓶的饮料、鲜奶及啤酒配送.

针对所有需求点均有取送两层需求的物流配送问题, Min^[1]首次定义了考虑同时取送货的VRP (VRP with simultaneous pickup and delivery, VRPSPD), 研究中心图书馆与地方图书馆之间图书发送与回库问题. 之后VRPSPD被广泛研究: 如多车场VRPSPD^[2]、不同车型VRPSPD^[3]、带时间窗VRPSPD^[4-5]、时间依赖型VRPSPD^[6]等, 以环保为出发点研究低碳约束^[7-8]和最小化燃油消耗^[9]的VRPSPD引起诸多学者的兴趣. 陈希琮等^[10]研究同时考虑车辆容量和距离约束的VRPSPD双目标模型, 并采用蚁群算法进行求解.

VRPSPD的广泛研究表明考虑同时取送货在物流决策中的重要性, 然而考虑同时取送货的LRP (LRP with simultaneous pickup and delivery, LRPSPD) 近才开始被少数学者关注. Karaoglan等^[11-12]最早提出LRPSPD并进行了持续研究, 构建了其初始模型, 先后采用分支切面法、混合遗传算法求解该模型; 随后, 提出了基于节点和基于弧的混合整数规划LRPSPD模型, 构造了LRPSPD测试算例集, 并设计两阶段模拟退火算法^[13]和模因算法^[14]求解模型. Yu等^[15]设计多起点模拟退火算法, 基于文献[13]构造的算例, 求解LRPSPD; 在进一步研究中^[16], 额外生成了客户数达318的4组算例, 设计改进的模拟退火算法进行求解. 孙青伟等^[17]基于实际案例研究了多车型LRPSPD. Kartal等^[18]研究单一指派 p -中位转运站LRPSPD, Karimi^[19]对枢纽选址VRPSPD进行研究, 不同于传统LRPSPD, 文献[18-19]允许枢纽间转运. Fan等^[20]基于构造算法、局部搜索和变邻域算法提出多起点混合启发式算法求解两阶段LRPSPD; 郭放等^[21]研究前置仓选址VRPSPD.

文献中考虑同时取送货的多目标优化研究主要针对VRPSPD, 几乎不能找到LRPSPD多目标优化研究. García-Nájera^[22]对取送货问题的多目标研究进行调查, 指出工作量平衡目标应被考虑, 但在其提出的六目标模型中并未包括工作量平衡目标. 同时, 多目标优化求解复杂, 由于对多目标进行转化处理具有不能兼顾多个目标的局限性, 但近年来同时兼顾多个目标的Pareto解集成为研究重点, 如Anderluh等^[23].

LRPSPD研究通常以最小化总成本为目标, 然而现实运作中决策者需要根据其他可能的情况进行决策, 如考虑客户满意度、碳排放、各车辆的工作量平衡

等. 本文在文献[10]的基础上, 集成选址决策, 考虑车辆容量和车辆行驶里程约束, 建立以最小化总成本与最小化各路径间最大长度差为目标的双目标模型, 该模型为文献中首次最小化总成本, 同时以平衡各车辆的工作量为目标的多目标LRPSPD研究. 鉴于变邻域搜索算法在求解组合优化问题中较好的性能, 本文构造四类邻域结构设计多起点变邻域搜索算法, 并设计多蚁群构造多个初始解作为变邻域搜索的多起点, 对文献中的LRPSPD算例进行求解分析.

2 问题描述和数学模型构建

2.1 问题描述与符号说明

定义图 $G = (N, A)$, 其中 $N = N_0 \cup N_C$, N_0, N_C 分别表示车场集合、需求点集合. 每个潜在车场 $k \in N_0$ 有固定的开放与营运成本 f_k , 容量为 C_k . 每个客户 i 的取送货需求分别为 p_i, d_i . 假定车场有足够可供派遣的完全相同的车队, 每辆车最大容量为 Q 、最大行驶距离为 L , 派遣每辆车的固定成本 F . 每辆车从车场出发, 通过恰当的行驶路线有序访问该线路上的所有客户, 然后回到车场, 每辆车最多执行一条路径的服务. $A = \{(i, j) : i, j \in V, i \neq j\}$ 为边集, 每条边 $(i, j) \in A$ 的行驶成本即距离为 c_{ij} . 本文研究的LRPSPD可描述为: 从一系列潜在车场中选择开放的车场, 并从开放的车场派出车辆, 在不超过车辆容量和最大行驶里程的约束下, 找到服务所有客户后回到原车场的路径, 使成本最小化, 同时最大可能地平衡各路径的总长度(即最小化最长路径与最短路径的里程差). 双目标LRPSPD模型中的变量定义如下.

决策变量: $x_{ij} = 1$ 表示先后连续访问客户 i, j , 否则 $x_{ij} = 0$; $y_k = 1$ 表示备选车场 k 被选中开放, 否则 $y_k = 0$; $z_{ik} = 1$ 表示节点 i 分配给车场 k , 否则 $z_{ik} = 0$.

其他变量: Y_{ij} 表示在 (i, j) 上时装载的送货量; Z_{ij} 表示在 (i, j) 上时装载的取货量; l_i 表示访问客户 i 时的累积行驶里程; $tL_{\max(\min)}$ 表示最长(最短)的被派遣车辆行驶路径长度.

2.2 模型构建

基于以上问题描述与符号, 构建以下双目标LRPSPD数学模型:

$$\min f_1 = \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} + \sum_{k \in N_0} f_k y_k + \sum_{k \in N_0} \sum_{i \in N_C} F x_{ki}, \quad (1)$$

$$\min f_2 = tL_{\max} - tL_{\min}, \quad (2)$$

s.t.

$$\sum_{j \in N} x_{ij} = 1, \quad \forall i \in N_C, \quad (3)$$

$$\sum_{k \in N_0} z_{ik} = 1, \quad \forall i \in N_C, \quad (4)$$

$$\sum_{j \in N} x_{ij} = \sum_{j \in N} x_{ji}, \quad \forall i \in N, \quad (5)$$

$$x_{ik} \leq z_{ik}, \forall i \in N_C, k \in N_0, \quad (6)$$

$$x_{ki} \leq z_{ik}, \forall i \in N_C, k \in N_0, \quad (7)$$

$$x_{ij} + z_{ik} + \sum_{m \in N_0, m \neq k} z_{jm} \leq 2, \quad (8)$$

$$\sum_{i \in N_C} d_i z_{ik} \leq C_k y_k, \forall k \in N_0, \quad (9)$$

$$\sum_{i \in N_C} p_i z_{ik} \leq C_k y_k, \forall k \in N_0, \quad (10)$$

$$\sum_{j \in N} Y_{ij} - \sum_{j \in N} Y_{ji} = -d_i x_{ij}, \forall i \in N_C, \quad (11)$$

$$\sum_{j \in N} Z_{ij} - \sum_{j \in N} Z_{ji} = p_i x_{ij}, \forall i \in N_C, \quad (12)$$

$$\sum_{i \in N_C} d_i \times z_{ik} - \sum_{j \in N_C} Y_{kj} = 0, \forall k \in N_0, \quad (13)$$

$$Y_{ij} + Z_{ij} \leq Q, \forall i, j \in N, \quad (14)$$

$$Y_{ij} \leq (Q - d_i) x_{ij}, \forall i \in N_C, j \in N, \quad (15)$$

$$Z_{ij} \leq (Q - p_i) x_{ij}, \forall i \in N, j \in N_C, \quad (16)$$

$$Y_{ij} \geq d_i x_{ij}, \forall i \in N, j \in N_C, \quad (17)$$

$$Z_{ij} \leq p_i x_{ij}, \forall i \in N_C, j \in N, \quad (18)$$

$$l_i - l_j + (L + c_{ij}) x_{ij} + (L - c_{ji}) x_{ji} \leq L, \quad (19)$$

$$\sum_{k \in N_0} c_{ki} x_{ki} \leq l_i \leq L + (c_{ki} - L) \sum_{k \in N_0} x_{ki}, \quad (20)$$

$$l_i \leq L - \sum_{k \in N_0} c_{ik} x_{ik}, \forall i \in N_C, \quad (21)$$

$$tL_{\max} \geq l_i + \sum_{k \in N_0} c_{ik} x_{ik}, \forall i \in N_C, \quad (22)$$

$$tL_{\min} \leq l_i + \sum_{k \in N_0} (c_{ik} - L) x_{ik} + L, \forall i \in N_C, \quad (23)$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in N, \quad (24)$$

$$y_k, z_{ik} \in \{0, 1\}, \forall i \in N_C, k \in N_0, \quad (25)$$

$$l_i \geq 0, \forall i \in N, \quad (26)$$

$$Y_{ij}, Z_{ij} \geq 0, \forall i \in N, j \in N. \quad (27)$$

目标函数(1)使总成本最小, 包括路径成本、车场的开放与运营固定成本和派遣车辆的成本, 目标式(2)最小化车辆的最大行驶里程差, 以平衡各车辆的工作量. 式(3)–(4)规定所有客户均被分配给开放的车场且仅被访问一次, 约束(5)保证进入某客户的弧总数等于从该客户出来的弧总数, 约束(6)–(8)消除了未始于车场或止于车场的子路径, 且当且仅当分配给相同的车场时, 才有可能被连续访问, 式(9)–(10)为车场容量约束, 式(11)–(12)为访问客户点前后的流量平衡约束, 同时表明客户的需求在一次访问中全部满足, 约束(13)保证从某开放车场运出的装载量等于分配给该

车场的所有客户的送货总需求, 约束(14)保证在任意弧上的装载量均不超过车辆容量, 约束式(15)–(18)是对上下界进行限定, 约束(19)–(21)保证车辆运送距离不超出最大可行驶距离, 且当且仅当车被派遣时才存在累积行驶距离; 式(22)–(23)界定了车辆行驶路径的长度, 式(24)–(27)为各变量的取值约束.

通常, 有效不等式的加入可缩紧模型约束, 从而消除模型解空间的无效片段. 式(28)为式(14)的有效替换, 其中 M 为极大常数, 该式表示当且仅当客户被连续访问时, 弧上存在流量, 且不超过车辆的最大容量.

$$Y_{ij} + Z_{ij} \leq \min(Q, Mx_{ij}), \forall i, j \in N. \quad (28)$$

3 多起点变邻域搜索算法

由于双目标模型并不存在绝对的最优解, 因此本文设计优化算法求解模型Pareto解. 变邻域搜索(variable neighborhood search, VNS)作为一种元启发式方法已被成功应用于解决多种优化问题, 尤其对于大规模组合优化问题效果较好. VNS求解性能受初始解影响较大, 因此本文构造有一定关联的多个初始解, 并基于各初始解进行变邻域搜索, 称为多起点变邻域搜索算法(multi-start VNS), 算法流程如图1所示. 分步说明如下:

步骤1 初始化VNS产生可行解及不可行解的次数 $n_F = 0$, $n_{\text{INF}} = 0$ 及多起点数 $n_{\text{MS}} = 0$, 并分别设定其最大值;

步骤2 分别基于选址蚁群、分配蚁群及车辆路径安排蚁群构造初始解;

步骤3 从初始解的多个邻域中随机选择一个邻域结构进行搜索;

步骤4 检查产生的邻域解是否为可行解.

若不可行: 则 $n_{\text{INF}} = n_{\text{INF}} + 1$ 并判断生成不可行解的次数是否达到最大, 否则回到步骤3, 是则跳到步骤5;

若可行: 则判断该可行解是否为Pareto解, 若是则产生了多目标问题的新解, 本轮搜索结束, 初始化 $n_F = 0$, $n_{\text{INF}} = 0$, 开始新一轮搜索, 回到步骤3;

若不是Pareto解, 则 $n_F = n_F + 1$ 并判断是否满足达到最大可生成可行解次数, 回到步骤3或转步骤5;

步骤5 判断是否达到最大起点数: 若 $n_{\text{MS}} = n_{\text{MS}} + 1$ 否则同时更新各蚁群信息素矩阵, 回到步骤2; 若是则输出并结束程序.

根据以上步骤说明, 变邻域搜索的每一个初始解(起点), 均在每一次迭代开始时, 基于此前迭代产生的解以及更新后的信息素由多蚁群算法产生, 从而使变邻域搜索的多个起点间相互关联, 且随迭代发生变邻域搜索的起点对应的解更优.

$$\phi_{ik} = \frac{1}{c_{ik}}, \Phi_{ik} = \min_{k \in O_s} \left| \sum_{i \in I_k} d_i z_{ik} - \sum_{i \in I_k} q_i z_{ik} \right|, \quad (34)$$

分配结束后, 检查分配给各车场的所有需求点总需求是否超出车场容量, 并对分配给容量超出的车场的需求点进行调整, 调整方案伪代码如下.

算法1:

```

初始化调整需求点分配的次数  $n_A = 0$ ;
检查车场容量是否超出, 分别记录超出及不超出的车场id_ex, id_unex
while (id_ex  $\neq \emptyset$  且 id_unex  $\neq \emptyset$ , 且  $n_A \leq n_{A-\max}$ )
    从id_ex车场中随机选择一个需求点重新分配到id_unex车场中;
     $n_A = n_A + 1$ 
    重新检查车场容量约束, 更新id_ex, id_unex;
end while
if (id_ex  $\neq \emptyset$  且 id_unex  $\neq \emptyset$ ) 或 ( $n_A = n_{A-\max}$ ) //所有
    开放的车场均超出容量或达到最大调整次数后仍有
    车场超出容量
     $p_s = p_s + 1$ , 回到选址阶段重新选择车场后进行
    重新分配;
else //该分配方案下所有车场均不超出容量
    输出分配方案, 基于分配方案进行路径选择;
end if

```

3.1.3 路径选择

选址、分配确定后, LRPSPD可视为多个单车场的VRPSPD, 该节分别确定分配给同一车场的需求点的访问路径. τ_{ij}^s 为构造第s个初始解时弧 (i, j) 上的信息素强度, 蚂蚁根据各条路径上的信息量和反映网络本身特性的效用函数 $(\psi_{ij})^{\beta'} (\zeta_{ij})^{\gamma'}$ 决定下一步要转移的点, P_{ij} 表示蚂蚁由节点*i*转移到节点*j*的概率. 其中 q'' 为均匀分布在 $[0, 1]$ 上的随机数, q_0'' 为预先设定的值, 当 $q \leq q_0$ 时, 蚂蚁直接移动到使效用函数 $(\psi_{ij})^{\beta'} (\zeta_{ij})^{\gamma'}$ 取最大值的点*j*如式(35), 否则按式(36)所示概率进行轮盘赌选择转移到*j*点, q_0'' 越大表明蚂蚁随机性越小, Z_{is} 表示第s次初始解蚂蚁到达节点*i*上时未访问的节点集合, 当达到行驶距离约束或容量约束条件时, 返回车场, 并再次从该车场开始构建新的路径, 直到所有节点都被访问, 即

$$j = \begin{cases} \arg \max_{j \in Z_{ir}} [\xi_{ij}^s (\psi_{ij})^{\beta'} (\zeta_{ij})^{\gamma'}], & q'' \leq q_0'', \\ J, & \text{其他,} \end{cases} \quad (35)$$

$$J: P_{ij}^s = \frac{\xi_{ij}^s (\psi_{ij})^{\beta'} (\zeta_{ij})^{\gamma'}}{\sum_{j \in Z_{ir}} \xi_{ij}^s (\psi_{ij})^{\beta'} (\zeta_{ij})^{\gamma'}}, \quad (36)$$

$$\psi_{ij} = \frac{d_{ki} + d_{jk} - d_{ik}}{d_{ij}}, \quad (37)$$

$$\zeta_{ij} = \frac{1}{|(p_i - d_i) - (d_j - p_j)|}, \forall j \in Z_{is}. \quad (38)$$

式(38)中当点的净装载量等于点的净卸载量时 $\zeta_{ij} = 1$. ψ_{ij} 与 ζ_{ij} 体现网络本身特征, 结合蚂蚁转移规则反应出蚂蚁具有偏好以最大节约距离服务取送货需求差最小的客户的特性. 参数 β' 和 γ' 分别表示 ψ_{ij} 与 ζ_{ij} 相对于信息素对转移概率的影响程度.

3.2 多目标变邻域搜索

本文变邻域搜索采用四类邻域结构: 路径内点序列操作、路径间点序列操作、从不同车场出发的两条路径交换、开放车场与未开放车场交换. 每次邻域操作均在随机选择的扰动机制下被执行, 且均以找到Pareto解为目标, 变邻域搜索过程定义了最大无效操作次数(产生不可行解次数 $n_{INF-\max}$ 、产生可行但为支配解的次数 $n_{F-\max}$), 即若产生 $n_{INF-\max}$ 个不可行解或 $n_{F-\max}$ 个支配解后, 仍未找到新的Pareto解, 则搜索结束; 若找到新的Pareto解, 则初始化 $n_{INF} = 0$, $n_F = 0$, 继续执行变邻域操作. 以下为第s次迭代中多目标变邻域搜索操作伪代码.

算法2:

```

输入多蚁群算法产生的初始解  $x_0^s$ , 以及对应的目标
 $f_1(x_0^s), f_2(x_0^s)$ ; 初始化  $n_{INF} = 0, n_F = 0$ ; 设置  $n_{INF-\max},$ 
 $n_{F-\max}$ ;
初始化产生的新Pareto解集合  $X_P = [x_0^s]$ ;
while ( $n_{INF} < n_{INF-\max}$  且  $n_F < n_{F-\max}$ )
    [ $n_P, -$ ] = size( $X_P$ ); //  $n_P$  记录Pareto解的个数
    随机选择一种邻域结构对解  $x_0^s$  进行操作, 产生新
    解  $x_{temp}^s$ ; // 邻域构造见第2.2.1-2.2.4节
    if (解  $x_{temp}^s$  为可行解)
        dominate = 0; // 初始化标注值
        for  $k = 1 : n_P$ 
            if  $f_1(x_{temp}^s) < f_1(X_P(k))$  且  $f_2(x_{temp}^s) \leq f_2(X_P(k))$ 
                // 新解支配Pareto解集中的解
                if (dominate = 0) // 新解未被记录到  $X_P$  中
                     $X_P(k) = x_{temp}^s$ ; // 被支配解用新解替换
                    dominate = 1; // 标注新解已记录到  $X_P$  中
                else // 新解已记录到  $X_P$  中
                     $X_P(k) = []$ ; // 被支配解从  $X_P$  中删除
                end if
            else if  $f_1(x_{temp}^s) < f_1(X_P(k))$  且  $f_2(x_{temp}^s) >$ 
 $f_2(X_P(k))$  或  $f_1(x_{temp}^s) > f_1(X_P(k))$  且  $f_2(x_{temp}^s) <$ 
 $f_2(X_P(k))$  // 不支配解集中任何一个解的新Pareto解
                 $X_P = [X_P; x_{temp}^s]$ ; // 新解保存至解集
                dominate = 1;
            end if
        end for
    if (dominate = 1) // 产生了新Pareto解

```

```

 $n_{\text{INF}} = 0; n_F = 0; //$ 无效操作次数归零
else
 $n_F = n_F + 1; //$ 未产生新Pareto解
end if
else  $//$ 解 $x_{\text{temp}}^s$ 为不可行解
 $n_{\text{INF}} = n_{\text{INF}} + 1;$ 
end if
end while
return  $X_P$ 

```

3.2.1 路径内操作

1) 插入: 随机选择路径上的需求点子序列 σ , 从当前位置删除, 插入到该路径的其他位置, $|\sigma|$ 小于该路径上总需求点个数. 如图2中a所示.

2) 交换: 随机选择路径上两个不重叠的子序列 (σ_1, σ_2) , 交换两个序列的位置, 子序列上需求点个数根据该路径上总需求点个数随机选择, $|\sigma_1| + |\sigma_2|$ 不

超过该路径上总需求点个数. 如图2中b所示.

3) 翻转: 随机选择路径上的需求点子序列 σ , 在当前位置进行翻转, $|\sigma|$ 小于该路径上总需求点个数. 如图2中c所示.

3.2.2 路径间操作

1) 插入: 随机选择一条路径 r_1 , 并随机选择子序列 σ 从当前位置删除, 随机选择另一条路径 r_2 , 将 σ 插入该路径中的任意位置, $|\sigma|$ 不超过该路径上总需求点个数, 若 $|\sigma|$ 等于 r_1 上总需求点个数, 则 r_1 整条路径被插入其他路径中, 删除路径 r_1 . 如图3中a所示.

2) 交换: 随机选择两条路径 r_1, r_2 , 分别选取子序列 (σ_1, σ_2) , 交换两个序列的位置. 如图3中b所示.

3) 翻转交换: 随机选择两条路径 r_1, r_2 , 分别选取子序列 (σ_1, σ_2) , 交换两个序列的位置同时分别将子序列进行翻转. 如图3中c所示.

$|\sigma_1|, |\sigma_2|$ 均不超过路径 r_1, r_2 上总需求点个数.

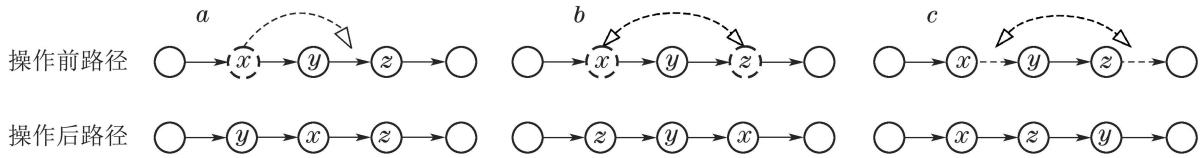


图2 路径内局部搜索操作示意图

Fig. 2 Procedure of Intra-route local search

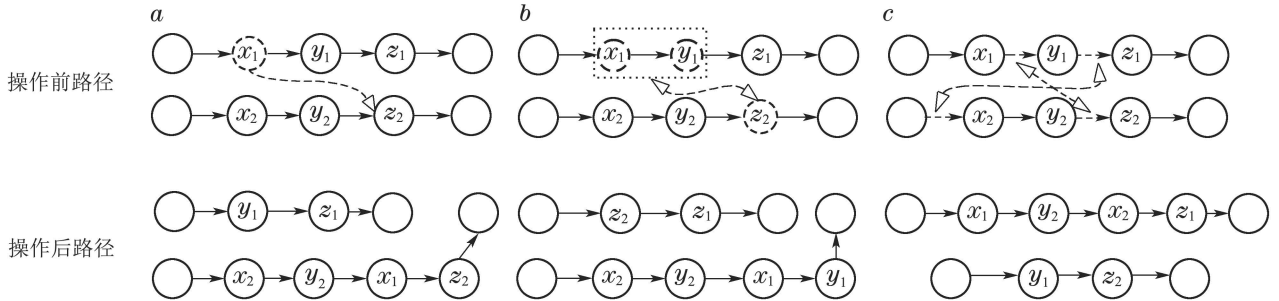


图3 路径间局部搜索操作示意图

Fig. 3 Procedure of Inter-route local search

3.2.3 路径操作

随机选择从不同车场出发的两条路径, 整条路径进行交换, 即交换两条路径的出发车场.

3.2.4 车场操作

若存在未开放的车场, 随机选择一个开放车场 D_1 与一个未开放车场 D_2 , 交换两个车场, 即开放车场 D_2 , 将从原开放车场 D_1 出发的所有路径, 全部调整为从 D_2 出发, 关闭车场 D_1 .

3.3 信息素更新准则

基于一个初始解达到最大变邻域搜索次数后, 输出产生的Pareto解集. 基于第3.1节中描述的多蚁群构造初始解的方法, 以及Pareto解集对多蚁群信息素矩

阵的影响, 构造新的初始解, 本文设计选址、分配和路径选址蚁群信息素更新主要受Pareto解集中总成本最小(各车辆行驶旅程差最大)及各车辆行驶旅程差最小(总成本最大)两个特殊的Pareto解的影响. 令该两个Pareto解为 X_1, X_2 , 任意Pareto解对应的两个目标函数值分别为 $f_1(\cdot), f_2(\cdot)$, 各蚁群的信息素更新准则如式(39)–(44), 式中 ρ, ρ', ρ'' 为各蚁群信息素挥发系数, n_k 为分配给车场的需求点数量, $tL(\cdot)$ 为对应解的总车辆行驶路径长. 设选址、分配和路径选址蚁群初始信息素水平为 $1/f_k$ 、极小的数和 $1/d_{ij}$,

$$\tau_k^{s+1} = (1 - \rho)\tau_k^s + \rho\Delta\tau_k, \quad (39)$$

$$\Delta\tau_k = \begin{cases} [f_1(X_2) - f_1(X_1)/f_1(X_2)] + \\ [f_2(X_1) - f_1(X_2)/f_2(X_1)], & i \in X_1 \cup X_2, \\ 0, & \text{其他}, \end{cases} \quad (40)$$

$$\vartheta_{ik}^{s+1} = (1 - \rho')\vartheta_{ik}^s + \rho'\Delta\vartheta_{ik}, \quad (41)$$

$$\Delta\vartheta_{ik} = \begin{cases} [f_1(X_2) - f_1(X_1)/f_1(X_2)] + \\ [f_2(X_1) - f_1(X_2)/f_2(X_1)] \times |n_k|, & i \in X_1 \cup X_2, \\ 0, & \text{其他}, \end{cases} \quad (42)$$

$$\xi_{ij}^{s+1} = (1 - \rho'')\xi_{ij}^s + \rho''\Delta\xi_{ij}, \quad (43)$$

$$\Delta\xi_{ij} = \begin{cases} tL(X_2) - tL(X_1)/tL(X_2), & i \in X_1 \cup X_2, \\ 0, & \text{其他}. \end{cases} \quad (44)$$

4 算例求解与结果分析

根据以往文献,对蚁群算法中的参数范围进行经验性设置:信息素挥发系数 $\rho, \rho', \rho'' \in [0.2, 0.8]$,启发式因子 $\alpha \in [2, 5]$, $\beta, \beta' \in [0, 5]$, $\gamma, \gamma' \in [0, 5]$,随机因子 $q_0, q'_0, q''_0 \in [0.2, 0.8]$.初次分配结束后,对车场容量进行检查,若容量超出对需求点的分配进行调整以确定是否需要额外开放一个车场,分配调整的最大次数 $n_{A-\max} \in [0.5, 1, 1.52] \times |N_C|$.MSVNS中起点数 $n_{MS} \in [0.5, 1, 1.52] \times |N|$,每一次VNS允许连续产生可行且为支配解的最大次数 $n_{F-\max} \in [0.5, 1, 1.52] \times |N|$,允许连续产生不可行解的最大次数为 $n_{INF-\max} = n_{F-\max}$.基于以上参数范围,从本文采用的Karaoglan等^[13]中随机选择4组不同规模的算例进行多组参数组合试验,获得最佳解的参数组合为: $\rho, \rho', \rho'' = 0.2, 0.2, 0.2$, $\alpha = 2$, $\beta, \beta' = 3, 3$, $\gamma, \gamma' = 1, 2$, $q_0, q'_0, q''_0 = 0.7, 0.7, 0.8$, $n_{A-\max} = |N_C|$, $n_{MS} = |N|$, $n_{F-\max} = 2 \times |N|$.

4.1 算例求解结果

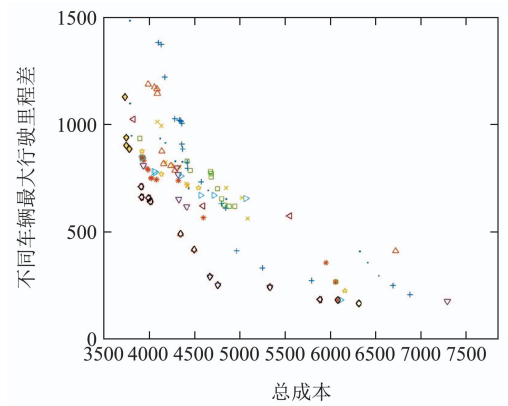
Karaoglan等^[13]根据两种不同的需求拆分策略(demand separate strategy, DSS)基于两组LRP算例构造了4组测试算例:BAM, BSN, PAM, PSN, 分别简称B, P系列,应用文献^[13]中的算例,并设置最大行驶里程为一个较大的常数.运用MATLAB (R2014b)编写代码,在Windows 7操作系统,处理器为Intel (R) Core (TM) i5-4460 (3.20 Ghz), 8 GB内存的个人电脑上运行程序,每组算例运行10次,不同次运行产生的Pareto解存在相互支配,因此综合10次运行结果得到最优Pareto前沿.基于综合后的最优Pareto前沿,决策者可根据实际偏好进行决策.

4.1.1 边界Pareto解分析

图4(a)(b)分别为BAM, PAM系列中随机算例的所有解集,最优Pareto前沿如菱形所示.图4(a)中不同解的最大行驶里程差波动较大,即不同的选址-路径决策会导致不同车辆的工作量相差较大,而图4(b)中不同决策下车辆的工作负荷相对平均.设近似Pareto解集中100%偏向最小化总成本时对应的两个目标函数值分别为 $f_{1-PCost}$ 与 $f_{2-PCost}$;100%偏向最小化车辆间最大行驶里程差时为 $f_{1-PTDGap}$ 与 $f_{2-PTDGap}$,则

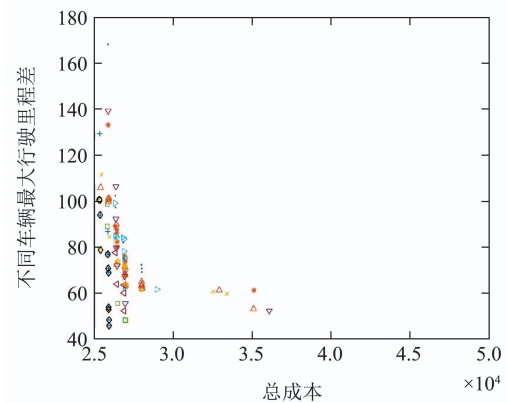
$$\Delta G/C = \frac{f_{2-PCost} - f_{2-PTDGap}}{f_{1-PTDGap} - f_{1-PCost}}$$

反映了车辆间最大行驶里程差随成本变化的最大波动率.



- + 第1次运行Pareto解集
- + 第2次运行Pareto解集
- + 第3次运行Pareto解集
- + 第4次运行Pareto解集
- + 第5次运行Pareto解集
- + 第6次运行Pareto解集
- + 第7次运行Pareto解集
- + 第8次运行Pareto解集
- + 第9次运行Pareto解集
- + 第10次运行Pareto解集
- ◇ 最优Pareto解前沿

(a) Min92-27 $\times$ 5-W



- + 第1次运行Pareto解集
- + 第2次运行Pareto解集
- + 第3次运行Pareto解集
- + 第4次运行Pareto解集
- + 第5次运行Pareto解集
- + 第6次运行Pareto解集
- + 第7次运行Pareto解集
- + 第8次运行Pareto解集
- + 第9次运行Pareto解集
- + 第10次运行Pareto解集
- ◇ 最优Pareto解前沿

(b) 50-5-3a-W

图4 不同系列随机两算例10次运行近似Pareto解集
Fig. 4 Pareto solutions of 10 runs for 2 random instances from different series

表1-2列出各算例的 $\Delta G/C$ 值, B系列算例车辆间最大行驶里程差随成本变化的最大波动率最大值分别达178.24, 235.34, 平均值为58.59, 86.5, 表明B系列算例不同Pareto解对应的选址-车辆路径方案, 在总成本变化不大时车辆间工作量相差较大, 体现出决策时考虑车辆间工作量平衡目标的重要性. 而表2中P系列算例的 $\Delta G/C$ 值较小, 平均值分别为4.48, 14.39, 即: 在不同决策下, 各车辆间工作量均相对平均.

表1 B系列算例边界Pareto解

Table 1 Pareto with extreme OFVs for instances sets B

算例	BAM		BSN	
	DSS	$\Delta G/C$	DSS	$\Delta G/C$
Srivastava86-8 $\times$ 2	W	0.00	X	57.38
	Z	172.66	Y	78.90
Perl83-12 $\times$ 2	W	10.04	X	37.25
	Z	14.31	Y	51.90
Gaskell67-21 $\times$ 5	W	178.24	X	67.75
	Z	62.29	Y	53.10
Gaskell67-22 $\times$ 5	W	39.36	X	102.94
	Z	101.43	Y	110.04
Min92-27 $\times$ 5	W	37.22	X	79.91
	Z	39.51	Y	235.31
Gaskell67-29 $\times$ 5	W	24.13	X	118.34
	Z	44.04	Y	235.34
Gaskell67-32 $\times$ 5.1	W	60.90	X	77.24
	Z	152.67	Y	178.26
Gaskell67-32 $\times$ 5.2	W	129.51	X	21.60
	Z	81.88	Y	91.52
Gaskell67-36 $\times$ 5	W	60.85	X	25.66
	Z	14.09	Y	40.73
Christofides 69-50 $\times$ 5	W	39.98	X	94.02
	Z	72.95	Y	162.38
Perl83-55 $\times$ 15	W	24.92	X	48.20
	Z	31.51	Y	107.47
Christofides 69-75 $\times$ 10	W	34.91	X	37.08
	Z	28.06	Y	161.11
Perl83-85 $\times$ 7	W	39.99	X	11.67
	Z	24.63	Y	31.78
Daskin95-88 $\times$ 8	W	60.00	X	26.35
	Z	47.77	Y	93.65
Christofides 69-100 $\times$ 10	W	50.84	X	79.60
	Z	78.88	Y	78.52
平均值		58.59		86.50

4.1.2 邻域结构性能分析

为了分析四类邻域结构对算法性能的影响, 随机选取不同规模的算例, 逐一删除四类邻域中的某一类邻域进行测试, 获得成本最小的解, 如表3所示, 其中Gap%表示删除邻域后获得的解与使用全四类邻域的解的偏差, 如第4列:

$$\text{Gap}\% = \frac{\text{Cost}_{.1} - \text{Cost}_{\text{All}}}{\text{Cost}_{\text{All}}} \times 100\%.$$

表2 P系列算例边界Pareto解

Table 2 Pareto with extreme OFVs for instances sets P

算例	PAM		PSN	
	DSS	$\Delta G/C$	DSS	$\Delta G/C$
20-5-1a	W	2.54	X	0.89
	Z	2.27	Y	4.23
20-5-1b	W	0.93	X	199.6
	Z	1.02	Y	99.90
20-5-2a	W	2.72	X	3.45
	Z	2.01	Y	2.99
20-5-2b	W	46.91	X	0.90
	Z	3.10	Y	0.87
50-5-1a	W	1.38	X	0.89
	Z	1.66	Y	1.08
50-5-1b	W	0.94	X	1.91
	Z	0.62	Y	0.91
50-5-2a	W	4.95	X	5.12
	Z	5.88	Y	45.68
50-5-2b	W	1.85	X	5.04
	Z	42.70	Y	89.53
50-5-3a	W	9.02	X	0.93
	Z	1.29	Y	0.89
50-5-3b	W	2.05	X	0.87
	Z	15.68	Y	0.84
100-5-1a	W	0.42	X	0.47
	Z	0.83	Y	0.71
100-5-1b	W	1.40	X	1.70
	Z	1.90	Y	1.85
100-5-2a	W	0.25	X	0.16
	Z	0.34	Y	0.43
100-5-2b	W	0.43	X	3.22
	Z	0.40	Y	5.05
100-10-1a	W	0.58	X	2.16
	Z	0.58	Y	4.06
100-10-1b	W	0.69	X	21.26
	Z	1.52	Y	5.87
100-10-2a	W	0.42	X	1.09
	Z	0.66	Y	0.76
100-10-2b	W	1.04	X	1.23
	Z	0.33	Y	1.41
平均值		4.48		14.39

由于算法包含多蚁群、多起点和变邻域搜索3种机制, 在3种机制的组合下算法达到较好的性能, 而非仅由邻域结构确定; 另外, 对于不同的算例, 某一种邻域操作产生的多样化具有较大的随机性, 即某一种邻域结构操作对某一算例可能产生更优的解、同时也未可能产生更优的解, 这解释了表3中删除单一邻域结构后对解的影响不甚明显的原因.

4.1.3 算法性能分析

本文以100%偏向总成本最小函数选取近似Pareto解集中成本最小的解与文献中以最小化总成本为目

标求得的解进行对比, 提取10次运行求得解集中成本最小的解, 分别以最优解和平均解与上界进行比较, 得到最小偏差(Gap%_{min})和平均偏差(Gap%_{Avg}).

B系列算例的解与Karaoglan等^[11]获得的上界、B&C算法求得的解、以及模因算法(MA)^[16]进行对比, 见表4-5, 结果显示本文提出的MSVNS可在极短的时间内(平均分别为4.3 s, 6.12 s)求解BAM, BSN系列所

有算例包含成本近似最优的Pareto解, 与上界相比最优解的平均偏差分别为17.57%, 17.39%, 平均解的平均偏差为17.94%, 17.78%, 最优解与平均解相差小于0.5%表明了算法的稳定性. 尽管整体上解与上界的偏差相较于B&C有一定的差距, 但在求解速度上有明显优势, 并且部分算例(Perl83-55 × 15及Perl83-85 × 7)的解比B&C更优.

表 3 邻域结构性能分析

Table 3 Performance analysis of neighborhoods

算例	全部邻域 Cost_All	删除一类邻域		删除二类邻域		删除三类邻域		删除四类邻域	
		Cost ₁	Gap%	Cost ₂	Gap%	Cost ₃	Gap%	Cost ₄	Gap%
Perl83-12 × 2_W	248.17	252.15	1.60	248.17	0.00	248.17	0.00	248.17	0.00
20-5-1a_W	26080.00	26125.00	0.17	26135.00	0.21	26123.00	0.16	27578.00	5.74
50-5-1a_W	35235.45	36127.00	2.53	36110.00	2.48	36123.00	2.52	39874.00	13.16
100-5-1a_W	160565.34	161412.87	0.53	161542.00	0.61	161474.91	0.57	163965.24	2.12
100-10-1a_W	225657.50	225692.33	0.02	225723.34	0.03	225657.50	0.00	230123.50	1.98

表 4 BAM算例MSVNS算法与B&C结果比较

Table 4 Results comparison by MSVNS against B&C for BAM instances

算例	DSS	UB	B&C		MA			MSVNS		
			Gap%	CPU	Gap% _{min}	Gap% _{Avg}	CPU	Gap% _{min}	Gap% _{Avg}	CPU
Srivastava86-8 × 2	W	873.58	0.00	0.00	0.00	0.00	6.49	0.00	0.03	0.17
	Z	806.06	0.00	0.00	0.00	0.00	10.16	4.22	4.50	0.16
Perl83-12 × 2	W	243.98	0.00	0.57	0.00	0.00	2.37	1.68	1.73	0.29
	Z	243.98	0.00	0.65	0.00	0.00	2.88	1.68	1.78	0.28
Gaskell67-21 × 5	W	528.42	0.00	290.18	0.00	0.00	13.59	8.32	8.94	0.63
	Z	513.30	0.00	89.43	0.00	0.00	14.71	2.33	3.02	0.69
Gaskell67-22 × 5	W	653.80	0.00	3.76	0.00	0.00	11.36	12.30	12.62	0.67
	Z	653.80	0.00	2.66	0.00	0.00	11.00	9.57	9.92	0.75
Min92-27 × 5	W	3142.02	0.00	19.66	0.00	0.00	17.07	15.73	15.76	0.93
	Z	3142.02	0.00	18.12	0.00	0.00	17.92	13.20	13.64	1.33
Gaskell67-29 × 5	W	592.10	0.00	6.76	0.00	0.00	15.53	15.20	15.58	1.09
	Z	592.10	0.00	100.91	0.00	0.00	13.36	17.82	18.19	1.22
Gaskell67-32 × 5.1	W	696.38	0.00	5694.09	0.00	0.00	22.86	17.12	17.42	1.54
	Z	643.37	0.00	259.67	0.00	0.00	24.87	21.71	21.90	1.58
Gaskell67-32 × 5.2	W	595.27	0.00	15.68	0.00	0.00	22.39	21.93	22.42	2.11
	Z	564.33	0.00	30.90	0.00	0.00	24.67	20.08	20.53	2.07
Gaskell67-36 × 5	W	540.37	0.00	27.20	0.00	0.00	18.24	29.08	29.73	3.06
	Z	540.37	0.00	46.46	0.00	0.00	18.80	29.92	30.63	2.52
Christofides 69-50 × 5	W	708.37	4.52	14400	4.73	4.74	44.27	35.47	35.72	3.75
	Z	701.91	3.67	14400	3.63	3.67	42.37	32.76	33.04	4.17
Perl83-55 × 15	W	1330.85	20.93	14400	26.10	26.11	429.14	15.01	15.69	4.46
	Z	1338.30	20.81	14400	24.79	24.79	429.08	10.68	11.34	4.39
Christofides 69-75 × 10	W	1177.65	17.55	14400	16.67	16.67	86.07	34.49	34.65	7.89
	Z	1108.82	12.81	14400	11.38	11.49	84.37	31.09	31.21	7.75
Perl83-85 × 7	W	1901.09	23.56	14400	27.61	27.61	299.91	16.85	17.35	13.16
	Z	1893.23	23.41	14400	27.88	27.96	307.91	14.63	15.09	11.35
Daskin95-88 × 8	W	533.37	21.96	14400	20.08	20.56	33.80	29.18	29.52	22.22
	Z	487.81	15.57	14400	16.55	16.55	29.73	29.87	30.46	20.12
Average			6.23	5980.22	6.41	6.43	73.39	17.57	17.94	4.30

表 5 BSN算例MSVNS算法与B&C结果比较

Table 5 Results comparison by MSVNS against B&C for BSN instances

算例	DSS	UB	B&C		MA			MSVNS		
			Gap%	CPU	Gap% _{min}	Gap% _{Avg}	CPU	Gap% _{min}	Gap% _{Avg}	CPU
Srivastava86-8 × 2	X	625.43	0.00	0.38	0.00	0.00	7.62	0.00	0.00	0.20
	Y	625.43	0.00	0.26	0.00	0.00	9.1	0.00	0.00	0.15
Perl83-12 × 2	X	242.41	0.00	0.60	0.00	0.00	1.97	2.23	2.33	0.22
	Y	242.41	0.00	0.55	0.00	0.00	2.57	3.96	4.24	0.24
Gaskell67-21 × 5	X	454.48	0.00	13.34	0.00	0.00	13.87	6.62	7.17	0.73
	Y	454.48	0.00	8.98	0.00	0.00	15.55	8.35	8.91	1.03
Gaskell67-22 × 5	X	629.51	0.00	109.62	0.00	0.00	7.48	12.94	13.20	0.79
	Y	629.51	0.00	64.99	0.00	0.00	7.23	7.89	8.05	0.49
Min92-27 × 5	X	2998.80	0.00	91.87	0.00	0.00	12.78	5.98	6.45	2.03
	Y	2998.80	0.00	190.62	0.00	0.00	13.286	7.21	7.67	1.83
Gaskell67-29 × 5	X	490.34	0.00	6.63	0.00	0.00	13.6	16.56	17.05	1.71
	Y	490.34	0.00	6.57	0.00	0.00	13.77	16.23	16.73	1.53
Gaskell67-32 × 5.1	X	563.47	0.00	18.77	0.00	0.00	25.15	20.50	20.64	1.99
	Y	563.47	0.00	27.31	0.00	0.00	25.21	20.39	20.81	2.70
Gaskell67-32 × 5.2	X	507.03	0.00	13.72	0.00	0.00	26.27	23.91	24.43	3.52
	Y	507.03	0.00	7.87	0.00	0.00	27.26	17.54	18.03	4.78
Gaskell67-36 × 5	X	494.86	0.00	18.36	0.00	0.00	17.85	26.83	27.39	2.73
	Y	494.86	0.00	18.13	0.00	0.00	18.64	24.40	25.06	3.93
Christofides 69-50 × 5	X	576.69	2.40	14400	2.46	2.94	41.64	36.66	36.70	5.75
	Y	578.97	2.69	14400	2.91	2.93	41.96	38.05	38.20	5.55
Perl83-55 × 15	X	985.56	2.94	14400	2.72	2.72	143.68	12.28	12.91	4.92
	Y	988.06	3.14	14400	2.66	2.66	143.46	10.95	11.63	6.43
Christofides 69-75 × 10	X	888.16	10.33	14400	7.77	7.93	76.79	33.62	34.38	9.87
	Y	884.00	8.90	14400	6.57	6.62	77.74	35.75	36.49	11.33
Perl83-85 × 7	X	1381.57	5.88	14400	3.69	3.78	147.21	15.45	15.84	14.36
	Y	1368.64	4.96	14400	3.67	3.68	143.57	17.07	17.73	14.08
Daskin95-88 × 8	X	399.17	11.20	14400	0.00	0.42	35.21	31.36	31.53	42.95
	Y	401.98	0.00	14400	0.00	0.69	30.62	34.17	34.38	25.44
Average			2.20	5779.75	1.16	1.23	40.75	17.39	17.78	6.12

P系列算例的解与文献[11]上界及B&C获得的解、以及混合遗传算法 (hybrid genetic algorithm, H-GA)^[14]求得的解进行对比, 如表6-7, 结果表明MSVNS对P系列算例极好的适应性, 72个算例中共46个算例获得了比上界更好的解, 最好解的平均偏差分别为-1.7, -2.08, 平均解的平均偏差分别为-1.59, -1.92, 表明算法稳定性, 同时说明本文提出的MSVNS求得平均解的结果优于文献中其他的算法, 且求解时间同样具有明显优势。

为进一步比较MSVNS求解性能, 表8从获得新最优解的数量及平均求解时间与文献[15]中两种模拟退火(simulated annealing, SA1, SA2)及多起点模拟退火算法(multi-start SA, MSA)进行比较. 对比P系列不同规模算例的求解结果, MSVNS 仅在50 × 5规模算例上获得的新最优解略少于SA, MSA, 在其他规模算例

上均获得了更多最优解, 解的精度和平均求解时间均体现出MSVNS求解性能更优。

5 结论

本文考虑车辆容量和最大行驶里程限制, 构建了最小化总成本和最小化不同车辆行驶距离最大差以平衡各车辆的工作负荷的双目标LRPSPD模型. 提出一种多起点变邻域搜索算法(MSVNS)进行求解, 首先基于多蚁群算法构建初始解, 基于初始解设计四类邻域结构进行随机变邻域多目标搜索, 根据所得Pareto邻域解更新信息素, 而后产生新的初始解作为变邻域搜索起点. 本文用MSVNS算法求解文献中的算例, 解得Pareto前沿的分布情况说明考虑最小化路径间最大长度差目标的重要意义. 与其他求解LRPSPD方法求解结果的比较表明了算法在求解速度上的绝对优势; 尤其对于P系列算例, MSVNS在求解精度和求解

时间均表现出更优性能,说明MSVNS算法可求得权衡各目标的Pareto解,并使得最小总成本目标近似最优.

文中算法对B系列与P系列求解结果精度差异较大,一方面,算法对于不同算例的普适性需进一步提高,另一方面,不同系列算例与不同算法匹配规律有

待进一步分析研究.采用容量较小的车辆并派遣其访问偏远节点,对于平衡车辆的行驶距离,充分利用车辆的装载空间有重要的意义,因此多车型的LRSPD为未来研究的重要方向之一.基于现实物流情形,同时考虑如时间窗约束、随机行驶时间等其他因素的LRSPD将是非常值得研究的领域.

表 6 PAM算例MSVNS算法与文献中已有结果比较

Table 6 Results comparison by MSVNS against existing results in literature for PAM instances

算例	DSS	B&C		MA			MSVNS		
		Gap%	CPU	Gap% _{min}	Gap% _{Avg}	CPU	Gap% _{min}	Gap% _{Avg}	CPU
20-5-1a	W	0.00	232.48	0.04	0.04	8.33	-1.45	-1.20	0.66
	Z	0.00	75.24	0.00	0.00	7.27	-5.56	-5.56	0.66
20-5-1b	W	0.00	3.22	0.00	0.00	8.45	-8.17	-8.10	0.72
	Z	0.00	0.94	0.00	0.00	8.04	-8.23	-8.23	0.70
20-5-2a	W	0.00	37.79	0.00	0.00	7.61	-1.57	-1.57	0.71
	Z	0.00	73.81	0.00	0.00	7.34	-5.38	-5.35	0.71
20-5-2b	W	0.00	7.09	0.00	0.00	7.22	-13.93	-13.65	0.71
	Z	0.00	4.40	0.00	0.00	7.56	-13.79	-13.36	0.69
50-5-1a	W	12.28	14400.0	10.79	12.75	46.46	4.11	4.11	3.47
	Z	0.30	14400.0	0.21	0.23	44.24	3.63	3.70	3.76
50-5-1b	W	0.21	14400.0	0.16	0.16	43.98	4.30	4.30	3.68
	Z	0.18	14400.0	0.14	0.15	39.53	0.49	0.49	3.67
50-5-2a	W	2.54	14400.0	2.57	2.59	45.06	-0.21	-0.21	3.36
	Z	0.16	14400.0	0.12	0.12	42.31	-2.66	-2.66	3.25
50-5-2b	W	0.13	14400.0	0.12	0.12	44.71	-3.32	-3.03	3.79
	Z	0.11	14400.0	0.11	0.1	42.69	-3.33	-3.33	4.27
50-5-3a	W	4.65	14400.0	4.77	4.81	34.91	7.06	7.09	3.83
	Z	0.26	14400.0	0.22	0.24	32.56	2.84	2.84	3.34
50-5-3b	W	0.21	14400.0	0.21	0.21	37.08	-1.38	-1.28	4.54
	Z	0.13	14400.0	0.13	0.13	36.46	-1.39	-1.39	4.24
100-5-1a	W	1.36	14400.0	0.78	1.11	228.54	0.73	0.88	12.07
	Z	0.73	14400.0	0.69	0.75	244.42	-1.13	-0.94	12.87
100-5-1b	W	0.78	14400.0	0.78	0.79	224.39	-1.48	-1.23	14.09
	Z	0.76	14400.0	0.75	0.76	204.08	-2.30	-2.30	15.04
100-5-2a	W	28.20	14400.0	1.82	2.21	235.75	-2.20	-2.20	11.91
	Z	30.38	14400.0	2.48	2.49	278.27	-7.26	-7.26	10.85
100-5-2b	W	29.57	14400.0	1.03	1.04	289.37	-4.58	-4.17	13.34
	Z	29.55	14400.0	1.00	1.01	280.93	-5.32	-5.32	13.61
100-10-1a	W	1.41	14400.0	0.97	1.34	222.86	0.55	0.88	13.46
	Z	0.53	14400.0	0.51	0.51	247.63	-0.31	-0.27	13.13
100-10-1b	W	0.10	14400.0	0.05	0.15	258.27	-0.41	0.09	16.34
	Z	0.04	14400.0	0.03	0.04	246.39	-1.21	-1.21	17.14
100-10-2a	W	22.30	14400.0	1.27	1.28	217.62	-1.78	-1.78	14.95
	Z	22.42	14400.0	1.45	1.46	234.41	-3.32	-3.32	13.85
100-10-2b	W	1.81	14400.0	1.18	1.19	213.72	19.68	20.14	15.52
	Z	22.78	14400.0	1.39	1.39	200.96	-2.91	-2.91	14.66
Average		6.11	11791.70	0.99	1.09	130.24	-1.70	-1.59	7.60

表7 PSN算例MSVNS算法与文献中已有结果比较

Table 7 Results comparison by MSVNS against existing results in literature for PSN instances

算例	DSS	B&C		MA			MSVNS		
		Gap%	CPU	Gap% _{min}	Gap% _{Avg}	CPU	Gap% _{min}	Gap% _{Avg}	CPU
20-5-1a	X	0.00	84.51	0.00	0.00	8.29	-8.73	-8.36	0.68
	Y	0.00	73.77	0.00	0.00	8.24	-8.41	-7.99	0.67
20-5-1b	X	0.00	1.86	0.00	0.00	7.73	0.16	0.63	0.52
	Y	0.00	0.27	0.00	0.00	7.61	0.13	0.13	0.46
20-5-2a	X	0.00	27.93	0.00	0.00	7.34	-7.84	-7.84	1.04
	Y	0.00	16.02	0.00	0.00	7.41	-7.76	-7.76	1.06
20-5-2b	X	0.00	1.61	0.00	0.00	6.67	-26.87	-26.47	1.05
	Y	0.00	1.38	0.00	0.00	7.18	-26.75	-26.72	0.74
50-5-1a	X	0.17	14400.0	0.17	0.20	43.29	9.44	9.95	3.50
	Y	0.22	14400.0	0.17	0.19	43.63	3.02	3.02	4.13
50-5-1b	X	0.04	14400.0	0.04	0.04	45.87	11.24	11.24	4.72
	Y	0.04	14400.0	0.05	0.05	46.02	4.74	4.74	5.33
50-5-2a	X	0.10	14400.0	0.09	0.09	46.81	0.48	0.48	4.20
	Y	0.09	14400.0	0.09	0.09	45.91	0.41	0.88	4.56
50-5-2b	X	0.00	213.75	0.00	0.00	44.11	0.55	0.73	5.31
	Y	0.00	1147.76	0.00	0.00	44.55	0.64	0.79	5.77
50-5-3a	X	0.24	14400.0	0.24	0.25	35.42	15.83	15.83	3.59
	Y	0.25	14400.0	0.23	0.24	35.74	9.59	10.04	3.41
50-5-3b	X	0.12	14400.0	0.09	0.09	42.30	2.25	2.47	5.27
	Y	0.03	14400.0	0.03	0.03	42.25	12.41	12.41	5.17
100-5-1a	X	0.18	14400.0	0.16	0.18	232.71	-1.06	-0.95	10.63
	Y	0.17	14400.0	0.16	0.18	231.53	-3.17	-3.17	13.93
100-5-1b	X	0.12	14400.0	0.09	0.10	185.47	-3.41	-3.41	16.39
	Y	0.12	14400.0	0.09	0.10	185.42	-3.54	-3.54	22.91
100-5-2a	X	0.11	14400.0	0.11	0.13	296.54	1.19	1.19	11.95
	Y	0.11	14400.0	0.12	0.13	299.33	-0.85	-0.85	15.33
100-5-2b	X	0.08	14400.0	0.06	0.07	286.23	0.21	0.21	20.25
	Y	0.07	14400.0	0.06	0.07	288.04	-0.84	-0.82	29.20
100-10-1a	X	1.07	14400.0	1.05	1.06	265.60	2.14	2.14	15.57
	Y	1.07	14400.0	1.04	1.05	256.65	2.26	2.76	17.98
100-10-1b	X	0.08	14400.0	0.06	0.06	232.63	1.37	1.91	23.14
	Y	0.07	14400.0	0.05	0.05	231.38	1.41	1.50	25.61
100-10-2a	X	32.08	14400.0	0.13	0.71	228.73	-46.85	-46.76	14.26
	Y	1.97	14400.0	0.15	0.90	230.26	-2.54	-2.54	16.65
100-10-2b	X	1.08	14400.0	1.08	1.09	194.59	-2.74	-2.24	23.21
	Y	1.08	14400.0	1.09	1.09	191.48	-2.81	-2.81	28.48
Average		0.98	11162.93	0.17	0.23	123.34	-2.08	-1.92	10.19

表8 不同方法获得新最优解的数量及平均求解时间比较

Table 8 Number of new best solutions obtained and computational time required by each approach

N _C	N ₀	总算例数	新最优解数量				平均CPU			
			SA ₁	SA ₂	MSA	MSVNS	SA ₁	SA ₂	MSA	MSVNS
20	5	16	16	16	14	14	54.1	54.3	4.8	0.74
50	5	24	7	11	17	7	110.7	111.8	49.5	4.17
100	5	16	4	5	11	14	190.3	192.3	217.2	14.79
100	10	16	0	3	12	13	371.2	381.8	232.8	17.15

参考文献:

- [1] MIN H. The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A General*, 1989, 23(5): 377 – 386.
- [2] LI J, PARDALOS P M, SUN H, et al. Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups. *Systems with Applications*, 2015, 42(7): 3551 – 3561.
- [3] AVCI M, TOPALOGLU S. A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery. *Systems with Applications*, 2016, 53: 160 – 171.
- [4] WANG C, MU D, ZHAO F, et al. A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup - delivery and time windows. *Computers & Industrial Engineering*, 2015, 83: 111 – 122.
- [5] ZHANG Xuanying, HU Rong, QIAN Bin. Hyper-heuristic estimation of distribution algorithm for solving vehicle routing problem with simultaneous pickup and delivery and soft time windows. *Control Theory & Applications*, 2021, 38(9): 1427 – 1441.
(张烜炎, 胡蓉, 钱斌. 超启发式分布估计算法求解带软时间窗的同时取送货车辆路径问题. 控制理论与应用, 2021, 38(9): 1427 – 1441.)
- [6] ZHANG Jingling, LIU Jinlong, ZHAO Yanwei, et al. Hyper-heuristic for time-dependent VRP with simultaneous delivery and pickup. *Computer Integrated Manufacturing Systems*, 2020, 26(7): 1905 – 1917.
(张景玲, 刘金龙, 赵燕伟, 等. 时间依赖型同时取送货VRP及超启发式算法. 计算机集成制造系统, 2020, 26(7): 1905 – 1917.)
- [7] QIN G, TAO F, LI L, et al. Optimization of the simultaneous pickup and delivery vehicle routing problem based on carbon tax. *Industrial Management & Data Systems*, 2019, 119(9): 2055 – 2071.
- [8] MA Yanfang, WANG Shan, HUANG Lingyu, et al. Algorithms for low-carbon pickup and delivery vehicle routing problem with fuzzy demand. *Journal of Computer Applications*, 2021, 41(3): 851 – 859.
(马艳芳, 王珊, 黄岭玉, 等. 模糊需求下低碳取送货车辆调度问题与算法. 计算机应用, 2021, 41(3): 851 – 859.)
- [9] OLGUN B, KOÇ Ç, ALTIPARMAK F. A hyper heuristic for the green vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, 2021, 153: 107010.
- [10] CHEN Xiqiong, HU Dawei, YANG Qianqian, et al. An improved ant colony algorithm for multi-objective vehicle routing problem with simultaneous pickup and delivery. *Control Theory & Applications*, 2018, 35(9): 1347 – 1356.
(陈希琼, 胡大伟, 杨倩倩, 等. 多目标同时取送货车辆路径问题的改进蚁群算法. 控制理论与应用, 2018, 35(9): 1347 – 1356.)
- [11] KARAOGLAN I, ALTIPARMAK F, KARA I, et al. A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery. *European Journal of Operational Research*, 2011, 211(2): 318 – 332.
- [12] KARAOGLAN I, ALTIPARMAK F. A hybrid genetic algorithm for the location-routing problem with simultaneous pickup and delivery. *Industrial Engineering & Management Systems*, 2011, 10(1): 1 – 6.
- [13] KARAOGLAN I, ALTIPARMAK F, KARA I, et al. The location-routing problem with simultaneous pickup and delivery: Formulations and a heuristic approach. *Omega*, 2012, 40(4): 465 – 477.
- [14] KARAOGLAN I, ALTIPARMAK F. A memetic algorithm for the capacitated location-routing problem with mixed backhauls. *Computers & Operations Research*, 2015, 55: 200 – 216.
- [15] YU V F, LIN S W. Multi-start simulated annealing heuristic for the location routing problem with simultaneous pickup and delivery. *Applied Soft Computing Journal*, 2014, 24: 284 – 290.
- [16] YU V F, LIN S Y. Solving the location-routing problem with simultaneous pickup and delivery by simulated annealing. *International Journal of Production Research*, 2016, 24(2): 1 – 24.
- [17] SUN Qingwei, ZHANG Yang. Study on location-routing problem with simultaneous pickup and delivery. *Journal of Transportation Engineering and Information*, 2017, 15(2): 100 – 104.
(孙青伟, 张杨. 带同时取送货的选址-多车型路径问题研究. 交通运输工程与信息学报, 2017, 15(2): 100 – 104.)
- [18] KARTAL Z, HASGUL S, ERNST A T. Single allocation p-hub median location and routing problem with simultaneous pickup and delivery. *Transportation Research Part E*, 2017, 108: 141 – 159.
- [19] KARIMI H. The capacitated hub covering location-routing problem for simultaneous pickup and delivery systems. *Computers & Industrial Engineering*, 2018, 116: 47 – 58.
- [20] FAN H, WU J, LI X, et al. Presenting a multi-start hybrid heuristic for solving the problem of two-echelon location-routing problem with simultaneous pickup and delivery (2E-LRPSPD). *Journal of Advanced Transportation*, 2020, 2020(1/2): 1 – 24.
- [21] GUO Fang, HUANG Zhihong, HUANG Weilai. Integrated sustainable planning of fast-pick area network and vehicle routing with simultaneous delivery and pickup. *Systems Engineering-Theory & Practice*, 2021, 41(4): 962 – 978.
(郭放, 黄志红, 黄卫来. 考虑前置仓选址与服务策略的同时取送货车辆路径问题研究. 系统工程理论与实践, 2021, 41(4): 962 – 978.)
- [22] GARCÍA-NÁJERA A, LÓPEZ-JAIMES A. An investigation into many-objective optimization on combinatorial problems: analyzing the pickup and delivery problem. *Swarm and Evolutionary Computation*, 2018, 38: 218 – 230.
- [23] ANDERLUH A, NOLZ P C, HEMMELMAYR V C, et al. Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and ‘grey zone’ customers arising in urban logistics. *European Journal of Operational Research*, 2021, 289(3): 940 – 958.
- [24] DORIGO M, GAMBARDELLA L M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1997, 1(1): 53 – 66.

作者简介:

陈希琼 博士, 讲师, 研究方向为运输网络优化、智能优化算法与启发式算法研究, E-mail: chenxiqiong123@163.com;

胡大伟 博士, 教授, 研究方向为物流系统工程与管理、交通运输系统及运输枢纽规划、一体化运输技术, E-mail: dwu@chd.edu.cn;

王 宁 博士, 教授, 研究方向为复杂系统分析与优化、交通网络建模与优化, E-mail: ningwang@chd.edu.cn.