

政府补贴和再制造收益影响下的集配一体化车辆路径问题研究^①

鲁建厦^② 李嘉丰 汤洪涛

(浙江工业大学机械工程学院 杭州 310023)

摘 要 针对废旧品回收过程中存在的政府补贴和再制造收益等问题,进行了以最大化企业利润为目标的集配一体化车辆路径问题(VRPSPD)研究,设计了改进混合蛙跳算法(ISFLA)进行求解。该算法利用改进扫描算法生成初始种群,采取多样性策略进行族群分配,设计深度领域搜索机制进行更优化探索。对比实验结果表明,考虑政府补贴和再制造收益的调度方案总利润平均增加 6%,可以有效提高企业利润,做出更优决策;所设计算法与遗传算法(GA)、模拟退火(SA)算法等经典算法相比,具有更好的收敛速度与求解精度,可以有效解决此类问题。

关键词 集配一体化;混合蛙跳算法(SFLA);政府补贴;再制造

0 引 言

国内经济已经从高速发展向高质量发展方向转变,实施绿色制造正是实现高质量发展的必然要求,而做好废旧品的回收以及再利用是绿色发展道路上十分重要的一环^[1]。2019 年发布的《中国废弃电器电子产品回收处理及综合利用行业白皮书 2018》显示,我国废弃电子产品的报废数量较往年呈持续增长,预计未来将会继续增加,面对规模继续扩大的废旧产品,回收物流的研究显得格外重要。

自 2011 年我国颁布实施《废弃电器电子产品回收处理管理条例》以来,已经有越来越多的企业投入到回收物流的行业中来。为提高资源利用率,将回收物流与配送物流联合起来进行调度是目前的研究趋势,该问题属于典型的集配一体化车辆路径问题(vehicle routing problem with simultaneous pick-up and delivery, VRPSPD),车辆在配送产品的同时,进行废旧品的回收,可以有效提高车辆利用效率、降低运作成本,实现绿色物流^[2]。VRPSPD 在 1989 年

由 Min^[3]提出,随后国内外学者的研究都是在此基础之上的扩展。Li 等人^[4]设计基于局部迭代的邻域搜索算法对多车场 VRPSPD 问题进行了研究。张江华等人^[5]以车辆数和行驶距离最小为目标,将拉格朗日松弛技术与禁忌搜索算法相融合,对多中心开放式的集配一体化车辆路径问题进行了研究,证明了开放式车辆调度较封闭式调度更加经济合理。Meng 等人^[6]研究了订单可拆分情况下的多批次集配一体化车辆路径问题,并提出禁忌搜索算法对其进行求解。Lin 等人^[7]研究了不同取货策略下的绿色集配一体化车辆路径问题,证明了全取策略较部分取货策略总成本更低。马艳芳等人^[8]引入模糊随机理论研究了客户需求双重不确定性下的同时送取货车辆路径问题。石建力等人^[9]以铁路物流为研究对象,提出了行驶时间和服务时间随机的可分批配送的集配一体化车辆路径问题。Chao 等人^[10]对带时间窗的集配一体化问题进行了研究,并设计大规模的算例进行了验证。Sebastian 等人^[11]研究了具有三维装载约束下的带回程取货的车辆路径问题,并提出了一种先装载货物再安排行驶路线的两

① 国家重点研发计划(2018YFB1308102),浙江省重点研发计划(2018C01003)和浙江省自然科学基金(LY15G010009)资助项目。
② 男,1963 年生,硕士,教授;研究方向:智能物流,物流装备和精益生产;联系人,E-mail: ljs@zjut.edu.cn
(收稿日期:2020-01-18)

阶段计算方法。

上述研究表明,学者们对 VRSPD 的研究主要集中在多车场、带时间窗、低碳和可分批取货等方面,却没有考虑政府补贴和再制造收益对车辆路线的影响。对于企业来说,政府补贴、废旧品的质量和回收价格很大程度上影响企业的回收收益,也决定着企业的回收选择,不同的回收选择对应着不同的车辆配送路线,影响企业的最终决策。比如,对于质量较低、再制造收益低但回收价格却偏高的废旧品,不能盲目进行回收,而是要综合考虑回收该产品所带来的总收益,并与回收该产品所要付出的总支出进行比较,然后决定是否进行回收。该问题较普通车辆路径问题更复杂,也更加具有现实意义,因此,本文以最大化企业收益为目标,建立了政府补贴和再制造收益影响下的集配一体化车辆路径问题(vehicle routing problem with simultaneous pickup and delivery and government subsidies and remanufacturing income, VRSPDGSRI)模型,设计了改进混合蛙跳算法(improved shuffled frog leaping algorithm, ISFLA)对该问题进行优化求解,并通过实例验证该模型与算法的有效性。

1 问题描述和数学建模

1.1 问题描述

VRSPDGSRI 可以描述为:某家电企业拥有不同类型的车辆若干辆,要为其客户提供产品的配送以及废旧品的回收服务,每个客户的废旧品质量不尽相同,回收价格以及再制造收益也不尽相同,每回收一件废旧产品,政府会进行相应的补贴,要求安排合理的车辆配送方案使企业的总收益最大。

同时,还要满足如下假设:(1)每辆车至少可以服务一个客户,即每个客户的需求量及回收量不会超过其车辆的最大承载量;(2)每一个客户仅被一辆车服务一次;(3)每辆车最多只被调度一次,车辆从配送中心出发,完成任务后返回到原配送中心;(4)在车辆路径上的任意点处,该车的承载量都不能超出其最大承载量。

1.2 回收定价

废旧产品的回收定价实质是企业与客户进行博

弈的问题,客户希望废旧品的回收定价越高越好,而定价高于废旧品的实际价值,则会影响企业利益,降低对废旧品进行回收的积极性,因此,政府出台了相应的补贴政策对企业进行回收补贴。文献[12,13]研究了在考虑政府补贴及产品质量的情况下,回收产品的定价问题,并证明了所建模型的正确性,因此,本文直接引用其回收产品最优定价模型,公式为

$$p' = \frac{(1 - q_0^2)\vartheta}{4(1 - \varepsilon)} - \frac{h}{2k} \quad (1)$$

其中, p' 表示废旧产品的最优回收价格; q_0 表示废旧产品的再制造质量门槛,质量高于 q_0 的产品有进行再制造的价值,而低于 q_0 的废旧品则只能被处理掉,获得处理收益 C_0 ; ϑ 表示质量价值系数,为大于0的实数; ε 为0到1之间的回收补贴系数; h 表示潜在顾客人群中有主动回收意识的客户数量; k 表示客户对回收价格的敏感系数。

质量高于 q_0 的回收产品经过再制造后再次进行销售,再制造成本主要与回收产品的质量有关,质量越高,则相应的再制造成本越低,再制造成本为

$$z = \tau \frac{p}{q} \quad (2)$$

其中, z 表示再制造成本, p 表示该产品的初始价格, q 表示回收产品的质量, τ 为再制造系数。

再制造产品通常会以折扣的形式再次出售给消费者,再销售价格为

$$p_2 = \mu p \quad (3)$$

其中, p_2 为二次销售价格, μ 为打折系数。

1.3 回收决策

废旧品的质量、回收价格、政府补贴以及前往回收点的距离代价等因素共同制约着企业的回收收益,间接影响着企业的回收决策,因此,企业在进行回收的过程中,必须加以综合考虑。

回收决策为

$$\omega = \begin{cases} p_2 + S - (p' + z) - d_e\beta & q > q_0 \\ S + C_0 - p' - d_e\beta & q \leq q_0 \end{cases} \quad (4)$$

其中, d_e 表示进行废旧品回收而额外行驶的距离, S 表示政府补贴, β 为车辆的单位距离行驶成本,只有 $\omega \geq 0$ 时,进行回收操作。

1.4 模型建立

VRSPDGSRI 问题可用加权图 $G = (V, E)$ 来

表示,其中,节点集 $V = \{C \cup D\}$, $C = \{1, 2, \dots, N\}$ 表示客户集, $D = \{1, 2, \dots, M\}$ 表示车场集, $K = \{1, 2, \dots, H\}$ 表示车辆集, $E = \{(i, j) \mid i, j \in V, i \neq j\}$ 表示节点 i 和 j 之间边的集合。

决策变量为

$$X_{ijk} = \begin{cases} 1 & \text{从节点 } i \text{ 到节点 } j \text{ 由车辆 } k \text{ 进行配送 } (i \neq j) \\ 0 & \text{其他} \end{cases}$$

$$Y_{ijk} = \begin{cases} 1 & \text{从节点 } i \text{ 到节点 } j \text{ 由车辆 } k \text{ 进行回收 } (i \neq j) \\ 0 & \text{其他} \end{cases}$$

根据以上条件分析,建立 VRSPDGSRI 数学模型如下:

$$\begin{aligned} \text{Max} Z = & \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} r_j p_j - \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} Y_{ijk} r'_j p'_j \\ & + \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} Y_{ijk} r'_j (p_{2j} - z_j) \\ & + \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} Y_{ijk} r'_j S_j \\ & - \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} \alpha_k - \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} X_{ijk} \beta_k d_{ij} \end{aligned} \quad (5)$$

s. t.

$$\sum_{i \in V} \sum_{k \in K} X_{ijk} = 1, \forall j \in C \quad (6)$$

$$\sum_{j \in V} \sum_{k \in K} X_{ijk} = 1, \forall j \in C \quad (7)$$

$$\sum_{i \in D} X_{ijk} = \sum_{i \in D} X_{jik} \leq 1, \forall j \in C, \forall k \in K \quad (8)$$

$$\sum_{i \in V} \sum_{j \in V} X_{ijk} r_j \leq Q_k, \forall k \in K \quad (9)$$

$$\sum_{i \in V} \sum_{j \in V} X_{ijk} Y_{ijk} r'_j \leq Q_k, \forall k \in K \quad (10)$$

$$w_{jk} \leq Q_k, \forall j \in C, \forall k \in K \quad (11)$$

$$w_{jk} - r_j + r'_j \leq Q_k, \forall j \in C, \forall k \in K \quad (12)$$

$$\sum_{i \in D} \sum_{k \in K} X_{ijk} \leq H, \forall j \in C \quad (13)$$

$$X_{ijk} \in \{0, 1\}, \forall i, j \in V, \forall k \in K \quad (14)$$

$$Y_{ijk} \in \{0, 1\}, \forall i, j \in V, \forall k \in K \quad (15)$$

其中, r_j 表示客户 j 的需求量, p_j 表示客户 j 的需求产品的单位价格, r'_j 表示客户 j 的回收量, p'_j 表示客户 j 的回收产品的单位回收价格, p'_{2j} 表示客户 j 的回收产品的二次销售价格, z_j 表示客户 j 回收产品的再制造成本, S_j 表示回收客户 j 产品的单位政府补贴; d_{ij} 表示节点 i 到节点 j 之间的距离; α_k 表示车辆 k 的固定使用成本, β_k 表示车辆 k 的单位距离行驶成本; w_{jk} 表示车辆 k 到达客户 j 时的车辆容量,

Q_k 表示车辆 k 的最大承载量。目标是在满足约束的条件下,确定每辆车的顾客服务顺序,使得企业的总收益最大。

式(5)为目标函数,表示此次集配货的总收益。其中,第1部分为销售产品的总收益,第2部分为废旧品回收总成本,第3部分为废旧品再制造收益,第4部分为政府补贴收益,第5部分为车辆的固定使用成本,第6部分为配送车辆的距离成本;式(6)和式(7)表示一个客户仅被一辆车服务一次;式(8)表示车辆从配送中心出发完成任务后返回原配送中心;式(9)表示车辆的单次配送总量不得超过车辆的最大承载量;式(10)表示车辆的单次回收总量不得超过车辆的最大承载量;式(11)和式(12)表示车辆在路径上的任意点处的承载量都不得超出车辆的最大承载量;式(13)表示调度所用车辆数不得超过可用车辆总数;式(14)和(15)为0~1的决策变量。

2 混合蛙跳算法设计

VRSPDGSRI 问题有着同时取送货的双重约束,较普通 VRP 类问题更加复杂,属于典型的 NP 难题^[14],对算法的要求更高。蛙跳算法(shuffled frog leaping algorithm, SFLA)^[15]是一种启发式优化算法,结合了模因算法和粒子群算法的优点,概念简单、调整参数少,具有很强的全局搜索能力,并已成功地在车辆路径规划^[16]、柔性车间调度^[17]等领域内应用。在蛙跳算法中,种群由一群青蛙组成,每一只青蛙代表一种可能的解决方案,将这些青蛙按照适应度大小分到不同的族群内,青蛙在各自的族群内部进行交流演化。一定时间后,所有族群的青蛙会聚集在一起,进行交流融合,然后再行划分、进化,循环以往,直至满足收敛条件。然而在传统蛙跳算法中,青蛙只向最优个体学习的行为,导致算法收敛速度慢,容易陷入局部最优解,因此,本文提出了一种改进的混合蛙跳算法对该问题进行求解。

2.1 编码

为扩大算法的适用范围,方便不同类型车辆之间的信息交互,本文采用多车辆的编码方式,即对每辆车分别进行编码。对于一个包含了 H 辆车、 N 个

客户的 VRSPDGSRI 问题,用编号 $1 \sim N$ 表示客户, $N+1 \sim N+H$ 分别表示不同类型的 H 辆车,分别对每辆车进行编码,每辆车服务客户的顺序称为该车的基因链。例如编号为 8 和 9 的两辆车要为编号为 $1 \sim 7$ 的客户进行配送与回收服务,则可能的编码方式为 $[8,2,1,5,7;9,4,6,3]$,分别表示车辆 8 从车场出发,依次服务客户 2、1、5、7,最后回到原车场;车辆 9 从车场出发依次服务客户 4、6、3,最后回到原车场。

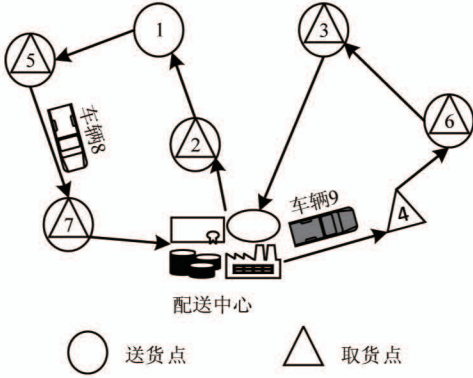


图1 车辆配送与取货示意图

2.2 生成初始解

为提高算法初始解的优异性,同时保证初始种群的多样性,本文设计了两种初始解的生成方式,共同构造初始种群,使初始解尽量广地覆盖解空间,提高寻优效率。多次实验结果表明,随机初始解与改进扫描算法产生的初始解比例为1:2时,算法运行效果最好。

2.2.1 随机生成初始解

将客户进行随机排序,然后将第一个客户随机分配给某辆车,并判断该车辆是否超载,如果在承载量之内,则该客户被该车服务,否则,随机选择另一辆车,并判断是否超载,以此类推,直至将该客户分配出去。然后按照上述过程分配序列中剩余的客戶,直至所有客戶都被分配出去,客戶被分配给车辆的先后顺序即为车辆的服务顺序。

2.2.2 改进扫描算法构造初始解

首先通过邻近数列进行初始服务客户的选择,然后利用扫描法构造初始种群。

邻近数列用来描述两个点的邻近关系,通过客戶与配送中心之间的距离构造邻近数列,公式为

$$\lambda(o \leftarrow i) = (\lambda_{oi})_{1 \times N} \quad i \in N, o \in D \quad (16)$$

其中, $\lambda_{oi} \in \{1, 2, 3, \dots, N\}$, 表示客戶 i 和车场 o 的邻近程度,如 $\lambda_{o4} = 3$ 表示客戶 4 与车场 o 邻近度为 3,即在所有客戶中,客戶 4 距离车场 o 第 3 近。

定义概率选择公式为

$$P_i = \frac{1/\lambda_{oi}}{\sum_{i=1}^N 1/\lambda_{oi}} \quad i \in C, o \in D \quad (17)$$

通过式(17)选择车辆的初始服务客戶,客戶距离车场越近,被选中的概率越大,然后,根据扫描算法的思想,以车场为圆心,顺时针方向选择与该客戶所成角度最小的客戶进行服务。以此类推,直至该车辆饱和,无法再服务更多客戶为止,对剩下车辆均执行上述操作,直至所有客戶均被分配出去。

2.3 族群分配

将初始种群按照适应度大小进行降序排列,位列第一的青蛙被分配给第一个族群,位列第二的青蛙被分配给第二个族群,以此类推,直至将所有青蛙分别分配到 m 个族群中,每个族群中包含 n 个青蛙,用公式表示为

$$Y_k = \{X_k^i \mid X_k^i = X_{k+m(i-1)}, i = 1, 2, \dots, n\} \quad k = 1, 2, \dots, m \quad (18)$$

其中, Y_k 表示第 k 个族群的青蛙, x_k^i 表示族群 k 中排序为 i 的青蛙, $X_{k+m(i-1)}$ 表示在整个种群中排序为 $k + m(i-1)$ 的青蛙。

传统蛙跳算法的族群分配方式中,只考虑了个体的适应度大小关系,并未考虑个体间的差异性,导致个体在族群进化的过程中,容易因为族群内个体相似性过大而提前陷入局部最优解。因此,本文提出一种考虑青蛙个体间基因差异性的族群分配策略,差异性判断公式为

$$\text{Div}(f_1, f_2) = 1 - \frac{\sum_{j=1}^H |\{cu \mid r_i = r_j, i \in \{1, 2, \dots, H\}, cu \in r_j\}|}{N} \quad r_i \in f_1, r_j \in f_2 \quad (19)$$

其中, f_1 、 f_2 分别表示两只青蛙个体, H 表示车辆数量, cu 表示客戶, r_i 表示车辆 i 的服务路线, N 表示总的客戶数量。

例如,一个包含了 9 个青蛙的种群 P 将要分为

3 个族群,每个族群 3 只青蛙,首先将 9 只青蛙按照适应度降序排列,然后根据式 (18) 进行第一轮分配。如图 2 所示,将青蛙 1、2、3 分别分配给族群 1、2 和 3,然后在探索范围 $s = 4$ 的 4 个青蛙中,根据

式(19)分别计算每只青蛙与族群 1、2、3 中青蛙的差异性,并将与族群差异性最大的青蛙分配到该族群中,以此类推,直至将所有青蛙分配到各个族群为止。

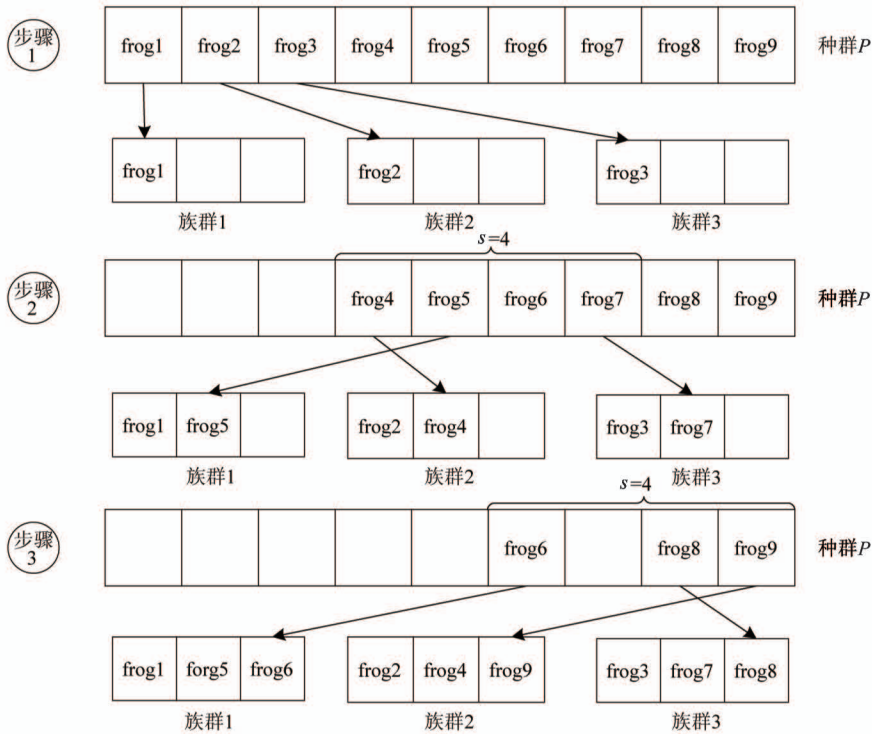


图 2 差异性族群分配过程

2.4 蛙跳策略

为充分利用族群内部的有效信息,增加算法的搜索宽度,在族群内部引入子群的概念,在大小为 n 的族群内部选择 s 只青蛙组成一个子群 ($s < n$),选择概率公式为

$$p_j = \frac{2(n + 1 - j)}{n(n + 1)} \tag{20}$$

其中, j 表示该青蛙在族群中的适应度排序数,适应度越大, j 值越小。

青蛙种群中最优个体为 P_g , 子群内的最优个体和最差个体分别为 P_b 和 P_w , P_w 向 P_b 进行跳跃,具体步骤如下。

- 步骤 1 P_b 中随机挑选一辆车的基因链,记为 b , 在 P_w 中选择相同车辆的基因链,记为 w 。
- 步骤 2 将两辆车的服务客户进行互换。
- 步骤 3 对比基因链 b 和 w , 将基因链 b 独有的客户基因保存在 F_b 中,基因链 w 独有的客户基因保

存在 F_w 中。

步骤 4 检查 P_w 中其余车辆的基因链,删除与基因链 b 中相同的客户基因,同样地,删除个体 P_b 中与基因链 w 中相同的客户基因。

步骤 5 随机选择 F_w 中的一个客户,并将其插入到 P_w 中任意一辆车的服务序列中,检查该车是否超载。如果超载,则将该客户随机分配给其他车辆进行服务;以此类推,直至将 F_w 中的客户全部分配出去为止;对个体 P_b 采取同样操作。

步骤 6 信息交流后的个体分别命名为 P'_b 和 P'_w , 分别计算其适应度 $f(P'_b)$ 和 $f(P'_w)$, 如果 $f(P'_b) > f(P_b)$, 则用 P'_b 替换 P_b , 否则保持 P_b 不变。

步骤 7 如果 $f(P'_w) > f(P_w)$, 则用 P'_w 替换 P_w , 否则,将步骤 1 中的 P_b 用 P_g 进行替换,然后重新进行信息交流操作,并再次比较信息交流后的适应度,如果 $f(P'_w) > f(P_w)$, 则用 P'_w 替换 P_w , 否则

重新生成一只新的青蛙代替 P_w 。

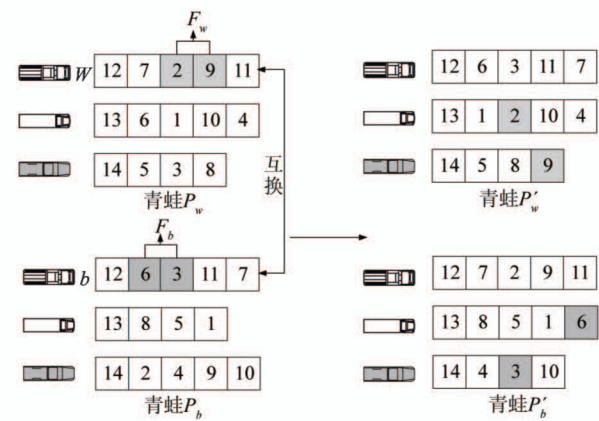


图3 蛙跳搜索示意图

2.5 邻域搜索策略

为了使算法跳出局部最优解,增加算法的搜索深度,设计了一种深度邻域搜索方法,帮助个体进行更为深入的进化,提高算法的寻优速度与寻优质量。根据本文多车辆编码的特点,主要采取以下 6 种邻域搜索策略。

(1) 车辆内路径单点插入。随机选择一条路线上的一个客户,然后按照最小成本法^[18]插入到路线上的其他位置。

(2) 车辆内路径路段互换。随机选取一条路线上的两段客户,将两段客户的位置互换。

(3) 车辆内路径反转。随机选择一条路径上的两个位置,然后将两位置间的所有客户进行反转。

(4) 车辆间单点转移。随机选择两条路径,将一条路径上的一个客户按照最小成本法插入到另一条车辆路径中。

(5) 车辆间路径两点互换。随机选择两条路径,从两条路径中分别选择一个客户按照最小成本法插入到另一条路径中。

(6) 车辆间路径路段互换。随机选择两条路径,从两条路径中分别选择一段客户随机插入到另一条路径中。

以上邻域搜索策略又可以分为车辆间邻域搜索和车辆内邻域搜索,车辆间邻域搜索对个体的变动幅度较车辆内更大,并且车辆内邻域搜索又可以做是车辆间邻域搜索的一部分。因此,本文在车辆间有效搜索的基础上,提出一种深度邻域搜索策略,

即在车辆间邻域搜索取得更好的解的前提下,继续向下执行车辆内邻域搜索策略,具体步骤如下:

1. 输入:青蛙 P

2. while 不满足邻域搜索次数

3. if $rand(0,1) < 0.5$

4. 随机执行一种车辆间邻域搜索策略,形成新解 P'

5. if $f(P') > f(P)$

6. $P = P'$

7. 随机执行一种车辆内邻域搜索策略,形成新解 P'

8. if $f(P') > f(P)$

9. $P = P'$

10. end if

11. end if

12. else

13. 随机执行一种车辆内邻域搜索策略,形成新解 P'

14. if $f(P') > f(P)$

15. $P = P'$

16. end if

17. end while

18. 输出:青蛙 P

3 实验对比分析

3.1 算例 1 分析

某经销商要对 50 个客户进行新空调的配送与废旧空调的回收服务,客户的基本需求、产品的具体参数和政府补贴如表 1 所示。编号 0 代表配送中心,配送中心可用车辆的具体信息如表 2 所示。废旧品的质量分为 1~5 共 5 个等级,等级为 1 的空调无法进行再制造,只能进行废弃处理,可以获得新品价格 1/5 的处理收益。除此之外,企业每回收一台空调,会获得相应的政府补贴,要求安排最合理的车辆路线、做出最好的回收选择,使得经销商综合利润最大。

根据文献[19]对蛙跳算法最优参数的分析,设置算法参数为:最大迭代次数 $G = 1000$, 种群规模 $F = 400$, 族群数 $m = 20$, 各族群内青蛙个数 $n = 20$, 子群规模 $s = 16$, 每个族群内蛙跳次数 $numLe = 5$, 邻域搜索次数 $numSe = 5$ 。

采用 Matlab2014a 作为实验仿真平台,对该问题进行优化求解,车辆的最优调度方案如表 3 所示。

表 1 信息表

编号	横坐标 /km	纵坐标 /km	配送量 /台	回收量 /台	新品价格 /千元	回收价格 /千元	质量	补贴 /千元
0	35	49	0	0	0	0	0	0
1	41	49	10	6	6	4	3	1.2
2	35	17	11	10	8	3	5	1.6
3	55	45	9	12	6	3	2	1.2
4	55	20	6	12	6	2	2	1.2
5	15	30	10	9	5	2	4	1.0
6	25	30	9	6	7	3	2	1.4
7	20	50	12	0	7	0	0	1.4
8	10	43	9	6	8	4	1	1.6
9	55	60	12	7	8	3	1	1.6
10	30	60	11	10	8	3	4	1.6
11	20	65	8	10	8	4	2	1.6
12	50	35	11	9	4	3	1	0.8
13	30	25	8	8	7	4	2	1.4
14	15	10	12	5	8	4	4	1.6
15	30	5	14	10	4	2	5	0.8
16	10	20	10	6	4	2	5	0.8
17	5	30	12	13	5	4	3	1.0
18	20	40	8	10	5	3	3	1.0
19	15	60	9	6	5	4	3	1.0
20	45	65	14	13	6	4	3	1.2
21	45	20	13	6	7	4	2	1.4
22	45	10	14	9	5	3	4	1.0
23	55	5	12	9	7	5	5	1.4
24	65	35	10	11	8	3	4	1.6
25	65	20	0	13	6	4	1	1.2
26	45	30	13	6	6	4	3	1.2
27	35	40	9	15	6	3	3	1.2
28	41	37	7	9	7	3	3	1.4
29	64	42	14	5	6	4	1	1.2
30	40	60	9	7	5	2	5	1.0
31	31	52	10	7	4	2	5	0.8
32	35	69	11	7	7	3	3	1.4
33	53	52	10	12	8	4	1	1.6
34	65	55	14	4	7	3	4	1.4
35	63	65	9	12	5	4	5	1.0
36	2	60	9	10	8	4	1	1.6
37	20	20	6	8	5	3	4	1.0
38	5	5	5	12	6	2	5	1.2
39	60	12	11	9	7	3	5	1.4
40	40	25	11	8	9	3	1	1.8
41	42	7	13	6	6	4	2	1.2
42	24	12	7	13	7	3	2	1.4
43	23	3	9	13	6	3	5	1.2
44	11	14	11	9	5	2	3	1.0
45	6	38	13	10	9	3	4	1.8
46	2	48	5	11	8	4	1	1.6

续表 1

47	8	56	8	11	6	3	3	1.2
48	13	52	7	10	8	2	1	1.6
49	6	68	13	10	7	2	4	1.4
50	47	47	12	6	5	4	1	1.0

表 2 配送中心信息表

车辆 编号	车辆 类型	车辆固定 成本/千元	车辆可变 成本/千元	承载量 /台
51	大型	3	3	100
52	大型	3	3	100
53	中型	2	2	90
54	中型	2	2	90
55	小型	1	1	80
56	小型	1	1	80
57	小型	1	1	80

表 3 最优调度方案

车辆 编号	车辆 路径	装载率	利润 /千元	路程 /km
51	0-0	0	0	0
52	0-27-1-50-3-29- 24-12-26-28-0	77%	384.69	83.40
53	0-13-41-22-23-39- 4-21-40-0	75%	462.11	94.52
54	0-31-10-11-19-49- 36-47-48-7-0	97%	477.50	106.87
55	0-37-44-38-14-42- 43-15-2-0	81%	505.70	110.58
56	0-6-5-16-17-45- 46-8-18-0	73%	459.50	98.06
57	0-33-34-35-9-20- 32-30-0	81%	456.90	114.50
	合计		2746.40	607.93

其中,黑色加粗的客户表示只进行配送服务,而不进行相应的废旧品回收服务,装载率取配送过程和回收过程中的平均值。最优路径如图 4 所示,车辆路线间交叉较少,平均装载率为 81%,配送方案质量较高。圆形表示只送不取客户,其中,客户 25 没有配送需求,同时因为其回收价格大于产品可创造的价值,因此不对该客户进行服务。综合决策后,车辆行驶总路程为 607.93 km,企业最大利润为 2746.40 千元。

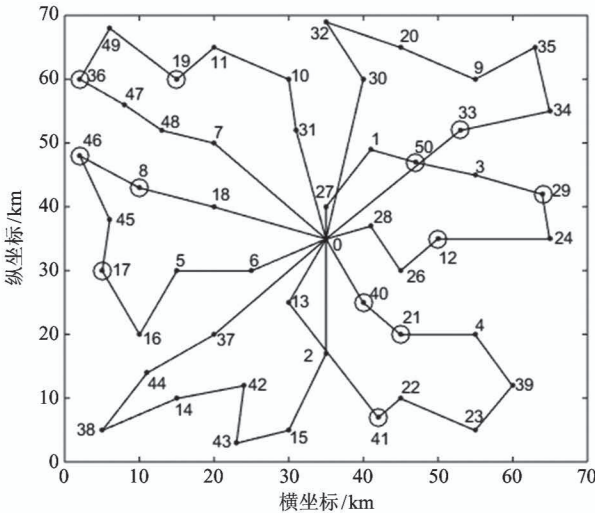


图 4 最优路线图

当不考虑政府补贴及再制造收益时,对该问题进行多次求解,最优调度方案如表 4 所示,最优路线如图 5 所示,车辆平均装载率为 92%,总路程为 632.97 km,企业最大利润为 2572.60 千元。对比表 3 和表 4 可知,不考虑政府补贴与再制造收益的调度

表 4 最优调度方案

车辆 编号	车辆 路径	装载率	利润 /千元	路程 /km
51	0-13-2-41-22-23- 39-4-21-40-0	88%	480.37	94.53
52	0-0	0	0	0
53	0-50-33-9-35-20- 32-30-1-0	87%	346.00	107.80
54	0-42-15-43-14- 38-44-16-37-6-0	92%	376.91	113.07
55	0-18-8-47-36-46- 45-17-5-0	96%	440.04	108.06
56	0-7-48-19-49-11- 10-31-27-0	92%	528.10	99.99
57	0-28-12-3-34-29- 24-25-26-0	92%	401.18	109.52
	合计		2572.60	632.97

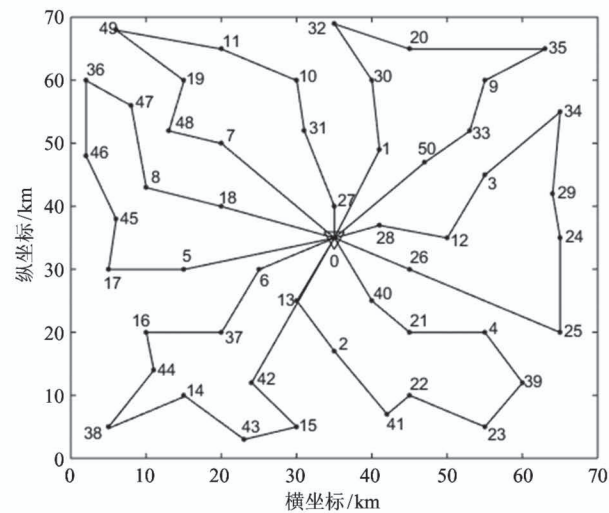


图 5 最优路线图

方案中,车辆的平均装载率虽然更高,但是其总路程也更大,与此同时,总利润却更少,相比于考虑政府补贴与再制造收益的调度方案总利润减少了 6%。这主要是由于不考虑政府补贴与再制造收益时,车辆会对所有客户提供回收服务,从而导致一些质量较差、再制造收益低、回收价格高的废旧品也进行了回收,造成了企业的部分亏损。比如客户 8 的回收价格为 4 千元,但是其废旧品质量等级为 1,企业只能获得补贴 1.6 千元,处理收益 1.6 千元,总收益 3.2 千元,回收价格大于总收益,回收造成了亏损。而考虑政府补贴与再制造收益的方案中,会对所有的客户进行回收决策的判断,不对回收价格大于潜在收益的客户进行回收,这样就避免了一部分不合理的回收现象,减少了车辆的总行驶距离,使企业获得更大的利润,这也证明了政府补贴和再制造收益对集配一体化车辆路径的影响。

分别采用传统蛙跳算法、变邻域模拟退火算法 (variable neighborhood search-simulated annealing, VNS-SA)^[20] 以及改进粒子群算法 (improved particle swarm optimization, IPSO)^[21] 对该问题进行求解,每种算法运行 30 次,对比结果如表 5 所示,平均进化曲线如图 6 所示。

本文算法在 400 代左右已经进行收敛,与传统蛙跳算法、改进粒子群算法以及变邻域模拟退火算法相比,初始解质量更好,收敛速度更快,求解精度更高,有较强寻优能力的同时,兼具了很好的稳定

性,显示了算法的有效性。

表 5 不同算法实验结果

算法	最小值 /千元	最大值 /千元	平均值 /千元	平均收敛 代数	标准差 /千元
SFLA	2278.5	2543.3	2426.6	768	72.9
IPSO	2580.9	2703.4	2658.1	679	32.3
VNS-SA	2493.5	2658.4	2580.0	756	48.1
ISFLA	2692.5	2746.4	2706.7	398	13.9

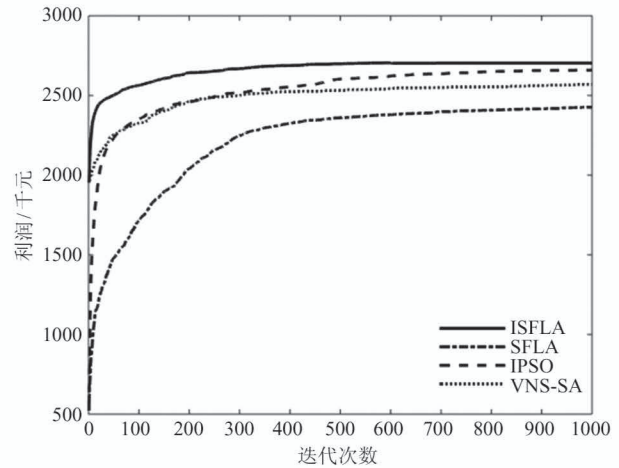


图 6 进化曲线

3.2 算例 2 分析

为充分验证本文算法在求解相关问题方面的优异性,从 Solomon 算例中分别选取 C 型、R 型、RC 型中不同客户规模的算例各 4 例,采用改进混合蛙跳算法、遗传算法 (genetic algorithm, GA)^[22]、模拟退火 (simulated annealing, SA) 算法^[23] 分别进行求解。其中,客户的需求量及回收量按照文献[24]所述方法进行设置,产品质量在 [1, 5] 中随机产生,产品价格 [3, 10] 之间随机产生,补贴为产品价格的 1/5,每种算法运行 30 次,停止条件为最优解连续 200 代没有变化或者迭代次数达到 2000 代,不同算法的对比结果如表 6 所示。

由表 6 可以看出,3 种算法在计算 C 型算例时,用时普遍较少,最优解差异性也最小,而在计算 RC 型算例时,用时最多,最优解差异性也最大,反映了不同类型问题对算法的影响。其中,部分客户聚集、部分客户离散的 RC 型分布方式问题最难求解。在不同规模、不同类型的所有算例中,ISFLA 均可以取

得更好的结果,说明本文算法通用性强,可以有效处理不同分布类型的问题。图 7 为 ISFLA 求解算例 R106 的最优结果,路线之间交叉较少,显示了结果的优异性。除了 C105 算例外,ISFLA 的标准差都是 3 种算法中最小的,体现了算法的稳定性。ISFLA

算法初始解更优,收敛速度快,除了 R102 算例之外,耗时都是最少的,并且随着算例规模的增大,算法求得最优解的时间虽然有所增长,但是都在可接受范围之内,充分说明了算法的实用性与有效性。

表 6 算法结果比较

算例		GA			SA			ISFLA		
名称	规模	最优值 /千元	标准差	时间/s	最优值 /千元	标准差	时间/s	最优值 /千元	标准差	时间/s
C101	50	3010.5	56.9	378.4	3005.7	91.9	430.0	3034.8	44.5	150.0
C105	50	3083.0	80.4	386.9	3038.6	26.1	468.5	3155.1	32.8	231.8
C102	100	5295.8	79.6	526.6	5386.4	75.6	673.3	5652.9	52.2	459.4
C106	100	5395.2	82.9	520.2	5335.9	78.1	668.1	5661.7	49.6	517.5
R101	50	2630.4	49.1	460.9	2690.4	58.4	641.9	2685.6	6.3	377.5
R105	50	2647.8	45.9	536.2	2707.5	31.3	559.7	2859.4	18.5	464.2
R102	100	5151.9	70.6	685.4	5498.8	40.6	714.7	5613.2	37.6	786.8
R106	100	5302.5	75.3	729.4	5497.2	54.7	812.1	5678.9	45.2	604.6
RC101	50	2443.6	63.2	485.7	2504.7	75.1	654.9	2520.1	35.6	370.2
RC105	50	2482.7	63.9	614.4	2517.7	60.2	610.0	2524.2	39.0	445.8
RC102	100	4874.1	58.4	771.6	5036.7	48.2	895.1	5288.2	25.9	642.2
RC106	100	5162.2	60.0	779.3	5112.4	56.7	796.6	5234.7	38.4	589.4

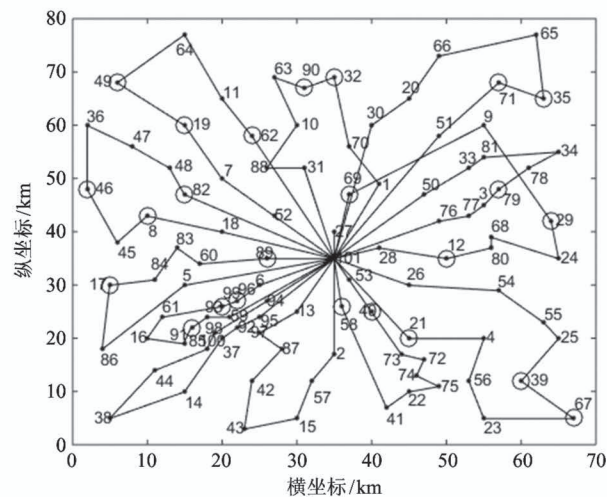


图 7 算例 R106 最优结果

4 结 论

政府补贴和废旧品的再制造收益会影响企业的回收选择,从而影响车辆的路线规划。为了解决政府补贴和再制造收益影响下的集配一体化车辆路径问题,建立了以总利润最大为目标的车辆调度模型,设计了混合蛙跳算法,该算法利用改进扫描算法生成初始解,引入差异性族群分配策略,设计了深层次

的邻域搜索机制,保证了算法搜索的宽度与深度。多组不同规模、不同类型的对比实验表明,模型与算法具有优异性与有效性,可以帮助企业做出合理的回收决策,制定更加科学有效的集配方案。考虑到废旧品的回收数量存在不确定性的特点,未来将会对模糊回收需求的集配一体化问题作进一步的研究。

参考文献

[1] 陶永,赵昱,王田苗,等. 以加快科技创新推动先进装备制造转型升级的策略思考[J]. 高技术通讯, 2016, 26(2):173-179

[2] Tong H F, Qu W S. Assessing China 2030 carbon emissions from fossil fuels:based on system dynamics model [J]. High Technology Letters,2015,21(2):178-184

[3] 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

[4] 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[J]. Expert Systems with Applications, 2015, 42(7):3551-3561

[5] 张江华,李进,高敏刚. 同时集散货物的开放式车辆路

- 径问题研究[J]. 中国管理科学, 2013, 21(4):187-192
- [6] Meng Q, Zhuo F, Richard E, et al. A tabu search algorithm for the vehicle routing problem with discrete split deliveries and pickups[J]. *Computers and Operations Research*, 2018, 100:102-116
- [7] Lin C, Choy K L, Ho G T S, et al. A genetic algorithm-based optimization model for supporting green transportation operations [J]. *Expert Systems with Applications*, 2014, 41(7):3284-3296
- [8] 马艳芳, 应斌, 康凯, 等. 模糊需求下绿色同时取送货问题与算法研究[J]. 计算机工程与应用, 2020, 56(16): 248-257
- [9] 石建力, 张锦. 行驶时间和服务时间随机的集配货一体的分批配送车辆路径优化[J]. 控制与决策, 2018, 33(4):657-670
- [10] Chao W, Dong M, Fu Z, et al. A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup-delivery and time windows[J]. *Computers and Industrial Engineering*, 2015, 83:111-122
- [11] Sebastian R, Andreas B, Lars M. Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints[J]. *European Journal of Operational Research*, 2018, 266(3):877-894
- [12] 程发新, 马方星, 邵汉青. 回收补贴下废旧产品质量不确定的闭环供应链定价决策及协调[J]. 软科学, 2018, 32(7):139-144
- [13] 黄帝, 周泓. 考虑不同回收质量等级的再制造系统回收生产决策[J]. 中国管理科学, 2018, 26(10):102-112
- [14] Liu J S, Luo Z W, Duan D Z, et al. A GA approach to vehicle routing problem with time windows considering loading constraints[J]. *High Technology Letters*, 2017, 23(1):54-62
- [15] Eusuff M M, Lansey K E. Optimization of water distribution network design using the shuffled frog leaping algorithm[J]. *Journal of Water Resources Planning and Management*, 2003, 129(3):210-225
- [16] Luo J, Li X, Chen M R, et al. A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows[J]. *Information Sciences*, 2015, 316:266-292
- [17] 杨冬婧, 雷德明. 新型蛙跳算法求解总能耗约束 FJSP [J]. 中国机械工程, 2018, 29(22):2682-2689
- [18] 禹鑫淼, 卢靓, 朱熠琛, 等. 基于启发式动态规划的自动化停车场调度方法[J]. 高技术通讯, 2019, 29(4):352-361
- [19] 孟凯露, 尚俊娜, 岳克强. 混合蛙跳算法的最优参数研究[J]. 计算机应用研究, 2019, 36(11):3321-3324
- [20] 李嘉, 杨东, 贾永基, 等. 装卸一体化电动汽车路径问题建模与优化[J]. 工业工程与管理, 2020, 25(1):29-37
- [21] 马艳芳, 闫芳, 康凯, 等. 不确定同时取送货车辆路径问题及粒子群算法研究[J]. 运筹与管理, 2018, 27(12):73-83
- [22] Tasan A S, Gen M. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries [J]. *Computers and Industrial Engineering*, 2012, 62(3):755-761
- [23] 陈妍, 单汨源, 王秋凤. 多车型集配货一体化车辆路径问题研究[J]. 中南大学学报(自然科学版), 2015, 46(5):1938-1945
- [24] 周蓉, 沈维蕾, 刘明周, 等. 带时间窗装卸一体化车辆路径问题的混合离散粒子群优化算法[J]. 中国机械工程, 2016, 27(4):494-502

Research on simultaneous pickup and delivery vehicle routing problem with government subsidies and remanufacturing income

Lu Jiansha, Li Jiafeng, Tang Hongtao

(College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023)

Abstract

Aiming at the problems of government subsidies and remanufacturing income in the process of recycling used waste, a research on the problem of the vehicle routing problem with simultaneous pickup and delivery (VRPSPD) with the goal of maximizing corporate profits is carried out, and an improved shuffled frog leaping algorithm (ISFLA) is designed to solve the problem. The algorithm uses an improved scanning algorithm to generate the initial population, adopts a diversity strategy for ethnic group assignment, and designs a deep local search mechanism for more optimized exploration. The results of multiple sets of comparative experiments show that the total profit of the solution considering government subsidies and remanufacturing income is increased by 6% on average, which can effectively improve corporate profits and make better decisions. Compared with classical algorithms such as genetic algorithm (GA) and simulated annealing (SA) algorithm, the designed algorithm has better convergence speed and solution accuracy, which can effectively solve such problems.

Key words: simultaneous pickup and delivery, shuffled frog leaping algorithm (SFLA), government subsidy, remanufacturing