

混合蛇优化算法求解带时间窗的车辆路径问题

袁杰 刘超 加尔肯别克 张震

(新疆大学智能科学与技术学院, 乌鲁木齐 830017)

摘要: 针对带时间窗的车辆路径问题 (Vehicle routing problem with time windows, VRPTW) 约束条件复杂、求解难度高, 以及蛇优化算法 (Snake optimization algorithm, SO) 在求解过程存在局部搜索能力弱、求解质量低等问题, 提出了一种混合蛇优化算法 (Hybrid snake optimization algorithm, HSO), 采用最小位置匹配值法进行离散化解码, 有效提升了求解效率。通过引入随机插入与基于密度带有噪声的空间聚类 (Density-based spatial clustering of applications with noise, DBSCAN) 算法相结合的混合初始化策略, 增强了初始种群多样性, 拓展了解空间覆盖范围; 融合蛇群探索食物、靠近食物、战斗与交配等行为, 并结合局部邻域搜索机制, 有效提升了算法局部搜索能力。利用 Solomon 数据集开展仿真试验, 并采用 Wilcoxon 秩和检验及 Friedman 检验对结果进行统计分析。试验结果表明, HSO 算法在求解 VRPTW 中表现出优异的求解精度与稳定性。最后, 通过实际城市农产品配送问题的应用, 验证了该算法在复杂组合优化问题中适应性与实际应用价值。

关键词: 带时间窗的车辆路径问题; 蛇优化算法; 混合初始化; 局部邻域搜索

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-1298(2026)20-0413-14

OSID:



Hybrid Snake Optimization Algorithm for Solving Vehicle Routing Problem with Time Windows

Yuan Jie Liu Chao Jia Erkenbieke Zhang Zhen

(School of Intelligent Science and Technology, Xinjiang University, Urumqi 830017, China)

Abstract: Aiming to address the complex constraints and high computational complexity of the vehicle routing problem with time windows (VRPTW), as well as the weak local search capability and limited solution quality of the snake optimization algorithm (SO), a hybrid snake optimization algorithm (HSO) was proposed. The minimum position matching value method was employed for discrete decoding, thereby improving computational efficiency. A hybrid initialization strategy combining random insertion and the density-based spatial clustering of applications with noise (DBSCAN) algorithm was introduced to enhance the diversity of the initial population and broaden the coverage of the search space. Furthermore, snake behaviors, including searching for food, approaching food, fighting, and mating, were integrated with a local neighborhood search mechanism to strengthen the local search capability. Simulation experiments were conducted on the Solomon dataset, and the results were statistically analyzed by using the Wilcoxon rank-sum test and Friedman test. The experimental results demonstrated that the proposed HSO algorithm exhibited superior solution accuracy and stability for solving the VRPTW. Finally, application of the proposed HSO algorithm to a real-world urban agricultural product distribution problem further demonstrated its adaptability and practical value for solving complex combinatorial optimization problems.

Key words: vehicle routing problem with time windows; snake optimization algorithm; hybrid initialization; local neighborhood search

收稿日期: 2025-07-22 修回日期: 2025-09-08

基金项目: 国家自然科学基金项目 (62263031) 和自治区高校基本科研业务费科研项目 (202532120001)

作者简介: 袁杰 (1975—), 男, 教授, 博士, 主要从事计算机应用研究, E-mail: yuanjie222@126.com

通信作者: 加尔肯别克 (1990—), 男, 讲师, 博士, 主要从事智能感知研究, E-mail: buickml@163.com

0 引言

物流运输业是连接生产与消费的重要环节,也是保障经济循环畅通的关键纽带,在推动产业协同发展、促进经济发展方式转型及优化产业结构等方面发挥着重要作用。车辆路径问题(Vehicle routing problem,VRP)^[1-2]是物流运输领域的核心问题,旨在规划车辆从配送中心出发的最优路线,在满足所有客户配送需求的同时,尽可能降低运输成本、缩短行驶距离或运输时间。车辆路径问题广泛应用于物流配送^[3]、冷链药品配送^[4]、餐厨垃圾收运^[5]等实际场景,通常涉及车辆载重限制、客户服务时间窗等复杂约束条件。

带时间窗的车辆路径问题(Vehicle routing problem with time windows,VRPTW)^[6-7]是车辆路径问题的典型拓展,属于组合优化问题,具有显著的NP-hard特性。在满足客户服务时间窗和车辆载重限制的前提下,合理安排配送车辆从配送中心出发,为多位客户完成货物配送,确保所有客户的需求在规定时间内得到满足,并尽可能缩短行驶距离或降低运输成本。VRPTW因在物流配送等实际应用中重要价值和复杂特性,已成为智能优化与运筹学领域的研究热点^[8-10]。近年来,众多学者围绕该问题展开深入研究,广泛应用各类元启发式算法进行求解,如遗传算法、禁忌搜索算法、蚁群算法等,并在提升求解精度等方面取得了显著成果^[11-13]。例如,为应对卡车与无人机协同配送问题,文献[14]构建了一种考虑客户时间窗口约束的路径优化模型,并采用变邻域搜索算法进行求解,有效降低了配送成本,但在部分测试实例中解的波动性较高。文献[15]提出了一种结合节约算法与大规模邻域搜索的混合遗传算法,仿真试验结果表明,在寻优精度方面表现良好,具有在应急物流配送中应用的可行性。文献[16]设计了一种融合多样性控制、改进克隆选择与极限优化策略的混合蛙跳算法,并引入自适应软时间窗惩罚机制以提升搜索能力,展现出良好的应用价值。文献[17]提出了一种改进的细菌觅食算法,通过K-means聚类构造初始解,并结合趋化操作与大邻域搜索中removal算子优化路径,但在求解客户规模较大问题时,算法仍有提升潜力。

蛇优化算法(Snake optimization algorithm,SO)是Hashim等^[18]受蛇群特殊交配行为启发,于2022年提出的一种新型元启发式算法,具有结构简单、寻优精度高、参数设置少、实现方便、更新机制灵活等优点^[19],已在图像分割^[20]、特征选择^[21]等领域得到应用,并展现出良好的优化性能。然而,目前关于该

算法在复杂约束路径优化问题中应用研究仍相对较少,特别是在带时间窗的车辆路径问题中,局部搜索能力与求解精度仍有待进一步提升。

针对蛇优化算法在求解VRPTW时存在局部搜索能力弱、收敛速度慢和求解质量低等问题,本文提出一种融合混合初始化策略与局部邻域搜索机制的混合蛇优化算法。采用最小位置匹配值法以实现求解结果的离散化解码,将实数编码转化为可执行的配送方案。引入随机插入与DBSCAN算法相结合的混合初始化策略,为优化过程提供高质量的初始解。在算法迭代过程融合局部邻域搜索机制,以提升寻优精度与搜索性能。并通过Solomon数据集和城市农产品配送问题对融合算法进行验证。

1 VRPTW 模型构建

1.1 问题描述

已知有1个物流配送中心和若干客户,配送中心及各客户点的地理坐标和客户的货物需求量均已知。运输车辆需从配送中心出发,完成若干客户的配送任务后返回,且所服务客户需求总和不超过车辆的最大载重。同时,每位客户均设有可接受的服务时间窗口,车辆必须满足客户的服务时间。当车辆提前到达客户位置时,须等待至最早服务时间方可开始服务;不允许车辆到达时间超过客户所规定的最晚服务时间,图1为VRPTW示意图。

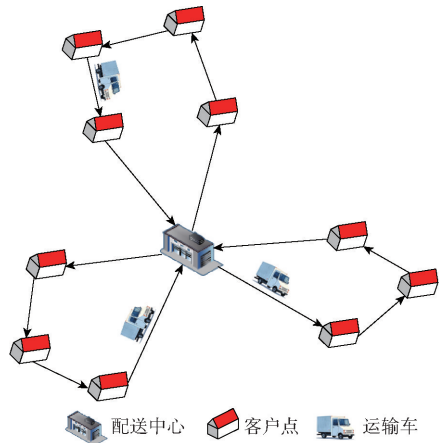


图1 VRPTW示意图

Fig. 1 VRPTW schematic

1.2 VRPTW 数学模型

为了对上述配送问题进行建模,将配送中心及客户点统一视为运输网络中节点,并在考虑车辆载重约束、时间窗限制及路线成本等因素基础上构建数学模型。设节点集合为 $V = \{0, 1, 2, \dots, n, n+1\}$,其中,0和 $n+1$ 表示配送中心, $1, 2, \dots, n$ 表示客户。 $N = V \setminus \{0, n+1\}$ 表示客户集合,车辆集合定义为 $K = \{1, 2, \dots, k_{\max}\}$,其中 $k_{\max}$ 表示最多可用车辆数

量。客户点 i 允许服务的时间为 $[a_i, b_i]$ 。为明确模型中符号含义,对主要参数与决策变量进行统一定义,如表 1 所示。

表 1 符号说明

Tab. 1 Description of symbols

类型	符号	说明
参数	A	配送路线集合
	V	所有节点集合,包含配送中心和所有客户点
	N	客户集合
	K	车辆集合
	c_{ij}	客户 i 和客户 j 之间距离
	v	车辆行驶速度
	C	车辆最大载重量
	s_i	客户 i 服务时间
	$\Delta^+(i)$	所有以 i 为起点的路线
	$\Delta^-(j)$	所有以 j 为终点的路线
	M	足够大的正数
	t_{ij}	由客户 i 到客户 j 行驶时间
	E	配送中心左时间窗
	l_i	车辆到达客户 i 时间
	L	配送中心右时间窗
	a_i	客户 i 左时间窗口
	b_i	客户 i 右时间窗口
	d_i	客户 i 需求量
决策变量	w_{ik}	车辆 k 对客户 i 开始服务时间
	x_{ijk}	车辆 k 是否从客户 i 出发前往客户 j , 如果是,则 $x_{ijk}=1$, 否则 $x_{ijk}=0$

基于参数定义与问题描述,VRPTW 数学模型为

$$\min \sum_{k \in K(i,j) \in A} c_{ij}x_{ijk} \tag{1}$$

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1 \quad (\forall i \in N) \tag{2}$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1 \quad (\forall k \in K) \tag{3}$$

$$\sum_{i \in \Delta^-(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{jik} = 0 \quad (\forall k \in K, \forall j \in N) \tag{4}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1 \quad (\forall k \in K) \tag{5}$$

$$t_{ij} = \frac{c_{ij}}{v} \tag{6}$$

$$w_{ik} + s_i + t_{ij} - w_{jk} \leq (1 - x_{ijk})M \quad (\forall k \in K, \forall (i,j) \in A) \tag{7}$$

$$a_i \left(\sum_{j \in \Delta^+(i)} x_{ijk} \right) \leq w_{ik} \leq b_i \left(\sum_{j \in \Delta^+(i)} x_{ijk} \right) \quad (\forall k \in K, \forall i \in N) \tag{8}$$

$$E \leq w_{ik} \leq L \quad (\forall k \in K \quad \forall i \in \{0, n+1\}) \tag{9}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C \quad (\forall k \in K) \tag{10}$$

在上述模型中,式(1)为目标函数,旨在最小化配送车辆总行驶里程,以提高运输效率并降低运营

成本。式(2)为客户服务约束,确保每位客户仅被分配到 1 条路线,避免重复配送,有助于提升整体运营效率。式(3)表示车辆均需从配送中心出发。式(4)用于确保车辆在访问任意客户节点 j 时,进入该节点的路线数量应等于从该节点离开的路线数量,即节点 j 流入与流出保持平衡。式(5)表示车辆必须最终返回配送中心,以保证配送路线形成闭环,便于车辆管理与任务安排。式(6)为计算车辆从节点 i 驶向节点 j 行驶时间,即为两节点之间距离除以车辆行驶速度。式(7)用于确保车辆在路线中行驶时间的连续性。当车辆 k 从客户 i 行驶至客户 j 时,到达节点 j 时间应等于从节点 i 出发时间加上两节点之间行驶时间。式(8)为客户时间窗约束,规定车辆 k 在为客户 i 提供服务时,开始服务时间必须落在客户 i 所设定的时间窗口范围内。该约束在实际配送中具有重要意义,能够反映客户对服务时间的具体要求;例如,在城市物流配送场景中,餐饮门店通常要求在上 午或中午收到食材以便当天使用,而零售超市则可能要求在下午或傍晚收货以补充库存。引入此约束可确保各类客户在指定时间段内准时收货,从而提升服务质量和客户满意度。式(9)为配送中心的时间窗约束,规定车辆从配送中心出发及返回时间均需满足配送中心所设定的时间窗口,从而确保配送任务按规定时间执行,提高管理效率。式(10)为车辆载重约束,确保车辆在初始装载阶段的装货量不超过最大载重能力,从而保障运输安全。

2 混合蛇优化算法

2.1 蛇优化算法原理

蛇优化算法遵循蛇群独特的交配行为,主要受环境温度和食物量因素影响。当环境温度较低且有食物可用时,蛇群会发生交配行为,否则蛇群会专注于寻找食物或消耗现有食物。蛇优化算法搜索过程分为勘探和开发阶段:在勘探阶段,蛇群主要通过探索周围环境寻找食物;在开发阶段,在有食物且温度很高的情况下,蛇群只会专注消耗现有食物。如果食物充足但环境温度低时,蛇群会进入战斗或交配模式。在战斗模式时,每只雄性蛇会为争取最好的雌性蛇而竞争,每只雌性蛇则会选择最好的雄性蛇。在交配模式时,雄性蛇与雌性蛇进行交配,雌性蛇会有一定的概率产卵并孵化出新蛇。

2.2 编码与解码

为了提高算法求解速度和执行效率,采用一种同时体现配送中心与客户信息的编码方式,既能够完整地表示所有客户的配送任务,又能有效地体现

车辆的分配信息,整体上实现对配送顺序与车辆路线的统一编码。通过这种方式,算法在优化过程能考虑任务的先后顺序以及车辆的合理调配,提升求解效率。当客户数目为 n ,最大车辆使用数目为 $k_{\max}$ 时,则编码长度设置为 $n + k_{\max} - 1$ 。

解码策略的核心思想:在蛇优化算法中,每条蛇个体以实数编码表示其在解空间中位置。为了将实数表示转化为可执行的配送方案,采用最小位置匹配值法^[22],对个体在各维度上数值进行排名,最小值对应整数 1,次小值对应整数 2,依此类推。排名后整数序列即为具体的配送路线方案。该解码过程实现了从连续优化空间向离散配送方案的有效映射,确保了解的可行性与实用性。

为便于理解上述编码与解码过程,如图 2 所示,直观展示该方法的执行逻辑。假设在配送问题中共有 8 位客户,最多允许配送车辆 3 辆。根据上述原则,编码长度应为 $8(\text{客户数}) + 3(\text{最大车辆数}) - 1 = 10$ 。设 1 个 10 维度个体,各维度值为 $[0.1, 1.5, -0.3, 2.2, 0.8, -1.2, 0.7, 3.6, 0.4, 1.3]$,对各个维度值排名可得序列 $[3, 8, 2, 9, 6, 1, 5, 10, 4, 7]$ 。其中 9 和 10 表示配送中心,用于划分客户集合,9 和 10 将客户 1~8 划分为 3 段,即划分为 3 条配送路线(0 表示配送中心),具体划分结果:第 1 条路线为 $0 \rightarrow 3 \rightarrow 8 \rightarrow 2 \rightarrow 0$;第 2 条路线为 $0 \rightarrow 6 \rightarrow 1 \rightarrow 5 \rightarrow 0$;第 3 条路线为 $0 \rightarrow 4 \rightarrow 7 \rightarrow 0$ 。

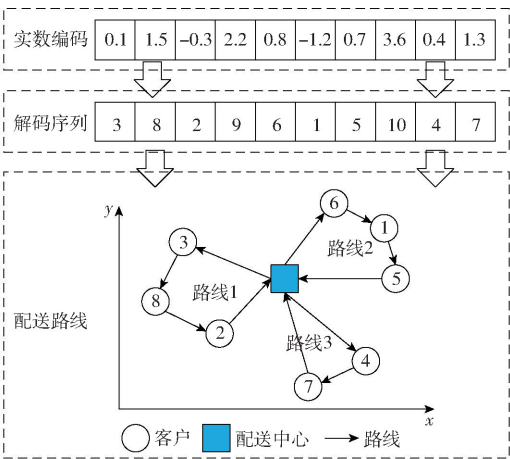


图 2 编码与解码示意图

Fig. 2 Schematic of encoding and decoding

2.3 适应度函数设计

为满足蛇优化算法在求解 VRPTW 时既要优化车辆行驶距离,又需满足车辆载重和客户时间窗等约束的需求,在数学建模基础上,依据构建的目标函数设计相应的适应度函数,并引入惩罚项机制,处理车辆载重限制与客户时间窗约束,计算式为

f(s) = c(s) + αq(s) + βw(s) (11)

其中

q(s) = \sum_{k=1}^{k_{\max}} \max \left\{ \left(\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} - C \right), 0 \right\} (12)

w(s) = \sum_{i=1}^n \max \{ (l_i - b_i), 0 \} (13)

- 式中 s——配送方案
- f(s)——配送方案总成本
- c(s)——车辆总行驶距离
- q(s)——违反装载量约束之和
- w(s)——违反客户时间窗约束之和
- α——违反载重约束的惩罚因子
- β——违反时间窗约束的惩罚因子

2.4 混合初始化策略

为提升初始种群的质量与多样性,采用一种融合随机插入与 DBSCAN 算法的混合初始化策略。假设种群规模为 N_x ,将种群按设定比例划分为 N_A 、 N_B 2 组。其中, N_A 组通过随机插入结合可行性检测构造可行路线; N_B 组通过 DBSCAN 算法按客户地理位置进行聚类,并在各簇内构造可行路线。最终,将 N_A 、 N_B 2 组合并形成初始种群,以提升解的多样性与整体质量。通过混合初始化策略,不仅保证了个体的可行性,还丰富了初始解空间,为优化过程提供高质量且多样化的初始解。

(1) 基于随机插入的初始解路线构造

为生成多样化的可行初始解,采用随机插入方法构造可行路线,具体步骤:①客户排序:对客户节点进行随机排序,以增加初始解的多样性。②客户插入:按排序顺序依次选择未访问客户,并尝试将其插入当前路线的所有可能位置。同时,依据车辆载重约束和时间窗约束判断插入的可行性。③插入方案选择:在所有满足约束的客户插入方案中随机选择 1 个方案执行,以兼顾解的质量与随机性。④路线更新:若当前路线无法插入更多客户,则关闭该路线并开启新路线,重复步骤②~③,直至所有客户均被服务。⑤初始解生成:多次运行步骤①~④,生成结构多样、约束可行的初始路线集合,为优化奠定良好基础。

(2) 基于 DBSCAN 的初始解路线构造

为提升初始解的质量,引入 DBSCAN 算法进行路线构造,具体步骤:①聚类阶段:采用 DBSCAN 算法将空间邻近的客户划分为若干簇,将被标记为噪声的客户作为单独簇处理。②簇内路径构造:在每个簇内,随机选择尚未访问的客户,根据时间窗和载重约束,尝试插入当前路线的各个可能位置;若存在满足约束的插入点,则选择增量成本最小的方式;若无合适插入点,则结束当前路线并开启新路线,直至

簇内客户全部被服务。③初始解生成:多次运行步骤①~②,通过有效控制路线的服务区域,生成紧凑且约束可行的初始路线集合,为优化提供高质量初始解。

2.5 局部邻域搜索机制

为增强蛇优化算法局部搜索与跳出局部最优能力,融合大规模邻域搜索算法 (Large neighborhood search, LNS)^[23],借助其中“破坏”与“修复”算子对蛇个体解进行重构,即使用破坏算子从当前解中移除若干客户,然后使用修复算子将被移除的客户重新插回到破坏的解中。

(1)破坏算子

破坏算子依据客户间的相似性指标,有选择性地移除若干相似客户,在保留解结构特征的同时引入扰动,以提升局部搜索的效果。随机选取 1 位客户作为初始参考点,再依据与其他客户的相关性,逐步筛选并移除若干客户,直到达到预设的移除数量。

破坏算子移除若干位相关的客户实现局部解的扰动,计算式为

$$R(i,j) = 1/(c'_{ij} + V_{ij}) \tag{14}$$

其中
$$c'_{ij} = \frac{c_{ij}}{\max c_{ij}} \tag{15}$$

$$V_{ij} = \begin{cases} 0 & (i,j \text{ 在同一条路径上}) \\ 1 & (\text{其他}) \end{cases} \tag{16}$$

式中 c'_{ij} —— c_{ij} 标准化后值

(2)修复算子

修复算子在满足车辆载重约束与时间窗约束前提下,优先考虑将被移除客户插入到对车辆总路线长度增幅最小的位置,从而提升解的质量。从待插入客户集合中选取 1 位与当前路线整体“距离最远”的客户,即插入成本最大的客户。然后,遍历所有可行路线及其插入位置,在满足车辆载重、客户时间窗、服务时间及配送中心返回时限等约束条件下,选择插入代价最小的位置将该客户合理分配至路线中,直至所有被移除客户均重新分配完毕。

2.6 HSO 算法流程

在蛇优化算法基础上引入混合初始化与局部邻域搜索机制,提出一种混合蛇优化算法 (HSO)。2 种改进方法有效弥补了原算法在迭代更新过程不足,提升了算法求解质量。HSO 算法具体步骤:①输入客户数据(客户需求量、时间窗等信息),并完成必要的数

据预处理,设定算法参数(种群规模 N_x 、最大迭代次数 $T_{\max}$ 等)。②载入解的编码与解码规则,并定义适应度函数。③采用混合初始化策略构建初始种群。④在 $T < T_{\max}$ 的条件下计算当前蛇群适应度函数,并按 SO 算法中探索食物、靠近食物、战斗与交配等更新规则更新蛇群位置;对更新后的蛇群执行局部邻域搜索机制。⑤当达到最大迭代次数 $T_{\max}$ 时,输出迭代过程最优解,并将其转换为配送方案。HSO 算法流程图如图 3 所示。

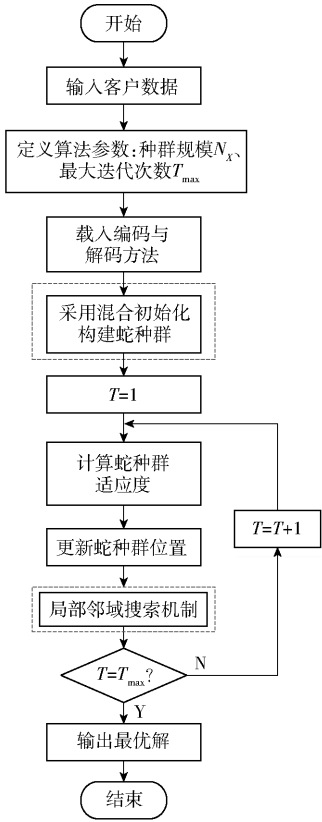


图 3 HSO 流程图

Fig. 3 Flowchart of HSO algorithm

3 仿真试验与结果分析

3.1 试验设置

仿真试验使用的计算机配置为 Intel(R) Core (TM) i7-9700F,主频 3 GHz、内存 16 GB,计算环境为 Matlab 2020b。为评估 HSO 算法的寻优性能,选用 Solomon 数据集^[24]进行仿真试验。数据集包含 56 个测试问题,分为 6 种类型:c1、c2、r1、r2、rc1 和 rc2。其中,c 类客户地理位置较为集中,r 类客户则呈随机分布,rc 类客户同时具备集中与随机的特征。每个测试问题均包含 100 个客户节点,涵盖了多样的时间窗特性,有助于全面评估算法在不同情境下适应能力。数据集当前最佳解决方案 (Best known solutions, BKS) 由 SINTEF 网站 (<https://www.sintef.no/projectweb/top/vrptw/>) 提供,作为结果评估的参考。所有算法均设置种群规模为 30,最大迭代次数为 500,每个测试问题均独立运行 20 次。

为评价算法运行速度,在测试环境和试验设置条件下,平均运行时间计算式为

$$S_{\text{AveTime}} = \frac{\sum_1^{N_{\text{num}}} \sum_1^{T_{\text{times}}} P_{\text{run}}}{N_{\text{num}} T_{\text{times}} T_{\text{max}}} \quad (17)$$

式中 S_{AveTime} ——平均运行时间
 N_{num} ——测试问题总数
 T_{times} ——独立试验次数
 P_{run} ——1 次独立试验时间

3.2 参数灵敏度

为分析混合初始化配比设置对 HSO 算法性能的影响,在固定种群规模为 30 前提下,设置多组不同混合比例 (40% : 60%、50% : 50%、60% : 40%、70% : 30%),如表 2 所示。在保持其他参数不变的条件 下,试验选取部分测试问题进行验证,每组试验均独立运行 20 次,以试验结果的平均值评估算法性能。为系统比较不同组合在多个测试问题中整体表现,图 4 以热力图形式展示了各初始化比例在不同测试问题上平均值排名。排名依据为各算法平均值,若平均值相等,则进一步比较标准差;若平均值与标准差均相等,则为并列排名。

表 2 混合初始化配比设置

Tab.2 Hybrid initialization parameter settings %

算法	DBSCAN 算法初始	
	随机插入初始化占比	化占比
HSO1	40	60
HSO2	50	50
HSO3	60	40
HSO4	70	30

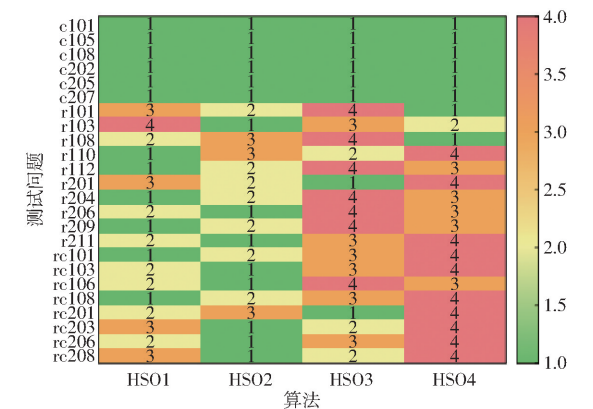


图 4 不同配比设置下 HSO 算法在测试问题上的平均值排名

Fig.4 Average rank of HSO algorithm on test problems under different parameter settings

由图 4 可见,HSO2 算法在测试问题中平均值排名优于其他算法,表明其参数配置较为合理。具体而言,采用比例 50% : 50% 有助于优化搜索空间分布,同时保持较高的种群多样性,从而避免算法陷入局部最优,提升整体寻优性能。因此,后续试验将

选择 HSO2 算法比例设置,即 50% : 50% 比例组合作为默认配置。

3.3 消融试验

为验证混合初始化策略和局部邻域搜索机制对蛇优化算法性能的提升效果,将仅引入混合初始化策略的蛇优化算法记为 HISO,在 Solomon 数据集上,对 SO、HISO、HSO 算法进行消融试验。试验结果如表 3 所示。

表 3 不同算法在部分测试问题上优化结果

Tab.3 Optimization results of different algorithms on selected test problems

测试问题	SO		HISO		HSO	
	平均值	标准差	平均值	标准差	平均值	标准差
c101	1 368.45	100.34	828.94	0	828.94	0
c103	1 579.73	84.68	993.28	0	828.06	0
c109	1 300.33	45.57	880.13	0	828.94	0
c201	759.11	35.11	703.60	2.04	591.56	0
c204	992.70	55.35	983.20	43.75	587.71	0
c206	859.76	47.09	756.48	0	588.49	0
r101	2 296.46	46.59	2 334.17	62.13	1 488.90	12.81
r103	1 788.12	37.23	1 800.86	90.82	1 069.58	10.53
r108	1 383.62	45.51	1 406.73	56.62	887.65	8.19
r201	1 995.87	48.29	2 008.16	71.40	1 237.65	14.61
r207	1 282.88	30.02	1 282.54	42.83	827.72	8.22
r211	1 146.98	33.31	1 163.32	27.71	777.74	11.51
rc101	2 129.38	76.09	2 151.27	59.04	1 441.65	24.12
rc103	1 711.37	59.42	1 638.18	6.81	1 146.31	8.35
rc104	1 524.16	45.89	1 545.20	50.83	1 067.51	12.09
rc107	1 732.16	65.14	1 566.14	0	1 148.96	17.24
rc202	2 039.33	49.22	1 789.46	0	1 086.41	7.36
rc205	2 105.15	51.50	1 855.66	0	1 131.45	8.90
rc206	1 718.94	63.76	1 729.91	30.18	1 048.30	9.88
rc208	1 269.78	43.80	1 262.01	26.75	741.48	7.83

由表 3 可见,在 c 类问题中,相较于原始 SO 算法,融合混合初始化策略的 HISO 在求解精度方面表现更为优越,表明该策略在客户聚集类问题中具有较强的适应性。混合初始化策略结合了随机插入与 DBSCAN 算法 2 种方式,在保证解的可行性前提下,有效提升了初始解的结构多样性,为优化过程提供了良好的搜索基础。在 r 类和 rc 类问题中,由于客户分布较为分散,混合初始化策略的优化效果相对有限。在 HISO 算法基础上引入局部邻域搜索机制的 HSO,通过对现有解的破坏与修复,增强了算法的跳出局部最优能力,在 r 类和 rc 类问题中表现尤为突出,能够灵活调整路线结构,显著改善求解性能;而在客户聚集的 c 类问题中,尽管性能提升幅度相对较小,但仍能有效细化解的局部结构,提高整体求解质量与算法的稳定

性。综上所述,混合初始化与局部邻域搜索机制 2 种改进策略在不同类型问题中均表现出良好的适应性,有效增强了算法的搜索能力,验证了在多类测试问题中的稳定性。

3.4 不同算法性能试验

为评估 HSO 算法的性能,将 HSO 算法与混合遗传算法 (Hybrid genetic algorithm, HGA)^[15]、混合灰狼优化算法 (Hybrid grey wolf optimizer, HGWO)^[10]、粒子群算法 (Particle swarm optimization, PSO)、大规模邻域搜索算法 (Large neighborhood search, LNS) 在 56 个测试问题上进行

试验。算法参数设置如表 4 所示,试验结果如表 5 所示。

表 4 算法参数设置
Tab.4 Algorithm parameter settings

算法	参数
HGA	$P_c = 0.9, P_m = 0.05$
PSO	$c_1 = 1.5, c_2 = 1.5, w_{\max} = 0.9, w_{\min} = 0.4$
HGWO	$a = [0, 2]$
LNS	$D = 15$
SO	$c_1 = 0.5, c_2 = 0.05$
HSO	$c_1 = 0.5, c_2 = 0.05, D = 15$

表 5 6 种算法在部分测试问题上优化结果

Tab.5 Optimization results of six algorithms on selected test problems

测试问题	BKS	指标	HGA	PSO	HGWO	LNS	SO	HSO
c101	828.94	最优值	828.94	1 117.39	828.94	828.94	1 124.66	828.94
		平均值	869.21	1 433.16	865.71	828.94	1 368.45	828.94
		标准差	19.06	129.44	45.22	0	100.34	0
c104	824.78	最优值	975.68	1 209.29	876.60	865.91	1 235.01	822.79
		平均值	1 036.83	1 311.82	993.29	897.85	1 321.65	829.16
		标准差	27.64	54.64	55.65	19.99	48.73	10.59
c109	828.94	最优值	902.70	1 240.45	828.94	828.94	1 204.84	828.94
		平均值	944.59	1 298.86	954.16	885.42	1 300.33	828.94
		标准差	21.17	27.34	58.96	33.56	45.57	0
c201	591.56	最优值	711.66	710.50	591.56	591.56	688.63	591.56
		平均值	752.05	762.57	591.56	591.56	759.11	591.56
		标准差	22.18	32.40	0	0	35.11	0
c203	591.17	最优值	738.44	843.42	588.49	588.49	840.37	588.49
		平均值	786.07	972.41	590.36	588.49	964.89	588.49
		标准差	22.13	55.32	5.76	0	73.52	0
c205	588.88	最优值	676.83	753.36	588.49	588.49	770.21	588.49
		平均值	754.07	861.18	588.49	588.49	853.58	588.49
		标准差	26.69	51.26	0	0	44.82	0
r101	1 650.8	最优值	1 544.32	2 101.52	1 465.08	1 460.21	2 207.36	1 454.77
		平均值	1 572.32	2 266.72	1 530.71	1 488.24	2 296.46	1 488.90
		标准差	14.05	69.09	30.70	13.02	46.59	12.81
r104	1 007.31	最优值	962.91	1 305.29	911.42	902.53	1 307.30	904.36
		平均值	973.31	1 449.84	952.75	922.22	1 442.54	917.39
		标准差	5.25	57.81	20.98	9.74	62.21	8.84
r107	1 104.66	最优值	1 043.21	1 513.98	1 008.14	974.11	1 498.01	973.11
		平均值	1 060.80	1 599.38	1 050.69	998.58	1 598.03	988.41
		标准差	8.62	36.58	21.76	10.08	47.40	11.66
r112	982.14	最优值	944.51	1 326.82	909.62	873.57	1 309.66	865.14
		平均值	972.98	1 377.45	946.81	897.07	1 368.04	898.82
		标准差	10.64	35.57	26.72	10.60	31.25	10.87
r201	1 252.37	最优值	1 304.24	1 907.25	1 199.57	1 201.15	1 898.66	1 203.75
		平均值	1 370.29	1 992.34	1 269.83	1 227.92	1 995.87	1 237.65
		标准差	28.32	48.86	53.15	12.11	48.29	14.61
r206	906.14	最优值	991.89	1 246.31	933.26	917.31	1 253.16	884.54
		平均值	1 046.24	1 370.20	962.00	936.56	1 361.91	913.19
		标准差	22.57	51.70	24.08	8.21	48.55	9.92

续表 5

测试问题	BKS	指标	HGA	PSO	HGWO	LNS	SO	HSO
r208	726. 82	最优值	830. 14	950. 29	705. 61	699. 37	975. 52	703. 46
		平均值	853. 98	1 020. 02	726. 00	709. 52	1 026. 15	711. 60
		标准差	13. 20	31. 29	11. 05	6. 84	29. 34	3. 83
r211	885. 71	最优值	839. 48	1 085. 31	783. 82	772. 73	1 091. 31	757. 28
		平均值	861. 70	1 141. 40	817. 97	793. 51	1 146. 98	777. 74
		标准差	11. 27	31. 84	14. 55	8. 49	33. 31	11. 51
rc101	1 696. 95	最优值	1 646. 75	1 939. 15	1 416. 43	1 451. 32	2 004. 05	1 388. 66
		平均值	1 686. 70	2 136. 35	1 565. 21	1 491. 92	2 129. 38	1 441. 65
		标准差	16. 92	93. 00	53. 53	22. 43	76. 09	24. 12
rc105	1 629. 44	最优值	1 440. 01	1 986. 00	1 360. 00	1 342. 65	1 958. 66	1 308. 17
		平均值	1 497. 84	2 130. 44	1 435. 71	1 372. 27	2 117. 48	1 330. 80
		标准差	18. 04	74. 22	35. 81	17. 66	78. 59	9. 37
rc106	1 424. 73	最优值	1 293. 34	1 725. 24	1 268. 23	1 215. 14	1 773. 49	1 161. 86
		平均值	1 309. 77	1 849. 29	1 395. 27	1 279. 89	1 850. 12	1 195. 81
		标准差	9. 80	63. 57	80. 83	30. 22	48. 44	20. 51
rc108	1 139. 82	最优值	1 111. 78	1 521. 14	1 153. 82	1 107. 81	1 492. 06	1 032. 71
		平均值	1 123. 65	1 591. 58	1 247. 39	1 158. 38	1 591. 29	1 073. 84
		标准差	7. 84	43. 36	48. 75	28. 02	45. 27	18. 03
rc201	1 406. 94	最优值	1 552. 44	1 990. 98	1 372. 87	1 371. 26	2 071. 08	1 245. 85
		平均值	1 615. 45	2 235. 52	1 464. 09	1 405. 03	2 201. 07	1 257. 50
		标准差	30. 68	85. 12	65. 47	22. 25	72. 98	7. 22
rc203	1 049. 62	最优值	1 072. 53	1 554. 56	934. 50	922. 14	1 541. 41	911. 56
		平均值	1 210. 07	1 655. 56	973. 80	941. 80	1 663. 06	916. 86
		标准差	50. 70	70. 72	25. 96	13. 02	62. 73	3. 48
rc205	1 297. 65	最优值	1 338. 96	2 004. 39	1 297. 93	1 264. 97	2 000. 49	1 109. 15
		平均值	1 391. 45	2 131. 77	1 334. 76	1 285. 56	2 105. 15	1 131. 45
		标准差	28. 31	71. 55	47. 65	10. 38	51. 50	8. 90
rc208	828. 14	最优值	897. 42	1 176. 70	803. 08	759. 06	1 171. 80	726. 76
		平均值	953. 97	1 281. 85	896. 28	835. 18	1 269. 78	741. 48
		标准差	28. 63	43. 49	37. 39	40. 29	43. 80	7. 83
平均运行时间			0. 461	0. 131	0. 066	1. 066	0. 147	1. 172

由表 5 可知,HSO 算法在 c 类问题上整体表现较为优越。除 c104 测试问题外,HSO 算法平均值均与 BKS 保持一致,体现了其较强的求解能力;在 c104 测试问题上,HSO 获得最优值优于 BKS;在 c203 ~ c205 测试问题上,HSO、HGWO、LNS 获得最优值优于 BKS。HSO 算法在解决 c 类问题中展现出良好的稳定性与求解能力,尤其在客户分布具有较强聚类特征的情况下,搜索能力优于其他算法。在 r 类问题上,除 r206 测试问题外,HSO 算法在其余测试问题中平均值均优于 BKS,体现出卓越的寻优能力。与 HGA、PSO、HGWO、SO 等算法相比,HSO 在平均值方面均有不同程度的提升;与 LNS 算法相比,HSO 在 r101、r108、r112、r201 ~ r204、r208 等测试问题上表现略逊一筹,但在其余大多数问题中表现更优,展现出良好的优化性能。在部分测试问题中,HSO 虽优于 LNS,但稳定性略低于 LNS,标准差稍大。在 r 类问题中,HSO 算法

在 10 个测试问题上获得了优于 BKS 的最优值,充分体现了其出色的求解能力。在 rc 类问题上,除 rc204 测试问题外,HSO 算法平均值优于其他算法,表现出不同程度的性能提升,整体表现仍具显著优势。HSO 算法在 16 个 rc 类测试问题中均获得了优于 BKS 的最优值,展现出卓越的优化能力。在 rc101、rc106 及 rc108 测试问题中,HSO 算法的标准差略逊于 HGA 算法;在 rc204 问题中,HSO 算法标准差略高于 LNS 算法,但在其余问题中,HSO 算法稳定性普遍优于其他算法。从算法的平均运行时间来看,HGWO 相较于其他算法在运行时间方面具有一定优势,而 HSO 与 LNS 在运行时间上差异并不显著,但与原始 SO 相比,耗时有所增加。结合表 5 可知,HSO 虽然在运行时间上并非最优,但在收敛精度和结果稳定性方面较其他算法表现出显著优势,且平均运行时间在 VRPTW 问题的实际应用中仍处于可接受范围内。

综合上述分析,相较于 HGA、PSO、LNS 等算法,

HSO 算法在求解精度和稳定性方面表现优异。HSO 算法通过混合初始化提升了初始种群质量,并利用“破坏-修复”算子增强了算法跳出局部最优的能力,使其在求解质量与稳定性之间实现良好平衡,从而展现出卓越的综合性能。

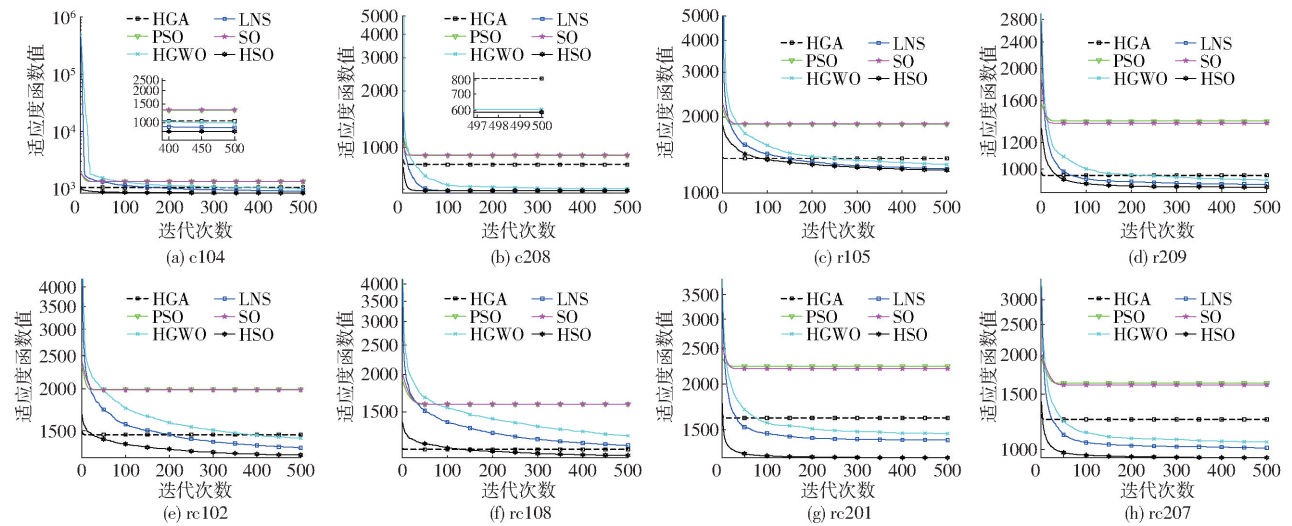


图 5 不同算法平均收敛曲线

Fig. 5 Average convergence curves of different algorithms

由图 5 可知,在 c104、c208 测试问题上,HSO 算法在初始阶段即可获得较低的适应度函数值,随着迭代的进行,逐步收敛到更高的精度。主要归因于混合初始化策略显著提升了初始种群的多样性,加快了算法的收敛速度,同时为优化阶段奠定了高质量的种群基础。在 r105、r209 测试问题上,HSO 收敛优势明显,寻优精度在所有算法中最高,展现出较强的搜索能力。与其他算法相比,HSO 算法在处理 rc102、rc108、rc201、rc207 测试问题时寻优性能同样表现出色。特别是在 rc108 测试问题上,当迭代次数达到第 133 次之前,HGA 寻优精度优于 HSO;但随着迭代次数的不断增加,HSO 寻优精度逐渐提升并最终超越 HGA。主要归因于在算法迭代过程,局部邻域搜索机制通过对解的破坏与修复,有效增强了算法跳出局部最优的能力,从而提升了寻优精度。综上所述,通过对算法平均收敛曲线的分析,进一步验证了 HSO 在求解 VRPTW 问题上卓越性能。借助混合初始化和局部邻域搜索机制,HSO 算法在保证收敛精度的同时显著加快了收敛速度,从而更高效地解决复杂组合优化问题。

3.6 不同算法稳定性

选取 c103、c208、r106、r209、rc102 和 rc206 共 6 个测试问题进行箱线图分析,既能展示算法所得的最优解与最差解,又能直观反映结果分布离散程度,为算法性能全面评估提供有力依据,结果如

3.5 不同算法收敛性

为更直观地分析不同算法的收敛特性,绘制了 HSO 与其他算法在 8 个测试问题(c104、c208、r105、r209、rc102、rc108、rc201、rc207)上平均收敛曲线,如图 5 所示。

图 6 所示。

由图 6 可知,HSO 算法均表现出显著优势。在 c103 问题中,HSO 结果波动较小且分布更为集中,表明在多次运行中能够持续获得较优解;在 c208 问题中,HSO 求解结果与 LNS 算法相当,但优于 SO 算法;在其他测试问题中,HSO 箱线图中位线普遍低于其他算法,且箱体高度较小,说明具有良好的稳定性与可靠性。此外,与其他算法相比,HSO 在箱线图中异常值数量更少且偏差幅度较小,结果分布更加集中,进一步验证了算法在多次运行下稳定性与鲁棒性。综合来看,HSO 算法稳定性在 6 种算法中最优,且相比 SO 有显著提升,充分说明了本文改进策略的有效性。

3.7 Wilcoxon 秩和检验

为进一步检验改进算法性能,对算法在 56 个测试问题上运行 20 次试验结果进行 Wilcoxon 秩和检验^[25-26],检验结果如表 6 所示。其中,当检验结果 p 小于 0.05 时,说明与 HSO 算法存在显著差异, S 的符号“+”、“-”以及“=”分别表示 HSO 算法性能优于、劣于和相当于对比算法,“N/A”表示不适用,即无法进行显著性判断。

由表 6 可知,大部分 p 均小于 0.05,表明 HSO 与其他算法的试验结果存在显著差异。此外,HSO 在 56、56、47、32、56 个测试问题上优于其他算法。在 56 个测试问题上,HSO 均与 HGA、PSO 存在显著差异。对于 HGWO 算法,在 c201、c203、c204、c208、

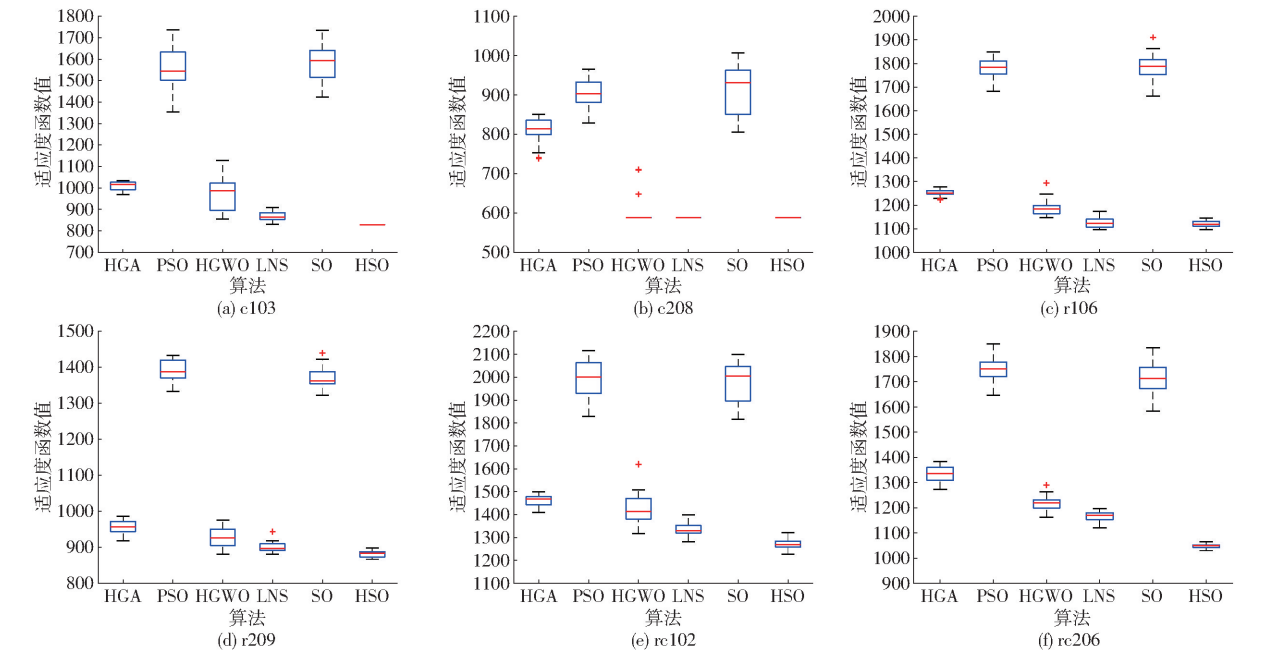


图 6 不同算法箱线图

Fig.6 Boxplots of different algorithms

表 6 Friedman 检验结果
Tab.6 Friedman test results

测试问题	HGA		PSO		HGWO		LNS		SO	
	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>
c101	7.17×10^{-9}	+	8.01×10^{-9}	+	1.02×10^{-7}	+	8.04×10^{-2}	=	8.01×10^{-9}	+
c102	1.58×10^{-8}	+	3.30×10^{-8}	+	3.05×10^{-4}	+	8.31×10^{-2}	=	3.30×10^{-8}	+
c103	7.99×10^{-9}	+	8.01×10^{-9}	+	8.01×10^{-9}	+	7.95×10^{-9}	+	8.01×10^{-9}	+
c104	6.49×10^{-8}	+	6.50×10^{-8}	+	6.50×10^{-8}	+	6.50×10^{-8}	+	6.50×10^{-8}	+
c105	1.80×10^{-8}	+	2.40×10^{-8}	+	2.40×10^{-8}	+	1.39×10^{-4}	+	2.40×10^{-8}	+
c106	7.02×10^{-9}	+	8.01×10^{-9}	+	8.01×10^{-9}	+	3.31×10^{-6}	+	8.01×10^{-9}	+
c107	5.80×10^{-9}	+	8.01×10^{-9}	+	8.01×10^{-9}	+	2.99×10^{-8}	+	8.01×10^{-9}	+
c108	7.98×10^{-9}	+	8.01×10^{-9}	+	8.01×10^{-9}	+	2.99×10^{-8}	+	8.01×10^{-9}	+
c109	7.72×10^{-9}	+	8.01×10^{-9}	+	8.01×10^{-9}	+	2.98×10^{-8}	+	8.01×10^{-9}	+
c201	7.90×10^{-9}	+	7.92×10^{-9}	+	8.04×10^{-2}	=	N/A	N/A	7.99×10^{-9}	+
c202	8.01×10^{-9}	+	8.01×10^{-9}	+	1.97×10^{-2}	+	N/A	N/A	8.01×10^{-9}	+
c203	7.99×10^{-9}	+	8.01×10^{-9}	+	1.63×10^{-1}	=	N/A	N/A	8.01×10^{-9}	+
c204	8.01×10^{-9}	+	8.01×10^{-9}	+	3.42×10^{-1}	=	N/A	N/A	8.01×10^{-9}	+
c205	8.01×10^{-9}	+	8.01×10^{-9}	+	N/A	N/A	N/A	N/A	8.01×10^{-9}	+
c206	8.01×10^{-9}	+	8.01×10^{-9}	+	N/A	N/A	N/A	N/A	8.01×10^{-9}	+
c207	8.01×10^{-9}	+	8.01×10^{-9}	+	N/A	N/A	N/A	N/A	8.01×10^{-9}	+
c208	8.01×10^{-9}	+	8.01×10^{-9}	+	8.06×10^{-2}	=	N/A	N/A	8.01×10^{-9}	+
r101	6.79×10^{-8}	+	6.80×10^{-8}	+	1.25×10^{-5}	+	7.97×10^{-1}	=	6.80×10^{-8}	+
r102	6.79×10^{-8}	+	6.80×10^{-8}	+	7.58×10^{-6}	+	5.08×10^{-1}	=	6.80×10^{-8}	+
r103	6.80×10^{-8}	+	6.80×10^{-8}	+	9.13×10^{-7}	+	7.71×10^{-3}	+	6.80×10^{-8}	+
r104	6.77×10^{-8}	+	6.80×10^{-8}	+	2.06×10^{-6}	+	1.02×10^{-1}	=	6.80×10^{-8}	+
r105	6.80×10^{-8}	+	6.80×10^{-8}	+	3.94×10^{-7}	+	1.01×10^{-3}	+	6.80×10^{-8}	+
r106	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	5.98×10^{-1}	=	6.80×10^{-8}	+
r107	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	7.11×10^{-3}	+	6.80×10^{-8}	+
r108	6.76×10^{-8}	+	6.80×10^{-8}	+	1.16×10^{-4}	+	9.79×10^{-3}	-	6.80×10^{-8}	+
r109	1.23×10^{-7}	+	6.80×10^{-8}	+	3.42×10^{-7}	+	4.99×10^{-2}	+	6.80×10^{-8}	+
r110	6.80×10^{-8}	+	6.80×10^{-8}	+	2.36×10^{-6}	+	1.33×10^{-1}	=	6.80×10^{-8}	+
r111	9.17×10^{-8}	+	6.80×10^{-8}	+	3.42×10^{-7}	+	5.98×10^{-1}	=	6.80×10^{-8}	+
r112	6.79×10^{-8}	+	6.80×10^{-8}	+	1.23×10^{-7}	+	5.79×10^{-1}	=	6.80×10^{-8}	+

续表 6

测试问题	HGA		PSO		HGWO		LNS		SO	
	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>	<i>p</i>	<i>S</i>
r201	6.80×10^{-8}	+	6.80×10^{-8}	+	7.11×10^{-3}	+	2.39×10^{-2}	−	6.80×10^{-8}	+
r202	6.78×10^{-8}	+	6.80×10^{-8}	+	5.90×10^{-5}	+	6.01×10^{-2}	=	6.80×10^{-8}	+
r203	6.80×10^{-8}	+	6.80×10^{-8}	+	8.10×10^{-2}	=	1.44×10^{-2}	−	6.80×10^{-8}	+
r204	6.80×10^{-8}	+	6.80×10^{-8}	+	2.30×10^{-5}	+	2.56×10^{-2}	−	6.80×10^{-8}	+
r205	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	6.80×10^{-8}	+
r206	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	3.94×10^{-7}	+	6.80×10^{-8}	+
r207	6.80×10^{-8}	+	6.80×10^{-8}	+	5.17×10^{-6}	+	5.63×10^{-4}	+	6.80×10^{-8}	+
r208	6.80×10^{-8}	+	6.80×10^{-8}	+	2.04×10^{-5}	+	1.90×10^{-1}	=	6.80×10^{-8}	+
r209	6.80×10^{-8}	+	6.80×10^{-8}	+	2.06×10^{-6}	+	4.17×10^{-5}	+	6.80×10^{-8}	+
r210	6.80×10^{-8}	+	6.80×10^{-8}	+	2.96×10^{-7}	+	1.81×10^{-5}	+	6.80×10^{-8}	+
r211	6.80×10^{-8}	+	6.80×10^{-8}	+	2.22×10^{-7}	+	5.25×10^{-5}	+	6.80×10^{-8}	+
rc101	6.79×10^{-8}	+	6.80×10^{-8}	+	6.92×10^{-7}	+	1.20×10^{-6}	+	6.80×10^{-8}	+
rc102	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	7.95×10^{-7}	+	6.80×10^{-8}	+
rc103	6.60×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+
rc104	6.79×10^{-8}	+	6.80×10^{-8}	+	2.22×10^{-7}	+	3.94×10^{-7}	+	6.80×10^{-8}	+
rc105	6.78×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	6.80×10^{-8}	+
rc106	6.79×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	1.66×10^{-7}	+	6.80×10^{-8}	+
rc107	6.80×10^{-8}	+	6.80×10^{-8}	+	7.90×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+
rc108	6.75×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+
rc201	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+
rc202	6.75×10^{-8}	+	6.75×10^{-8}	+	6.75×10^{-8}	+	6.75×10^{-8}	+	6.75×10^{-8}	+
rc203	6.78×10^{-8}	+	6.78×10^{-8}	+	6.78×10^{-8}	+	6.78×10^{-8}	+	6.78×10^{-8}	+
rc204	6.67×10^{-8}	+	6.78×10^{-8}	+	1.89×10^{-1}	=	9.67×10^{-5}	−	6.78×10^{-8}	+
rc205	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+	6.80×10^{-8}	+
rc206	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+
rc207	6.78×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+
rc208	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+	6.79×10^{-8}	+
+ / − / = / N / A	56/0/0/0		56/0/0/0		47/0/6/3		32/5/11/8		56/0/0/0	

r203 和 rc204 上,HSO 与 HGWO 之间差异不明显,在 c205 ~ c207 上,由于两者数据过于接近,HSO 与 HGWO 之间未表现出显著差异。与 LNS 算法相比,HSO 算法在 32 个测试问题上整体性能优于 LNS,尤其在处理 rc 类问题时展现出更强的搜索能力,但在 r108、r201、r203、r204、rc204 上,HSO 结果略逊于 LNS 算法。在 11 个测试问题上,HSO 与 LNS 的性能差异不明显。在 r201 ~ r208 测试问题中,由于两者所得结果差异较小,HSO 与 LNS 之间无法进行显著性检验。总体而言,HSO 在寻优能力方面相较于其他 5 种算法表现出显著优势,表明提出的 2 种改进策略有效提升了算法的优化性能,增强了其跳出局部最优的能力,从而使 HSO 在复杂组合优化问题上展现出更为优异的性能。

3.8 Friedman 检验

为了进一步验证 HSO 算法与其他 5 种算法之间的差异程度,采用 Friedman 检验^[27]对 6 种算法在

测试问题上运行 20 次所得的平均值进行非参数检验,以评估不同算法之间的差异是否具有统计学上的显著性。

检验结果为 HSO 的平均秩为 1.29,排名第 1,其次是 LNS(1.80)和 HGWO(3.16),分别位居第 2 和第 3,HGA(3.75)排名第 4,SO(5.38)、PSO(5.63)依次位列第 5、第 6。整体来看,HSO 在数据集上的表现与其他算法存在显著性差异,表明在寻优精度方面具有明显优势。主要得益于采用融合随机插入与 DBSCAN 算法的混合初始化策略,有效提升了初始解的质量,尤其在客户聚集的 c 类问题中表现出较强的适应性和优化效果;另一方面,引入局部邻域搜索机制中破坏与修复算子,增强了算法局部搜索能力,提高了算法在 r 类和 rc 类问题上求解质量。综上所述,2 种策略的有机结合不仅提升了算法的寻优能力,还增强了算法在不同类型问题中适应性及稳定性,验证了 HSO 算法在复杂组合优化问题上的优越性。

4 HSO 算法在城市农产品配送问题中应用

为进一步验证 HSO 算法在实际应用场景中适应性与实用价值,基于某区域的实际农产品配送数据,构建了 1 个城市农产品配送问题的测试实例。该实例涵盖客户地址分布、需求量、服务时间窗以及配送车辆的载重与数量等核心要素,具体数据如表 7 所

表 7 配送中心与客户信息

Tab.7 Information of distribution center and customers

编号	x 坐标/ km	y 坐标/ km	需求量/ kg	时间窗	服务时 间/min
0	40	50	0	07:00—21:00	0
1	45	68	40	17:00—17:30	20
2	45	70	10	16:30—17:00	20
3	42	66	40	08:00—09:30	35
4	42	68	10	16:00—16:30	20
5	20	80	31	12:40—13:40	40
6	10	90	22	08:40—10:00	20
7	35	69	17	13:30—14:20	25
8	25	85	3	15:10—16:10	50
9	22	75	16	08:30—09:30	25
10	22	85	23	14:10—15:10	20
11	60	30	25	10:20—11:20	20
12	15	25	18	12:00—13:00	10
13	70	40	15	13:00—14:00	20
14	40	66	47	09:40—10:20	10
15	38	68	30	10:20—11:00	20
16	25	90	32	14:00—15:00	50
17	80	20	20	16:00—17:00	20
18	95	50	18	19:40—20:20	10
19	55	10	30	18:00—19:00	15
20	35	92	26	19:00—20:00	20
21	90	35	14	08:40—09:40	28
22	20	10	16	09:40—10:40	20
23	75	60	24	10:40—11:40	20
24	10	40	19	11:40—12:40	20
25	65	90	21	12:40—13:30	35
26	5	10	12	14:00—15:00	25
27	15	60	17	13:20—14:20	20
28	85	10	20	14:20—15:20	30
29	10	75	28	17:00—17:50	10
30	25	40	23	15:40—16:40	20
31	20	95	30	18:40—19:30	20
32	30	25	11	12:00—13:00	20
33	60	80	20	13:00—13:50	20
34	10	55	22	14:00—15:00	10
35	50	66	20	07:20—08:00	20
36	90	80	15	16:40—17:40	20
37	40	10	12	17:40—18:40	25
38	40	69	10	14:00—14:40	20
39	38	70	10	13:40—14:20	30
40	35	66	5	12:00—12:30	20

示。其中编号 0 表示配送中心,车载最大装载质量为 200 kg,车速为 15 km/h,最大车辆使用数目为 25。配送问题需满足以下约束条件:车辆从配送中心出发并返回形成闭环路线,且车辆在配送过程中载重不得超过最大装载质量,以保证运输安全和载重合理;此外,车辆必须在客户指定的时间窗内完成配送,以确保客户能够按时收到产品。针对该实例,设置种群规模为 30,最大迭代次数为 500,并进行 20 次独立试验,以测试和评估所提算法的优化能力及其在实际问题中应用潜力。试验结果如表 8 所示,不同算法求解的平均收敛曲线如图 7 所示,配送车辆路线如图 8 所示。

表 8 不同算法在城市农产品配送问题中优化结果

Tab.8 Optimization results of different algorithms for urban agricultural product distribution problem

算法	HGA	PSO	HGWO	LNS	SO	HSO
最优值	937.18	1 113.10	914.94	911.02	1 115.15	902.70
平均值	981.18	1 185.64	940.81	917.07	1 183.98	914.21
标准差	16.04	36.92	17.37	3.47	47.86	4.25
最差值	1 009.36	1 251.46	980.69	927.51	1 272.17	918.23

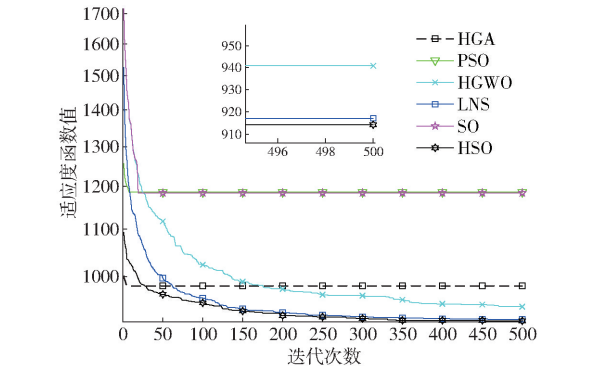


图 7 不同算法平均收敛曲线

Fig. 7 Average convergence curves of different algorithms

根据表 8 可知,HSO 算法平均值较 SO 算法降低 22.79%,较 HGA、PSO、HGWO、LNS 算法分别降低 6.83%、22.89%、2.83%、0.31%;虽然 HSO 算法的标准差略逊于 LNS 算法,但较 SO 算法降低 91.12%,较 HGA、PSO、HGWO 算法分别降低 73.50%、88.49%、75.53%。结果表明,HSO 算法表现出良好的求解精度与稳定性,进一步验证了在复杂组合优化问题中的有效性和优越性。由图 7 可知,HGA、PSO 和 SO 的平均收敛曲线随着迭代次数的增加逐渐趋于平缓。相比之下,尽管 HGWO、LNS 也表现出一定的收敛趋势,但收敛速度明显低于 HSO。总体来看,HSO 在寻优精度和收敛速度方面表现出更优的性能,主要得益于混合初始化策略生成了高质量的初始种群,同时,融合局部邻域搜索机制中“破坏-修复”算子,有助于

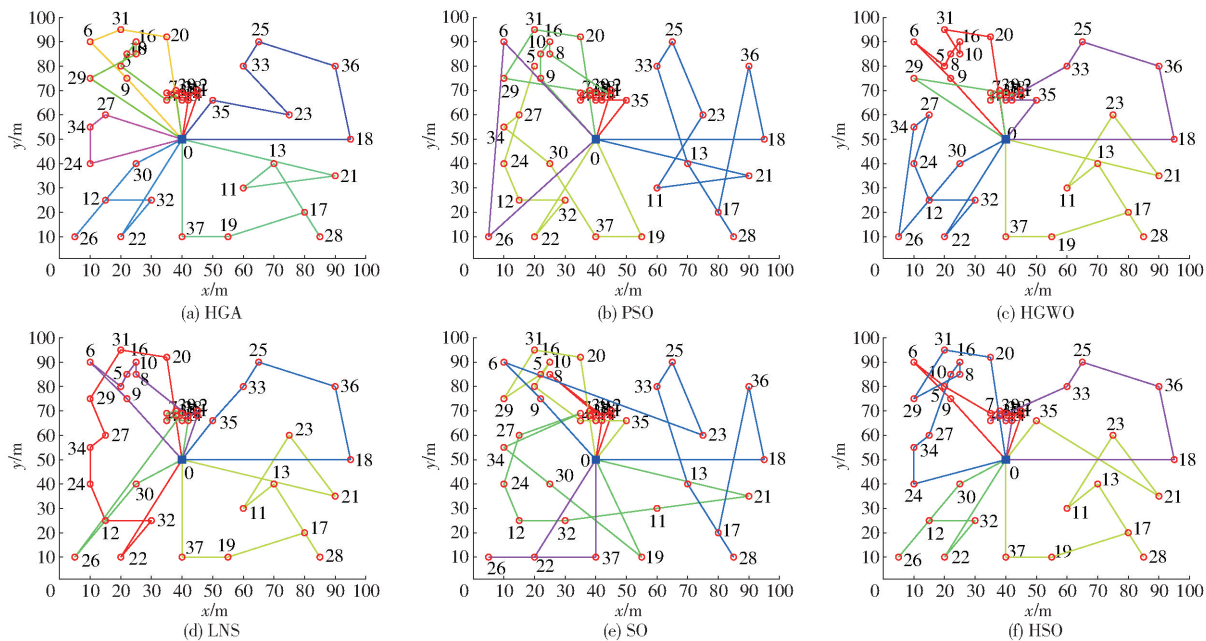


图 8 不同算法求解的配送车辆路线

Fig. 8 Delivery vehicle routes solved by different algorithms

动态调整可行解,进一步增强算法的局部搜索能力。

5 结论

(1) 针对带时间窗的车辆路径问题约束条件复杂、求解难度大的特点,在蛇优化算法的基础上进行了改进,提出了一种混合蛇优化算法。在求解精度和稳定性方面均有显著提升,展现出良好的应用前景和实际价值。

(2) 通过引入最小位置匹配值方法对求解结果进行离散化,使蛇优化算法具备处理离散编码的能力,从而提升了算法在复杂约束问题中的求解效率。为增强初始种群的质量与多样性,设计了一种融合

随机插入与 DBSCAN 算法的混合初始化策略,有效拓展了解空间覆盖范围。同时,引入局部邻域搜索机制,并结合“破坏-修复”算子对可行解进行结构性调整,增强了局部搜索能力,提高了算法跳出局部最优的能力,进一步优化了整体求解性能。

(3) 借助参数灵敏度分析确定了混合初始化最佳配比,并开展消融试验验证了改进策略有效性。在 Solomon 数据集上进行仿真试验,并采用 Wilcoxon 秩和检验及 Friedman 检验对结果进行了评估。试验结果验证了所提策略在提升算法求解精度与稳定性方面具有显著效果。最后通过应用于城市农产品配送实例,进一步证明了算法在求解复杂组合优化问题上的优势。

参 考 文 献

[1] Gutiérrez-Sánchez A, Rocha-Medina L B. VRP variants applicable to collecting donations and similar problems: a taxonomic review[J]. Computers & Industrial Engineering, 2022,164:107887.

[2] 蒋华伟, 郭陶, 杨震. 车辆路径问题研究进展[J]. 电子学报, 2022,50(2):480-492.

Jiang Huawei, Guo Tao, Yang Zhen. Research progress of vehicle routing problem[J]. Acta Electronica Sinica, 2022,50(2):480-492. (in Chinese)

[3] Rezaei B, Guimaraes F G, Enayatifar R, et al. Combining genetic local search into a multi-population imperialist competitive algorithm for the capacitated vehicle routing problem[J]. Applied Soft Computing, 2023,142:110309.

[4] 陈理, 赖有春, 王帅北, 等. 基于改进遗传算法的智能实时餐厨垃圾收运路径优化[J]. 农业机械学报, 2025,56(6):119-129.

Chen Li, Lai Youchun, Wang Shuaipei, et al. Intelligent real-time optimization of food waste collection and transportation route based on improved genetic algorithm[J]. Transactions of the Chinese Society for Agricultural Machinery, 2025,56(6):119-129. (in Chinese)

[5] Zhang J, Li Y. Vehicle routing problem for cold-chain drug distribution with epidemic spread situation[J]. Expert Systems with Applications, 2025,262:125186.

[6] 雷金美, 孙宇, 朱洪杰. 改进蚁群算法在带时间窗车辆路径规划问题中的应用[J]. 计算机集成制造系统, 2022,28(11):3535-3544.

- Lei Jinxian, Sun Yu, Zhu Hongjie. Improved ant colony optimization algorithm for vehicle routing problems with time window[J]. Computer Integrated Manufacturing Systems, 2022,28(11):3535–3544. (in Chinese)
- [7] Liu J, Tong L, Xia X. A genetic algorithm for vehicle routing problems with time windows based on cluster of geographic positions and time windows[J]. Applied Soft Computing, 2025,169:112593.
- [8] Wu Q, Xia X, Song H, et al. A neighborhood comprehensive learning particle swarm optimization for the vehicle routing problem with time windows[J]. Swarm and Evolutionary Computation, 2024,84:101425.
- [9] Kuo R J, Luthfiansyah M F, Masruroh N A, et al. Application of improved multi-objective particle swarm optimization algorithm to solve disruption for the two-stage vehicle routing problem with time windows[J]. Expert Systems with Applications, 2023, 225:120009.
- [10] 马祥丽, 马良, 张惠珍. 不确定车辆数的多约束车辆路径问题[J]. 计算机工程与应用, 2025,61(6):369–376.
Ma Xiangli, Ma Liang, Zhang Huizhen. Multi-constraint vehicle routing problem with variable fleets[J]. Computer Engineering and Applications, 2025,61(6):369–376. (in Chinese)
- [11] 陈玉全, 冯丽曼, 孙克璇, 等. 基于改进分数阶粒子群算法的多无人车取送货任务调度方法[J]. 农业机械学报, 2025,56(6):109–118.
Chen Yuquan, Feng Liman, Sun Kexuan, et al. Task scheduling for multi-unmanned vehicle delivery using improved fractional-order particle swarm optimization algorithm[J]. Transactions of the Chinese Society for Agricultural Machinery, 2025,56(6):109–118. (in Chinese)
- [12] Dong W, Zhou K, Qi H, et al. A tissue p system based evolutionary algorithm for multi-objective VRPTW[J]. Swarm and Evolutionary Computation, 2018,39:310–322.
- [13] Tan L, Lin F, Wang H. Adaptive comprehensive learning bacterial foraging optimization and its application on vehicle routing problem with time windows[J]. Neurocomputing, 2015,151:1208–1215.
- [14] Kuo R J, Lu S, Lai P, et al. Vehicle routing problem with drones considering time windows[J]. Expert Systems with Applications, 2022,191:116264.
- [15] 韩孟宜, 丁俊武, 陈梦覃, 等. 基于混合遗传算法的应急物资配送路径优化[J]. 科学技术与工程, 2021,21(22):9432–9439.
Han Mengyi, Ding Junwu, Chen Mengtan, et al. Optimization of emergency material distribution path based on hybrid genetic algorithm[J]. Science Technology and Engineering, 2021,21(22):9432–9439. (in Chinese)
- [16] Luo J, Li X, Chen M, et al. A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows[J]. Information Sciences, 2015,316:266–292.
- [17] 李珺, 郝丽艳, 何奕涛, 等. 求解带时间窗车辆路径优化问题的改进细菌觅食算法[J]. 计算机工程, 2021,47(11): 44–53.
Li Jun, Hao Liyan, He Yitao, et al. Improved bacterial foraging algorithm for solving vehicle routing optimization problem with time windows[J]. Computer Engineering, 2021,47(11):44–53. (in Chinese)
- [18] Hashim F A, Hussien A G. Snake optimizer: a novel meta-heuristic optimization algorithm[J]. Knowledge-Based Systems, 2022,242:108320.
- [19] Zhu Y, Huang H, Wei J, et al. ISO: an improved snake optimizer with multi-strategy enhancement for engineering optimization[J]. Expert Systems with Applications, 2025,281:127660.
- [20] Song H, Wang J, Bei J, et al. Modified snake optimizer based multi-level thresholding for color image segmentation of agricultural diseases[J]. Expert Systems with Applications, 2024,255:124624.
- [21] Wang J, Wang Y. Multi-strategy enhanced snake optimizer for quantitative structure-activity relationship modeling[J]. Applied Mathematical Modelling, 2024,132:531–560.
- [22] 蔡延光, 方春城, 吴艳林, 等. 求解 TSP 的离散野马优化算法[J]. 计算机工程与应用, 2024,60(1):145–153.
Cai Yanguang, Fang Chuncheng, Wu Yanlin, et al. Discrete wild horse optimizer for TSP[J]. Computer Engineering and Applications, 2024,60(1):145–153. (in Chinese)
- [23] Koyuncuoglu M U, Demir L. An adaptive hybrid variable-large neighborhood search algorithm for profit maximization problem in designing production lines[J]. Computers & Industrial Engineering, 2023,175:108871.
- [24] Wang Y, Wang L, Peng Z, et al. A multi ant system based hybrid heuristic algorithm for vehicle routing problem with service time customization[J]. Swarm and Evolutionary Computation, 2019,50:100563.
- [25] Lakshmi A V, Mohanaiah P. WOA – TLBO: whale optimization algorithm with teaching-learning-based optimization for global optimization and facial emotion recognition[J]. Applied Soft Computing, 2021,110:107623.
- [26] 苏力德, 张永, 王健, 等. 基于改进动态时间规整算法的奶牛步态分割方法[J]. 农业机械学报, 2020,51(7):52–59.
Su Lide, Zhang Yong, Wang Jian, et al. Segmentation method of dairy cattle gait based on improved dynamic time warping algorithm[J]. Transactions of the Chinese Society for Agricultural Machinery, 2020,51(7):52–59. (in Chinese)
- [27] Duan Y, Yu X. A collaboration-based hybrid GWO – SCA optimizer for engineering optimization problems[J]. Expert Systems with Applications, 2023,213:119017.