Computer Science and Application 计算机科学与应用, 2024, 14(7), 51-65
Published Online July 2024 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2024.147163

Hans汉斯

基于ALNS改进的蜣螂优化算法求解带时间窗的车路径问题

贾悦栋，张隆浩，罗　晶

河北地质大学信息工程学院，河北　石家庄

收稿日期：2024年6月4日；录用日期：2024年7月8日；发布日期：2024年7月15日

摘　要

针对带时间窗的车辆路径问题(Vehicle Routing Problems with Time Windows, VRPTW)，提出一种混合大规模领域搜索的改进蜣螂优化算法(Improved Dung Beetle Optimization of ALNS, ALSN-IDBO)进行求解。本文主要的改进点为：1) 设计新的编码解码方式实现连续蜣螂位置向量向离散客户序列的转化；2) 对于蜣螂优化算法的初始化采用随机、贪婪、最邻近而策略；3) 在ALNS中设计了3个移除算子和3个重插算子；4) 在传统的DBO中针对繁育的蜣螂和小蜣螂分别改进为螺旋搜索策略和三角游走策略。通过在标准Solomon数据集的部分算例进行实验，将本文算法与GA、DBO、ALNS算法进行对比，实验结果表明，本文所提出的混合大规模领域搜索的改进蜣螂优化算法能找到更好的解，并且寻优能力和稳定性均优于对比算法。

关键词

自适应大规模领域搜索算法，蜣螂优化算法，车辆路径问题，螺旋搜索，三角游走

Improved Dung Beetle Optimization Algorithm Based on ALNS for Vehicle Routing Problem with Time Windows

Yuedong Jia, Longhao Zhang, Jing Luo

College of Information Engineering, Hebei GEO University, Shijiazhuang Hebei

Received: Jun. 4th, 2024; accepted: Jul. 8th, 2024; published: Jul. 15th, 2024

Abstract

For the Vehicle Routing Problems with Time Windows (VRPTW), an Improved Dung Beetle Opti-

文章引用: 贾悦栋, 张隆浩, 罗晶. 基于ALNS改进的蜣螂优化算法求解带时间窗的车路径问题[J]. 计算机科学与应用, 2024, 14(7): 51-65. DOI: 10.12677/csa.2024.147163

mization of ALNS (ALSN-IDBO) with hybrid large-scale domain search is proposed to for solving. The main improvements in this paper are 1) designing a new encoding and decoding method to realize the transformation of continuous dung beetle position vectors to discrete client sequences; 2) adopting random, greedy, and nearest-neighbor while strategies for the initialization of the dung beetle optimization algorithm; 3) designing three removal operators and three re-insertion operators in ALNS; and 4) improving the traditional DBO for the breeding dung beetles and the small dung beetles with a spiral search strategy and a triangular wandering strategy. The algorithm in this paper is compared with GA, DBO, and ALNS algorithms by conducting experiments on some arithmetic cases in the standard Solomon dataset. The experimental results show that the improved dung beetle optimization algorithm for hybrid large-scale domain search proposed in this paper can find better solutions, and the optimality finding ability and stability are better than the compared algorithms.

Keywords

Adaptive Large Neighborhood Search, Dung Beetle Optimization, Vehicle Path Problems, Spiral Search, Triangular Wandering

1. 引言

车辆路径规划问题(Vehicle Routing Problems, VRP)是在 20 世纪 60 年代提出来的，由于出行交通在大家生活中的占比越来越重，VRP 问题引起国内外研究人员的重视[1]。随着客户对时间的敏感度越来越强烈，带时间窗的车辆路径问题(Vechile Routing Problem with Time Windows, VRPTW)逐渐成为了车辆路径问题研究的一大热点。VRPTW 问题是在 VRP 问题的基础上，对地理空间维度上进行合理规划路径的最小长度，同时在时间层面上根据客户规定好的被服务时间安排有效的服务次序。VRPTW 根据时间的不同要求可以划分为硬时间窗[2]和软时间窗[3]。硬时间窗要求车辆必须严格在客户指定的时间窗内提供服务。如果车辆提前到达，需要等待直到时间窗开始，而如果迟到，则会被客户拒收。相比之下，软时间窗允许车辆在规定的时间窗外提供服务，但会因在时间窗外服务而受到惩罚。软时间窗和硬时间窗的最大区别在于，软时间窗用惩罚机制代替了等待和拒收的规定，本文是在硬时间窗的基础上进行研究。

对于复杂的车辆路径问题，单纯的元启发式算法往往难以取得理想的结果。因此，目前通常采用混合元启发式算法来解决这一问题，其目的是平衡不同算法的局部搜索和全局搜索能力，从而获得更佳效果。为此，Wang 等人[4]提出一种基于多蚂蚁系统的服务时间定制的混合启发式算法求解 VRPTW。曹二保等人[5]提出一种混合禁忌搜索和差分进化的算法对 VRPTW 采用次序优化进行求解。Zhen 等人[6]考虑带时间窗和交货时间的多站点多行程的车辆路径问题，提出了一种混合粒子群优化算法和一种混合遗传算法用于求解该模型。Jin 等人[7]提出一种结合禁忌搜索和人工免疫算法的混合算法用于求解 VRPTW。梁承姬等[8]对遗传算法进行了优化，在算法中加入基因修复，并对种群拥挤度进行了优化。雷金羡等[9]借鉴了最大最小蚁群算法，同时调整信息挥发因子的大小，并加入多种局部策略完成带时间窗车辆路径问题的优化。李楠等[10]设计了两阶段混合优化算法对初始化的路线进行动态调整并优化。R.GOEL 等人[11]提出萤火虫算法与蚁群算法的混合算法求解 VRP，用蚁群算法作为基本框架，萤火虫算法搜索解的空间，提高了解的质量。Zhang [12]等人将快速采样策略全局搜索和基于路径序列差异局部搜索相结合，

设计了一个混合多目标进化算法用于求解 VRPTW。

本文提出了一种自适应大邻域搜索算法与混合螺旋搜索三角游走算子的蜣螂优化算法相结合的方法，即 ALNS-IDBO 算法。通过在 Solomon 标准测试数据集上的实验，并进行多角度的比较，结果显示，ALNS-IDBO 算法显著增强了局部搜索能力，获得了更优质的解，有效提升了遗传算法在带时间窗车辆路径问题(VRPTW)中的应用效果，该研究具有重要的实际应用价值。同时，考虑到混合元启发式算法的多变量性和随机性，初始解的质量对群体搜索算法的搜索效率和最终解的质量有重要影响，本文还对 VRPTW 问题的初始解进行了深入研究。

2. 问题描述及数学模型建立

2.1. 问题描述

在带时间窗的车辆路径问题(VRPTW)中，主要是根据客户的时间窗和货物载重量限制，为配送中心制定决策，确定所需的车辆数量，并规划多辆配送车辆的行驶路径。路径规划考虑客户提供的服务时间窗口，确保在规定时间内完成客户的服务需求。配送车辆需要从仓库出发，经过各个商家或客户进行服务，按规划路线完成环形配送任务(返回仓库)，优化目标包括最小化所需配送车辆的数量以及总运输路径的长度。

针对该问题的约束条件和限制如下：

1) 配送时间选在交通较为顺畅的非高峰时段，不考虑交通堵塞因素；
2) 仓库或配送中心提前获取所有待服务客户的位置信息；
3) 每个商家或客户只能由一辆车提供服务；
4) 事先获取所有客户指定的服务时间窗口，允许配送车辆在最早服务时间前到达并等待；
5) 严格控制运输车辆的最大工作时间；
6) 运输车辆的最大载重量能够满足任何单个商家的货物需求；
7) 可用运输车辆数量充足。

2.2. 模型构建

2.2.1. 符号定义

Table 1. Symbol definitions and descriptions
表 1. 符号定义与说明

符号	说明
Q	车辆的最大载重量
N_0	顾客节点数
V	同质车辆数目
c_i	要服务的第 i 个顾客
r_i	顾客 i 的货物需求量
e_i	开始服务客户 c_i 的最早时间。
l_i	顾客 i 接受服务的时间窗
c_0	配送中心
e_0	为客户 c_i 提供服务的最晚时间
l_0	允许任何车辆返回配送中心 c_0 的最晚时间

VRPTW 由用 V 表示的同质车辆车队表示，有向图 G = (C, A)。在有向图 G 中，顶点集合由 C 和弧集合 A 表示。在顶点集合 $c_i \in C$ 中，$\{I \in [0,n] \mid i \in N_0\}$，其中 c_0 表示仓库，c_i 中 $\forall i \in [1,n]$ 表示要服务的 n 个顾客。每个客户 c_i 都有与之相关的送货详情。在上表 1 中会的定义符号和相对应的说明，并规定所有这些变量都假定为非负数。

在 G 中，弧集给出为 $A = \{\langle c_i, c_j \rangle | c_i, c_j \in C, i \neq j\}$。每个弧 $\langle c_i, c_j \rangle$ 代表的是两个顶点 c_i 和 c_j 之间的欧几里得距离 d_{ij}，并且 $d_{ij} = d_{ji}$。由于要配送的路线是闭合式，所以说起点和终点是同一个点 c_0。同时根据业务车辆配送的需求要设计一个二元决策 x_{ijk} 变量来满足不同客户之间的线路约束条件，代表的含义是：在两个顶点 c_i 和 c_j 客户之间是通过 k 车来运输货物的。同时满足要求的车辆 v 由 k 索引，其中 $\{k \in [1, |V|] \mid k \in N\}$，设计改变量用于标识哪辆车 k 为客户 c_i 提供服务。

$$x_{ijk} = \begin{Bmatrix} 1, \text{客户} i \text{ 到} j \text{ 是由车辆} k \text{ 运输} \\ 0, \text{客户} i \text{ 到} j \text{ 非由车辆} k \text{ 运输} \end{Bmatrix} \tag{2-1}$$

$$y_{ik} = \begin{Bmatrix} 1, \text{客户 } i \text{ 由车辆 } k \text{ 提供服务} \\ 0, \text{客户 } i \text{ 非车辆 } k \text{ 提供服务} \end{Bmatrix} \tag{2-2}$$

2.2.2. 数学模型

在 VRPTRW 中的目标是优化配送线路，使得路线距离最短、启用车辆最少，约束条件是在规定的时间窗内、不超所车辆的容量限制完成对所有客户的单次服务。那么针对于车辆使用数量最少、配送车辆的路线最短采用下方的公式(2-4)。

$$\min F_1 = |V| \tag{2-3}$$

$$\min F_2 = \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{|V|} x_{ijk} d_{ij} \tag{2-4}$$

模型的约束条件：

公式(2-5)和公式(2-6)中给出的约束表示，对于每一个客户点只有唯一的入度和出度，也就是每个客户点只允许一辆车进行服务。

$$\sum_{i=0}^{n} x_{ijk} = y_{jk} \forall k = 1, \cdots, |V| \forall j = 1, \cdots, n \tag{2-5}$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik} \forall k = 1, \cdots, |V| \forall i = 1, \cdots, n \tag{2-6}$$

方程(2-7)给出的约束代表每个顾客只能由一辆车服务。

$$\sum_{i=1}^{n} y_{ik} = 1, \ \forall i = 1, \cdots, n \tag{2-7}$$

公式(2-8)给出的约束表示所有路线均从车站出发。

$$\sum_{k=1}^{|V|} y_{0k} = |V| \tag{2-8}$$

公式(2-9)给出的约束表示每辆车的装载量不得超过其承载能力 Q。

$$\sum_{i=0}^{n} y_{ik} \times r_i \leq Q, \ \forall i = 1, \cdots, |V| \tag{2-9}$$

公式(2-10)中表示的是从客户 i 到客户 j 需要的时间表示。具体是定义 t_j，即开始为顾客 c_j 提供服务的时间；t_j 的表示方式是求和得到的：① 当车辆 k 到达客户 c_i 的时间是用 t_i 表示、② 在指定时间窗口内

开始服务客户 c_i 之前的等待时间或空闲时间是 w_i、③ 服务客户 c_i 的时间是用 s_i 表示、车辆在客户 c_i 和 c_j 之间的行程时间用 t_{ij} 表示。

$$t_i + w_i + s_i + t_{ij} = t_j \quad \forall i, j = 0, \cdots, n \quad i \neq j \tag{2-10}$$

公式(2-11)中表示的客户 c_j 开始得到服务的时间窗口。对于客户 c_j 来说需要有一个对应时间窗口，定义 e_j 是客户 c_j 可以开始服务的最早时间，l_j 是客户可以得到服务的最晚时间，因此，客户 c_j 开始接受服务的时间是车辆到达客户 c_j 的时间，加上车辆在服务客户 c_j 之前的等待时间或者扣或者空闲时间。

$$e_j \leq t_j + w_j \leq l_j \quad \forall j = 0, \cdots, n \tag{2-11}$$

公式(2-12)中表示的是车辆早到服务客户需要等待的时间或者是空闲时间。

$$w_i = \max\{e_i - t_i, 0\} \quad \forall i = 0, \cdots, n \tag{2-12}$$

3. 算法设计

3.1. 自适应大规模领域搜索算法

在解决复杂高维问题时，自适应大邻域搜索算法(ALNS)展现出了强大的适应性，能够在迭代过程中产生高质量的可行解。该算法的基本操作流程包括从破坏算子池和修复算子池中根据权重选择算子，用以生成新的解。破坏算子用于移除当前解中的个体，而修复算子则用于选择性地插入新的个体。ALNS根据破坏修复原则重新组合路径以达到配送目标。通过引入自适应层，该算法在算子选择上表现出智能性。

本文提出了 3 种破坏算子和 3 种修复算子，以丰富 ALNS 算法的算子选择池。初始化所有破坏算子的权重为 0.33，以确保在搜索初始阶段破坏修复算子被选中的可能性相同。在选择算子后，解将根据打分规则进行标记。打分规则基于以下原则：1) 破坏/修复后得到新的全局最优解增加 30 分；2) 破坏/修复后未得到全局最优解，但比当前解好增加 15 分；3) 破坏/修复后得到的解比当前解差，但为了保持解的多样性，根据模拟退火算法的概率增加 5 分。通过算子的得分和选择次数，计算出算子的权重。

3.1.1. 破坏算子

1) Random Removal

从初始解中选择需要处理的路径，并在该路径的原有排列方案中随机挑选若干客户进行移除。在本文的操作中，至少选择 3 个节点进行删除，随后通过修复算子对删除后的路径进行修复。Random Removal方法没有明确的选择顺序，这可能导致选择的节点不够理想，从而生成质量较低的解。然而，这种随机性也有助于避免算法陷入局部最优解，从而增加探索性，有利于发现更优的解决方案。

2) Random-Importance Removal

在初始解中的路径中选择某一条路径，对该路径中的客户中随机选择一个客户 i，针对于该客户 i 需要按照特定公式算出针对于该 i 客户的重要度，随后进行降序排列选择出重要度最高的三个客户进行移除，并将其放入至待处理客户集合，代修复算子进行修复。重要度计算公式(3-1)如下：

$$\xi_i = \frac{\eta(r_i)}{\eta(l_i - e_i)} + \eta(d_{0i}) \tag{3-1}$$

其中：ξ_i 代表的是关于客户点 i 的重要度，η 的函数是归一化函数，d_{oi} 表示的是配送中心到客户 i 的距离。

3) Worst Removal

该算子的核心思想是删除选定路径中不合适的客户，实现逻辑是：根据删除的客户点计算出删除前后的路径长度差值的绝对值，选择绝对值最大的至少前两个客户点，排名越靠前边的表示被删除的客户

点影响越大，也就是浪费的资源越多(ΔZ 表示浪费的资源)，越不合适。

$$\Delta Z = Z_a - Z_b \tag{3-2}$$

为了引入算法中的随机元素，在一定范围内随机选择移除的客户数量，这种策略的目标是通过剔除那些增加总成本的客户，来降低整体成本。

3.1.2. 修复算子

1) Random Repair

在破坏算子操作后，对待处理客户集合(remove_node)中的客户，采用随机插入的方法将其依次插入到路径中的随机位置从而构建新的解。

2) Greedy Repair

该算子将被移除的客户点集合置于指定的破坏路径(removal_path)中，依次遍历每个可能的插入位置。如果成本增加值小于当前最小值，则更新最小值和插入位置。在原始的破坏路径(removal_path)中插入节点，然后将节点从待恢复列表中移除。

3) Regret Repair

针对于 Regret repair 算子，主要是在 Greedy Repair 的基础上多加了一步后悔值的比较来决定客户最终的插入位置，直到待处理客户集合(remove_node)中的客户全部重新插入到新构成的解中。后悔值(RE)是指客户在最佳插入位置和次佳插入位置之间的成本差异，其数值大小代表的是未在当前位置进行插入操作所带来的未来成本增加，进而反映了客户的后悔程度。后悔值 RE 的计算方式(3-3)如下所示：

$$\Delta \mathrm{RE} = \mathrm{R}Z_{best} - \mathrm{R}Z_{nextbest} \tag{3-3}$$

3.2. 蜣螂优化算法

蜣螂优化算法(dung beetle optimizer, DBO)是 Jianka Xue 和 Bo Shen 在 2022 年提出的一种新型群体智能优化算法[13]，其灵感来自于蜣螂的滚球、跳舞、觅食、偷窃和繁殖行为。该算法同时考虑了全局探索和局部开发，从而具有收敛速度快和准确率高的特点，可以有效地解决复杂的寻优问题。

在自然界中，蜣螂以其独特的行为方式著称：它们将粪便滚成球，并依靠天体的线索进行导航，使粪球沿着直线滚动。然而，当光源消失时，蜣螂的路径便会变得曲折不定，当然各种其他的自然因素也可能导致蜣螂偏离原有的方向。蜣螂具有一种独特的“舞蹈”行为，通过这种舞蹈，它们能够重新校准自己的方向。粪球不仅是蜣螂的重要食物来源，还被用作繁殖的场所，蜣螂会在粪球中产卵。值得一提的是，蜣螂之间存在一种“偷窃”行为，一些蜣螂会抢夺其他蜣螂的粪球，作为自己的食物和繁殖资源，这样种群之间的交流竞争为蜣螂的后代繁衍，以及研究算法特性，提供现实依据。

在 DBO (Dung Beetle Optimization)算法中，每只蜣螂的位置对应一个解。蜣螂在觅食时会表现出五种典型行为：

1) 滚球：蜣螂将粪便滚成一个球，利用天体线索导航，使粪球沿直线滚动。

2) 跳舞：蜣螂通过舞蹈重新定位自己的方向。

3) 觅食：一些成年蜣螂会从地下钻出来寻找食物。

4) 偷窃：一些被称为“小偷”的蜣螂会从其他蜣螂那里偷取粪球。

5) 繁殖：蜣螂会将粪球滚到安全的地方，藏起来进行繁殖。

在此算法中，蜣螂种群被分为四类，这种分类和行为模拟了蜣螂的自然生态，有助于优化问题的解决：

1) 滚球蜣螂：受各种自然环境因素影响，不断更新其运行方向，初步寻找安全觅食的位置。

2) 繁育雏球：粪球在已知的安全区域内进行繁殖。

3) 小蜣螂：在最佳觅食区觅食，成长为成虫的蜣螂。

4) 小偷蜣螂：根据其他蜣螂的位置和最佳觅食区寻找食物。

3.2.1. 编码

为了解决 VRPTW(车辆路径问题带时间窗)问题，本文采用十进制编码方式。具体来说，每个顾客的序号作为染色体基因，其中序号 0 代表配送中心，用来表示环形配送任务。同时，序号 0 还用作分界线，以标识车辆的数量。染色体的长度设置为 X + Y + 1，其中 X 表示需要服务的顾客数量，Y 表示实际使用的车辆数量。例如，对于需要服务 8 个客户，使用了 3 辆车的情况，假设配送的 3 条路径为：0-1-5-2-0，0-3-7-0，0-4-6-8-0，那么染色体的构成为：0-1-5-2-3-7-4-6-8-0。

3.2.2. 适应度函数

VRPTW(车辆路径问题带时间窗)是一个多目标优化问题，其主要目标是最大限度地降低路线成本，这包括减少派遣车辆的数量以及它们的累积行驶距离。本文中设定了如下的适应度函数，该方法通过将问题目标函数与加权系数相加来评估该问题的解，从而将多目标函数转化为单个优化问题。适应度函数用 f 表示，定义如下(3-4)：

$$f = a \cdot |\mathrm{V}| + b \cdot \mathrm{D} \tag{3-4}$$

上方公式中 $|\mathrm{V}|$ 表示的是车辆数量之和，D 表示的是累计行驶距离的总和，a 和 b 是常量系数，分别是 100、0.001。

3.2.3. 种群初始化

为了更深探究初始化解对 DBO 的影响，本文提出了三种方法进行初始化，分别是随机初排列始化、贪婪初始化和最邻近初始化。

1) 随机初排列始化：随机选择要服务的下一个客户，直到全部的客户被服务。

2) 贪婪初始化：在一定的区域内，需要随机选择一个要服务的客户，以此类推直到客户全都被服务完毕，区域划定又称为贪婪半径，定义如下(3-5)：

$$rad = \frac{\max d_{ij} - \min d_{ij}}{2} \tag{3-5}$$

3) 最近邻初始化：非随机选择要服务的下一个客户，而是考虑距离当前选中的最近的点，直至服务完成所有客户。

3.2.4. 改进蜣螂优化算法(WDBO)

1) 滚球蜣螂

蜣螂生活中有一个习性，会把粪便滚成圆球，滚到一定位置后会停下来，滚球蜣螂以太阳为导航以保证粪球在直线路径上滚动，但是也会受到像光源强度、风等自然因素影响蜣螂的行进路线，滚球蜣螂在滚球过程中的位置更新方式如下(3-6)：

$$\begin{cases} x_i(t+1) = x_i(t) + \alpha \times k \times x_i(t-1) + b \times \Delta x \\ \Delta x = \left| x_i(t) - X^w \right| \end{cases} \tag{3-6}$$

上式中，t 代表当前迭代次数，$x_i(t)$表示第 t 次迭代时第 i 只蜣螂的位置信息，α 是一个可控制系数，表示是方向偏离程度，其中 $\alpha \in (-1,1)$，k 表示的是偏转系数 $k \in (0, 0.2)$，b 是可规大小的常量 $b \in (0,1)$，在本为中，$k = 0.1$ 和 $b = 0.3$，X^w 表示当前迭代中全局解的最差值，Δx 的作用是仿照光强。

在实际的蜣螂活动中会遇到障碍物，它需要调整自己的行径路线绕过障碍物，这个过程称为跳舞。主要实现方法是：利用切线函数进行模拟，为了获得新的前进方向，即公式(3-7)表示蜣螂跳舞行为获取新的位置。

$$x_i(t+1)=x_i(t)+\tan(\theta)\left|x_i(t)-x_i(t-1)\right| \tag{3-7}$$

上边的式中 θ 表示跳舞旋转角度，它的大小是 $\theta\in[0,\pi]$，在 θ 等于 0、$\pi/2$ 或 π 时，蜣螂的位置不会更新。

2) 改进的繁育蜣螂

对于第二类繁育蜣螂，需要针对在原始的算法区域内产卵会造成快速收敛的问题，并且缺乏多样性，达不到最优解，往往陷入局部最优。所以会对该类蜣螂做出算法改进，融合鲸鱼算法的螺旋搜索策略，鲸鱼在搜寻猎物的时候，会根据目标位置与自身位置之间的距离，螺旋式移动寻找并更新位置，由单线或者单点的搜索变成了螺旋的区域搜索，可以提高解的质量。

① 原始的繁育蜣螂

该类蜣螂是通过繁殖产生新的解，主要是采用边界选择策略模拟雌性蜣螂产卵区域，产卵区域的定义为如下(3-8)：

$$\begin{cases} Lb^*=\max\left(X^*\times(1-R),Lb\right) \\ Ub^*=\min\left(X^*\times(1+R),Ub\right) \\ R=1-t/T_{\max} \end{cases} \tag{3-8}$$

上式中的变量，X^* 表示当前迭代的局部最佳位置，Lb^* 和 Ub^* 分别表示产卵区的下限和上限，$T_{\max}$ 表示最大迭代数，当然 max 函数时两个变量之间取较大的值，min 和 max 是相反的，Lb 和 Ub 分别代表优化问题的原始参数的下上界。

在迭代更新中，育雏球位置是动态变化的，其定义为(3-9)：

$$B_i(t+1)=X^*+b_1\times\left(B_i(t)-Lb^*\right)+b_2\times\left(B_i(t)-Ub^*\right) \tag{3-9}$$

上式中，$B_i(t)$ 代表的是 t 次迭代中第 i 个育雏球的位置记录，b_1 和 b_2 代表两个大小为 1 × D 的独立随机向量，D 表示优化问题的维度大小，也就是服务客户的数量。

② 融合螺旋搜索的繁育蜣螂

针对上方原始的繁育蜣螂提出优化，借鉴鲸鱼算法中的螺旋搜索策略，会使得每一次迭代跳出局部，再扩大搜索范围寻找优质解，那么鲸鱼围捕猎物的公式如下(3-10)：

$$X(t+1)=\left|X^*(t)-X(t)\right|\cdot e^{cl}\cdot\cos(2\pi l)+X^*(t) \tag{3-10}$$

上式中的围捕位置更新是根据上一代最优位置、cos 的取值和参数 c 来影响。当 c 的值过小时，会导致搜索过程缓慢，反而当 c 的值过大又会收敛过快，为此可以采用动态的自适应的有区间限制的螺旋搜索策略来控制变量的取值，公式如下(3-11)：

$$m=e^{c\cdot g\cdot\cos\left(\frac{\pi\cdot t}{T_{\max}}\right)} \tag{3-11}$$

即采用上诉算子改进后，在迭代更新中，育雏球位置是动态变化的，其定义为(3-12)：

$$B_i(t+1)=X^*+e^{ml}\cdot\cos(2\pi l)\times b_1\left(B_i(t)-Lb^*\right)+e^{ml}\cdot\cos(2\pi l)\times b_2\times\left(B_i(t)-Ub^*\right) \tag{3-12}$$

3) 改进的小蜣螂

① 原始的小蜣螂

对于小蜣螂主要是模拟它的觅食，这个过程中首先建立一个觅食区间，在区间中小蜣螂找到最近最大最好的食物，如下公式是区间的设定(3-13)：

$$\begin{cases} Lb^b = \max\left(X^b \times (1-R), Lb\right) \\ Ub^b = \min\left(X^b \times (1+R), Ub\right) \end{cases} \tag{3-13}$$

小蜣螂在上方区域内的原始位置更新如下(3-14)：

$$x_i(t+1) = x_i(t) + C_1 \times \left(x_i(t) - Lb^b\right) + C_2 \times \left(x_i(t) - Ub^b\right) \tag{3-14}$$

② 融合三角游走的小蜣螂

针对小蜣螂的觅食行为做改进，不需要直接接近食物，而是在食物周围成三角形模式游走，目的是为了增强算法的搜索能力，方式陷入局部最优。则游走更新如下(3-15)：

$$\begin{cases} \vec{L}_1 = x_i(t) - x_j(t) \\ \vec{L}_2 = rand(\,) \times \vec{L}_1 \\ P = \vec{L}_1^{\,2} + \vec{L}_2^{\,2} - 2\vec{L}_1 \cdot \vec{L}_2 \cdot \cos\left(2\pi \cdot rand(\,)\right) \end{cases} \tag{3-15}$$

小蜣螂在父代的位置上进行更新如下(3-16)：

$$x_i(t+1) = x_i(t) + rand(\,) \cdot P \tag{3-16}$$

③ 小偷蜣螂

小偷蜣螂是种群中个体之间的一种沟通过方式，也是互相竞争的一种模式，那么小偷蜣螂的未知性更新如下(3-17)：

$$x_i(t+1) = X^b + S \times g \times \left(\left|x_i(t) - X^*\right| + \left|x_i(t) - X^b\right|\right) \tag{3-17}$$

上边的式中，X^b 是最佳的小偷蜣螂位置，g 表示一个遵循正态分布的大小为 n 的随机向量，S 表示一个常数值。

3.3. 混合 ALNS-IDBO 算法

对于 VPRTW 问题具有多约束、高复杂性的特点，同时需要保证最终解的高质量性，本文将融合 ALNS 与加入搜索策略和三角游走思想的 DBO 相结合，动态平衡全局搜索和局部搜索能力。如下为 ALNS-IDBO 流程图如图 1 所示。

4. 实验仿真及结果分析

本文所有计算在配置为 Intel(R) Pentium(R) Gold G5400 CPU @ 3.70GHz 3.70 GHz，16GB 内存的微型计算机上进行，操作系统为 Microsoft Windows11，并且利用 Python 3.79 仿真环境下进行分析实验。

4.1. 环境模型设置与算法参数

为了验证 ALNS-IDBO 求解 VRPRTW 问题的优势性，通过分别取 Solomon 数据集算例的 C 型、R 型和 RC 型进行测试，并且在两类问题中选择 25 维、50 维客户同遗传算法 GA、蜣螂优化算法 DBO，自适应大领域搜索算法 ALNS 进行比较。其中 C 型是：客户集中型，R 型是：客户随机型，RC 型：客户混合型。ALNS-IDBO 的种群规模设置 100，算法的迭代次数为 200，将所有算法均独立运行 30 次，以消除偶然因素的影响。

4.2. ALNS-GGA 与其他算法的比较

本节中，分别对 ALNS-IDBO 与其他 3 个算法求解 VRPTW 的计算结果进行比较和分析。在表 2 给

出各个算法在 C 型、R 型、RC 型的实例中，求解 25 个客户在车辆载重量为 200、700、1000 的条件下需要的车辆数和实际路径长度结果。在表 3 给出各个算法在 C 型、R 型、RC 型的实例中，求解 50 个客户在车辆载重量为 200、700、1000 的条件下需要的车辆数和实际路径长度结果。在结果图中红色点代表仓点，黑色点是服务的客户点，连线是车辆行走的路径，x 轴为客户横坐标，y 轴是客户点纵坐标，25 个客户的求解图为图 2~5、50 个客户的求解图为图 6~9 所示。

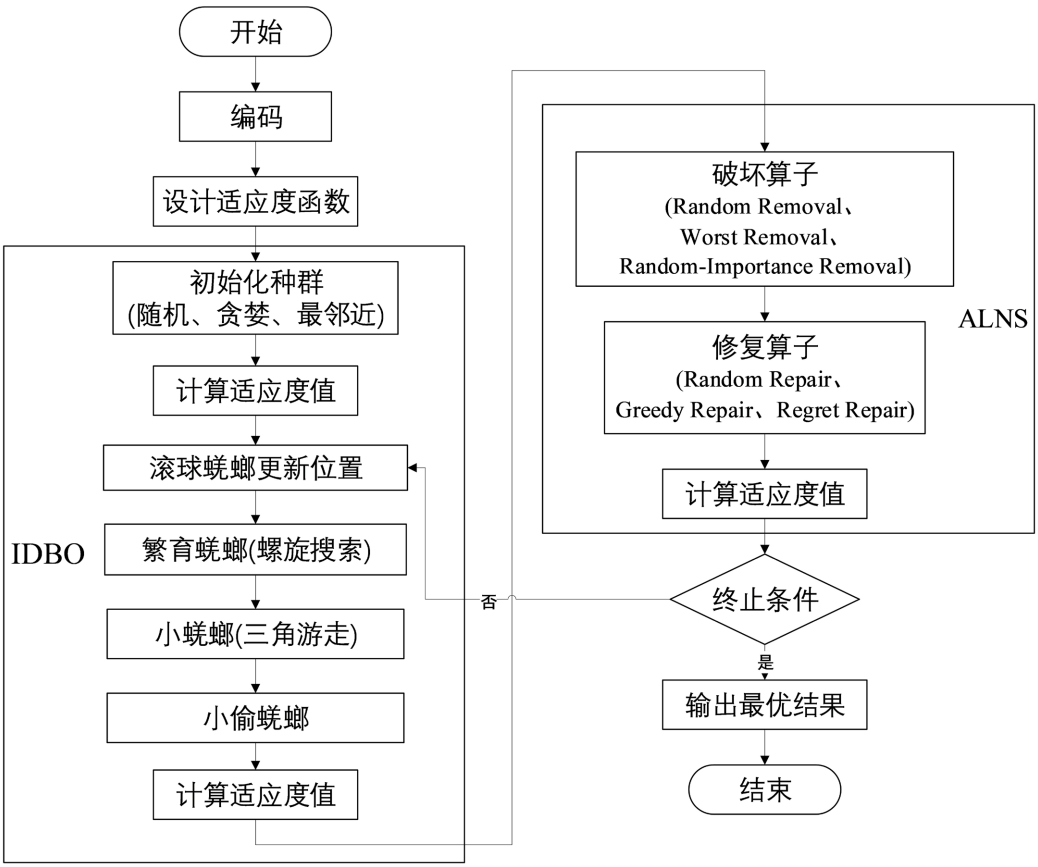

Figure 1. ALNS-IDBO flowchart
图 1. ALNS-IDBO 流程图

Table 2. Experimental results of 4 algorithms in 6 instances of 25 dimensions
表 2. 四种算法在 25 维的六个实例中的实验结果

数据集	标准		ALNS-IDBO		GA		DBO		ALNS	
	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度
C101	3	191.3	3	191.81	3	191.81	3	191.81	4	257.47
C204	2	213.1	1	213.93	1	223.93	2	219.20	1	220.36
R103	5	454.6	4	476.05	4	473.39	5	473.52	5	461.56
R203	3	391.4	2	406.24	3	452.21	2	630.98	3	415.18
RC103	3	332.8	3	333.92	3	412.32	3	340.95	3	334.39
RC203	3	326.9	1	432.82	2	453.22	3	431.23	3	330.23

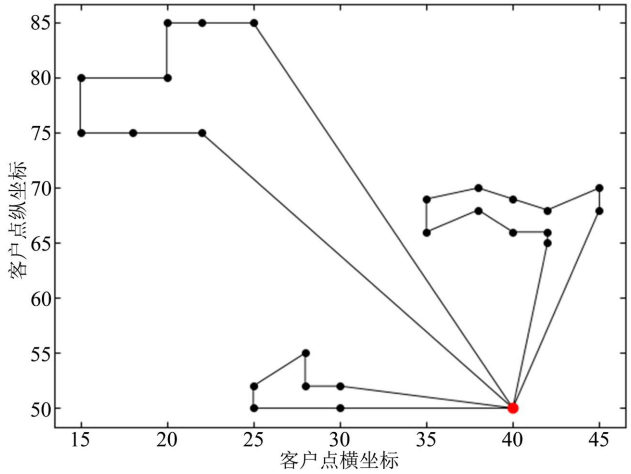

Figure 2. ALNS-IDBO serves 25 customers at C101
图 2. ALNS-IDBO 在 C101 服务 25 个客户

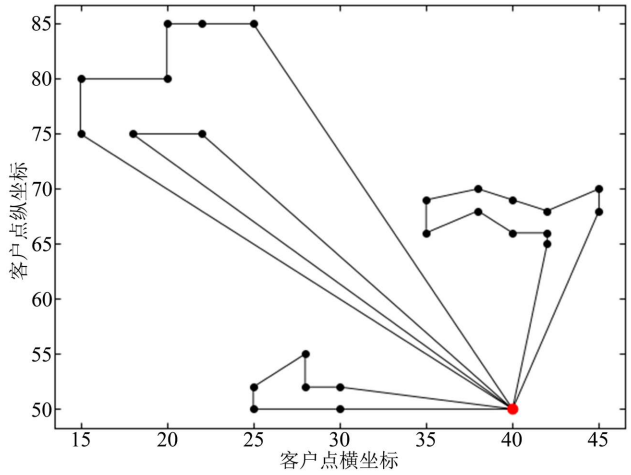

Figure 3. ALNS serves 25 customers at C101
图 3. ALNS 在 C101 服务 25 个客户

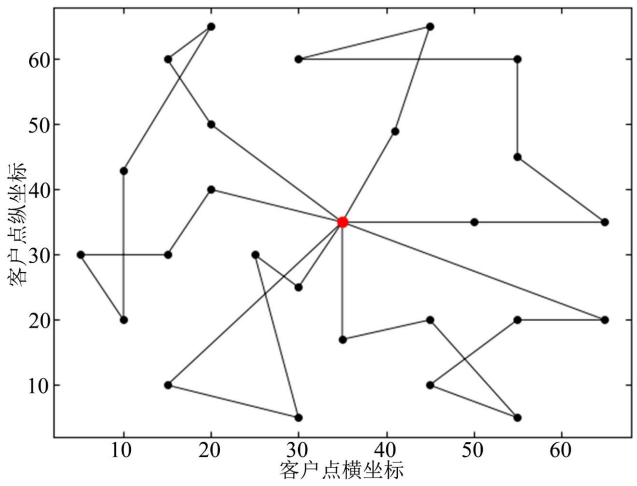

Figure 4. ALNS-IDBO serves 25 customers at R103
图 4. ALNS-IDBO 在 R103 服务 25 个客户

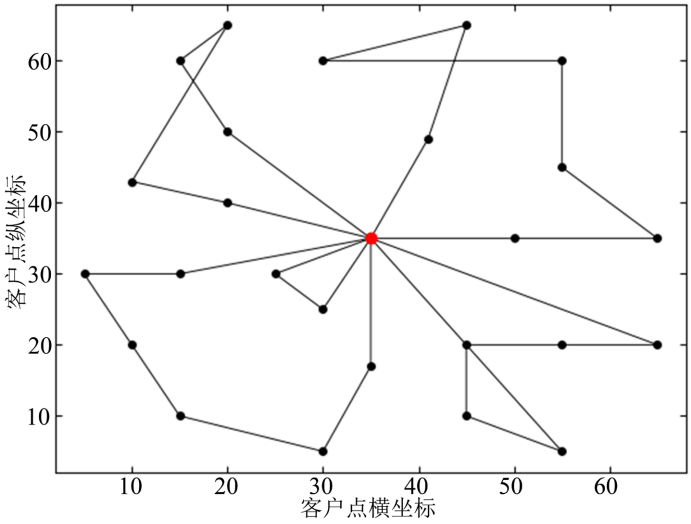

Figure 5. ALNS serves 25 customers at R103
图 5. ALNS 在 R103 服务 25 个客户

Table 3. Experimental results of 4 algorithms in 6 instances of 50 dimensions
表 3. 四种算法在 50 维的六个实例中的实验结果

数据集	标准		ALNS-IDBO		GA		DBO		ALNS	
	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度	车辆数	路径长度
C103	5	361.4	5	392.65	6	579.16	6	425.32	6	407.85
C203	3	359.8	2	403.25	4	674.60	4	582.60	4	778
R102	11	909	10	951.64	12	932.49	13	995.3	12	954.20
R107	7	711.1	6	765.34	8	782.65	9	963.28	9	798.03
RC201	5	684.8	3	848.4	7	826.52	8	1064.41	6	787.20
RC202	5	613.6	2	897.43	6	863.28	7	968.20	6	743.71

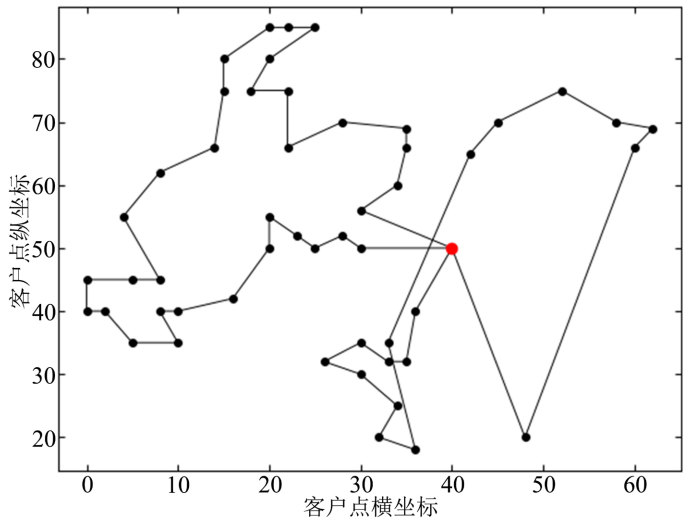

Figure 6. ALNS-IDBO serves 50 customers at C203
图 6. ALNS-IDBO 在 C203 服务 50 个客户

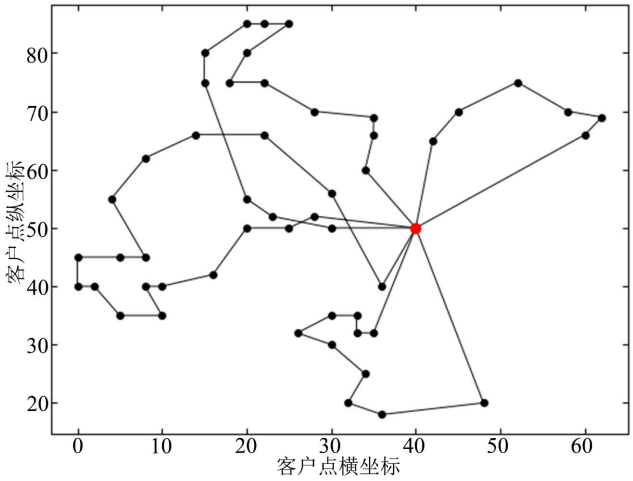

Figure 7. GA serves 50 customers at C203
图 7. GA 在 C203 服务 50 个客户

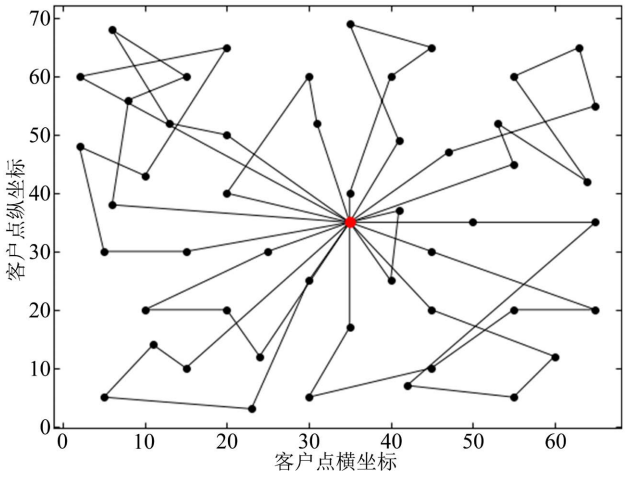

Figure 8. ALNS-IDBO serves 50 customers at R102
图 8. ALNS-IDBO 在 R102 服务 50 个客户

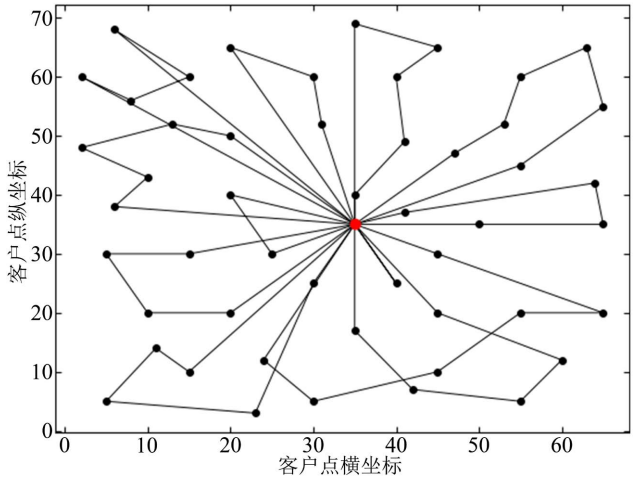

Figure 9. GA serves 50 customers at R102
图 9. GA 在 R102 服务 50 个客户

上述实验是 ALNS-IDBO 算法应用于求解带时间窗口的车辆路径问题，表 2、表 3 是在 Solomon 数据集的某些实例的实验结果，这表明 DBO 算法同其他出传统算法一样，可以应用于求解带时间窗口的车辆路径问题，并且取得的效果还是不错的，为路径规划领域提供了一种新方法。

通过表 2 和表 3 的路径长度和所用车辆数目可知，在车辆数目中优化的结果是比较明显的，面对两个衡量标准，规定中设置参数车辆数目为优先优化的目标，同时也取得了较好的效果，而传统 DBO 算法在 VRRPTW 的稳定性不高，进而改进的 DBO 算法的搜索能力变强了，平衡了全局搜索和局部搜索。可以观察表 2 中，车辆服务 25 个客户点的实例中，4 种算法都在某个实例中相比较标准解有所提升，但是 ALNS-IDBO 的改进获益者是最大的，在 6 个实例中，有 4 个都可以减少用车数量，占比 67%，同时图 2~5 是抽取其中具有代表性的车辆运行轨迹实验结果并进行直观比较，显示出该进算法的优势。在表 3 中，车辆服务 50 个客户点的实例中，相对于其他比较算法，ALNS-IDBO 的改进取得的效果是显著的，可以说明该算法对于求解高维问题存在较大的优势，并且尤其在 RC202 实例中由 5 辆车的运送，改进为 2 辆，提升明显。综上所述，本文提出的混合大规模领域搜索的改进蜣螂优化算法求解带时间窗口的车辆路径问题能找到更好的解。

5. 总结与展望

本文针对求解带时间窗车辆路径问题提出 ALNS-IDBO 算法。首先通过建立求解问题的模型，然后将两个算法相混合，前期突出 DBO 的全局搜索能力，但是对于繁育蜣螂和小蜣螂的位置更新分别改进为螺旋搜索策略和三角游走策略做到位置更新，后期为了快速收敛发挥 ALNS 的强局部搜索能力，最终得到高质量的解。为了验证算法的合理性和实用性在公认数据集 Solomon 中进行分维度的多实例的实验，实验结果对比表明：ALNS-IDBO 在大多数实例中的求解质量有所提高，尤其是较高维模型中有更好的表现。

在未来的工作中，仍需进一步提升 IDBO 的性能，同时还可以结合一些动态客户需求保证实时的效率和路径规划。

参考文献

[1] Molina, J.C., Salmeron, J.L. and Eguia, I. (2020) An ACS-Based Memetic Algorithm for the Heterogeneous Vehicle Routing Problem with Time Windows. *Expert Systems with Applications*, **157**, Article ID: 113379. https://doi.org/10.1016/j.eswa.2020.113379

[2] Belhaiza, S. (2018) A Game Theoretic Approach for the Real-Life Multiple-Criterion Vehicle Routing Problem with Multiple Time Windows. *IEEE Systems Journal*, **12**, 1251-1262. https://doi.org/10.1109/jsyst.2016.2601058

[3] Zhang, K., He, F., Zhang, Z., Lin, X. and Li, M. (2020) Multi-vehicle Routing Problems with Soft Time Windows: A Multi-Agent Reinforcement Learning Approach. *Transportation Research Part C*: *Emerging Technologies*, **121**, Article ID: 102861. https://doi.org/10.1016/j.trc.2020.102861

[4] Wang, Y., Wang, L., Peng, Z., Chen, G., Cai, Z. and Xing, L. (2019) A Multi Ant System Based Hybrid Heuristic Algorithm for Vehicle Routing Problem with Service Time Customization. Swarm and Evolutionary Computation, **50**, Article ID: 100563. https://doi.org/10.1016/j.swevo.2019.100563

[5] 曹二保, 赖明勇, 聂凯. 带时间窗的车辆路径问题的改进差分进化算法研究[J]. 系统仿真学报, 2009, 21(8): 2420-2423.

[6] Zhen, L., Ma, C., Wang, K., *et al*. (2020) Multi-Depot Multi-Trip Vehicle Routing Problem with Time Windows and Release Dates. Transportation Research Part E: Logistics and Transportation Review, **135**, Article ID: 101866. https://doi.org/10.1016/j.tre.2020.101866

[7] Jin, H. and Li, J.-Q. (2023) A Hybrid Complementary Metaheuristic for VRPTW with Product Classification and Pickup-Delivery Constraints. *Journal of Intelligent & Fuzzy Systems*, **44**, 1305-1322.

[8] 梁承姬, 黄涛, 徐德洪, 等. 改进遗传算法求解带模糊时间窗冷链配送问题[J]. 广西大学学报(自然科学), 2016, 41(3): 826-835.

[9] 雷金羡, 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. https://doi.org/10.13196/j.cims.2022.11.017

[10] 李楠, 胡蓉, 钱斌, 等. 两阶段混合优化算法求解模糊需求下多时间窗车辆路径问题[J]. 控制与决策, 2022, 37(6): 1573-1582. https://doi.org/10.13195/j.kzyjc.2021.0022

[11] Goel, R. and Maini, R. (2018) A Hybrid of Ant Colony and Firefly Algorithms (HAFA) for Solving Vehicle Routing Problems. *Journal of Computational Science*, **25**, 28-37. https://doi.org/10.1016/j.jocs.2017.12.012

[12] Zhang, W., Gen, M. and Jo, J. (2013) Hybrid Sampling Strategy-Based Multiobjective Evolutionary Algorithm for Process Planning and Scheduling Problem. *Journal of Intelligent Manufacturing*, **25**, 881-897. https://doi.org/10.1007/s10845-013-0814-2

[13] Xue, J. and Shen, B. (2022) Dung Beetle Optimizer: A New Meta-Heuristic Algorithm for Global Optimization. *The Journal of Supercomputing*, **79**, 7305-7336. https://doi.org/10.1007/s11227-022-04959-6