Advances in Applied Mathematics 应用数学进展, 2023, 12(9), 3899-3919
Published Online September 2023 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2023.129383

基于免疫克隆的蚁狮算法及其桁架优化

吴　悦

河北工程大学土木工程学院，河北　邯郸

收稿日期：2023年8月9日；录用日期：2023年9月3日；发布日期：2023年9月11日

摘　要

蚁狮优化算法具有较好的搜索和开发能力，但是在寻优后期精英蚁狮的影响权重减小，导致算法收敛速度较慢且易于陷入局优，对此提出一种免疫克隆的蚁狮优化算法。前期使用反向学习策略初始化蚂蚁种群，提高种群多样性；在精英蚁狮更新中加入柯西变异算子，提高算法后期的开发能力；最后结合免疫克隆选择算法对蚁狮进行克隆变异，改变蚁狮的位置和适应度值，进一步提高算法全局寻优能力以及收敛精度。通过10个测试函数、0~1背包来评估算法的寻优能力，并运用到桁架结构的尺寸、布局优化问题中，通过受力效果图发现优化效果良好，验证了ICALO应用于组合优化问题中收敛速度更快、精度更高的特点，为结构优化提供了一种新的方法。

关键词

桁架，组合优化，蚁狮优化算法，免疫克隆算法

Antlion Algorithm Based on Immune Cloning and Its Truss Optimization

Yue Wu

School of Civil Engineering, Hebei University of Engineering, Handan Hebei

Received: Aug. 9th, 2023; accepted: Sep. 3rd, 2023; published: Sep. 11th, 2023

Abstract

Antlion optimization algorithm has good search and development capabilities, but the influence weight of elite ant lions is reduced in the later stage of optimization, which leads to slower algorithm convergence and easy fall into local optimization. For this purpose, an antlion optimization algorithm based on immune cloning was proposed. In the early stage, the reverse learning strategy was used to initialize the ant population. The Cauchy mutation operator was added to the elite antlion update to improve the later development ability of the algorithm; finally, the antlion was

文章引用: 吴悦. 基于免疫克隆的蚁狮算法及其桁架优化[J]. 应用数学进展, 2023, 12(9): 3899-3919.
DOI: 10.12677/aam.2023.129383

cloned and mutated with the immune clone selection algorithm to change the position and fitness value of the antlion, and further improved the algorithm's global optimization ability and convergence accuracy. 10 test functions and 0~1 backpack were used to evaluate the optimization ability of the algorithm, and applied to the size and layout optimization problems of the truss structure. The optimization effect was found to be good through the force effect diagram. It is verified that ICALO is applied to combinatorial optimization problems with faster convergence speed and higher accuracy. It provides a new method for structural optimization.

Keywords

Truss, Combinatorial Optimization, Antlion Optimization Algorithm, Immune Cloning Algorithm

Copyright © 2023 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

群智能优化算法是通过模拟自然界中的一些动物或植物的行为规律，在解空间中进行全局寻优的一种方法。近几年来，学者们研究了一系列的新智能优化算法，如：蚁群算法(Ant Clony Optimization, ACO)，狮群优化算法(Loin Swarm Optimization, LSO)，灰狼优化算法(Grey Wolf Optimizer, GWO)，鲸鱼优化算法(Whale Optimization Algorithm, WOA)等等，这些算法鲁棒性[1]较强，有较广适用范围[2]并且操作简单，被广泛运用于各种领域[3]。其中受蚁狮捕捉蚂蚁的过程启发，2015 年澳大利亚作者 Seyedali 提出一种群智能算法即蚁狮优化算法(Ant Lion Optimizer Algorithm, ALO)，目的是为了解决函数优化问题[4]。虽然 ALO 算法提出时间较短，但因其具有全局优化，收敛精度高，鲁棒性较好的优点，与新型的元启发式算法鲸鱼算法(Whale Optimization Algorithm, WOA)、正余弦算法(Sine Cosine Algorithm, SCA)以及樽海鞘算法(Slap Swarm Algorithm, SSA)具有一定的可比性。目前，蚁狮优化算法已被应用到改进无人机[5]，SVM (Supported Vector Machine) [6]、Elman 神经网络以及并行机调度[7]求解等方向，并取得了较好的成果。

但是蚁狮优化算法后期精英蚁狮影响因子较小，易受较差蚁狮影响，造成算法易陷入局部最优，收敛速度较慢求解精度较低等问题[7]，近年来很多科研工作者对蚁狮优化算法进行了改进，为了提高算法的全局搜索能力和寻优精度，李宗妮等[8]通过蚁狮的位置改变蚂蚁的搜索边界，使其与精英蚁狮更好地搜索有效区域；吴伟民等[9]通过自适应反馈调整蚁狮收缩的范围使算法收敛精度得到提高；赵小国等[10]通过蚂蚁的动态搜索和范围收缩改进蚁狮优化算法，使算法的收敛速度得到提高。为了增加种群多样性：王亚东等[11]将差分进化思想引入蚁狮优化算法，通过反向学习策略[12]改进了蚂蚁的更新位置。为了解决蚁狮优化算法的早熟和局部最优问题：Haydar Kilic 等[7]使用锦标赛选择策略改变 ALO 算法中蚂蚁和蚁狮的更新公式的同时自适应改变其步长，使之能够较好地平衡全局搜索能力及局部搜索能力；Alaa Tharwat 等[13]提出了基于混沌映射和罚函数的蚁狮优化算法，减少蚂蚁搜索范围，加快算法寻优速度。

以上对蚁狮优化算法的改进，虽然在一定范围内上减少了算法落入局部最优的可能性。但是改进后的算法仍然存在收敛精度较低，开发能力较差等缺点，仍需进一步研究，鉴于此，考虑到柯西变异能够对当前较差蚁狮进行扰动[14]，防止算法陷入局优，将反向学习策略应用到蚂蚁种群中，丰富种群多样性，最后将具有较强探索能力的 ALO 算法和较好的开发能力的 ICS 算法相结合[15] [16]，提出一种基于免疫克隆的蚁狮优化算法，通过对蚁狮进行克隆、变异、选择等操作有效的平衡算法的探索与开发能力。通

过 10 个基准测试函数[17]仿真实验和组合优化实验[18]，体现出改进后蚁狮优化算法相比其它算法有更强的收敛性和寻优性能，并且延展算法在实际工程中的应用领域。

2. 基本的蚁狮优化算法

具体运算步骤如下[19] [20]：

1) 首先将蚂蚁的位置参数初始化，自定义维数为 D，迭代次数为 T，同时随机初始化 N 个蚂蚁位置的解

2) 通过轮盘赌策略选取一个蚁狮，并安排一个蚂蚁在蚁狮周围，同时记录其位置

$$X(t)=\left[0,cumsum\left(2p\left(t_1\right)-1,cumsum\left(2p\left(t_2\right)-1,\cdots,cumsum\left(2p\left(t_{n-1}\right)-1\right]\right.\right.\right. \tag{1}$$

其中，$cumsum$ 是计算存在的蚂蚁移动位置和，n 是最大迭代次数，t 表示随机行走的步数(在本研究中指是迭代数)，$p(t)$ 是一个随机函数。

3) 对公式进行归一化处理

$$X_i^t=\left(X_i^t-a_i\right)\left(d_i^t-c_i^t\right)/\left(b_i-a_i\right)+c_i^t \tag{2}$$

式中 X_i^t 表示蚂蚁在第 i 维第 t 次迭代时的位置 a_i,b_i 分别为蚂蚁步长的上下界，c_i^t,d_i^t 为蚂蚁游走的上下界，为了使迭代的蚂蚁在指定范围内移动，运用 I 对其约束。

$$I=10^q t/T \tag{3}$$

其中 I 是一个比率，表示在第 t 次迭代中存在变量的最小值向量，T 指最大迭代次数，q 与迭代次数相关，同时也定义了在第 t 次迭代中存在变量的最大值向量。

4) 最后一个过程是假设当蚂蚁比其对应的蚁狮适应度更好(进入沙子内)时，就会存在一只对应的蚁狮将当前蚂蚁的位置占据，使蚁狮能够捕捉到迭代后适应度更好的蚂蚁。

5) 由于精英蚁狮是适应度值最优的蚁狮，因此假定每个蚂蚁都通过轮盘赌轮和精英策略同时随机地绕着选定的蚁狮行走，如下所示：

$$Ant_i^t=\left(R_A^t+R_E^t\right)/2 \tag{4}$$

其中 R_A^t 表示第 t 次迭代时蚂蚁绕轮盘赌选择的蚁狮周围的随机游动，R_E^t 是在第 t 次迭代中绕着精英蚁狮的随机游走。

6) 蚂蚁位置更新后计算对应目标函数的适应度并与精英蚁狮进行比较，选择较好的适应度值为全局最优解，如果蚂蚁适应度大于蚁狮的值，则更新蚁狮位置。

7) 当算法达到最大的迭代次数时使则停止搜索，并输出最优解，否则循环上述过程。

3. 基于免疫克隆的蚁狮优化算法

3.1. 反向学习策略

ALO 初始种群是随机生成，由于蚂蚁分布不均匀，导致种群多样性减少，导致算法收敛速度下降，受 Tizhoosh [21] (2005)提出的反向学习策略启发，首先随机生成初始种群，然后通过初始种群生成反向种群，从中选取较优的种群作为下一代种群。反向学习策略能够选择更靠近个体的种群作为最初个体，由于每个个体距离最优解的位置更近，有利于提高算法收敛速度，并且由于反向学习策略能够对有效区域进行搜索，能够提高种群的多样性，增强算法全局搜索能力。

首先对蚂蚁种群的反向种群进行初始化定义 $X_i^*=\left[x_{i1}^*,\cdots,x_{id}^*,\cdots,x_{iD}^*\right]$

$$X_i^* = rand \cdot \left(c_i^t + d_i^t\right) - X_i \tag{5}$$

$$M = N^2 / N + 1 \tag{6}$$

将随机生成的蚂蚁种群和其反向种群合并为一个新的种群，求新种群的适应度函数，并将适应度值按升序排列，取前 M 个最优初始解作为新的蚂蚁初始种群。生成的蚂蚁反向种群和随机生成的种群重组为一个新的种群，重新计算新种群的适应度值，并按照适应度值进行排列，取排列的前 M 项作为蚂蚁的最初种群。

3.2. 基于柯西变异的精英蚁狮扰动机制

判断一个算法的计算效率和准确性主要权衡其开发能力和搜索能力，根据蚂蚁更新公式可知，其受蚁狮和精英蚁狮的影响概率是相同的，因此在后期寻优能力较差，易受较差蚁狮的影响，过早陷入局优，因此，本文在精英蚁狮的更新时引入柯西变异算子[22]。柯西变异能够在精英蚁狮周围区域进行搜索，通过在最优解周围随机扰动，增强蚂蚁局部寻优的能力，柯西概率密度如式 7 所示。

$$g\left(x; x_0, \gamma\right) = \frac{1}{\pi\gamma\left[1 + \left(\frac{x - x_0}{\gamma}\right)^2\right]} \tag{7}$$

其中，x_0 是位置参数，γ 是一个大于 0 的随机变量，γ 是一个实数。本文中取 $x_0 = 0$，$\gamma = 1$，其为标准柯西分布。通过分析其概率密度函数，可知其没有特定的均值和方差，但众数和中值都等于位置参数，即 x_0 其分布函数如式(8)所示：

$$G\left(x\right) = \frac{1}{2} + \frac{1}{\pi}\arctan\left(x\right) \tag{8}$$

柯西分布与正态分布和高斯分布相比，其具有更均匀的整体分布；极值更高并且尾部的概率较大，有较大的散布特性，能够更好的对极值点周围进行搜索对精英蚁狮的扰动公式如下所示：

$$\boldsymbol{X}_{ibset}\left(t\right) = \boldsymbol{X}_i\left(t\right) + \boldsymbol{X}_i\left(t\right) * G\left(x\right) \tag{9}$$

$$\boldsymbol{X}_i\left(t+1\right) = \begin{cases} \boldsymbol{X}_{ibest}\left(t\right), f\left(\boldsymbol{X}_{ibest}\left(t\right)\right) > f\left(\boldsymbol{X}_i\left(t\right)\right) \\ \boldsymbol{X}_i\left(t\right), f\left(\boldsymbol{X}_{ibest}\left(t\right)\right) \le f\left(\boldsymbol{X}_i\left(t\right)\right) \end{cases} \tag{10}$$

其中，$f\left(\boldsymbol{X}_i\left(t\right)\right)$ 表示精英蚁狮在第 t 次迭代时的适应度值。通过局部扰动，引导算法跳出局部最优。

精英蚁狮的个体迁移有利于算法的加速收敛，但是如果精英蚁狮的更换频率较高则会使最优蚂蚁群体迁移至另一蚂蚁种群，在之后的迭代过程中，会出现较优的精英蚁狮捕获所有的蚂蚁种群，使算法陷入局优，因此设计新的策略在收敛性和多样性之间实现平衡是下一步改进的重点。

3.3. 算法融合策略

3.3.1. 免疫克隆选择算法(Immune Clone Algorithm, ICA)

免疫克隆选择算法[23] (Immune Clone Algorithm, ICA)：首先将初始的抗体群进行克隆并从中选择优秀的抗体，通过免疫基因操作提高优秀抗体的多样性，同时使用抗体修正操作改变较差的抗体，提高算法局部寻优能力。如果把优化中后期较差蚁狮视为抗体，把蚁狮的候选解视作抗体，那么生物免疫系统的免疫识别和免疫相应过程就可以与蚁狮优化算法联系起来，通过克隆操作来提高蚁狮优化算法中种群的多样性，在使用免疫操作改变较差蚁狮提高算法的收敛性，见图 1 所示。

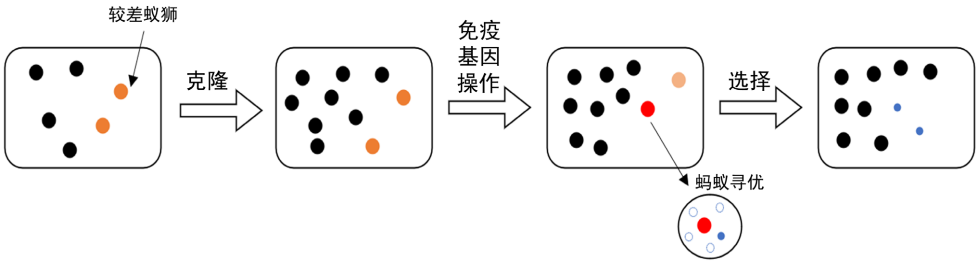


Figure 1. The search process of the antlion optimization algorithm based on immune cloning
图 1. 基于免疫克隆的蚁狮优化算法的搜索过程

3.3.2. ICALO 的执行

该算法以基本的蚁狮优化算法为框架，以进化停滞步数 $n > m$ (进化停滞步数阈值)，作为调用 ICA 算法的条件，当停滞步数大于进化停滞步数阈值时，将克隆、免疫、选择等操作引入蚁狮的更新中，通过不断调用 ICA 算法使蚁狮变异，更新蚂蚁的搜索空间，扩大蚂蚁的搜索范围，最终获得可行解集。具体操作如下所示：

1) 克隆操作：通过锦标赛选择策略，选择出适应度较高的蚁狮，保证其蚂蚁的搜索空间能围绕较优的位置，对选出的较优蚁狮按照一定的比例进行复制，增加局部寻优的多样性以及扩大蚂蚁的寻优空间。锦标赛选择策略的实现过程为，首先设置队伍规模为 B1，蚁狮群提供的选择规模为 B2，通过在 B2 中随机选择 B1 个蚁狮进行比较，保留非支配等级较高且拥挤度距离大的蚁狮进行下一轮选择，重复选择 B2 次，最终选出的 B2 个蚁狮，即为克隆选择的抗体群。

克隆得到的蚁狮如下：

$$\boldsymbol{A}^{(1)}(t)=\left\{\boldsymbol{A}_1^{(1)}(t),\boldsymbol{A}_2^{(1)}(t),\cdots,\boldsymbol{A}_{B_2}^{(1)}(t)\right\} \tag{11}$$

克隆操作后存在的蚁狮抗体群如式(12)所示：

$$\begin{aligned}\boldsymbol{A}^{(2)}(t)&=\boldsymbol{A}^{(1)1}(t)+\boldsymbol{A}^{(1)2}(t)+\cdots+\boldsymbol{A}^{(1)m_c}(t)\\&=\left\{\boldsymbol{A}_1^{(1)1}(t),\cdots,\boldsymbol{A}_{B_2}^{(1)1}(t)\right\}+\left\{\boldsymbol{A}_1^{(1)2}(t),\cdots,A_{B_2}^{(1)2}(t)\right\}+\cdots+\left\{\boldsymbol{A}_1^{(1)m_c}(t),\cdots,\boldsymbol{A}_{B_2}^{(1)m_c}(t)\right\}\\&=\left\{\boldsymbol{A}_1^{(2)}(t),\boldsymbol{A}_2^{(2)}(t),\cdots,\boldsymbol{A}_{n(B_2)}^{(2)}(t)\right\}\end{aligned} \tag{12}$$

其中，蚁狮数量为 $N(a_1)=A_1\times k$，A_1 为克隆选择时所保留的蚁狮数量，k 为克隆的比例，经大量测试证明 $k=3$ 效果最优。

2) 自适应免疫基因变异操作

免疫操作是指将基因进行重组和变异，能够较好的提高算法局部微调能力和搜索精度，因此本文在蚁狮位置中加入自适应扰动免疫变异算子，计算公式如下：

$$\begin{cases}\mathrm{AL}_i'=\left(\mathrm{AL}_i-\dfrac{\mathrm{AL}_i-\mathrm{AL}_{\min}}{\mathrm{AL}_{\max}-\mathrm{AL}_{\min}}\cdot r(t)\right)\times(1+\lambda),rand\geq 0.5;\\ \mathrm{AL}_i'=\left(\mathrm{AL}_i+\dfrac{\mathrm{AL}_i-\mathrm{AL}_{\min}}{\mathrm{AL}_{\max}-\mathrm{AL}_{\min}}\cdot r(t)\right)\times(1+N(0,1)),rand<0.5;\end{cases} \tag{13}$$

$$r(t)=1-\frac{\mathrm{e}^{2(1-t/T)}-\mathrm{e}^{-2(1-t/T)}}{\mathrm{e}^{2(1-t/T)}+\mathrm{e}^{-2(1-t/T)}}rand^{(1-t/t_{\max})^3} \tag{14}$$

其中：λ 为[−1, 1]的随机变量，$N(0, 1)$为服从标准正态分布的柯西变量，t 为当前循环次数，$\mathrm{AL}_{\max}$ 和 $\mathrm{AL}_{\min}$ 为当前最优蚁狮和最差蚁狮的位置。

3) 编辑选择

通过克隆和变异后，较优蚁狮保持当前位置，较差蚁狮对应的蚂蚁继续寻优，通过局部择优实现较差蚁狮的压缩，同时，结合免疫网络调剂机制[24]，进化停滞步数 $n > 3$ 时对较差蚁狮进行调整，经计算选择如式：

$$a_c\left(\mathrm{AL}_i\right)=\left(\sum_{j=1}^{n} fitness\left(\mathrm{AL}_j\right)\right)\Big/ fitness\left(\mathrm{AL}_j\right) \tag{15}$$

其中：$fitness\left(\mathrm{AL}_j\right)$ 为精英蜂群中的蜜源亲和度值，且 $a_c^{\min} \le a_c \mathrm{AL}_i \le a_c^{\max}$ 该策略能够使较优的蚁狮在与其他蚁狮的竞争中被多次选择克隆，从而提高算法的全局收敛速度。

每次对较优蚁狮进行克隆选择操作后，结合免疫系统记忆机制，将精英蚁狮的位置以及适应度共享至蚂蚁和蚁狮群中，当满足下式时，对精英蚁狮的位置进行更新

$$\left| fitness\left(\mathrm{AL}\right) - fitness\left(\text{elite-AL}\right)\right| \le \varepsilon \tag{16}$$

免疫克隆算法的收敛速度较快、目标函数多样性较高度进化算法，能够为精英蚁狮提供较多的选择。

3.3.3. 融合免疫克隆的蚁狮优化算法(Antlion Optimization Algorithm Fused with Immune Cloning, ICALO)

对于基本的 ALO 在前期使用反向学习策略初始化种群，作为算法寻优的基础，通过加入柯西变异提高了算法的搜索性能和开拓性能，最后融合克隆免疫算法使算法避免陷入局部最优，提高算法的精度，算法流程图见图 2 所示：

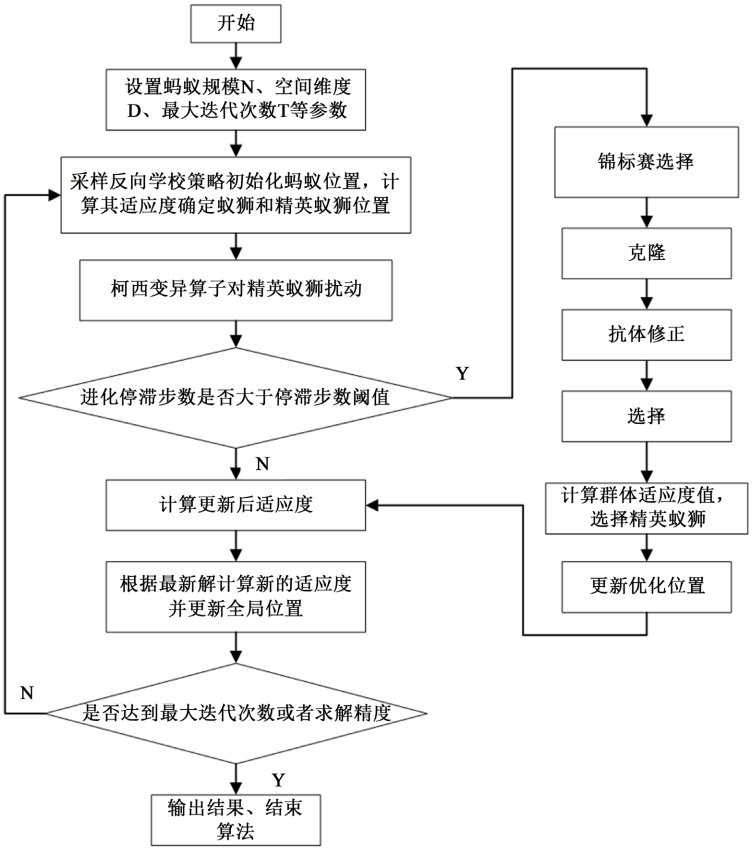


Figure 2. Flowchart of ICALO algorithm
图 2. ICALO 算法流程图

4. 仿真实验

4.1. 测试函数

为验证改进蚁狮优化算法的有效性，选取 10 个标准测试函数以及经典 NP 问题来对改进算法的性能进行数值测试，将计算结果与基本的 ALO；Rajan 等提出基于加权精英策略的蚁狮优化算法(Weighted Elitism Based ALO, WALO)；黄长强等提出反调节混沌蚁狮优化算法(Anti Adjust Chaos Based ALO, HALO)；ICA 进行对比。实验环境为 Win10 系统，8GB 内存，Matlab2020a 软件。参数设置均为：初始化种群数量 500，最大迭代次数为 500 次。为避免偶然性，每种算法在 Matlab2020a 软件中独立运行 30 次。

表 1 为各测试函数的基本参数，分别选取单峰[25]和多峰[26]来评价算法收敛速度[27]和跳出局部最优的能力。式中 F_1 函数和 F_2 函数为单峰值函数，主要用来测试文中改进算法在运算过程中的收敛性能；F_3~F_{10} 函数均为多峰值函数，其中 F_8 为复杂的多峰函数，全局最小值位于非常狭窄的山谷中，F_9 为移动、旋转、不可分、可拓展的多峰函数，F_{10} 的多峰函数中有无数个极小值点，函数具有强烈震荡形态，很难找到全局最优值。上述函数在寻优特征上各具特性，不仅可以测试改进的算法在处理“早熟”问题上的能力，也适合用来全面测试改进算法的计算精度、收敛稳定性、收敛速度。

Table 1. Test functions
表 1. 测试函数

函数名称	特征	维度	区间	最优值
$F_1(x)=\sum_{i=1}^{n}x_i^2$	单峰	100	[−100, 100]	0
$F_2(x)=\sum_{i=1}^{n}\left[100\left(x_{(i+1)}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	单峰	100	[−10, 10]	0
$F_3(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos\left(2\pi x_i\right)+10\right]$	多峰	100	[−32, 32]	0
$F_4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left\{\frac{1}{n}\sum_{i=1}^{n}\cos\left(2\pi x_i\right)\right\}+20+e$	多峰	100	[6, 6]	0
$F_5(x)=\frac{1}{4000}\sum_{i=1}^{n}X_i^2-\prod_{i=1}^{n}\cos\left(x_i/\sqrt{i}\right)+1$	多峰	100	[−600, 600]	0
$F_6(x)=\sum_{i=1}^{n}X_i^2+\left(\sum_{i=1}^{n}0.5iX_i\right)^2+\left(\sum_{i=1}^{n}0.5iX_i\right)^4$	多峰	100	[−100, 100]	0
$F_7(x)=\sum_{i=1}^{n}\left(\left(x_{4i-2}^2-10x_{4i-2}\right)^2+5\left(x_{(4i-1)}-x_4 i\right)^2+\left(x_{(4i-2)}-2x_{(4i-1)}\right)^4+10\left(x_{(4i-3)}-2x_4 i\right)^4\right)$	多峰	100	[−1000, 1000]	0
$F_8(x)=\sum_{i=1}^{D-1}\left(100\left(x_i^2-x_{i+1}\right)^2+\left(x_i-1\right)^2\right)$	多峰	100	[−100, 100]	0
$F_9(x)=\sum_{i=1}^{D}\left(10^6\right)^{\frac{i-1}{D-1}x_i^2}$	多峰	100	[−1300, 1300]	0
$F_{10}(x)=g\left(x_1,x_2\right)+g\left(x_2,x_3\right)+\cdots+g\left(x_{D-1},x_D\right)+g\left(x_D,x_1\right)$	多峰	100	[−600, 600]	0

10 个函数在 100 维的实验结果见表 2，图 3 为各算法在 10 种函数下的迭代曲线，可分别对 ICA、ALO、HALO、WALO 和 ICALO 具体分析：

ALO 在单峰函数求解结果相比其他的算法均较差，但是在 F_9、F_{10} 中求得的最优值和 F_7、F_8 中求得的最差值与平均值相比 ICA 较优。

WALO 在单峰函数中的最优解、最差解、平均值均优于对 ALO、HALO、ICA。在多峰函数 F_4~F_6 和 F_{10} 中，最优值和平均值相比其他算法更优，但是 F_4，F_7 中最差值均比 HALO 差，可以看出 WALO 有陷入局优的可能。在函数收敛图中可以看出 WALO 有效的提高了算法的寻优速度，相比基本的 ALO 和 ICA 寻优的迭代次数减少一半。

HALO 在 F_1~F_6、F_9、F_{10} 的最优值、最差值、平均值均优于 ALO、ICA。在 F_3、F_5、F_6~F_{10} 迭代图中也可看出收敛速度较慢。

ICALO 在 F_1、F_3、F_5~F_{10} 函数的最优值、最差值、平均值相比 ALO、HALO、WALO、ICA 有更好的精度，且可以看出受维度影响较小，在高维函数中也能体现出求解的结果远优于其它算法，并且可以在迭代图中看出算法的收敛速度较快。

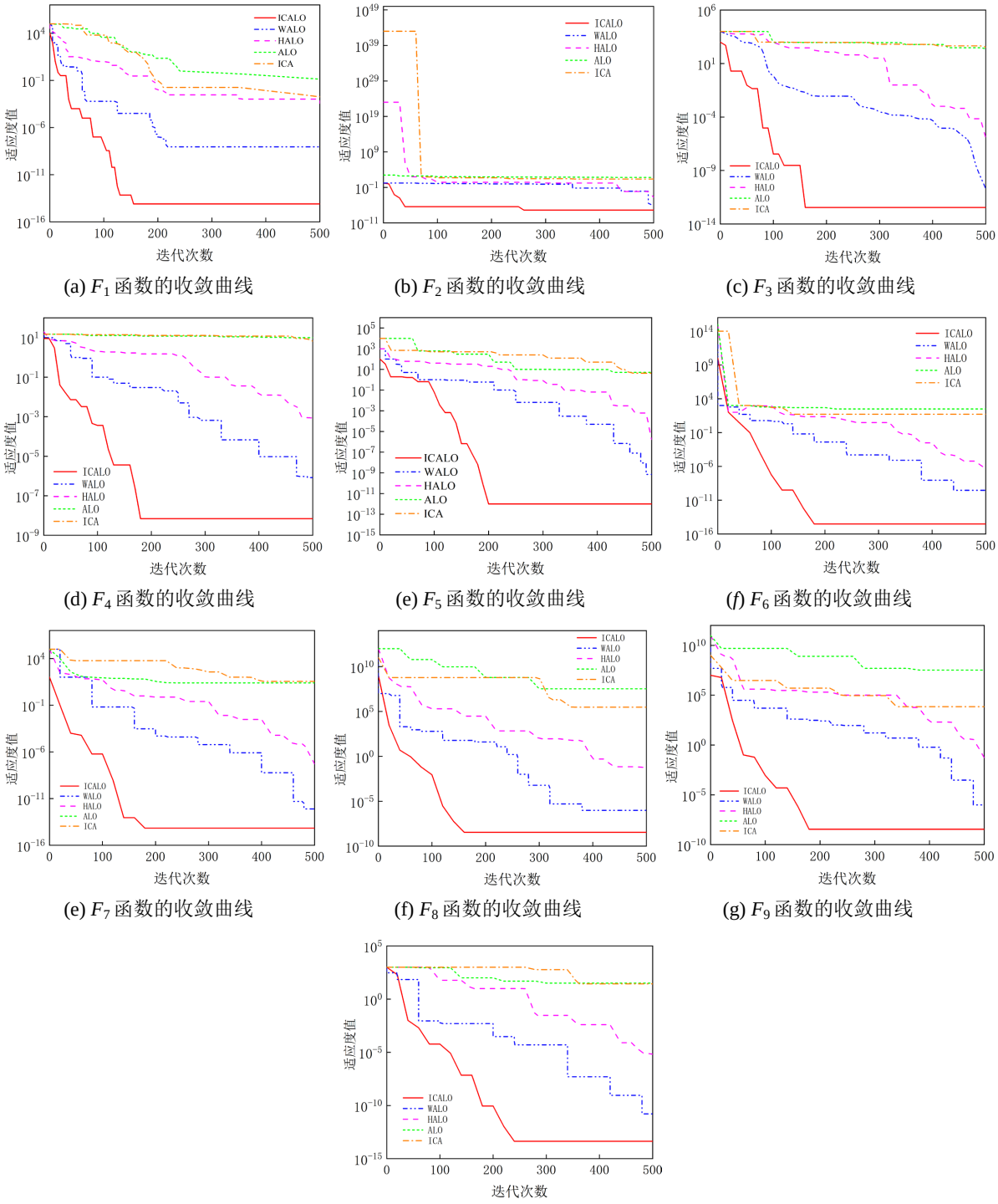
从图像和数据可以得知，不论是算法对函数的收敛性能，运行效率，对于“局优值”的处理问题还是求得的解的质量，ICALO 都明显优于其他算法，并且在可以看出求解高维函数时有较好的稳定性，这主要是因为在蚁狮的位置中加入了动态权重因子，有效的缓解了轮盘赌策略减少种群多样性的问题，收敛速度较快也可以体现出免疫克隆的引入很大的提高了蚁狮优化算法跳出局优性能，充分说明 ICALO 求解高维问题的可行和有效性。

Table 2. Comparison of standard function 100-dimensional optimization performance
表 2. 标准函数 100 维优化性能对比

函数	维数	算法	最优值	最差值	平均值	均方差
F_1	100	ALO	0.137 39	3.8784	0.927	2.4624
		WALO	1.09×10^{-12}	1.31×10^{-9}	2.63×10^{-10}	6.28×10^{-20}
		HALO	3.71×10^{-7}	1.32×10^{-6}	8.59×10^{-7}	4.17×10^{-14}
		ICA	0.00183	1.0422	7.92×10^{-2}	3.68×10^{-2}
		ICALO	6.58×10^{-15}	1.29×10^{-10}	7.26×10^{-11}	2.29×10^{-25}
F_2	100	ALO	64.456	$4.54 \times 10^{+29}$	$1.51 \times 10^{+28}$	$6.88 \times 10^{+57}$
		WALO	1.60×10^{-6}	3.03×10^{-5}	9.01×10^{-2}	4.18×10^{-11}
		HALO	0.000 19	0.000 74	2.90×10^{-4}	1.03×10^{-8}
		ICA	24.125	44.733	36.62	63.615
		ICALO	4.09×10^{-8}	6.84×10^{-5}	5.96×10^{-7}	6.84×10^{-15}
F_3	100	ALO	252.13	525.33	348.0637	$4.64 \times 10^{+3}$
		WALO	1.99×10^{-11}	1.57×10^{-9}	3.49×10^{-10}	1.92×10^{-19}
		HALO	1.20×10^{-6}	369.05	$1.55 \times 10^{+2}$	$2.17 \times 10^{+4}$
		ICA	131.47	264.94	187.9973	$1.35 \times 10^{+3}$
		ICALO	$0.00 \times 10^{+00}$	3.51×10^{-13}	1.89×10^{-15}	4.18×10^{-19}

Continued

F_4	100	ALO	10.035	16.018	$1.27 \times 10^{+1}$	23.0
		WALO	8.33×10^{-7}	5.63×10^{-4}	2.79×10^{-5}	1.67×10^{-12}
		HALO	8.88×10^{-6}	0.00031	2.30×10^{-4}	5.01×10^{-9}
		ICA	6.6864	10.179	8.59	23.15
		ICALO	6.94×10^{-9}	6.32×10^{-7}	9.72×10^{-7}	6.58×10^{-16}
F_5	100	ALO	5.245 5	31.378	17.733 93	52.62
		WALO	7.82×10^{-12}	1.49×10^{-9}	3.32×10^{-10}	1.54×10^{-19}
		HALO	1.69×10^{-6}	0.275 6	1.79×10^{-2}	4.63×10^{-3}
		ICA	4.314 4	12.694	7.713 64	20.061
		ICALO	9.84×10^{-14}	6.18×10^{-10}	4.94×10^{-13}	9.58×10^{-24}
F_6	100	ALO	307.5	1 128.2	701.763	$4.04 \times 10^{+4}$
		WALO	2.22×10^{-13}	4.03×10^{-10}	1.23×10^{-10}	1.51×10^{-20}
		HALO	4.48×10^{-7}	1.27×10^{-6}	8.25×10^{-7}	4.49×10^{-14}
		ICA	51.612	492.81	158.068 8	$1.85 \times 10^{+4}$
		ICALO	3.04×10^{-15}	5.18×10^{-11}	9.48×10^{-12}	9.14×10^{-24}
F_7	100	ALO	24.856	94.647	47.35	$3.02 \times 10^{+2}$
		WALO	7.92×10^{-9}	1.24×10^{-6}	3.40×10^{-8}	1.26×10^{-21}
		HALO	5.22×10^{-8}	1.24×10^{-7}	8.33×10^{-8}	3.92×10^{-16}
		ICA	35.394	103.38	64.35	$1.89 \times 10^{+3}$
		ICALO	6.95×10^{-15}	6.94×10^{-12}	9.59×10^{-13}	5.21×10^{-28}
F_8	100	ALO	$3.99 \times 10^{+6}$	$6.87 \times 10^{+7}$	$2.25 \times 10^{+7}$	$2.72 \times 10^{+14}$
		WALO	5.96×10^{-9}	4.72×10^{-7}	1.18×10^{-7}	1.51×10^{-14}
		HALO	105.95	123.91	$1.13 \times 10^{+2}$	153.9
		ICA	$1.75 \times 10^{+6}$	$2.27 \times 10^{+7}$	$7.36 \times 10^{+6}$	$2.07 \times 10^{+13}$
		ICALO	9.58×10^{-13}	5.45×10^{-10}	4.43×10^{-10}	9.84×10^{-21}
F_9	100	ALO	$3.35 \times 10^{+7}$	$1.15 \times 10^{+8}$	$7.50 \times 10^{+7}$	$5.17 \times 10^{+14}$
		WALO	9.59×10^{-7}	9.79×10^{-5}	3.31×10^{-5}	8.78×10^{-10}
		HALO	5.47×10^{-2}	1.54×10^{-1}	9.79×10^{-2}	6.77×10^{-4}
		ICA	4.28×10^{-2}	1.34×10^{-1}	8.95×10^{-2}	4.80×10^{-4}
		ICALO	3.48×10^{-9}	5.78×10^{-8}	1.64×10^{-8}	9.45×10^{-16}
F_{10}	100	ALO	31.65	46.62	41.645	59.462
		WALO	1.65×10^{-11}	4.30×10^{-9}	9.21×10^{-10}	1.12×10^{-18}
		HALO	6.80×10^{-4}	40.651	25.352	130.926
		ICA	27.611	37.62	32.641	47.561
		ICALO	4.29×10^{-14}	9.59×10^{-11}	2.29×10^{-12}	3.23×10^{-24}



(a) F_1 函数的收敛曲线 (b) F_2 函数的收敛曲线 (c) F_3 函数的收敛曲线

(d) F_4 函数的收敛曲线 (e) F_5 函数的收敛曲线 (f) F_6 函数的收敛曲线

(e) F_7 函数的收敛曲线 (f) F_8 函数的收敛曲线 (g) F_9 函数的收敛曲线

(h) F_{10} 函数的收敛曲线

Figure 3. Iteration curve of the algorithm under 10 kinds of functions
图 3. 算法在 10 种函数下的迭代曲线

4.2. 0~1 背包问题

在组合优化领域，0~1 背包问题是较为经典的 NP 问题，国内外学者分别采用模拟退火算法[28]、粒

子群算法[29]、遗传算法[30]等算法对其进行分析研究。本节通过运用 ICALO 算法对其进行优化，并与 ALO、WALO、HALO 进行对比，验证 ICALO 算法在求解组合问题中的可行性。实验环境为 Win10 系统，8GB 内存，Matlab2020a 软件。初始种群数为 500，最大迭代次数为 1000，其他参数设置与上一节相同，每组数据独立运行 10 次取最优解。

0~1 背包问题数学模型如下：

设一共有 D 个物品，其中第 i 个物品的的体积为 w_i 对应的价格为 p_i，背包容量为 C，x_i 作，变量设为 0 或 1，若物品可以装入背包为 1，否则为 0。

$$\max f = RX = \sum_{i=1}^{D} R_i x_i \tag{17}$$

$$\text{s.t. } WX = \sum_{i=1}^{D} w_i x_i \le C \tag{18}$$

式中：$R = (R_1, R_2, \cdots, R_D)$ 代表物体的价值向量，$W = (w_1, w_1, \cdots, w_D)$ 代表物体的体检向量，$X = (x_1, x_2, \cdots, x_D)$ 代表解向量：

算法的参数设置表 3 所示：

Table 3. Reference table for basic parameters of 0~1 problem
表 3. 0~1 问题基本参数参照表

参数	参数值
物品数量	$D = 100$
背包容量	$C = 1173$
物品价值	$R = [199, 194, 193, 191, 189, 178, 174, 169, 164, 164, 161, 158, 157, 154, 152, 152, 149, 142, 131, 125, 124, 124, 124, 122, 119, 116, 114, 113, 111, 110, 109, 100, \cdots, 97, 94, 91, 82, 82, 81, 80, 80, 80, 79, 74, 72, 71, 70, 69, 68, 65, 65, 61, 56, 55, 54, 53, 47, 47, 46, 41, 36, 34, 32, 32, 30, 29, 77, 76, 29, 26, 25, 23, 22, 20, 11, 10, 9, 5, 4, 3, 1]$
物品重量	$W = [40, 27.5, 21, 51, 16.42, 18, 52, 28, 57, 34, 44, 43, 52, 55, 53, 42, 47, 56.57, 44, 16, 2, 12, 9, 40, 23, 56, 3, 39, 16, 54, 36, 52, 5, 53, 48.23, 47, 41, 49, \cdots, 22, 42, 10, 16, 53, 58, 40, 1, 43, 56, 40, 32, 44, 35, 37, 45, 52, 56, 40, 2, 23, 49, 50, 26, 11, 35, 32, 34, 58, 6, 52, 26, 31, 23, 4, 52, 53, 19]$

将 ICALO 与基本的 ALO、WALO 与 HALO 对 0~1 背包问题进行优化对比，得到测试结果如表 4 所示。

Table 4. Comparison table of results of 0~1 knapsack problem
表 4. 0~1 背包问题结果对照表

算法	最优值	最差值	平均值	方差
ALO	4327	3932	4193.5	8865.426
ICALO	5187	5020	5146.6	4647.0621
ICA	5023	4378	4878.7	6786.526
WALO	5150	4026	4735.3	10318.159

由表 4 和收敛曲线(见图 4)可得，WALO 虽然最优值较高，但是得到的较差值和方差相比 ICALO 更

差，说明 WALO 在 0~1 背包问题中算法稳定性较差，且结合收敛曲线也可看出在第 169，432 代陷入局优时间较长，不易于跳出局优；ICA 相比 ALO 得到的结果更优，结合方差看可以看出算法也更稳定，但是 ICA 的求得的值相比 WALO 和 ALO 较差，结合图像可以看出收敛较慢，迭代至 412 次才能够收敛，而 ICALO 在第 87 代就已经收敛，最优值为 5187，相比其他算法结果更优。

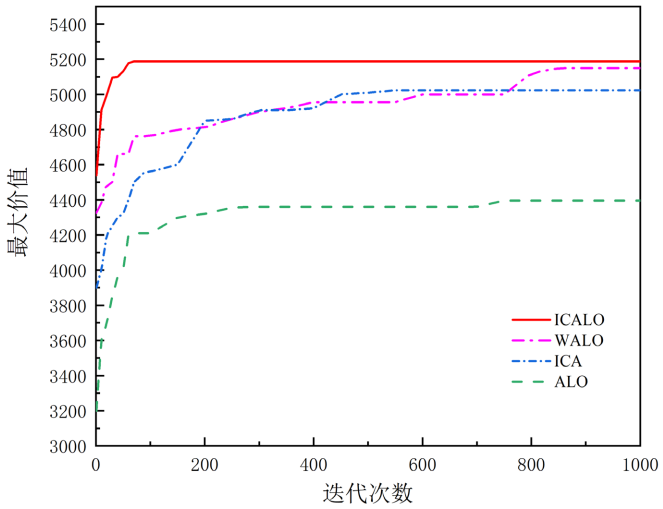


Figure 4. 0~1 knapsack convergence diagram
图 4. 0~1 背包收敛曲线图

综上所诉，ICALO 在 0~1 背包问题中稳定性较高，能够得到较好的值，且具备较好的跳出局优的能力。

5. 桁架结构优化

5.1. 桁架尺寸的优化

选取典型的空间桁架结构模型来验证 ICALO 在求解带有约束的桁架优化问题的可行性。并将 ICALO 的桁架优化与基本的 ALO、WALO、ICA 算法优化结果进行分析比较。通过使用 MATLAB2020a 软件来实现 ICALO，算法计算 10 次，使用其平均值作为计算的结果值。各算法参数的取值与第三节相同。

1) 桁架尺寸优化应用

算例 1：600 杆空间桁架结构如图 5 所示。材料弹性模量 $E = 6.895\times10^{7}$ kpa；材料密度 $\rho = 2768\ \mathrm{kg\cdot m^{-3}}$。约束条件：所有节点的 x 及 y 方向最大位移不得超过 6.35 mm；各杆件的最大许用应力 ±172. 375 MPa。

目标函数如下

$$W(A) = \sum_{i=1}^{n} \rho A_i l_i \tag{19}$$

其中 A_i 是第 i 个单元的截面积 l_i 为第 i 根杆的长度，ρ 为材料密度。

考虑到对桁架的强度和稳定性，对于应力约束和位移约束条件如下：

$$\sigma_{ik} \le \overline{\sigma_i}\left(i = 1,2,\cdots,N; K = 1,2,\cdots,P\right) \tag{20}$$

$$\mu_{jik} \le \overline{\mu_{jl}}\left(j = 1,2,\cdots,m; l = 1,2,3; K = 1,2,\cdots,P\right) \tag{21}$$

式中 σ_{ik} 是第 i 个单元在第 k 个载荷工况下的应力，μ_{jik} 是第 k 个载荷工况下节点 j 在 l 方向的位移，$\overline{\sigma_i}$ 和 μ_{jl} 分别为第 i 个单元的许用应力值和节点 j 在 l 方向的位移允许值，P 为载荷工况总数，m 为结构的节点总数。

将杆件子结构见表 5，ICALO 算法初始设置：初始种群个体 200；迭代次数 500。由于 600 杆空间桁架结构的数据分析时间较长，并且在每个非固定节点有 10 KN 的荷载工况对结构产生影响，在此仅与相关算法和文献进行比较。

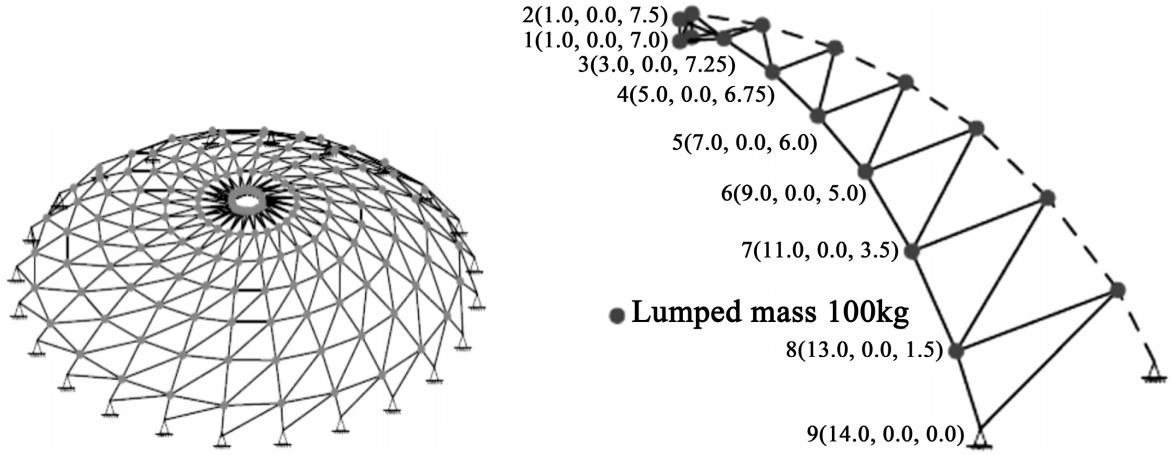


Figure 5. 600-bar space truss structure
图 5. 600 杆空间桁架结构

Table 5. Results of 10 random optimizations
表 5. 10 次随机优化结果

三角形变量号	横截面积/mm^2				
	ICALO	ALO	HALO	ICA	WALO
1	72.155	102.523	91.523	101.355	88.258
2	84.161	115.15	86.073	145.548	132.967
3	113.526	241.351	175.395	164.258	120.258
4	194.46	175.26	145.249	167.290	191.999
5	304.238	346.645	345.528	326.903	170.516
6	326.415	321.782	331.357	335.483	353.548
7	364.516	464.625	364.412	364.516	364.516
8	368.151	465.234	364.412	464.516	397.355
9	319.15	425.156	372.382	325.869	313.999

本文算法优化结构为 6347.65 kg；ALO 优化结果为 7215.12 kg；HALO 优化结果为 6698.14 kg。ICA、WALO 质量分别为 6725.43 kg 和 6451.78 kg。

迭代的收敛过程如图 6 所示。可以看出 27 代前迭代频率较低，迭代幅度较大，32 代后迭代频率加快，迭代幅度降低，迭代逐渐平稳，134 代收敛于最优解附近，分析上图可以体现出改进的蚁狮算法在求解桁架问题上具有收敛速度快、收敛精度高的优点，并且，免疫克隆算法和柯西变异的加入可以减少蚂蚁在局部优化区域时间的浪费，能够很大程度提高算法的收敛速度，增加蚂蚁的活跃程度以及种群多样性，使其能跳出局优，因此在寻优程度上相比基本蚁狮算法有较大提高。

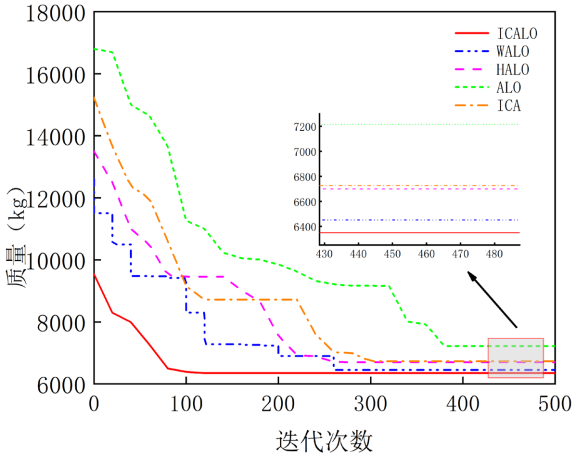


Figure 6. Convergence curve of 600-bar space truss
图 6. 600 杆空间桁架收敛曲线

5.2. 桁架结构动力特性优化

算例 1：18 杆平面桁架结构如图 7 所示。18 杆平面桁架包含 11 个节点，在结构每个非固定节点上有 10kg 荷载，本桁架存在 13 个设计变量，其中有 4 个节点位置变量，且都为连续变量，要求精度误差为 10^{-2} cm 和 10^{-2} m，其中 18 个截面的上下限在 1 cm^2 和 10 cm^2。杆件的材料弹性模量 $E = 210$ GPa，材料密度 $\rho = 7800\ \mathrm{kg/m^3}$ 。

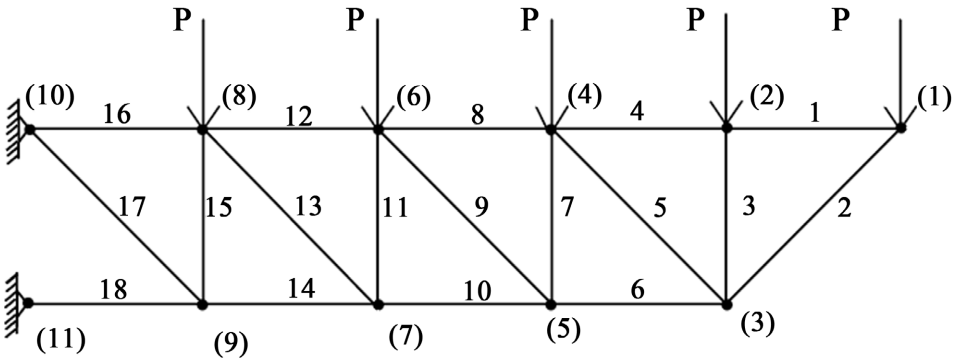


Figure 7. 18-bar planar truss structure
图 7. 18 杆平面桁架结构

目标函数如下

$$W(A) = \sum_{i=1}^{n} \rho A_i l_i + QM \qquad (Q = 0 \text{ or } 1) \tag{22}$$

本文设计变量增加节点位置变量 $\bar{X}$，通过对于各杆件间纲量间的耦合作用进行处理，对杆件进行约束如下：

$$\text{s.t.1}\begin{cases} g_i^{\sigma}(A_i, \bar{X}) = [\sigma_i] - \sigma_i \geq 0 (i = 1, 2, \cdots, N) \\ g_{jl}^{N}(A_i, \bar{X}) = [\mu_{jl}] - \mu_{jl} \geq 0 (j = 1, 2, \cdots, m; l = 1, 2, \cdots, m) \end{cases} \tag{23}$$

式中 $g_i^{\sigma}(A_i, \bar{X})$ 为应力约束，$g_{jl}^{N}(A_i, \bar{X})$ 为位移约束。

动力学约束如下：

$$\text{s.t.2}\begin{cases}\omega_k \geq \omega_{k\max}^* \\ \omega_k \leq \omega_{k\min}^*\end{cases} \tag{24}$$

由于桁架结构优化较为复杂，因此仅与 ALO、ICA、WALO 进行比较。ICALO 算法初始设置：初始种群个体 200；迭代次数 500。其余算法参数设置同上，10 次随机优化收敛曲线见图 8，结果如表 6 所示，最终优化图形如图 9 所示。

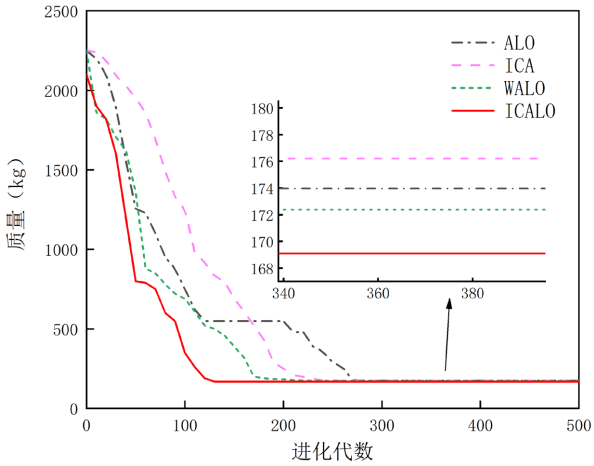


Figure 8. Convergence curve of 18-bar plane truss
图 8. 18 杆平面桁架收敛曲线

Table 6. Comparison of 10 random shape optimization results
表 6. 10 次随机形状优化结果对比

变量号	ICALO	ALO	ICA	WALO
A_1	3.92	6.21	6.32	2.87
A_2	2.65	3.21	2.12	2.47
A_3	1.01	2.64	3.21	1.21
A_4	3.84	4.54	5.65	3.64
A_5	2.14	1.21	2.32	2.05
A_6	3.25	3.02	2.65	2.97
A_7	2.10	1.05	0.98	1.14
A_8	3.47	3.89	3.68	3.54
A_9	2.14	3.10	3.00	1.12
A_{10}	4.21	3.55	2.99	3.84
A_{11}	1.87	1.22	1.35	2.66
A_{12}	3.74	3.88	3.68	3.87
A_{13}	2.14	2.54	2.66	2.04
A_{14}	3.54	3.21	3.54	2.98
A_{15}	2.00	2.55	2.31	3.21

Continued

A_{16}	3.64	3.87	4.10	3.65
A_{17}	2.78	3.54	2.95	2.68
A_{18}	4.62	6.75	5.96	3.99
桁架质量(kg)	169.51	173.94	176.24	172.46

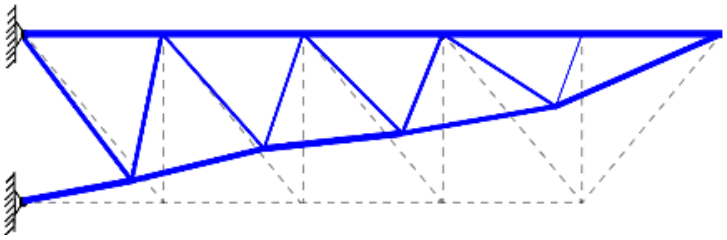


Figure 9. Structure diagram of the planar 18-bar truss after ICALO layout optimization
图 9. ICALO 布局优化后的平面 18 杆桁架结构图

由表 6 可以看出 ICALO 算法所求的结果均比 ALO、WALO、ICA 更优，并且在平均值和最优值数值上有明显的提高，从表 6 可以看出 ICALO 算法的最优解相比 ALO，ICA，WALO 算法分别轻了 4.43 kg、6.73 kg、3.95 kg。

在图 8 可以看出相比 ALO，ICALO 收敛速度更快，并且 ICALO 在第 114 代就达到最优值附近，而 ALO 和 WALO 分别在 283 代和 178 代趋于最优值，ICALO 精度也明显优于其他算法。优化后空间桁架结构图见图 9。

算例 2：145 杆平面桁架结构如图 10 所示。结构外围存在固定约束，每个非固定节点上有 100 kg 荷载，本桁架存在 145 个设计变量，其中有 56 个节点位置变量，且都为连续变量，要求精度误差为 10^{-2} m 以内，其中杆截面的上下限在 1 cm^2 和 200 cm^2，标准模块和桁架分组见图 11。

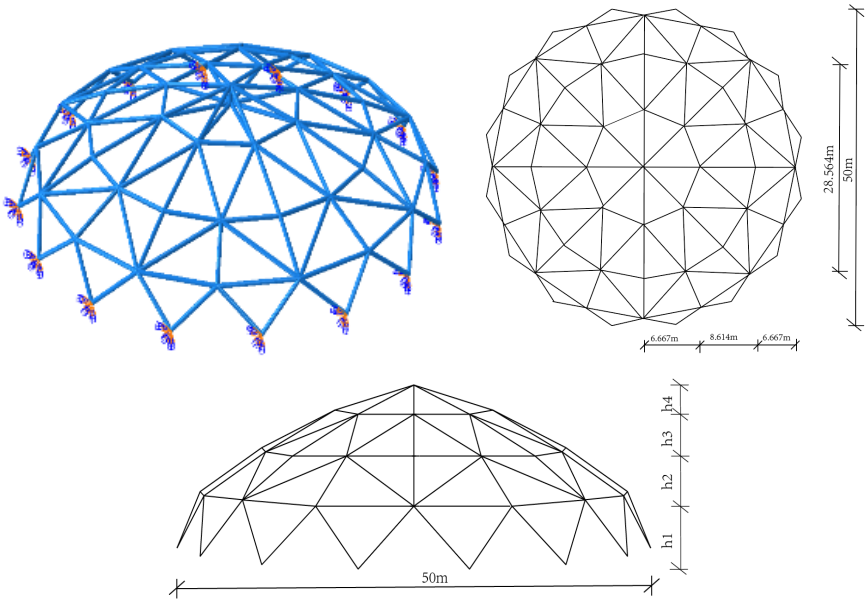


Figure 10. Structure diagram of 145-bar space truss
图 10. 145 杆空间桁架结构图

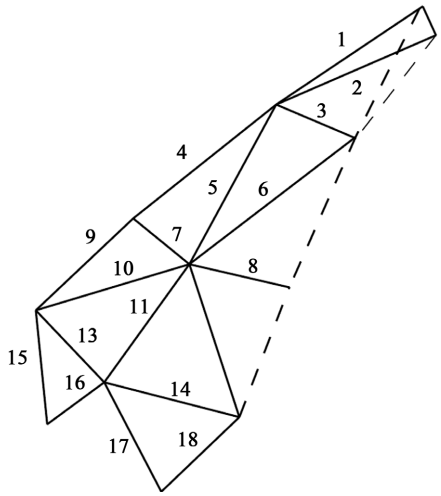


Figure 11. Standard modules and trusses grouping
图 11. 标准模块和桁架分组

本次优化考虑三种设计变量

1) 考虑桁架横截面积和高度变量

其中各层高度约束如下：

$$1.56\ \text{m} \le h_4 \le 3.98\ \text{m}, 3.65\ \text{m} \le h_3 \le 6.12\ \text{m}, 2.35\ \text{m} \le h_2 \le 9.14\ \text{m}, 1.23\ \text{m} \le h_1 \le 4.32\ \text{m}$$

2) 考虑桁架横截面积和布局变量

(布局变量是指同一高度相邻节点之间距离的设计，其中每层杆件数小于 48 根，由于每层节点位置的改变会引起总荷载的改变，因此将每层荷载设置为一个总量，平均分配至每个节点处。)

3) 考虑桁架横截面积、布局变量和布局变量(高度和布局约束同上)

本桁架优化时需要在左右对称的条件下，按左右对称分为 8 组，每组 18 根杆，如图 10 所示，其中 12 个节点位置变量，位移可以在 x，y，z 三个方向移动，精度在 1 cm 以内。结构第一约束频率为 $6.92\ \text{Hz} \le \omega_1 \le 13.84$，第二固有频率 $\omega_2 \le 28.65\ \text{Hz}$，为了对结构在不同频率下质量进行研究，将结构频率为 $13.84\ \text{Hz} \le \omega_1 + \omega_2 \le 35.57\ \text{HZ}$ 和 $35.57\ \text{Hz} \le \omega_1 + \omega_2 \le 57.84\ \text{Hz}$ 的结果进行比较。桁架优化结果和变形图如下：

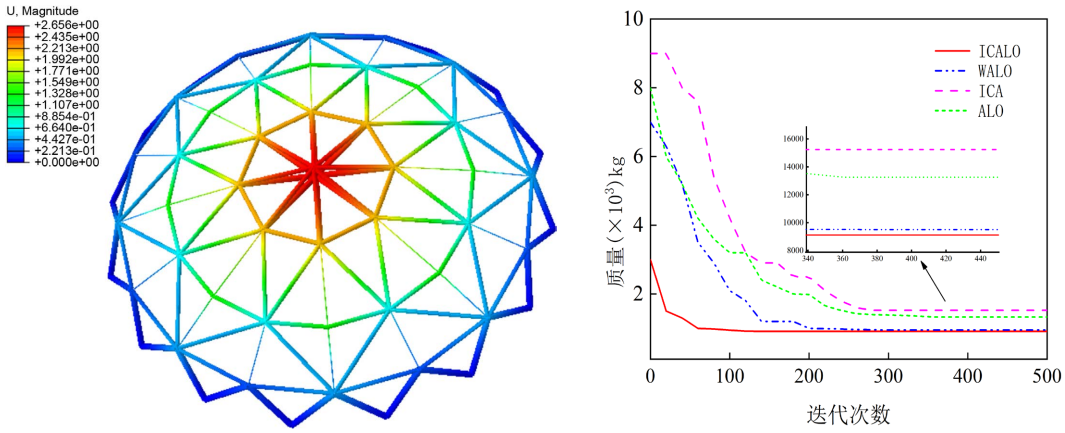


Figure 12. Deformation diagram and convergence curve of the highly optimized 145-bar space truss structure
图 12. 高度优化 145 杆空间桁架结构变形图及收敛曲线

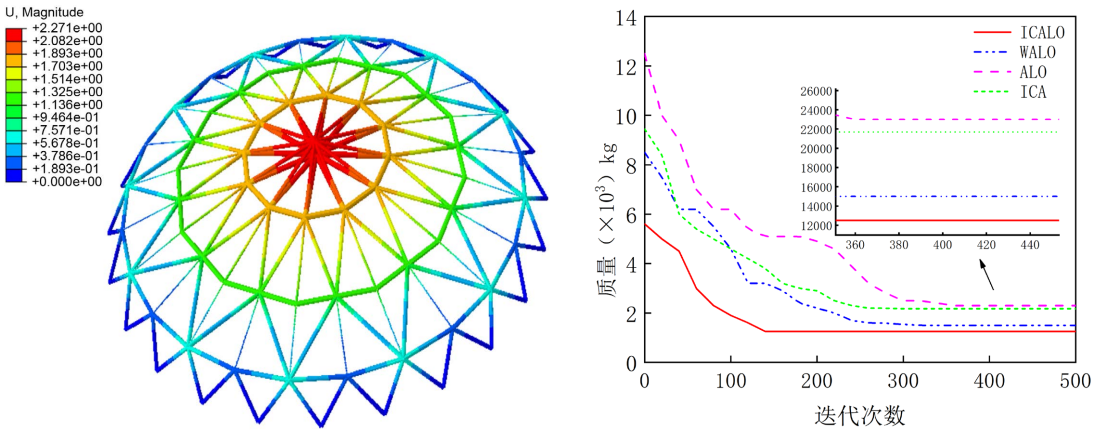


Figure 13. Displacement deformation diagram and convergence curve of 145-bar space truss structure after layout optimization
图 13. 布局优化后 145 杆空间桁架结构位移变形图及收敛曲线

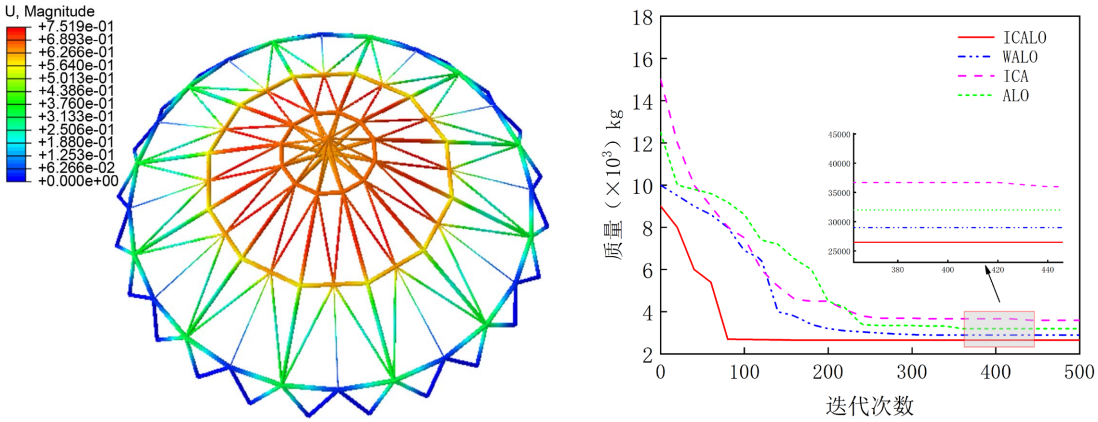


Figure 14. Displacement deformation diagram and convergence curve of low-frequency 145-bar space truss structure
图 14. 低频率 145 杆空间桁架结构位移变形图及收敛曲线

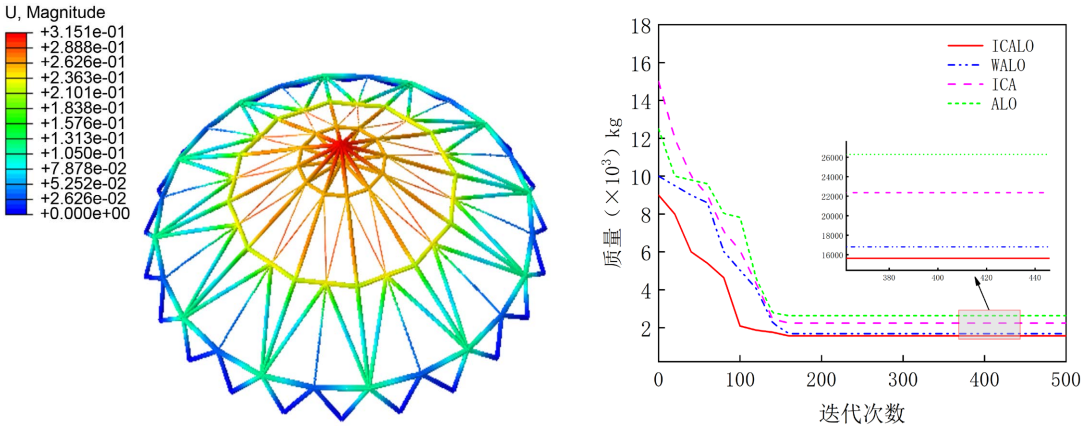


Figure 15. Deformation diagram and convergence curve of high frequency 145-bar space truss structure
图 15. 高频率 145 杆空间桁架结构变形图及收敛曲线

图 12~15 分别为不同算法在不同设计变量下桁架优化最终变形图及优化收敛曲线，从表 7 中可以看出不同频率方案下的 10 次的优化结果，相比 WALO、ALO、ICA，ICALO 在高频率和低频率优化结果下都能使得质量更小，同基本的 ALO 相比，质量最高能够减小 80%，并且从收敛曲线中可以看出 ICALO 收

敛速度较快，能够更好的跳出局优。桁架优化最终变形图可以看出施加荷载后桁架位移都在 3 cm 以内，但是在 ALO 优化后，整体桁架频率超出约束范围，说明本文的改进策略能够较好的适应桁架的力学特征。并且从桁架的位移图中可以看出，布局优化后每层杆件的位移相比高度优化减少 15.04%，杆件受力相比只考虑横截面和高度的桁架结果更优；从图 14、图 15 可以看出在考虑频率约束后，相比小频率桁架频率较大的结构在优化后，位移减少 58.04%，质量增加 30.61%，每层桁架的位移相比小频率桁架有较好的改善。但是，由于优化的单体杆数从 18 根变为 23 根，桁架总重量从 9120 kg 变为 12,426 kg，优化前后质量相差过大，若对桁架稳定性要求较低，出于经济性考虑，对于桁架的优化仅从高度和横截面积等变量进行优化。

Table 7. Optimization results of 10 times under different frequency schemes
表 7. 不同频率方案下的 10 次的优化结果

方案		算法	最优值	最差值	平均值
1		ICALO	9120	10,658	93,218
		WALO	9504	12,847	98,115
		ICA	15,243	20,681	17,944
		ALO	13,265	16,311	14,218
2		ICALO	12,426	14,147	12,988
		WALO	15,521	17,784	16,358
		ICA	21,657	25,921	24,581
		ALO	23,482	25,691	23,648
3	$13.84\ \text{Hz} \le \omega_1 + \omega_2 \le 35.57\ \text{Hz}$	ICALO	15,628	18,916	15,968
		WALO	16,814	17,518	17,214
		ICA	22,351	26,585	23,814
		ALO	26,289	28,681	26,948
	$35.57\ \text{Hz} \le \omega_1 + \omega_2 \le 57.84\ \text{Hz}$	ICALO	22,531	24,518	22,987
		WALO	29,047	33,629	32,628
		ICA	36,121	48,817	39,481
		ALO	32,022	39,281	36,176

6. 结论

1) 本文对于蚁狮算法易陷入局部最优，后期收敛速度较慢求解精度较低等问题，提出基于免疫克隆选择的蚁狮算法，改进机制是通过在精英蚁狮更新中加入柯西变异算子，提高算法后期的开发能力，改变自适应权重和高斯变异，减少后期较差蚁狮对蚂蚁的影响，进一步提高算法全局寻优能力以及收敛精度，通过 10 个标准测试函数和 0~1 背包问题分析出改进后算法收敛速度更快，更易于跳出局优。

2) 本文以桁架的横截面积、高度、布局为设计变量，对桁架结构的应力、位移和频率进行约束，将 ICALO 算法应用到桁架结构的优化中，并通过最优解对桁架结构形状进行优化，得出桁架结构形状和受力变形图，体现出改进蚁狮算法应用于实际工程的可行性。但是如何运用算法对桁架材料以及对算法本身的时间复杂度进行优化是下一步需要具体研究和探索的方向。

基金项目

河北省教育厅在读研究生创新能力培养资助项目(CXZZSS2022028)。

参考文献

[1] 李雪岩，李学伟，李静. 基于混沌集群智能优化算法的多目标粗糙集属性约减[J]. 运筹与管理，2020，29(12): 30-37.

[2] 叶小岭，姚锦松，熊雄，巩灿灿，金瞳宇. 基于余弦相似度和移动曲面拟合的地面气温质量控制算法[J]. 中国科技论文, 2020, 15(5): 577-584+592.

[3] Feo, G., Sharma, J. and Cunningham, S. (2020) Integrating Fiber Optic Data in Numerical Reservoir Simulation Using Intelligent Optimization Workflow. *Sensors*, **20**, 506-524. https://doi.org/10.3390/s20113075

[4] Ma, Z.T., Yuan, X.F., Han, S., Sun, D.Y. and Ma, Y. (2019) Improved Chaotic Particle Swarm Optimization Algorithm with More Symmetric Distribution for Numerical Function Optimization. *Symmetry*, **11**, 106-110. https://doi.org/10.3390/sym11070876

[5] 赵克新，黄长强，王渊，等. 基于混沌蚁狮算法的无人机航迹规划[J]. 飞行力学, 2018, 36(1): 93-96.

[6] 简春菲，王晓峰，王司. 基于 ALO-SVM 的模拟电路故障诊断研究[J]. 自动化技术与应用, 2019, 38(12): 113-118.

[7] Kilic, H., Yüzgeç, U., *et al*. (2019) Improved Antlion Optimization Algorithm via Tournament Selection and Its Application to Parallel Machine Scheduling. *Computers & Industrial Engineering*, **132**, 166-186. https://doi.org/10.1016/j.cie.2019.04.029

[8] 李宗妮，吴伟民，林志毅. 一种采用改进蚁狮优化算法的图像增强方法[J]. 计算机应用研究，2018，35(4): 1258-1260+1265.

[9] 吴伟民，张晶晶，林志毅，等. 双重反馈机制的蚁狮算法[J]. 计算机工程与应用, 2017, 53(12): 31-35, 75.

[10] 赵小国，刘丁，景坤雷. 融合改进蚁狮算法和 T-S 模糊模型的噪声非线性系统辨识[J]. 控制与决策，2019，34(4): 759-766.

[11] 王亚东，石全，尤志锋，等. 差分准对立学习多目标蚁狮算法[J]. 计算机工程与应用, 2020, 56(13): 156-163.

[12] 肖子雅，刘升. 精英反向黄金正弦鲸鱼算法及其工程优化研究[J]. 电子学报, 2019, 47(10): 2177-2186.

[13] Tharwat, A. and Hassanien, A.E. (2018) Chaotic Antlion Algorithm for Parameter Optimization of Support Vector Machine. *Applied Intelligence*, **48**, 670-686. https://doi.org/10.1007/s10489-017-0994-0

[14] 俎云霄，周杰. 基于组合混沌遗传算法的认知无线电资源分配[J]. 物理学报, 2011, 60(7): 894-901.

[15] Ryu, N., Min, S., *et al*. (2019) Multiobjective Optimization with an Adaptive Weight Determination Scheme Using the Concept of Hyperplane. *International Journal for Numerical Methods in Engineering*, **118**, 303-319. https://doi.org/10.1002/nme.6013

[16] Calsina, À., Cuadrado, S., Desvillettes, L. and Raoul, G. (2016) Asymptotic Profile in Selection-Mutation Equations: Gauss versus Cauchy Distributions. *Journal of Mathematical Analysis and Applications*, **444**, 1515-1541. https://doi.org/10.1016/j.jmaa.2016.07.028

[17] 刘静秋，杜文莉，张飞. 基于粒子群算法的函数复杂度分类法[J]. 控制工程, 2020, 27(8): 1337-1345.

[18] 康俊涛，邹立，曹鸿猷，张亚州. 基于樽海鞘群算法的桁架结构优化设计[J]. 空间结构, 2020, 26(3): 51-58+83.

[19] Subhashini, K.R., Satapathy, J.K., *et al*. (2017) Development of an Enhanced Ant Lion Optimization Algorithm and Its Application in Antenna Array Synthesis. *Applied Soft Computing*, **59**, 153-173. https://doi.org/10.1016/j.asoc.2017.05.007

[20] 李彦苍，吴悦. 基于高斯变异的蚁狮算法及其在组合优化中的应用[J]. 中国科技论文, 2022, 17(3): 295-304.

[21] Chen, H., Li, W. and Yang, X. (2020) A Whale Optimization Algorithm with Chaos Mechanism Based on Quasi-Opposition for Global Optimization Problems. *Expert Systems with Applications*, **158**, Article ID: 113612. https://doi.org/10.1016/j.eswa.2020.113612

[22] 张凤斌，王超，杨泽. 基于疫苗算子和柯西变异的克隆选择算法改进[J]. 计算机工程与应用, 2017, 53(1): 118-123.

[23] 方鹏飞，黄陆光，娄苗苗，蒋昆. 基于二维 Logistic 混沌映射与 DNA 序列运算的图像加密算法[J]. 中国科技论文, 2021, 16(3): 247-252.

[24] Valandar, M.Y., Ayubi, P. and Barani, M.J. (2017) A New Transform Domain Steganography Based on Modified Logistic Chaotic Map for Color Images. *Journal of Information Security and Applications*, **34**, 142-151.

https://doi.org/10.1016/j.jisa.2017.04.004

[25] 陆玉佳, 倪志伟, 朱旭辉, 许力分, 伍章俊. 基于MapReduce改进离散型萤火虫算法和多重分形的属性约简方法[J]. 模式识别与人工智能, 2018, 31(6): 537-547.

[26] Basu, M. (2016) Improved Particle Swarm Optimization for Global Optimization of Unimodal and Multimodal Functions. *Journal of the Institution of Engineers* (*India*): *Series B*, **97**, 525-535. https://doi.org/10.1007/s40031-015-0204-6

[27] 石晟, 杜东升, 王曙光, 李威威. 基于高斯函数的完全非平稳地震动多峰模型及其应用[J]. 建筑结构学报, 2020, 41(5): 198-206.

[28] Moradi, N., Kayvanfar, V. and Rafiee, M. (2021) An Efficient Population-Based Simulated Annealing Algorithm for 0-1 Knapsack Problem. *Engineering with Computers*, **96**, 1245-1365. https://doi.org/10.1007/s00366-020-01240-3

[29] Sun, W.Z., Zhang, M., Wang, J.S., Guo, S.S., Wang, M. and Hao, W.K. (2021) Binary Particle Swarm Optimization Algorithm Based on Z-Shaped Probability Transfer Function to Solve 0-1 Knapsack Problem. *IAENG International Journal of Computer Science*, **48**, 236-246.

[30] 王娜, 向凤红, 毛剑琳. 改进的自适应遗传算法求解 0/1 背包问题[J]. 计算机应用, 2012, 32(6): 1682-1684.