工程科学学报, 第 46 卷, 第 10 期: 1864−1879, 2024 年 10 月
Chinese Journal of Engineering, Vol. 46, No. 10: 1864−1879, October 2024
https://doi.org/10.13374/j.issn2095-9389.2023.12.02.001; http://cje.ustb.edu.cn

求解多维复杂函数及真实世界工程设计问题的高效海洋捕食者算法

刘景森[1,2)], 范文凯[1,2)], 孙　琳[2)✉], 周　欢[3)]

1) 河南大学河南省智能网络理论与关键技术国际联合实验室, 开封 475004　2) 河南大学软件学院, 开封 475004　3) 河南大学商学院, 开封 475004

✉通信作者, E-mail: sunlin@henu.edu.cn

摘　要　针对海洋捕食者算法在面对复杂函数和工程设计优化问题时存在的自适应能力有限、寻优精度有时较低、局部桎梏概率较高等缺点, 提出一种新的高效自适应海洋捕食者算法. 首先在海洋记忆存储阶段引入学习自动机引导的教与学搜索机制, 更好地平衡算法在不同迭代时期对探索和挖掘能力的不同需求; 然后在局部开发阶段, 引入对数螺旋探索机制, 加强算法在最优解附近的精细挖掘能力, 进一步提高收敛精度; 最后在算法中每次迭代末尾处加入改进的自适应相对反射策略, 提升种群跳出局部最优的能力, 降低局部桎梏概率. 为了分析和验证该改进算法的性能, 将其和 6 种代表性算法在进化计算大会(CEC)2017 测试套件上进行 100 维的函数极值测试, 并在 4 个具有挑战性的工程设计优化问题上进行测试. 测试结果表明在求解多维复杂函数和工程设计优化问题时, 本文改进算法的寻优精度、收敛性能和求解稳定性明显优于其他 6 种代表性算法, 尤其在高维复杂函数下, 其寻优性能的优越性更为显著.

关键词　优化; 海洋捕食者算法; 学习自动机; 对数螺旋机制; 相对反射机制; 工程设计优化问题

分类号　TP301.6

Efficient marine predators algorithm for solving multidimensional complex functions and real-world engineering design problems

LIU Jingsen[1,2)], *FAN Wenkai*[1,2)], *SUN Lin*[2)✉], *ZHOU Huan*[3)]

1) International Joint Laboratory of Intelligent Network Theory and Key Technology, Henan University, Kaifeng 475004, China

2) College of Software, Henan University, Kaifeng 475004, China

3) Business School, Henan University, Kaifeng 475004, China

✉Corresponding author, E-mail: sunlin@henu.edu.cn

ABSTRACT　As optimization problems grow increasingly complex, characterized by their intricate difficulty, larger-scale, and diverse constraints, swarm intelligence optimization algorithms have emerged as an effective solution for addressing these multifaceted challenges. Among these, the marine predators algorithm, a recent innovation in intelligent optimization algorithms, has demonstrated remarkable efficacy in solving optimization issues. However, its application to complex CEC test function sets and engineering constraint problems reveals several limitations, including limited adaptive ability, low optimization accuracy, and high local shackle probability. This paper proposes an enhanced version of the marine predators algorithm designed to overcome its inherent shortcomings. The enhancement begins with the integration of a learning automata guided teaching–learning search mechanism during the marine memory stage. This adjustment aims to strike a better balance between exploration and exploitation across different iteration periods.

收稿日期: 2023−12−02

基金项目: 国家自然科学基金资助项目 (72104069); 河南省重点研发与推广专项 (222102210065)

Subsequently, the introduction of a logarithmic spiral exploration mechanism phase strengthens the algorithm's ability to conduct nuanced searchers around the optimal solution, thereby improving convergence accuracy. Finally, an improved adaptive relative reflection strategy is added at the end of each iteration to enhance the algorithm's capability to escape local optima and reduce the risk of local shackling. The optimization performance of this refined algorithm is evaluated through parameter sensitivity analysis, determining the optimal parameter values. To validate its effectiveness, the improved algorithm undergoes testing against six benchmark algorithms, including the basic marine predators algorithm and its variants, as well as other improved algorithms and those recognized with awards in the CEC2017 test suite across 100 dimensions. The evaluation focuses on optimization accuracy, the Wilcoxon rank sum test, and boxplot analysis. The test results indicate that the improved algorithm proposed in this paper outperforms the other six benchmark algorithms in optimization precision, convergence rate, and solution stability, particularly when solving complex functions in high-dimensional (100 dimensions) spaces. Furthermore, the applicability and superior performance of the improved algorithm are demonstrated through comparative analysis with four established algorithms on challenging engineering design optimization problems. These include welded beam design, process synthesis, heat exchanger network design, and design optimization of industrial refrigeration systems. The findings unequivocally showcase the enhanced algorithm's exceptional ability to solve various engineering constraint problems effectively.

KEY WORDS optimization; marine predators algorithm; learning automata; logarithmic spiral mechanism; relative reflection mechanism; engineering design optimization

随着科学技术的进步, 优化问题广泛出现在生活中各个领域, 同时优化问题也呈现出难度更加复杂、规模更加庞大、约束条件更加多变等特点, 在传统优化方法中像梯度下降法、枚举法、牛顿法等的求解精度和求解效率已经很难满足实际需求, 因此为求解复杂优化问题, 一系列基于群体搜索的元启发式智能优化算法被提出, 群智能优化算法具有执行效率高, 全局搜索能力强等特点, 在解决复杂优化问题上具备出色的竞争力.

海洋捕食者算法 (MPA) 是由 Faramarzi 等[1] 在 2020 年提出的一种模拟海洋捕猎行为的元启发式算法. 算法的灵感来自于海洋捕食者的觅食策略. 其寻优过程分为三个阶段, 捕食者和猎物在这三个阶段中按照莱维运动或者布朗运动进行位置更新. 猎物在被捕食的同时也充当捕食者身份, 使得算法更具有动态特性, 且其独有的海洋记忆存储阶段与涡流效应阶段, 可以进一步提高更新后种群的质量. 该算法拥有较为优秀的全局搜索能力, 且控制参数少, 层次结构清晰, 具有趋优避差的导向性特征, 现已成功应用于预测多国 COVID-19 感染人数[2]、医学图像分类[3] 、光伏模型参数提取[4] 、焊接接头拉伸性能预测[5] 、电力系统经济调度[6] 和网络功率分配[7] 等一系列实际应用领域.

虽然 MPA 在优化方面具有一定优势, 但依然存在着群智能算法容易陷入局部最优、收敛精度有时不高和收敛速度较慢的问题, 仍需进行改进以提高其优化性能. 因此, 为了进一步提升 MPA 寻优性能, 许多学者对 MPA 进行了改进. 在国内, 张磊等[8] 提出了精英反向黄金正弦海洋捕食者算法, 旨在增强算法的局部开发能力, 从而提高算法的求解精度以及收敛速度. 李守玉和何庆[9] 通过邻域学习、维度变异机制以及正余弦与步长因子来增强海洋捕食者算法的全局与局部搜索能力, 并将其应用于数据集的分类. 王逸文等[10] 针对海洋捕食者算法寻优精度较低、收敛速度较慢等缺陷, 提出了一种多策略融合改进的海洋捕食者算法. 付华等[11] 提出阶段化改进的海洋捕食者算法, 提升了海洋捕食者算法的自适应能力, 降低了局部桎梏概率. 在国外, Sadiq 等[7] 提出一种非线性海洋捕食者算法, 通过引入自适应步长和非线性控制参数来平衡算法的探索和开发阶段, 增强海洋捕食者算法的探索和开发能力, 并应用在结合非正交多址和可见光通信的超 5G 网络(NOMA−VLC−B5G)中实现功率公平分配. Oszust[12] 提出一种具有局部逃逸机制的海洋捕食者算法, 使算法探索和开发能力之间更加平衡, 防止过早陷入局部收敛. Fan 等[13] 为提高海洋捕食者算法中的种群多样性, 提升算法的收敛精度而提出了一种引入对立学习策略、惯性权重系数与非线性步长控制参数的海洋捕食者算法. Yousri 等[14] 提出了一种新的综合学习海洋捕食者算法, 增强了种群多样性, 提升了算法的勘探能力和开发能力, 并用来识别超级电容等效电路的最优参数. Houssein 等[15] 提出了一种基于对立学习与灰狼优化器的海洋捕食者算法, 来防止算法陷入局部收敛, 提升算法的局部探索效率和收敛速度, 并成功将其应用于光伏系

统的最大功率点跟踪问题. Abd Elaziz 等[16] 通过使用量子理论来改进海洋捕食者算法中的位置更新公式, 从而提出了量子海洋捕食者算法, 提高了算法的勘探能力和开发能力, 并成功应用于解决多级图像分割问题.

上述学者的研究促进了海洋捕食者算法的快速发展, 但该算法在解决高维度复杂问题方面仍面临着像勘探与开发能力不足、局部桎梏概率高、收敛精度低等挑战. 为了更好地提升海洋捕食者算法的应用能力和寻优性能, 本文提出一种基于学习自动机引导的教与学搜索、对数螺旋探索和自适应相对反射机制的海洋捕食者算法(LLAMPA). 在海洋记忆存储阶段引入学习自动机引导的教与学搜索机制, 教与学机制具备较强的交互性, 能充分利用来自其他个体的有用信息进而增强勘探与开发能力, 同时加入的学习自动机 (LA) 是一个自适应决策系统, 可以增加捕食者根据环境选择最优学习机制的概率, 提高算法优化性能; 在迭代后期的局部开发阶段, 引入对数螺旋探索机制, 可以更加充分利用最佳个体的位置信息, 加强算法的局部挖掘能力, 提升算法的收敛精度; 在算法中每次迭代末尾处加入改进的自适应相对反射策略, 将较差部分个体以其他随机个体位置为支点向其周围进行反射, 增强算法逃离局部极值区域的能力, 防止种群容易陷入局部收敛, 克服算法早熟问题.

本文将 LLAMPA 与基本海洋捕食者算法 MPA 及其变体, 以及其他改进算法和进化计算大会 (CEC) 优胜算法等多种具有代表性的对比算法在 CEC2017 函数集[17] 上进行 100 维的函数极值优化实验, 通过寻优精度分析、关于迭代终止条件的讨论分析、Wilcoxon 秩和检验统计分析以及箱线图分析, 对优化结果进行讨论, 结果表明, LLAMPA 算法的寻优性能和求解稳定性明显优于其他 6 种代表性算法, 且相对于其他 6 种算法的优化结果具有显著性差异. 对焊接梁设计、过程合成、热交换网络设计以及工业制冷系统优化设计等 4 个具有挑战性的工程设计优化问题[18] 的求解, 其实验结果也充分显示了 LLAMPA 算法优越的求解效果和良好的适用性.

1　基本海洋捕食者算法

基本海洋捕食者算法 MPA 流程如下:

Step1: 根据搜索空间中每一维的上下限, 初始化猎物种群 $\boldsymbol{X}_i^j(i=1,2,\cdots,N, j=1,2,\cdots,D)$, 即

$$\boldsymbol{X}_{\mathrm{i}}^j=\mathrm{rand}\times(\mathbf{ub}(j)-\mathbf{lb}(j))+\mathbf{lb}(j) \tag{1}$$

其中, N为猎物种群的规模, D为空间维度, $\mathbf{ub}(j)$和$\mathbf{lb}(j)$分别为搜索空间中第 j维的上、下限, rand为[0,1]上均匀分布的随机数.

Step2: 对猎物种群中个体的每一维进行边界处理, 并根据目标函数计算每个猎物的适应度值.

Step3: 根据目标函数适应度值从猎物种群中挑选出最优个体, 并由此复制N次构建成捕食者种群矩阵 $\boldsymbol{E}_i^j\ (i=1,2,\cdots,N, j=1,2,\cdots,D)$.

Step4: 执行海洋记忆存储功能, 将当前猎物种群和上次执行海洋记忆功能时存储的旧猎物种群进行适应度对比, 用更好解替换先前解.

Step5: 根据当前的迭代阶段来更新猎物种群中的个体位置.

当$t<1/3\cdot\mathrm{Max_}t$时, 猎物种群中个体位置更新如下:

$$\begin{aligned}&\mathbf{stepsize}=\boldsymbol{R}_{\mathrm{B}}\otimes(\boldsymbol{E}_i-\boldsymbol{R}_{\mathrm{B}}\otimes\boldsymbol{X}_i(t))\\&\boldsymbol{X}_i'(t)=\boldsymbol{X}_i(t)+P\times\mathrm{rand}\otimes\mathbf{stepsize}\end{aligned} \tag{2}$$

当$1/3\cdot\mathrm{Max_}t\leqslant t<2/3\cdot\mathrm{Max_}t$时, 猎物种群分为两个部分进行个体位置更新.

猎物种群中前1/2个体位置更新如下:

$$\begin{aligned}&\mathbf{stepsize}=\boldsymbol{R}_{\mathrm{L}}\otimes(\boldsymbol{E}_i-\boldsymbol{R}_{\mathrm{L}}\otimes\boldsymbol{X}_i(t))\\&\boldsymbol{X}_i'(t)=\boldsymbol{X}_i(t)+P\times\mathrm{rand}\otimes\mathbf{stepsize}\quad i=1,2,\cdots,n/2\end{aligned} \tag{3}$$

猎物种群中后1/2个体位置更新如下:

$$\begin{aligned}&\mathbf{stepsize}=\boldsymbol{R}_{\mathrm{B}}\otimes(\boldsymbol{R}_{\mathrm{B}}\otimes\boldsymbol{E}_i-\boldsymbol{X}_i(t))\\&\boldsymbol{X}_i'(t)=\boldsymbol{E}_i+P\times\mathrm{CF}\otimes\mathbf{stepsize}\quad i=1,2,\cdots,n/2\end{aligned} \tag{4}$$

当$t\geqslant 2/3\cdot\mathrm{Max_}t$时, 猎物种群中个体位置更新如下:

$$\begin{aligned}&\mathbf{stepsize}=\boldsymbol{R}_{\mathrm{L}}\otimes(\boldsymbol{R}_{\mathrm{L}}\otimes\boldsymbol{E}_i-\boldsymbol{X}_i(t))\\&\boldsymbol{X}_i'(t)=\boldsymbol{E}_i+P\times\mathrm{CF}\otimes\mathbf{stepsize}\end{aligned} \tag{5}$$

其中, t 是当前迭代次数, $\mathrm{Max_}t$为最大迭代次数, $\otimes$为逐项乘法运算, $\boldsymbol{X}_i$为猎物种群中第i个体, $\boldsymbol{E}_i$为捕食者种群中第i个体. P为常数, 算法中取值 0.5. $\mathbf{stepsize}$为移动步长向量, $\boldsymbol{R}_{\mathrm{B}}$为布朗运动随机向量, $\boldsymbol{R}_{\mathrm{L}}$为莱维运动随机向量 , rand是[0,1]上均匀分布的随机数, CF是自适应参数, 用于控制捕食者运动的步长, CF的更新公式如下:

$$\mathrm{CF}=\left(1-\frac{t}{\mathrm{Max\_}t}\right)^{\frac{2\times t}{\mathrm{Max\_}t}} \tag{6}$$

Step6: 对更新后猎物个体的每一维进行边界处理, 根据目标函数计算适应度值, 并记录最优个体.

Step7: 执行海洋记忆存储功能, 将更新后的猎

物种群和旧的猎物种群进行适应度对比, 用更好解替换先前解.

Step8: 对猎物种群中的个体根据涡流形成和鱼类聚集装置的影响进行位置更新, 即

$$\boldsymbol{X}_i(t+1)=\begin{cases}\boldsymbol{X}_i'(t)+\mathrm{CF}[\mathbf{lb}+\boldsymbol{R}_{\mathrm{L}}\otimes(\mathbf{ub}-\mathbf{lb})]\otimes\boldsymbol{U} \\ \qquad\qquad r\leqslant \mathrm{FADs} \\ \boldsymbol{X}_i'(t)+[\mathrm{FADs}(1-r)+r]\times(\boldsymbol{X'}_p(t)-\boldsymbol{X'}_q(t)) \\ \qquad\qquad r>\mathrm{FADs}\end{cases} \tag{7}$$

其中, FADs是对优化过程产生影响的概率, 算法中FADs取值为 0.2, **ub**和**lb**分别为搜索空间上、下限, $\boldsymbol{U}$是二进制数组向量, r为[0, 1]上均匀分布的随机数, $\boldsymbol{X}_p$、$\boldsymbol{X}_q$分别为猎物种群中两个随机个体.

Step9: 判断是否达到最大迭代次数, 若是, 则输出结果; 否则, 转到 Step2 继续进行迭代.

2 LLAMPA 改进算法

2.1 引入学习自动机引导的教与学搜索机制

在海洋记忆阶段, 为了使那些觅食不成功的捕食者个体能够通过向同伴或者精英个体学习获得更多关于猎物的知识来提高觅食的成功率, 本文在海洋记忆阶段引入学习自动机引导的教与学搜索机制. 其中教机制能够使捕食者个体位置向精英个体进行学习, 从而引导种群中个体向着更优解方向进化. 而“学”机制则可以使个体位置向其他随机个体进行学习, 从而使算法中种群保持多样性. 教机制与学机制的交互使用, 可以使种群个体向更优解方向进化的同时又避免了基本算法容易陷入局部最优的问题. 而学习自动机 (LA) 是一个自适应决策系统, 具有从一组与环境交互的预定义行动中选择最优行动的学习策略[19]. 本文中学习自动机包含教机制和学机制两种行动机制. 这两种机制根据特定的概率分布来进行选择, 同时通过适应度值来评估执行所选择机制的影响, 判断该机制是否促进个体向更优解方向进化, 并向自动机进行反馈. 自动机利用反馈的信息来更新教机制和学机制的概率分布, 从而增加捕食者在当时环境下选择最优学习机制的概率.

本文在海洋记忆阶段引入学习自动机引导的教与学搜索机制, 将当前猎物种群与前一次海洋记忆存储的猎物种群进行适应度值比较, 若当前个体的适应度值不如前一次存储的对应个体, 则按照概率$\mathrm{Pr}_{i,\mathrm{act}}$选择教与学算法[20] 中的教机制或者学机制进行位置更新. 更新后的个体适应度值如果优于前一次存储的对应个体, 则使用学习自动机中奖励公式 (11) 对$\mathrm{Pr}_{i,\mathrm{act}}$进行更新. 如果不优于前一次存储的对应个体, 则将更新后的个体用前一次存储的对应个体进行替换, 并使用学习自动机中惩罚公式 (12) 对$\mathrm{Pr}_{i,\mathrm{act}}$进行更新.

教、学机制的数学模型如下:

$$\boldsymbol{X}_i'=\boldsymbol{X}_i+\mathrm{rand}\otimes(\boldsymbol{X}_{\mathrm{teacher}}-\mathrm{TF}\times\boldsymbol{X}_{\mathrm{mean}}) \tag{8}$$

$$\boldsymbol{X}_i'=\begin{cases}\boldsymbol{X}_i+\mathrm{rand}\cdot(\boldsymbol{X}_i-\boldsymbol{X}_k) & f(\boldsymbol{X}_i)<f(\boldsymbol{X}_k) \\ \boldsymbol{X}_i+\mathrm{rand}\cdot(\boldsymbol{X}_k-\boldsymbol{X}_i) & \text{else}\end{cases} \tag{9}$$

其中, rand为[0, 1]上均匀分布的随机数; $\boldsymbol{X}_{\mathrm{teacher}}$为当前代最优学生(即个体); $\boldsymbol{X}_i$为当前代第i个学生; $\boldsymbol{X}_k$为当前代随机选取的一个学生, 且$k\neq i$; $\boldsymbol{X}_{\mathrm{mean}}$表示班级(即种群)平均水平; TF为教学因子, 表示班级平均水平影响学生水平的权重, 随机取 1 或 2, 即:

$$\mathrm{TF}=\mathrm{round}[1+\mathrm{rand}(0,1)] \tag{10}$$

学习自动机的奖、惩机制数学模型如下:

$$\mathrm{Pr}_{i,\mathrm{act}}(t+1)=\begin{cases}\mathrm{Pr}_{i,\mathrm{act}}(t)+\mathrm{RS}(1-\mathrm{Pr}_{i,\mathrm{act}}(t)) & \text{if act = action} \\ \mathrm{Pr}_{i,\mathrm{act}}(t)(1-\mathrm{RS}) & \text{if act} \neq \text{action}\end{cases} \tag{11}$$

$$\mathrm{Pr}_{i,\mathrm{act}}(t+1)= \begin{cases}\mathrm{Pr}_{i,\mathrm{act}}(t)(1-\mathrm{PB}) & \text{if act = action} \\ \mathrm{Pr}_{i,\mathrm{act}}(t)(1-\mathrm{PB})+\mathrm{PB}/(\mathrm{ar}-1) & \text{if act} \neq \text{action}\end{cases} \tag{12}$$

其中, $\mathrm{Pr}_{i,\mathrm{act}}$表示当前第$i$个体执行动作集中各个动作act的概率; action表示当前执行的动作; t为当前迭代次数; ar为自动机中定义的动作数; RS和PB分别表示奖励和惩罚步长, 经反复测试算法中分别取值为 0.6、0.6 时效果最佳.

改进后的海洋记忆存储功能代码段如以下Code1所示

如果当前为首次迭代, 则将初始猎物种群$\boldsymbol{X}_i^j$赋值给用于存放猎物个体历史最优位置的旧猎物种群$\boldsymbol{X}_\mathbf{old}_i^j$ $(i=1,2,\cdots,N,\ j=1,2,\cdots,D)$

```
FOR i = 1 : N
  IF f(X_i) > f(X_old_i)
    利用轮盘赌规则根据P_{i,act}生成动作序号act
    IF act=1
      由“教”公式(8)将X_i更新为X_i'
    ELSE
      由“学”公式(9)将X_i更新为X_i'
    END IF
    IF f(X_i') < f(X_old_i)
      由奖励公式(11)对Pr_{i,act}进行更新
    ELSE
      X_i' = X_old_i
```

由惩罚公式(12)对$\mathrm{Pr}_{i,\mathrm{act}}$进行更新

END IF

ELSE

$\boldsymbol{X}_i' = \boldsymbol{X}_i$

END IF

END FOR

FOR $i = 1{:}N$

对更新后猎物种群中的个体进行边界处理

计算猎物种群中个体的适应度值, 比较并记录最优个体

END FOR

将更新后的猎物种群$\boldsymbol{X}'$重新存储到旧猎物种群. $\boldsymbol{X}$**_old**.

2.2　引入对数螺旋探索机制

在基本算法迭代后期的局部开发阶段, 捕食者种群单一地使用莱维飞行机制生成的随机步长进行局部探索, 因此会存在最优解附近局部精细挖掘能力不强的问题, 造成算法有时寻优精度不高. 为了提高海洋捕食者算法的局部搜索能力, 本文在局部开发阶段引入对数螺旋探索机制. 将个体的位置信息, 由原来的仅基于莱维飞行探索机制生成, 变为按照一定概率从莱维飞行探索机制和对数螺旋探索机制中进行选择, 然后基于所选择的机制生成. 对数螺旋探索机制[21]可以选择当前种群中的最佳个体作为目标对象, 并在最佳个体附近进行螺旋搜索, 具有较强的局部挖掘能力. 本文在局部开发阶段引入对数螺旋探索策略, 丰富了算法在局部搜索上的探索策略, 同时可以在此阶段有效地和莱维飞行出色的跳跃能力进行局部搜索上的互补, 从而可以更加充分地利用最佳个体的位置信息. 改进后的局部开发阶段的位置更新公式, 由基本算法中的公式 (5) 改为公式 (14).

$$\boldsymbol{\mathrm{dist}} = \mathrm{abs}(\boldsymbol{E}_i(t) - \boldsymbol{X}_i(t)) \tag{13}$$

$$\boldsymbol{X}_i'(t) = \begin{cases} \boldsymbol{E}_i(t) + P \times \mathrm{CF} \otimes \boldsymbol{\mathrm{stepsize}} & \mathrm{rand} > 0.5 \\ \boldsymbol{\mathrm{dist}} \times \mathrm{e}^{cl} \times \cos(2\pi l) + \boldsymbol{E}_i(t) & \mathrm{else} \end{cases} \tag{14}$$

其中, **dist**是从猎物种群当前个体到捕食者种群当前个体的距离; $\boldsymbol{X}_i(t)$是猎物种群第t次迭代中第i个个体; $\boldsymbol{E}_i(t)$为捕食者种群第t次迭代中第i个个体; abs()是绝对值函数; c是一个等于 1 的常数; l为 [−1, 1] 上均匀分布的随机数; rand为 [0, 1] 上均匀分布的随机数.

2.3　引入改进的自适应相对反射策略

在 LLAMPA 算法局部开发阶段中引入对数螺旋探索策略, 虽然提高了算法的局部搜索和挖掘能力, 但也使算法中种群的同化程度有所升高, 增加了陷入局部最优的风险, 因此为防止种群容易陷入局部收敛, 克服算法早熟问题, 我们在算法末端改进并加入一种自适应相对反射策略[22]来跳出局部最优.

在该改进策略中, 首先将猎物种群按适应度值进行降序排序, 并选取适应度值最差的K个个体, 然后将其中w个个体以当前最优解作为支点, 将剩下的$K-w$个个体以其他随机个体作为支点, 而后根据其各自的当前位置分别围绕支点进行反射得到新的位置, 如公式 (15)、(16) 所示. 其中w利用线性人口增加策略随着迭代次数的增加而递增, 如公式 (19) 所示. 这样在算法的迭代前期w值较小, 保证了最差的K个个体中大多数个体以其他随机个体为支点向周围进行反射, 增强了全局的探索能力, 保证了种群的多样性. 而随着迭代次数的增加, w值逐渐增大, 适应度值最差的K个个体中越来越多的个体以当前找到的最优解为支点向周围进行反射, 以此来增强这些个体在最优解周围的局部挖掘能力, 同时因为个体是以最优解为支点向周围进行反射探索, 也有助于在迭代后期种群仍有跳出局部最优的机会和能力.

$$\boldsymbol{X}\_\boldsymbol{\mathrm{new}}_i(t+1) = \lambda_3 \cdot \boldsymbol{X}_i(t) + F \cdot (\lambda_1 \cdot \boldsymbol{X}_{\mathrm{best}}(t) - \lambda_2 \cdot \boldsymbol{X}_i(t)) \tag{15}$$

$$\boldsymbol{X}\_\boldsymbol{\mathrm{new}}_i(t+1) = \lambda_3 \cdot \boldsymbol{X}_i(t) + F \cdot (\lambda_1 \cdot \boldsymbol{X}_v(t) - \lambda_2 \cdot \boldsymbol{X}_i(t)) \tag{16}$$

$$F = (F_{\max} - F_{\min}) \times (t/T_{\max}) + F_{\min} \tag{17}$$

$$\lambda_1(t) = \frac{s}{4}\sin(\pi \cdot \lambda_3(t-1)),\ \lambda_2(t) = \frac{s}{4}\sin(\pi \cdot \lambda_1(t)),$$
$$\lambda_3(t) = \frac{s}{4}\sin(\pi \cdot \lambda_2(t)) \tag{18}$$

$$w = \mathrm{round}(w_{\min} + (t/T_{\max})(K - w_{\min})) \tag{19}$$

其中, t表示当前迭代次数; $T_{\max}$表示最大迭代次数; $\boldsymbol{X}_\boldsymbol{\mathrm{new}}_i(t+1)$是由最优解反射得到的新位置; $\boldsymbol{X}_{\mathrm{best}}(t)$是当前最优解的位置; $\boldsymbol{X}_i(t)$是当前第i个个体位置; $\boldsymbol{X}_v(t)$是当前随机第$v(v \neq i)$个个体位置; F为自适应相对反射因子, $F_{\max} = 3$和$F_{\min} = 1$为相对反射因子的初始值和最终值; λ_1、λ_2和λ_3是由 Sine 混沌映射生成的 [0, 1] 区间内的随机数, s 为 Sine 映射的系数, 这里取值为 4; K表示选取适应度值最差的个体数, 经反复测试, 算法中取值为种群个体数的 1/10; w表示K个个体中以最优解为支点的个体数; $w_{\min}$为w的最小值, 算法中取值为$K/2$, 此时寻优效果最佳.

上述自适应相对反射策略的代码段如以下Code 2所示

FOR $i = 1 : N$

计算猎物种群中个体的适应度值, 比较并记录最优个体

END FOR

对猎物种群进行降序排序, 并确定K个适应度值最差的个体

由公式(17)计算自适应相对反射因子F

由公式(18)生成随机数λ_1、λ_2和λ_3

由公式(19)计算自适应个体数w

FOR $i = 1 : K$

IF $i < w$

由公式(15)更新猎物种群中个体位置

ELSE

由公式(16)更新猎物种群中个体位置

END IF

END FOR

LLAMPA算法流程

设置算法各参数

初始化猎物种群$\boldsymbol{X}_i^j (i = 1,2,\cdots,N,\ j = 1,2,\cdots,D)$

$t = 0$

WHILE ($t <$ Max_t)

FOR $i = 1 : N$

对猎物种群中的个体进行边界处理

计算猎物种群中个体的适应度值, 比较并记录最优个体

END FOR

对猎物种群中的N个个体执行代码段Code1所示的海洋记忆存储功能.

由最优个体复制构造捕食者种群

由公式(6)计算自适应参数CF

FOR $i = 1 : N$

IF $t < 1/3 \cdot$ Max_t

由公式(2)更新猎物种群中个体位置

ELSE IF $t < 2/3 \cdot$ Max_t

IF $i < 1/2 \cdot N$

由公式(3)更新猎物种群中个体位置

ELSE

由公式(4)更新猎物种群中个体位置

END IF

ELSE

由公式(13)计算dist

由公式(14)更新猎物种群中个体位置

END IF

END FOR

FOR $i = 1 : N$

对更新后猎物种群中的个体进行边界处理

计算猎物种群中个体的适应度值, 比较并记录最优个体

END FOR

对猎物种群中的N个个体执行代码段Code1所示的海洋记忆存储功能.

FOR $i = 1 : N$

由公式(7)对猎物种群中N个个体进行位置更新

END FOR

执行代码段Code 2所示的自适应相对反射策略.

$t = t + 1$

END WHILE

输出结果

3 仿真实验

为了全面检验本文改进算法 LLAMPA 的整体性能, 选取具有挑战性的 CEC2017 测试函数集套件[17], 将 LLAMPA 算法与基本海洋捕食者算法 (MPA, 2020)[1], 两种新颖优秀的 MPA 改进算法 Enhanced marine predators algorithm with local escaping operator (LEO-MPA, 2021)[12]、Modified self-adaptive marine predators algorithm (MMPA, 2021)[15], CEC 2017 的优胜算法之一 Covariance matrix learning with Euclidean neighborhood is used for LSHADE-EpSin (LSHADE-cnEpSin, 2017)[23], CEC2022 优胜算法之一 Adaptive differential evolution algorithm (NL-SHADE-LBC, 2022)[24], 以及在数值优化与工程约束优化问题上求解性能优越的 Improved grey wolf optimizer (I-GWO, 2020)[25], 共 7 种算法在 100 维进行对比测试.

3.1 测试函数与参数设置

CEC2017 测试集函数分为四类共 30 个函数, 即$f_1(x)$ ~ $f_3(x)$为单峰函数, $f_4(x)$ ~ $f_{10}(x)$为多峰函数, $f_{11}(x)$ ~ $f_{20}(x)$为混合函数, $f_{21}(x)$ ~ $f_{30}(x)$为组合函数. 测试套件中对这 30 个函数都加入了旋转和偏移, 使函数变得更具复杂性, 大大提高了寻求全局最优解的难度. CEC2017 测试集具体函数如文献 [17] 所示, 由于套件中各函数分别使用了 +100 ~ +3000 的偏移操作, 因此它们的理论最优值均不再为 0, 而是从 100 ~ 3000. 基于算法性能比较的公平性原则, 7 种算法采用相同的软、硬件平台, 运行环境为 Windows11 操作系统、CPU 为 Intel i5-11400H、测试平台为 Matlab R2019b. 仿真实验中, 7 种算法的种群规模N、空间维度dim、最大迭代次数Max_iter和运行次数都保持一致, 即$N = 50$, dim = 100, Max_iter = 1000, 每种算法独立运行 50 次. 7 种算法的参数设置在表 1 中列出, 其中: 在 I-GWO

中, a为线性参数; 在 LSHADE-cnEpSin 算法中, μF、μCR 为控制参数, ps、pc 为协方差矩阵参数, H为记忆参数; 在 NL-SHADE-LBC 算法中, $M_{\mathrm{F,r}}$、$M_{\mathrm{Cr,r}}$为交叉变异参数, k为存储器索引参数, n_{A}为位置更新参数.

表 1　7 种算法的参数设置

Table 1　Parameter settings of the 7 algorithms

Algorithm	Parameter setting
LLAMPA	P = 0.5, FADs = 0.2, RS = 0.6, PB = 0.6, c = 1, $F_{\min}$ = 1, $F_{\max}$ = 3, $k = N/10$, $l_{\min} = k/2$
I-GWO	a linear decline from 2 to 0
MPA	P = 0.5, FADs = 0.2
LEO-MPA	P = 0.5, FADs = 0.2, $k = N/10$
MMPA	P = 0.5, FADs = 0.2
LSHADE-cnEpSin	μF = 0.5, μCR = 0.5, H = 5, ps = 0.5, pc = 0.4
NL-SHADE-LBC	$M_{\mathrm{F,r}}$ = 0.5, $M_{\mathrm{Cr,r}}$ = 0.9, k = 1, n_{A} = 0.5

3.2　参数敏感度分析

学习自动机中的奖励步长RS是 LLAMPA 算法的重要参数之一, 其取值会对算法在优化过程中对学习策略的选择产生一定影响. 为了分析不同步长取值对算法性能的影响, 本节进行了参数敏感性仿真实验. 表 2 展示了 Dim=100 维时, LLAMPA 算法在不同奖励步长取值下独立运行 50 次求解 CEC2017 测试集函数得到的平均值对应的排名情况. 为了更直观地展示实验结果, 根据表 2 中求解得到的不同步长参数值对应的 Friedman 平均排名绘制了柱状图, 如图 1 所示. 在本节中我们仅选取了奖励步长参数RS作为示例, 对于惩罚步长PB的确定方法与之类似, 不再赘述.

从表 2 可以看到, 所提出的改进算法 LLAMPA 相较于基本 MPA 算法在绝大多数测试函数上的寻优结果更优. 同时, 根据不同奖励步长的平均排名, 可以看出不同的步长取值对 LLAMPA 算法的性能有一定影响, 而当奖励步长RS = 0.6时, LLAMPA 算法的平均排名为 3.14, 排在第一位. 此外, 通过图 1 的柱状图也可以更为直观地观察到, 当奖励步长RS = 0.6时, 其柱体明显低于其他步长参数值. 这表明当奖励步长取值为 0.6 时, LLAMPA 算法在大多数函数上的优化效果更佳, 能够更大程度地提升改进算法的寻优精度. 因此, 通过实验结果分析, 本文将奖励步长的取值确定为 0.6.

3.3　寻优精度分析

表 3 分别统计了 LLAMPA、MPA、MMPA、LEO-MPA、I-GWO、LSHADE-cnEpSin 和NL-SHADE-LBC 这 7 种算法在空间维度 100 维的情况下, 各自独立运行 50 次对 CEC2017 测试集求解得到的最佳值、平均值和最差值, 并将各函数上求得的最佳结果进行了加粗标注. 本文对 CEC2017 测试集中 30 个函数都进行了测试和分析, 因篇幅原因下面只讨论按序给出的单峰函数 $f_1(x)$, 多峰函数 $f_9(x)$、$f_{10}(x)$, 混合测试函数 $f_{14}(x)$ ~ $f_{16}(x)$, 组合测试函数 $f_{23}(x)$、$f_{24}(x)$和 $f_{26}(x)$、$f_{27}(x)$, 其他函数的测试结果与之类似, 不再一一赘述.

由表 3 可知, 在 100 维条件下, 对于函数 $f_1(x)$、$f_9(x)$、$f_{10}(x)$、$f_{14}(x)$ ~ $f_{16}(x)$、$f_{23}(x)$、$f_{24}(x)$和 $f_{27}(x)$, LLAMPA 在这 9 个函数上的最佳值、平均值、最差值均为 7 种算法中最优, 表现出十分优秀的维度适应性以及相当出色的高维求解能力. 而对于函数 $f_{26}(x)$, LLAMPA 的最佳值、平均值也均为 7 种算法中最优, 最差值则仅次于 LEO-MPA.

由以上寻优结果分析可知, 相比于其他 6 种代表性对比算法, 本文改进算法 LLAMPA 表现出较好的求解性能, 在 CEC 复杂函数上的整体寻优结果显著更优.

3.4　关于迭代终止条件的讨论分析——以适应度评估最大次数为终止条件的精度分析

本节测试以适应度评估最大次数(Fes)作为终止条件时, 对比 7 种算法(同 3.3 节)求解 CEC2017 测试集的优化结果, 并与 3.3 节以最大迭代次数作为终止条件的求解结果进行比较, 证明了这两种方式在本文所采用的对比算法中进行求解比较的结果是相对一致的. 为了保持实验的公平性和客观性原则, 各算法中除了迭代终止条件均设置为 FEs=200000 外, 其他参数和实验环境均与 3.3 节保持一致. 表 4 给出了这 7 种算法在不同维度下分别独立运行 50 次求解 CEC2017 各函数的平均值、最佳值和最差值, 最好的结果以粗体展示.

由表 4 可知, 在 100 维的高维条件下, 对于函数 $f_9(x)$、$f_{10}(x)$、$f_{14}(x)$ ~ $f_{16}(x)$、$f_{23}(x)$、$f_{24}(x)$和 $f_{27}(x)$, LLAMPA 在这 8 个函数上的最佳值、平均值以及最差值均为 7 种算法中最优. 对于剩余两个函数, 在函数 $f_1(x)$上, LLAMPA 的最佳值、平均值和最差值仅次于 LSHADE-cnEpSin, 优于剩余全部对比算法. 在函数 $f_{26}(x)$上, LLAMPA 的最佳值和平均值为 7 种算法中最好的, 最差值则优于部分对比算法.

通过观察表 3 和表 4 的数据并结合各自的分析, 可以得出以下结论: 虽然使用最大迭代次数和适应度评估最大次数作为迭代终止条件运行得到

表 2 不同步长参数值对 LLAMPA 算法优化结果的影响(Dim=100)

Table 2 Influence of different step size parameters on the optimization results of LLAMPA (Dim = 100)

Algorithm function	Rank									
	MPA	LLAMPA								
		RS = 0.1	RS = 0.2	RS = 0.3	RS = 0.4	RS = 0.5	RS = 0.6	RS = 0.7	RS = 0.8	RS = 0.9
f_1	10	7	6	9	5	8	3	2	1	4
f_3	10	8	1	9	2	4	5	6	7	3
f_4	10	2	1	7	4	8	3	5	6	9
f_5	10	9	7	4	5	2	1	3	8	6
f_6	10	4	5	6	1	2	3	7	8	9
f_7	10	4	8	5	1	3	6	2	9	7
f_8	10	1	2	8	4	5	3	6	9	7
f_9	10	5	8	4	6	3	2	7	9	1
f_{10}	10	7	9	2	5	3	4	8	1	6
f_{11}	10	4	3	1	5	7	8	9	2	6
f_{12}	10	6	9	3	5	7	2	8	4	1
f_{13}	10	2	5	4	1	6	3	8	7	9
f_{14}	3	5	2	9	7	8	6	1	10	4
f_{15}	10	3	9	5	1	4	2	8	7	6
f_{16}	10	2	5	6	7	4	8	3	9	1
f_{17}	10	4	8	6	9	5	3	2	7	1
f_{18}	1	4	8	2	5	6	9	7	10	3
f_{19}	10	4	1	8	6	2	3	9	5	7
f_{20}	10	8	3	9	5	2	1	6	4	7
f_{21}	10	3	4	8	5	9	1	6	2	7
f_{22}	10	9	5	1	2	7	3	8	6	4
f_{23}	10	6	7	2	9	3	4	8	5	1
f_{24}	10	2	9	3	7	8	1	4	5	6
f_{25}	10	2	3	4	9	6	1	5	7	8
f_{26}	10	1	8	9	5	7	3	2	6	4
f_{27}	10	6	1	2	7	4	3	8	9	5
f_{28}	10	5	3	6	4	7	1	9	8	2
f_{29}	10	8	3	5	4	9	1	6	2	7
f_{30}	10	3	8	4	9	2	1	5	7	6
Average ranking	9.45	5.05	5.36	5.41	5.21	5.22	3.14	5.59	5.91	4.66

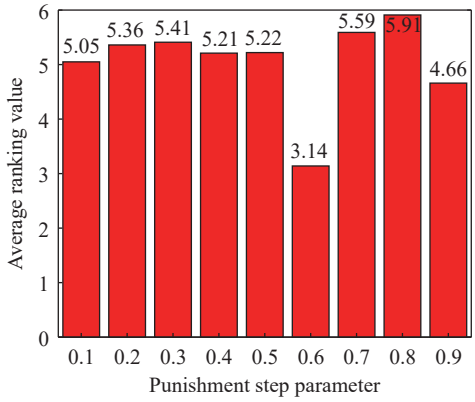

图 1 不同步长参数的平均排名

Fig.1 Average ranking of different step size parameters

的数据有些许差异,但总体差异并没有影响算法优劣的对比结论. 这说明在本文所采用的各算法中使用最大迭代次数或适应度评估最大次数作为迭代终止条件时,算法的运行结果相对稳定且一致. 尽管具体的数据可能存在一些差异,但整体上可以认为这两种迭代终止方式在本文所提出和使用的算法求解和对比中都是有效且基本等价的.

3.5 实验结果的秩和检验统计分析

非参数统计检验方法——Wilcoxon 秩和检验[26]可以统计分析两种算法之间的差异是否具有显著性,这种方法通常用于进一步评估改进算法的优

表 3 7 种算法在 100 维测试函数上的求解结果

Table 3 Results of 7 algorithms on the test function for100 dimensions

Function	Algorithm	100-dim			Function	Algorithm	100-dim		
		Best	Avg	Worst			Best	Avg	Worst
	LLAMPA	**1.54×10^5**	**8.15×10^5**	**3.96×10^6**		LLAMPA	**2.82×10^3**	**4.52×10^3**	**5.46×10^3**
	MPA	5.67×10^7	2.12×10^8	4.90×10^8		MPA	4.12×10^3	5.41×10^3	6.65×10^3
	MMPA	3.91×10^7	1.71×10^8	9.09×10^8		MMPA	3.60×10^3	5.41×10^3	6.71×10^3
f_1	LEO-MPA	1.99×10^7	6.25×10^7	1.92×10^8	f_{16}	LEO-MPA	3.94×10^3	5.26×10^3	6.18×10^3
	LSHADE-cnEpSin	2.74×10^6	2.00×10^7	6.30×10^7		LSHADE-cnEpSin	3.05×10^3	4.94×10^3	6.92×10^3
	I-GWO	1.94×10^9	9.99×10^9	2.59×10^{10}		I-GWO	3.25×10^3	4.93×10^3	1.02×10^4
	NL-SHADE-LBC	4.95×10^9	1.06×10^{10}	2.34×10^{10}		NL-SHADE-LBC	3.90×10^3	5.18×10^3	6.39×10^3
	LLAMPA	**3.06×10^3**	**7.52×10^3**	**1.95×10^4**		LLAMPA	**3.01×10^3**	**3.09×10^3**	**3.16×10^3**
	MPA	9.41×10^3	1.96×10^4	2.65×10^4		MPA	3.10×10^3	3.22×10^3	3.31×10^3
	MMPA	7.49×10^3	2.22×10^4	2.74×10^4		MMPA	3.06×10^3	3.16×10^3	3.28×10^3
f_9	LEO-MPA	3.86×10^3	7.76×10^3	1.96×10^4	f_{23}	LEO-MPA	3.05×10^3	3.14×10^3	3.24×10^3
	LSHADE-cnEpSin	7.61×10^3	1.86×10^4	4.24×10^4		LSHADE-cnEpSin	3.22×10^3	3.38×10^3	3.61×10^3
	I-GWO	5.33×10^3	1.90×10^4	4.21×10^4		I-GWO	3.11×10^3	3.23×10^3	3.72×10^3
	NL-SHADE-LBC	2.01×10^4	4.24×10^4	5.64×10^4		NL-SHADE-LBC	3.59×10^3	3.83×10^3	4.17×10^3
	LLAMPA	**1.14×10^4**	**1.36×10^4**	**1.55×10^4**		LLAMPA	**3.51×10^3**	**3.60×10^3**	**3.78×10^3**
	MPA	1.16×10^4	1.46×10^4	1.80×10^4		MPA	3.63×10^3	3.77×10^3	3.90×10^3
	MMPA	1.28×10^4	1.53×10^4	1.79×10^4		MMPA	3.61×10^3	3.74×10^3	3.87×10^3
f_{10}	LEO-MPA	1.24×10^4	1.46×10^4	1.64×10^4	f_{24}	LEO-MPA	3.56×10^3	3.66×10^3	3.86×10^3
	LSHADE-cnEpSin	1.78×10^4	1.98×10^4	3.15×10^4		LSHADE-cnEpSin	3.68×10^3	3.96×10^3	4.35×10^3
	I-GWO	1.39×10^4	2.66×10^4	3.26×10^4		I-GWO	3.53×10^3	3.72×10^3	4.29×10^3
	NL-SHADE-LBC	2.16×10^4	2.67×10^4	3.04×10^4		NL-SHADE-LBC	4.42×10^3	4.84×10^3	5.38×10^3
	LLAMPA	**1.92×10^3**	**2.27×10^3**	**3.40×10^3**		LLAMPA	**3.99×10^3**	**9.24×10^3**	1.88×10^4
	MPA	1.99×10^3	2.31×10^3	4.49×10^3		MPA	1.00×10^4	1.11×10^4	1.31×10^4
	MMPA	2.02×10^3	2.58×10^3	1.36×10^4		MMPA	5.86×10^3	1.13×10^4	2.39×10^4
f_{14}	LEO-MPA	1.99×10^3	2.37×10^3	4.02×10^3	f_{26}	LEO-MPA	8.89×10^3	9.72×10^3	**1.09×10^4**
	LSHADE-cnEpSin	5.34×10^3	5.30×10^4	1.40×10^5		LSHADE-cnEpSin	9.58×10^3	1.18×10^4	1.54×10^4
	I-GWO	2.50×10^5	1.33×10^6	3.01×10^6		I-GWO	8.96×10^3	1.09×10^4	1.29×10^4
	NL-SHADE-LBC	1.31×10^5	6.97×10^5	1.33×10^6		NL-SHADE-LBC	8.36×10^3	2.01×10^4	2.47×10^4
	LLAMPA	**1.79×10^3**	**3.57×10^3**	**1.14×10^4**		LLAMPA	**3.33×10^3**	**3.42×10^3**	**3.54×10^3**
	MPA	1.15×10^4	2.19×10^4	4.05×10^4		MPA	3.46×10^3	3.57×10^3	3.72×10^3
	MMPA	1.03×10^4	2.56×10^4	7.36×10^4		MMPA	3.47×10^3	3.66×10^3	3.94×10^3
f_{15}	LEO-MPA	3.51×10^3	8.71×10^3	2.25×10^4	f_{27}	LEO-MPA	3.43×10^3	3.52×10^3	3.71×10^3
	LSHADE-cnEpSin	3.79×10^3	7.99×10^3	3.24×10^4		LSHADE-cnEpSin	3.47×10^3	3.70×10^3	3.94×10^3
	I-GWO	5.23×10^4	1.13×10^5	2.27×10^5		I-GWO	3.41×10^3	3.51×10^3	3.64×10^3
	NL-SHADE-LBC	3.31×10^3	1.17×10^4	2.74×10^4		NL-SHADE-LBC	3.96×10^3	4.41×10^3	4.88×10^3

化性能. 因此为了验证 LLAMPA 的优化结果是否明显优于其他 6 种对比算法, 采用 Wilcoxon 秩和检验进行统计分析.

表 5 为 CEC2017 测试套件 100 维条件下, 对应 3.3 节中的 10 个代表性函数, 将 LLAMPA 与其他各对比算法进行 Wilcoxon 秩和检验的统计结果, 其中“+”、“−”分别表示 LLAMPA 的寻优结果显著优于、劣于对比算法, 而“=”表示 LLAMPA 与

表 4 7 种算法在 100 维测试函数上的求解结果

Table 4 Results of 7 algorithms on the test function for 100 dimensions

Function	Algorithm	100-dim			Function	Algorithm	100-dim		
		Best	Avg	Worst			Best	Avg	Worst
	LLAMPA	1.17×10^{6}	6.99×10^{5}	5.71×10^{6}		LLAMPA	$\mathbf{2.82 \times 10^{3}}$	$\mathbf{3.97 \times 10^{3}}$	$\mathbf{5.01 \times 10^{3}}$
	MPA	6.86×10^{6}	1.46×10^{7}	3.48×10^{7}		MPA	3.57×10^{3}	4.86×10^{3}	5.69×10^{3}
	MMPA	7.85×10^{6}	3.53×10^{7}	6.81×10^{7}		MMPA	3.44×10^{3}	4.83×10^{3}	5.72×10^{3}
f_1	LEO-MPA	1.03×10^{7}	2.14×10^{7}	4.78×10^{7}	f_{16}	LEO-MPA	3.37×10^{3}	5.01×10^{3}	5.92×10^{3}
	LSHADE-cnEpSin	$\mathbf{1.89 \times 10^{2}}$	$\mathbf{1.01 \times 10^{4}}$	$\mathbf{4.11 \times 10^{4}}$		LSHADE-cnEpSin	2.95×10^{3}	3.99×10^{3}	5.02×10^{3}
	I-GWO	2.21×10^{7}	1.48×10^{8}	1.03×10^{9}		I-GWO	3.06×10^{3}	4.48×10^{3}	5.63×10^{3}
	NL-SHADE-LBC	1.18×10^{6}	5.11×10^{11}	6.27×10^{11}		NL-SHADE-LBC	2.44×10^{4}	3.58×10^{4}	4.48×10^{4}
	LLAMPA	$\mathbf{2.67 \times 10^{3}}$	$\mathbf{7.17 \times 10^{3}}$	$\mathbf{1.51 \times 10^{4}}$		LLAMPA	$\mathbf{3.02 \times 10^{3}}$	$\mathbf{3.10 \times 10^{3}}$	$\mathbf{3.20 \times 10^{3}}$
	MPA	8.39×10^{3}	1.54×10^{4}	2.39×10^{4}		MPA	3.06×10^{3}	3.14×10^{3}	3.26×10^{3}
	MMPA	5.54×10^{3}	7.33×10^{3}	1.65×10^{4}		MMPA	3.03×10^{3}	$\mathbf{3.10 \times 10^{3}}$	3.23×10^{3}
f_9	LEO-MPA	3.40×10^{3}	7.49×10^{3}	1.58×10^{4}	f_{23}	LEO-MPA	3.03×10^{3}	3.11×10^{3}	3.24×10^{3}
	LSHADE-cnEpSin	5.45×10^{3}	8.32×10^{3}	1.18×10^{4}		LSHADE-cnEpSin	3.12×10^{3}	3.25×10^{3}	3.48×10^{3}
	I-GWO	3.06×10^{3}	1.20×10^{4}	2.84×10^{4}		I-GWO	3.08×10^{3}	3.15×10^{3}	3.24×10^{3}
	NL-SHADE-LBC	1.66×10^{5}	1.99×10^{5}	2.28×10^{5}		NL-SHADE-LBC	6.85×10^{3}	8.34×10^{3}	9.73×10^{3}
	LLAMPA	$\mathbf{1.04 \times 10^{4}}$	$\mathbf{1.15 \times 10^{4}}$	$\mathbf{1.38 \times 10^{4}}$		LLAMPA	$\mathbf{3.50 \times 10^{3}}$	$\mathbf{3.60 \times 10^{3}}$	$\mathbf{3.64 \times 10^{3}}$
	MPA	1.05×10^{4}	1.37×10^{4}	1.62×10^{4}		MPA	3.55×10^{3}	3.66×10^{3}	3.85×10^{3}
	MMPA	1.06×10^{4}	1.41×10^{4}	1.89×10^{4}		MMPA	3.54×10^{3}	3.62×10^{3}	3.82×10^{3}
	LEO-MPA	1.15×10^{4}	1.40×10^{4}	1.58×10^{4}		LEO-MPA	3.52×10^{3}	3.64×10^{3}	3.78×10^{3}
f_{10}	LSHADE-cnEpSin	1.07×10^{4}	1.19×10^{4}	1.43×10^{4}	f_{24}	LSHADE-cnEpSin	3.69×10^{3}	3.96×10^{3}	4.31×10^{3}
	I-GWO	1.06×10^{4}	2.32×10^{4}	3.19×10^{4}		I-GWO	3.53×10^{3}	3.61×10^{3}	3.72×10^{3}
	NL-SHADE-LBC	3.44×10^{4}	3.60×10^{4}	3.78×10^{4}		NL-SHADE-LBC	1.18×10^{4}	1.41×10^{4}	1.63×10^{4}
	LLAMPA	$\mathbf{1.83 \times 10^{3}}$	$\mathbf{2.02 \times 10^{3}}$	$\mathbf{2.34 \times 10^{3}}$		LLAMPA	$\mathbf{4.28 \times 10^{3}}$	$\mathbf{9.09 \times 10^{3}}$	1.84×10^{4}
	MPA	1.99×10^{3}	2.29×10^{3}	2.37×10^{3}		MPA	9.27×10^{3}	1.07×10^{4}	$\mathbf{1.21 \times 10^{4}}$
	MMPA	1.86×10^{3}	2.51×10^{3}	2.56×10^{3}		MMPA	4.64×10^{3}	1.08×10^{4}	2.05×10^{4}
	LEO-MPA	1.87×10^{3}	2.05×10^{3}	2.49×10^{3}		LEO-MPA	8.56×10^{3}	9.57×10^{3}	1.08×10^{4}
f_{14}	LSHADE-cnEpSin	3.80×10^{3}	2.10×10^{4}	6.27×10^{4}	f_{26}	LSHADE-cnEpSin	9.62×10^{3}	1.18×10^{4}	1.49×10^{4}
	I-GWO	3.16×10^{5}	9.44×10^{5}	1.99×10^{6}		I-GWO	8.38×10^{3}	9.91×10^{3}	1.21×10^{4}
	NL-SHADE-LBC	2.56×10^{8}	6.54×10^{8}	1.29×10^{9}		NL-SHADE-LBC	6.92×10^{4}	8.71×10^{4}	1.10×10^{5}
	LLAMPA	$\mathbf{1.74 \times 10^{3}}$	$\mathbf{3.30 \times 10^{3}}$	$\mathbf{1.38 \times 10^{4}}$		LLAMPA	$\mathbf{3.26 \times 10^{3}}$	$\mathbf{3.41 \times 10^{3}}$	$\mathbf{3.60 \times 10^{3}}$
	MPA	3.08×10^{3}	5.03×10^{3}	2.15×10^{4}		MPA	3.35×10^{3}	3.48×10^{3}	3.63×10^{3}
	MMPA	1.93×10^{3}	3.88×10^{3}	2.10×10^{4}		MMPA	3.27×10^{3}	3.44×10^{3}	3.65×10^{3}
f_{15}	LEO-MPA	2.66×10^{3}	6.01×10^{3}	2.53×10^{4}	f_{27}	LEO-MPA	3.39×10^{3}	3.48×10^{3}	3.61×10^{3}
	LSHADE-cnEpSin	2.02×10^{3}	4.15×10^{3}	2.06×10^{4}		LSHADE-cnEpSin	3.53×10^{3}	3.74×10^{3}	4.08×10^{3}
	I-GWO	7.76×10^{3}	1.77×10^{4}	4.61×10^{4}		I-GWO	3.33×10^{3}	3.45×10^{3}	3.62×10^{3}
	NL-SHADE-LBC	2.29×10^{10}	3.95×10^{10}	6.00×10^{10}		NL-SHADE-LBC	1.33×10^{4}	1.73×10^{4}	2.05×10^{4}

对比算法的寻优结果相当或没有显著性差异. 由文献 [27] 可知, p小于 0.05 的结果可以认为是拒绝零假设, 是具有显著性差异的有力验证.

根据表 5 的统计检验结果, 本文改进算法 LLAMPA 除了在少数几个函数上与个别算法求解结果之间没有显著性差异, 在剩余绝大多数函数上的检验 p值均远远小于 0.05, 且符号几乎都为“+”. 因此, LLAMPA 算法相对于其他 6 种对比算

表 5 LLAMPA 与各对比算法之间 Wilcoxon 秩和检验的 p 值

Table 5 p-values of the Wilcoxon rank sum test between LLAMPA and each comparison algorithm

Function	p-value					
	LLAMPA *vs* MPA	LLAMPA *vs* MMPA	LLAMPA *vs* LEO-MPA	LLAMPA *vs* I-GWO	LLAMPA *vs* LSHADE-cnEpSin	LLAMPA *vs* NL-SHADE-LBC
f_1	7.07×10^{-18}+	7.07×10^{-18}+	7.07×10^{-18}+	7.07×10^{-18}+	7.50×10^{-18}+	7.07×10^{-18}+
f_9	1.13×10^{-16}+	8.99×10^{-17}+	6.12×10^{-1}=	3.77×10^{-12}+	1.10×10^{-13}+	7.07×10^{-18}+
f_{10}	6.02×10^{-6}+	2.53×10^{-10}+	2.05×10^{-7}+	6.72×10^{-17}+	7.07×10^{-18}+	7.07×10^{-18}+
f_{14}	9.81×10^{-1}=	1.60×10^{-2}+	3.90×10^{-2}+	7.07×10^{-18}+	7.07×10^{-18}+	7.07×10^{-18}+
f_{15}	7.07×10^{-18}+	7.50×10^{-18}+	1.27×10^{-12}+	7.07×10^{-18}+	3.56×10^{-11}+	2.17×10^{-15}+
f_{16}	3.77×10^{-12}+	1.69×10^{-10}+	5.66×10^{-11}+	6.22×10^{-2}=	8.17×10^{-4}+	4.32×10^{-8}+
f_{23}	5.98×10^{-17}+	2.95×10^{-12}+	3.56×10^{-8}+	7.12×10^{-17}+	7.07×10^{-18}+	7.07×10^{-18}+
f_{24}	4.76×10^{-16}+	1.11×10^{-15}+	2.21×10^{-7}+	6.20×10^{-11}+	8.99×10^{-18}+	7.07×10^{-18}+
f_{26}	2.95×10^{-11}+	6.52×10^{-4}+	3.40×10^{-2}+	2.05×10^{-9}+	6.00×10^{-13}+	1.51×10^{-16}+
f_{27}	2.27×10^{-16}+	1.54×10^{-17}+	3.26×10^{-12}+	3.09×10^{-13}+	1.84×10^{-17}+	7.07×10^{-18}+
+/=/−	**9/1/0**	**10/0/0**	**9/1/0**	**9/1/0**	**10/0/0**	**10/0/0**

法的优化结果具有显著性差异, 且 LLAMPA 的寻优结果显著更优.

3.6 箱线图分析

箱线图是一种用于显示数据分布情况的统计方法, 利用最大值、最小值、中位数以及上、下四分位数这 5 个统计量来描述数据的分布和性质. 箱线图由箱子和箱子上下两端的两条细线组成, 箱子的高度代表上、下四分位的间距, 箱子中的横线代表中位数, 上下两端的细线代表最大值和最小值. 由于 CEC2017 测试套件中函数的求解结果往往容易受到局部最小值的干扰, 因此为了更好地了解 LLAMPA 和各对比算法对函数求解结果的数据分布, 采用箱线图进行统计分析, 如图 2 所示. 图 2 是根据 LLAMPA、MPA、MMPA、LEO-MPA、LSHADE-cnEpSin、IGWO 以及 NL-SHADE-LBC 这 7 种算法在 100 维条件下独立运行 50 次, 对 3.3 节中 10 个代表性函数求解得到的寻优结果所绘制的箱线图.

由图 2 可以看出, 除 $f_9(x)$、$f_{26}(x)$函数以外, LLAMPA 在其余 8 个函数上的箱线图相较于 MPA、MMPA、LEO-MPA、LSHADE-cnEpSin、NL-SHADE-LBC 以及 IGWO 这 6 种对比算法, 其箱体更窄, 说明对于绝大多数函数, LLAMPA 求解结果的数据分布更为集中, 波动程度更小, 体现了 LLAMPA 算法较为优越的稳定性. 而对于 $f_9(x)$、$f_{26}(x)$这两个函数, 虽然 LLAMPA 的箱线图中箱体长度大于部分算法, 求解的函数结果分布范围较宽, 但其中位数却低于其他对比算法, 说明 LLAMPA 算法的寻优结果在一般水平上仍优于另外 6 种对比算法.

4 工程设计优化问题

为了进一步检验本文改进算法 LLAMPA 的寻优性能, 验证其求解工程设计优化问题的应用能力, 本节将上述对比算法用于求解 CEC2020 真实世界工程设计优化问题[18]. 为了使实验更具针对性和说服力, 将对比算法中的 CEC2017 优胜算法 LSHASDE-cnEpSin 替换为求解 CEC2020 真实世界工程约束优化问题的冠军算法 SASS[28]. 同时, 为了保证实验的公平和一致性, 将 SASS 算法的迭代中止条件也设置为与本文中所有对比算法一样的最大迭代次数. 然后将 LLAMPA 算法与包含 SASS 算法在内的 5 种对比算法在焊接梁设计、过程合成、热交换网络设计、工业制冷系统优化设计等 4 个工程设计约束优化问题上进行求解实验和比较分析, 采用文献 [18] 中所提到的平均违反约束(MV)和可行率(FR)这两个统计指标, 评估 LLAMPA 及各对比算法在求解工程约束问题时对约束条件的满足情况. 这 4 个工程约束问题具有各不相同的数学模型和约束条件, 可以很好地测试出各算法求解工程设计优化问题的能力和适用性. 在求解这 4 个工程优化问题时, 6 种算法的种群规模、算法独立运行次数、最大迭代次数均保持一致, 设为 50、50、1000.

4.1 焊接梁设计

焊接梁设计[29] 的目标是在考虑剪应力、弯曲应力、屈曲载荷、端部挠度和侧面约束情况下, 通

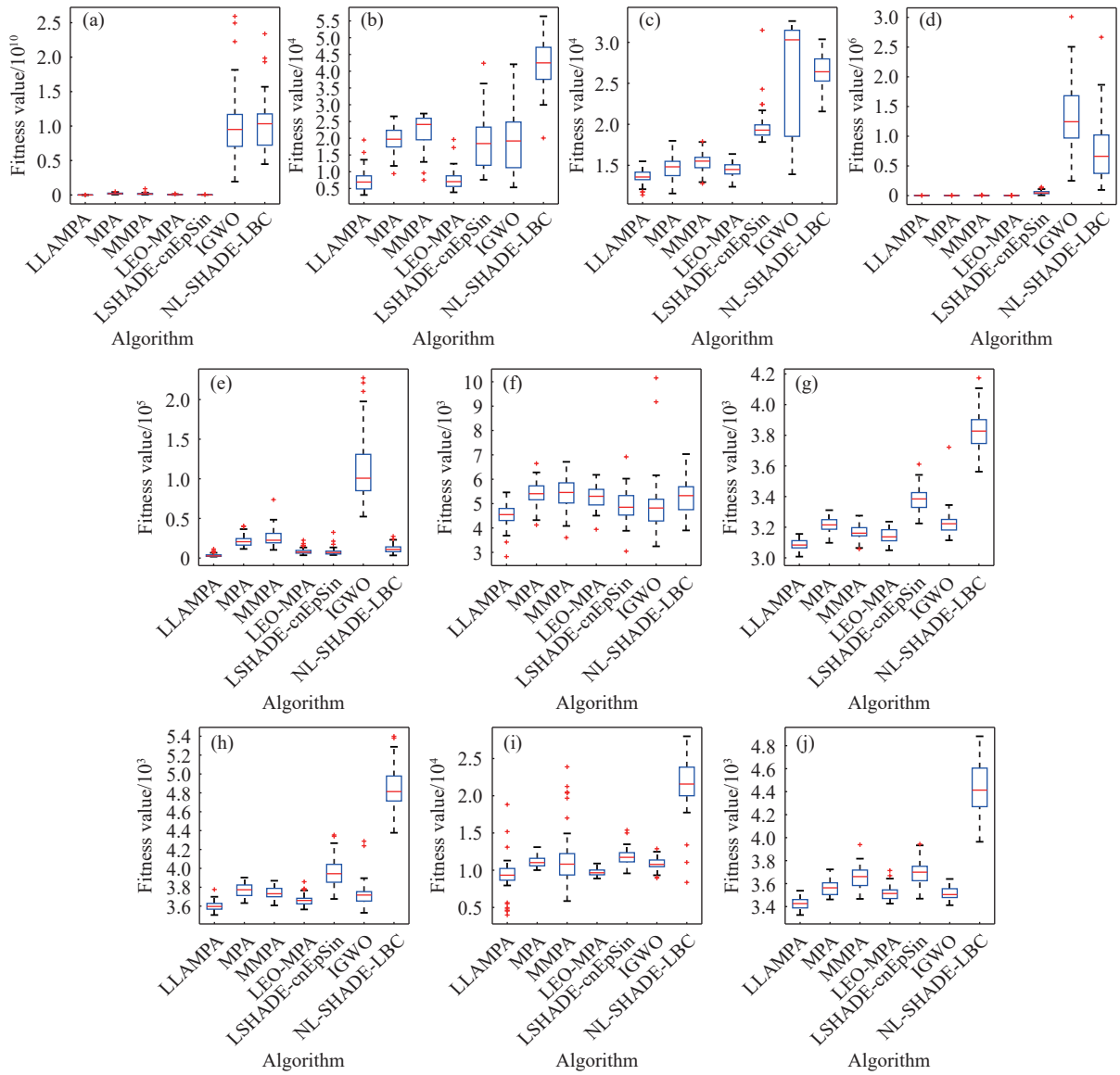

图 2 7 种算法寻优结果的箱线图. (a) $f_1(x)$; (b) $f_9(x)$; (c) $f_{10}(x)$; (d) $f_{14}(x)$; (e) $f_{15}(x)$; (f) $f_{16}(x)$; (g) $f_{23}(x)$; (h) $f_{24}(x)$; (i) $f_{26}(x)$; (j) $f_{27}(x)$

Fig.2 Boxplots of 7 algorithms' optimization results: (a) $f_1(x)$; (b) $f_9(x)$; (c) $f_{10}(x)$; (d) $f_{14}(x)$; (e) $f_{15}(x)$; (f) $f_{16}(x)$; (g) $f_{23}(x)$; (h) $f_{24}(x)$; (i) $f_{26}(x)$; (j) $f_{27}(x)$

过调整焊缝厚度 (tk)、夹子长度 (lg)、钢筋的高度 (hg) 和钢筋 (rb) 的厚度等决定变量, 找到制造成本的最低方案. 这个问题包含 4 个设计变量和 5 个约束条件, 数学模型如下:

设计变量: $x=\{x_1,x_2,x_3,x_4\}=\{\text{tk},\text{lg},\text{hg},\text{rb}\}$;

目标函数:

$$f(\bar{x})=0.04811x_3x_4(x_2+14)+1.10471x_1^2x_2;$$

约束条件:

$$g_1(\bar{x})=x_1-x_4\leqslant 0,\ \ g_2(\bar{x})=\delta(\bar{x})-0.25\leqslant 0,$$
$$g_3(\bar{x})=6000\leqslant P_c(\bar{x}),\ \ g_4(\bar{x})=13600\geqslant \tau(x),$$
$$g_5(\bar{x})=\sigma(\bar{x})-30000\leqslant 0.$$

其中: $\tau(x)=\sqrt{\tau'^2+\tau''^2+2\tau'\tau''\dfrac{x_2}{2R_m}}$ 为剪切应力,

$$\tau''=\frac{R_{\text{m}}M}{J},\ M=6000\left(\frac{x_2}{2}+14\right),\ \tau'=\frac{6000}{\sqrt{2}x_2x_1},$$

$$R_{\text{m}}=\sqrt{\frac{x_2^2}{4}+\left(\frac{x_1+x_3}{2}\right)^2},\ J=2\left(\left(\frac{x_2^2}{4}+\left(\frac{x_1+x_3}{2}\right)^2\right)\sqrt{2}x_1x_2\right),$$

$\sigma(\bar{x})=\dfrac{504000}{x_4x_3^2}$ 为悬臂梁弯曲应力,

$\delta(\bar{x})=\dfrac{65856000}{(30.10)^6x_3^3x_4}$ 为梁弯曲度,

$P_{\text{c}}(\bar{x})=\dfrac{4.013(30.10)^6x_3x_4^3}{6(14)^2}\left(1-\dfrac{x_3}{28}\sqrt{\dfrac{(30.10)^6}{4(12.10)^6}}\right)$ 为临界屈曲载荷, $0.1\leqslant x_3,x_2\leqslant 10$, $0.1\leqslant x_4\leqslant 2$, $0.125\leqslant x_1\leqslant 2$.

表 6 列出了解决焊接梁设计问题时 6 种算法各自独立运行 50 次得到的最佳值、平均值、方差 (Var)、平均违反约束 (MV) 和可行率 (FR).

由表 6 可知, LLAMPA 算法的最佳值、平均值和方差均为 6 种算法中最优, 同时最佳值达到了焊接梁设计问题的理论最优值 1.670217726, 说明 LLAMPA 算法在这一工程约束问题上有着突出的求解性能和稳定性.

4.2　过程合成问题

化学工程中过程合成问题[30]可以定义为混合整数非线性约束优化问题. 这个问题包含 7 个自由变量($x_1 \sim x_7$)和 9 个非线性约束条件($g_1 \sim g_9$). 数学模型如下:

目标函数:

$$f(\bar{x}) = (1-x_4)^2 + (1-x_5)^2 + (1-x_6)^2 - \ln(1+x_7) + (1-x_1)^2 + (2-x_2)^2 + (3-x_3)^2;$$

约束条件:

$$\begin{aligned}
&g_1(\bar{x}) = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 - 5 \leqslant 0,\\
&g_2(\bar{x}) = x_6^3 + x_1^2 + x_2^2 + x_3^2 = 5.5 \leqslant 0,\\
&g_3(\bar{x}) = x_1 + x_4 - 1.2 \leqslant 0,\\
&g_4(\bar{x}) = x_2 + x_5 - 1.8 \leqslant 0,\\
&g_5(\bar{x}) = x_3 + x_6 - 2.5 \leqslant 0,\\
&g_6(\bar{x}) = x_1 + x_7 - 1.2 \leqslant 0,\\
&g_7(\bar{x}) = x_5^2 + x_2^2 - 1.64 \leqslant 0,\\
&g_8(\bar{x}) = x_6^2 + x_3^2 - 4.25 \leqslant 0,\\
&g_9(\bar{x}) = x_5^2 + x_3^2 - 4.64 \leqslant 0;
\end{aligned}$$

边界条件: $0 \leqslant x_2, x_3, x_1 \leqslant 100$, $x_7, x_6, x_5, x_4 \in \{0, 1\}$.

表 7 列出了解决过程合成问题时 6 种算法各自独立运行 50 次得到的最佳值、平均值、方差、平均违反约束和可行率.

由表 7 可知, LLAMPA 算法运行 50 次得到的最佳值、方差和平均值均为 6 种算法中最优, 且最佳值达到了过程合成问题的理论最优值 2.924830554, 说明 LLAMPA 算法在解决过程合成问题上有着十分优秀的求解能力和稳定性.

4.3　热交换网络设计问题

热交换网络设计问题[31]的主要目的是设计热交换器的最佳构造, 使三股热流体加热一股冷流体, 并最大程度减小热交换器的传热面积. 这个问题包含 11 个设计变量($x_1 \sim x_{11}$)和 9 个约束条件($h_1 \sim h_9$), 设计变量包含热负荷 Q_1、Q_2、Q_3, 传热系数 U_1、U_2、U_3, 冷流体入口温度 T_{cl}, 三股热流体温度 T_{hl1}、T_{hl2}、T_{hl3}, 以及冷流体出口温度 T_{co}, 其数学模型如下:

设计变量:

$$x = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}\} = \{Q_1, Q_2, Q_3, U_1, U_2, U_3, T_{\mathrm{c1}}, T_{\mathrm{hl1}}, T_{\mathrm{hl2}}, T_{\mathrm{hl3}}, T_{\mathrm{co}}\};$$

目标函数:

$$f(\bar{x}) = \left(\frac{x_1}{120x_4}\right)^{0.6} + \left(\frac{x_2}{80x_5}\right)^{0.6} + \left(\frac{x_3}{40x_6}\right)^{0.6};$$

约束条件:

表 6　解决焊接梁设计问题时 6 种算法的寻优结果

Table 6　Optimization results of 6 algorithms for welded beam design problems

Algorithm	Best	Avg	Var	MV/%	FR/%
LLAMPA	**1.670217726**	**1.670217726**	**1.33×10^{-15}**	0.00	100
MPA	1.670217760	1.670217827	6.41×10^{-8}	0.00	100
MMPA	1.670217758	1.670217828	5.68×10^{-8}	0.00	100
LEO-MPA	1.670217729	1.670217743	1.27×10^{-8}	0.00	100
I-GWO	1.670281707	1.670404257	7.65×10^{-5}	0.00	100
SASS	1.67043865	1.67105273	4.39×10^{-4}	0.00	100

表 7　解决过程合成问题时 6 种算法的寻优结果

Table 7　Optimization results of 6 algorithms for the process synthesis problem

Algorithm	Best	Avg	Var	MV/%	FR/%
LLAMPA	**2.924830554**	**2.926158386**	**2.818625×10^{-5}**	0.00	100
MPA	2.924830857	3.125049886	1.949274×10^{-1}	0.00	100
MMPA	3.017707469	5.043021947	8.366092×10^{-1}	0.00	100
LEO-MPA	2.924830676	2.989807601	4.238314×10^{-2}	0.00	100
I-GWO	2.924874269	2.952942190	1.91408×10^{-2}	0.00	100
SASS	2.924834184	2.931218216	9.71232×10^{-4}	0.00	100

$$
\begin{aligned}
&h_1(x)=x_1-10^4(x_7-100)=0,\\
&h_2(x)=x_2-10^4(x_8-x_7)=0,\\
&h_3(x)=x_3-10^4(500-x_8)=0,\\
&h_4(x)=x_1-10^4(300-x_9)=0,\\
&h_5(x)=x_2-10^4(400-x_{10})=0,\\
&h_6(x)=x_3-10^4(600-x_{11})=0,\\
&h_7(x)=x_4\ln(x_9-100)-x_4\ln(300-\\
&\qquad x_7)-x_9-x_7+400=0,\\
&h_8(x)=x_5\ln(x_{10}-x_7)-x_5\ln(400-\\
&\qquad x_8)-x_{10}+x_7-x_8+400=0,\\
&h_9(x)=x_6\ln(x_{11}-x_8)-x_6\ln(100)-\\
&\qquad x_{11}+x_8+100=0;
\end{aligned}
$$

边界条件:

$$
\begin{aligned}
&10^4\leqslant x_1\leqslant 81.9\times10^4,\quad 10^4\leqslant x_2\leqslant 113.1\times10^4,\\
&10^4\leqslant x_3\leqslant 205\times10^4,\quad 0\leqslant x_4,x_5,x_6\leqslant 5.074\times10^{-2},\\
&100\leqslant x_7\leqslant 200,\quad 100\leqslant x_8,x_9,x_{10}\leqslant 300,\\
&100\leqslant x_{11}\leqslant 400.
\end{aligned}
$$

表 8 列出了解决热交换器网络设计问题时 6 种算法各自独立运行 50 次得到的最佳值、平均值、方差、平均违反约束和可行率.

由表 8 可知, SASS 算法 50 次求解得到的最佳值达到了热交换网络设计问题的理论最优值 7.04903695×10^3, 而 LLAMPA 算法的最佳值、平均值虽稍逊于 SASS 算法, 但也十分接近该问题的理论最优值, 同时明显优于其余 4 种对比算法多个数量级, 方差则优于其他全部 5 种对比算法, 展现出 LLAMPA 算法对解决这一问题有着相对出色的求解性能.

4.4 工业制冷系统优化设计问题

工业制冷系统优化设计问题[32−33]可以定义为非线性不等式约束优化问题. 该问题包括 14 个自由变量($x_1\sim x_{14}$)和 15 个非线性约束条件($g_1\sim g_{15}$), 数学模型如下:

目标函数:

$$
\begin{aligned}
f(\bar{x})=\ &63098.88x_2x_4x_{12}+5441.5x_2^2x_{12}+\\
&115055.5x_2^{1.664}x_6+6172.27x_2^2x_6+\\
&63098.88x_1x_3x_{11}+5441.5x_1^2x_{11}+\\
&115055.5x_1^{1.664}x_5+6172.27x_1^2x_5+\\
&140.53x_1x_{11}+281.29x_3x_{11}+\\
&70.26x_1^2+281.29x_1x_3+281.29x_3^2+\\
&14437x_8^{1.8812}x_{12}^{0.3424}x_{10}x_{14}^{-1}x_1^2x_7x_9^{-1}+\\
&20470.2x_7^{2.893}x_{11}^{0.316}x_1^2;
\end{aligned}
$$

约束条件:

$$
\begin{aligned}
&g_1(\bar{x})=1.524x_7^{-1}\leqslant 1,\\
&g_2(\bar{x})=1.524x_8^{-1}\leqslant 1,\\
&g_3(\bar{x})=0.07789x_1-2x_7^{-1}x_9-1\leqslant 0,\\
&g_4(\bar{x})=7.05305x_9^{-1}x_1^2x_{10}x_8^{-1}x_2^{-1}x_{14}^{-1}-1\leqslant 0,\\
&g_5(\bar{x})=0.0833x_{13}^{-1}x_{14}-1\leqslant 0,\\
&g_6(\bar{x})=47.136x_2^{0.333}x_{10}^{-1}x_{12}-1.333x_8x_{13}^{2.1195}+\\
&\qquad 62.08x_{13}^{2.1195}x_{12}^{-1}x_8^{0.2}x_{10}^{-1}-1\leqslant 0,\\
&g_7(\bar{x})=0.04771x_{10}x_8^{1.8812}x_{12}^{0.3424}-1\leqslant 0,\\
&g_8(\bar{x})=0.0488x_9x_7^{1.893}x_{11}^{0.316}-1\leqslant 0,\\
&g_9(\bar{x})=0.0099x_1x_3^{-1}-1\leqslant 0,\\
&g_{10}(\bar{x})=0.0193x_2x_4^{-1}-1\leqslant 0,\\
&g_{11}(\bar{x})=0.0298x_1x_5^{-1}-1\leqslant 0,\\
&g_{12}(\bar{x})=0.056x_2x_6^{-1}-1\leqslant 0,\\
&g_{13}(\bar{x})=2x_9^{-1}-1\leqslant 0,\\
&g_{14}(\bar{x})=2x_{10}^{-1}-1\leqslant 0,\\
&g_{15}(\bar{x})=x_{12}x_{11}^{-1}-1\leqslant 0;
\end{aligned}
$$

边界处理: $0.001\leqslant x_i\leqslant 5,\ i=1,\cdots,14$.

表 9 列出了解决工业制冷系统优化设计问题时 6 种算法各自独立运行 50 次得到的最佳值、平均值、方差、平均违反约束和可行率.

由表 9 可知, LLAMPA 算法 50 次求解得到的最佳值、平均值、方差均优于其他 5 种对比算法, 且最佳值(3.2213004×10^{-2})十分接近该问题的理论最优值 3.2213001×10^{-2}, 说明 LLAMPA 算法在工业

表 8 解决热交换器网络设计问题时 6 种算法的寻优结果

Table 8 Optimization results of 6 algorithms for the heat exchanger network design problem

Algorithm	Best	Avg	Var	MV/%	FR/%
LLAMPA	7.04903711×10^3	7.04903712×10^3	$\mathbf{1.3704\times10^{-10}}$	0.00	100.00
MPA	3.305264×10^{14}	2.189810×10^{17}	5.2904×10^{17}	1.92	0.00
MMPA	9.227617×10^{24}	1.024461×10^{25}	9.7796×10^{24}	4.00	0.00
LEO-MPA	9.396361×10^{13}	7.796936×10^{17}	2.2890×10^{18}	2.62	0.00
I-GWO	9.475101×10^{15}	1.132290×10^{17}	8.6954×10^{16}	2.12	0.00
SASS	$\mathbf{7.04903695\times10^3}$	$\mathbf{7.04903707\times10^3}$	4.7743×10^{-4}	0.00	100.00

表 9 解决工业制冷系统优化设计问题时 6 种算法的优化结果

Table 9 Optimization results of 6 algorithms for the optimal design of an Industrial refrigeration System problem

Algorithm	Best	Avg	Var	MV/%	FR/%
LLAMPA	$\mathbf{3.2213004 \times 10^{-2}}$	$\mathbf{3.2300370 \times 10^{-2}}$	$\mathbf{8.38 \times 10^{-8}}$	0.00	100
MPA	3.2215393×10^{-2}	3.2573308×10^{-2}	3.69×10^{-7}	0.00	100
MMPA	$3.22146060 \times 10^{-2}$	3.2365403×10^{-2}	1.69×10^{-7}	0.00	100
LEO-MPA	3.2217598×10^{-2}	3.2436210×10^{-2}	1.31×10^{-7}	0.00	100
I-GWO	3.2763016×10^{-2}	3.4186955×10^{-2}	9.75×10^{-7}	0.00	100
SASS	$3.44057845 \times 10^{-2}$	$3.65493377 \times 10^{-2}$	1.99×10^{-6}	0.00	100

制冷系统的优化设计问题上有着十分突出的求解性能和稳定性.

LLAMPA 算法对上述 4 个工程优化设计问题的求解结果证明, LLAMPA 在面对不同类型和复杂程度的工程优化设计问题时, 相较于其他 5 种代表性算法展现出更为出色的求解精度和稳定性, 且通过平均违反约束(MV)和可行率(FR), 可以看出 LLAMPA 算法对上述 4 个工程优化设计问题的求解结果均满足约束条件, 从而能够提供更为优越的解决方案.

5 结论

本文提出一种基于学习自动机引导的教与学搜索、对数螺旋探索和自适应相对反射机制的海洋捕食者算法(LLAMPA). 在海洋记忆存储阶段引入学习自动机引导的教与学搜索机制, 增强了算法的勘探与开发能力, 提高了算法优化性能; 在迭代后期的局部开发阶段, 引入对数螺旋探索机制, 进一步加强了算法的局部挖掘能力, 提升算法的收敛精度; 在算法中每次迭代末尾处加入改进的自适应相对反射策略, 增强了算法逃离局部极值区域的能力, 防止种群容易陷入局部收敛. 将 LLAMPA 算法应用于 CEC2017 复杂函数的全局优化求解中, 其 50 次运行得到的寻优结果最佳值、平均值和最差值明显优于其他 6 种性能优越的代表性对比算法; 而 Wilcoxon 秩和检验、以及箱线图分析则验证了 LLAMPA 相对于其他 6 种对比算法的优化结果具有显著性差异和求解稳定性. 最后, 将该算法对 4 个工程设计优化问题进行求解, 取得了很好的优化效果.

上述研究表明, LLAMPA 算法展现出了更好的搜索能力, 其寻优性能得到了明显提升, 有效排解了 MPA 存在的勘探与开发能力不足、局部桎梏概率高和收敛精度不足等问题. 该算法在求解全局复杂函数和工程设计约束优化问题时表现出了显著的可行性和有效性. 然而, 由于 LLAMPA 算法增加了新的参数, 因此在面对其他更多优化问题时, 可能存在参数值选择的问题, 需要重新测试和确定参数值. 此外, 随着问题规模及目标函数维度的增加, 虽然 LLAMPA 具有良好的解决问题能力, 但仍有进一步提升的空间. 因此, 在未来的研究中, 我们将继续探索改进 LLAMPA 和 MPA 的优化机制, 进一步验证和提高算法性能, 完善改进机制, 并将其应用于微电网优化配置、图像分割、柔性工作车间调度等更多领域.

参 考 文 献

[1] Faramarzi A, Heidarinejad M, Mirjalili S, et al. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst Appl*, 2020, 152: 113377

[2] Al-Qaness M A A, Saba A I, Elsheikh A H, et al. Efficient artificial intelligence forecasting models for COVID-19 outbreak in Russia and Brazil. *Process Saf Environ Prot*, 2021, 149: 399

[3] Abdel-Basset M, Mohamed R, Elhoseny M, et al. A hybrid COVID-19 detection model using an improved marine predators algorithm and a ranking-based diversity reduction strategy. *IEEE Access*, 2020, 8: 79521

[4] Ridha H M. Parameters extraction of single and double diodes photovoltaic models using Marine Predators Algorithm and Lambert W function. *Sol Energy*, 2020, 209: 674

[5] Abd Elaziz M, Shehabeldeen T A, Elsheikh A H, et al. Utilization of Random Vector Functional Link integrated with Marine Predators Algorithm for tensile behavior prediction of dissimilar friction stir welded aluminum alloy joints. *J Mater Res Technol*, 2020, 9(5): 11370

[6] Pan J S, Shan J, Chu S C, et al. A multigroup marine predator algorithm and its application for the power system economic load dispatch. *Energy Sci Eng*, 2022, 10(6): 1840

[7] Sadiq A S, Dehkordi A A, Mirjalili S, et al. Nonlinear marine predator algorithm: A cost-effective optimizer for fair power allocation in NOMA-VLC-B5G networks. *Expert Syst Appl*, 2022, 203: 117395

[8] Zhang L, Liu S, Gao W X, et al. An elite opposition-based golden

sine marine predator algorithm. *Comput Eng Sci*, 2023, 45(2): 355
(张磊, 刘升, 高文欣, 等. 精英反向黄金正弦海洋捕食者算法. 计算机工程与科学, 2023, 45(2): 355)

[9] Li S Y, He Q. Improved feature selection for marine predator algorithms. *Comput Eng Appl*, 2023, 59(11): 168
(李守玉, 何庆. 改进海洋捕食者算法的特征选择. 计算机工程与应用, 2023, 59(11): 168)

[10] Wang Y W, Wang W L, Yang Y G, et al. Improved marine predators algorithm with multi-strategy fusion and its engineering applications. *Comput Integr Manuf Syst*, http://kns.cnki.net/kcms/detail/11.5946.TP.20230515.1111.008.html
(王逸文, 王维莉, 杨宇鸽, 等. 多策略融合改进的海洋捕食者算法及其工程应用. 计算机集成制造系统, http://kns.cnki.net/kcms/detail/11.5946.TP.20230515.1111.008.html)

[11] Fu H, Liu S L, Guan Z F, et al. Phased-improvement marine predators algorithm and its application. *Contr Decis*, 2023, 38(4): 902
(付华, 刘尚霖, 管智峰, 等. 阶段化改进的海洋捕食者算法及其应用. 控制与决策, 2023, 38(4): 902)

[12] Oszust M. Enhanced marine predators algorithm with local escaping operator for global optimization. *Knowl Based Syst*, 2021, 232: 107467

[13] Fan Q S, Huang H S, Chen Q P, et al. A modified self-adaptive marine predators algorithm: Framework and engineering applications. *Eng Comput*, 2022, 38(4): 3269

[14] Yousri D, Fathy A, Rezk H. A new comprehensive learning marine predator algorithm for extracting the optimal parameters of supercapacitor model. *J Energy Storage*, 2021, 42: 103035

[15] Houssein E H, Mahdy M A, Fathy A, et al. A modified Marine Predator Algorithm based on opposition based learning for tracking the global MPP of shaded PV system. *Expert Syst Appl*, 2021, 183: 115253

[16] Abd Elaziz M, Mohammadi D, Oliva D, et al. Quantum marine predators algorithm for addressing multilevel image segmentation. *Appl Soft Comput*, 2021, 110: 107598

[17] Wu G H, Mallipeddi R, Suganthan P N. Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization [EB/OL]. *Technical Repore Online* (2017-09-01) [2023-12-02]. https://www.researchgate.net/publication/317228117_Problem_Definitions_and_Evaluation_Criteria_for_the_CEC_2017_Competition_and_Special_Session_on_Constrained_Single_Objective_Real-Parameter_Optimization

[18] Kumar A, Wu G H, Ali M Z, et al. A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm Evol Comput*, 2020, 56: 100693

[19] Narendra K S, Thathachar M A L. *Learning Automata: An Introduction*. Chelmsford: Courier corporation, 2012

[20] Rao R V, Savsani V J, Vakharia D P. Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput Aided Des*, 2011, 43(3): 303

[21] Mirjalili S, Lewis A. The whale optimization algorithm. *Adv Eng Softw*, 2016, 95: 51

[22] Zou T T, Wang C Y. Adaptive relative reflection Harris Hawks optimization for global optimization. *Mathematics*, 2022, 10(7): 1145

[23] Awad N H, Ali M Z, Suganthan P N. Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems // 2017 *IEEE Congress on Evolutionary Computation* (*CEC*). Donostia, 2017: 372

[24] Stanovov V, Akhmedova S, Semenkin E. NL-SHADE-LBC algorithm with linear parameter adaptation bias change for CEC 2022 Numerical Optimization // 2022 *IEEE Congress on Evolutionary Computation* (*CEC*). Padua, 2022: 1

[25] Nadimi-Shahraki M H, Taghian S, Mirjalili S. An improved grey wolf optimizer for solving engineering problems. *Expert Syst Appl*, 2021, 166: 113917

[26] Manalo T A, Biermann H D, Patil D H, et al. The temporal association of depression and anxiety in young men with erectile dysfunction. *J Sex Med*, 2022, 19(2): 201

[27] Supraba A, Musfirah M, Sjachrun R A M, et al. The students' response toward indirect corrective feedback used by the lecturer in teaching writing at Cokroaminoto Palopo University. *J Pendidikan Bahasa Dan Sastra*, 2021, 1(2): 148

[28] Kumar A, Das S, Zelinka I. A self-adaptive spherical search algorithm for real-world constrained optimization problems // *Proceedings of the* 2020 *Genetic and Evolutionary Computation Conference Companion*. Cancún, 2020: 13

[29] He X Y, Zhou Y R. Enhancing the performance of differential evolution with covariance matrix self-adaptation. *Appl Soft Comput*, 2018, 64: 227

[30] Angira R, Babu B V. Optimization of process synthesis and design problems: A modified differential evolution approach. *Chem Eng Sci*, 2006, 61(14): 4707

[31] Babu B V, Angira R. *Optimization of Industrial Processes Using Improved and Modified Differential Evolution*. Berlin: Springer Berlin Heidelberg, 2008

[32] Andrei N. *Nonlinear Optimization Applications Using the GAMS Technology*. New York: Springer, 2013

[33] Pant M, Thangaraj R, Singh V P. Optimization of mechanical design problems using improved differential evolution algorithm. *Int J Recent Trends Eng*, 2009, 1(5): 21