文章编号:1671-6833(2022)06-0008-07

一种具有学习机制的海鸥优化算法

王培崇[1,2], 尹欣洁[1,2], 李丽荣[3]

(1.河北地质大学 信息工程学院,河北 石家庄 050031;2.河北地质大学 人工智能与机器学习研究室,河北 石家庄 050031;3.河北地质大学 艺术学院,河北 石家庄 050031)

摘 要:为了克服海鸥优化算法在求解高维问题时存在的收敛速度慢、容易早熟和解精度低等问题,提出一种具有学习机制的海鸥优化算法(ISOAL)。首先,设计了一种基于当前粒子 $\boldsymbol{X}_i$ 与种群均值状态 $\boldsymbol{X}_m$ 差异的迁移算子,提升早期个体对解空间的搜索范围。其次,引入非线性自适应参数 A 保证算法适合于复杂问题解空间的搜索,避免算法过早地陷入局部最优。最后,通过引入部分精英粒子执行反向学习,加强对种群内的最优粒子所在空间的勘探,提高算法的解精度。实验选择了 CEC2017 中的 10 个无约束测试函数检测算法的性能,并与 HPSO-TS、V-DVGA、DADE、CMA-ES 等算法进行对比,该组实验结果显示,ISOAL 比其他算法具有更高的解精度和稳定性。针对张力弹簧问题进行实验,结果表明:ISOAL 所获得的弹簧总代价比 SOA 降低了 3.5%,弹簧的线圈直径和平均直径分别下降了 5.7%和 3.5%。ISOAL 算法具有收敛速度快、精度高和鲁棒性的特点,适合求解较高维度的连续函数优化问题和带有约束的工程优化问题。

关键词:海鸥优化算法;学习机制;非线性参数;反向学习

中图分类号:TP301.6;O234 **文献标志码**:A **doi**:10.13705/j.issn.1671-6833.2022.06.010

0 引言

近些年,元启发算法[1]发展迅速,2019 年的海鸥优化算法(seagull optimization algorithm, SOA)[2]就是元启发算法的代表算法之一。该算法通过模拟海鸥种群的协作飞行和捕食猎物实现对问题的求解,并在多个工程领域内得到了成功应用。但是,由于该算法缺少严格的数学理论支持,在解决较高维度的函数优化问题时出现了收敛速度慢、解精度低和容易早熟等问题[3-5]。

针对标准 SOA 容易早熟、解精度低等问题,毛清华等[6]提出一种融合改进 Logistics 混沌和正弦、余弦算子的自适应 t 分布海鸥算法,该算法采用改进的 Logistics 混沌映射机制初始化种群,在种群的搜索阶段引入正弦、余弦算子更新个体的位置,并引入非线性收敛因子协调算法的局部搜索和全局搜索,加快算法的收敛速度;同时,为了克服算法早熟,对种群内最优个体所在的位置引入自适应 t 分布变异策略加以扰动。王宁等[7]提出一种黄金正弦引导与 Sigmoid 连续化的海鸥优化算法,该算法在海鸥迁徙阶段,引入 Sigmoid 函数的惯性权重引导海鸥种群的全局搜索;在捕食阶段,将置信度低的区域设置为搜索禁忌,保证种群一直向置信度高的优秀解空间区域飞行;并且加入黄金正弦机制帮助种群寻找搜索范围,指引种群的位置更新,提高寻优能力。秦维娜等[8]提出一种基于惯性权重的海鸥优化算法,采用非线性递减的惯性权重计算附加变量的值来调整海鸥的位置,通过 Levy 飞行和随机指数值增加海鸥飞行的随机性,增强算法搜索寻优的全局能力,避免算法寻优搜索陷入局部最优。Che 等[9]提出了一种混合鲸鱼-海鸥优化算法,该算法首先在 SOA 螺旋式攻击行为中融入鲸鱼优化算法的空间收缩策略,以提高 SOA 的计算精度;其次,在 SOA 的搜索算子中引入 Levy 飞行策略,引导算法跳出局部最优的约束。上述改进机制在一定程度上改善

收稿日期:2022-02-01;**修订日期**:2022-06-29

基金项目:国家自然科学基金资助项目(61806069);河北省高等学校科学技术研究项目(ZD2020344)

作者简介:王培崇(1972—),男,河北辛集人,河北地质大学教授,博士,主要从事信息安全、机器视觉研究,E-mail:wpeichong@163.com。

通信作者:李丽荣(1973—),女,河北张家口人,河北地质大学副研究员,主要从事智能决策的研究,E-mail:lllrong@163.com。

了算法的性能,但是均没有考虑种群内粒子之间的信息交流问题。

本文提出一种具有学习机制的改进海鸥优化算法(improved SOA with learning,ISOAL)。该算法在迁移算子中引入基于当前粒子和种群均值间差异的学习机制,在算法的后期引入精英粒子的反向学习(opposition-based learning,OBL)[10-11],以提升算法性能。

1　标准海鸥优化算法

标准海鸥优化算法流程如图 1 所示。

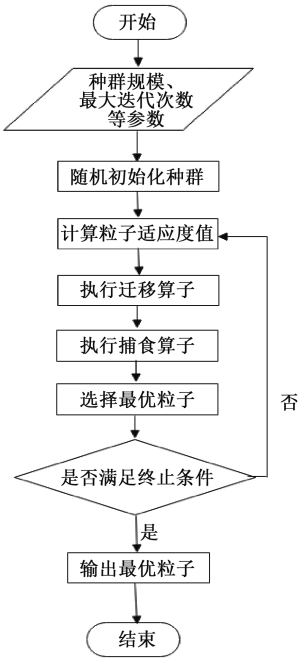

图 1　标准海鸥优化算法流程

Figure 1　Flow chart of standard SOA

1.1　迁移算子

在迁移算子中设置 2 个搜索代理 $\boldsymbol{X}_i^s(t)$ 和 $\boldsymbol{X}_i^b(t)$。前者用于搜索解空间内不与其他粒子发生碰撞的位置;后者用于搜索当前种群内的最优粒子所在位置,记 $\boldsymbol{X}_b(t)$ 为种群内的最优粒子,则有

$$\boldsymbol{X}_i^s(t) = A \cdot \boldsymbol{X}_i(t); \tag{1}$$

$$\boldsymbol{X}_i^b(t) = B \cdot (\boldsymbol{X}_b(t) - \boldsymbol{X}_i(t)) 。 \tag{2}$$

式中:$A = f_s - [t \cdot (f_s / T_{\max})]$,$f_s = 2$,$T_{\max}$ 为最大迭代次数;$B = 2A^2 \cdot \mathrm{rand}(0,1)$。

当前粒子与最优粒子之间的相对距离为

$$\boldsymbol{D}_i(t) = | \boldsymbol{X}_i^s(t) + \boldsymbol{X}_i^b(t) | 。 \tag{3}$$

1.2　捕食算子

海鸥采用一种螺旋式飞行的方式对猎物进行攻击,在三维空间的运动为

$$\begin{cases} g_x = r\cos\theta; \\ g_y = r\sin\theta; \\ g_z = r\theta; \\ r = u\mathrm{e}^{\theta v}。 \end{cases} \tag{4}$$

式中:r 为海鸥在螺旋飞行时的半径;$\theta \in [0, 2\pi]$ 为随机角度;u、v 为常数。

当前粒子的捕食算子为

$$\boldsymbol{X}_i(t+1) = \boldsymbol{D}_i(t) \cdot g_x g_y g_z + \boldsymbol{X}_b(t)。 \tag{5}$$

综上,算法迭代过程中的当前粒子轨迹为

$$\boldsymbol{X}_i(t+1) = | (A - B) \cdot \boldsymbol{X}_i(t) + B \cdot \boldsymbol{X}_b(t) | \cdot g_x g_y g_z + \boldsymbol{X}_b(t)。 \tag{6}$$

2　具有学习机制的海鸥优化算法

由式(6)可知,海鸥优化算法的迭代过程仅有 2 个参数。其中参数 A 为线性变化,在一定程度上提升了算法的收敛速度,但对于解空间为多峰的复杂优化问题,算法容易忽略部分解空间。此外,该算法没有考虑种群内部非最优粒子之间的信息交流和互动行为,而自然界中的生命群体均会存在内部信息的沟通和学习行为,这种学习行为在很大程度上可以主导种群的演化方向。基于此,本文将从以下几个方面对算法进行改进。

2.1　基于学习机制的迁移算子

记 $\boldsymbol{X}_m(t)$ 为当前种群全部粒子的平均状态值,则式(1)的空间搜索机制更新为

$$\boldsymbol{X}_i^s(t) = A[\boldsymbol{X}_i(t) + \alpha(\boldsymbol{X}_m(t) - \boldsymbol{X}_i(t))]。 \tag{7}$$

式中:α 为(0,1)的随机数;$\boldsymbol{X}_m(t) - \boldsymbol{X}_i(t)$ 表示当前粒子向种群均值学习,这样可以在算法早期有效扩大搜索范围,避免过早的收敛。

2.2　参数 A 的改进

标准 SOA 中的参数 A 受 f_s 的制约,从 2 线性递减到 0,收敛速度较快。但是,大多数优化问题的解空间为非线性,线性参数并不适合对空间进行有效搜索,容易忽略部分空间。因此将参数 A 调整为非线性以保证粒子对解空间的非线性搜索,见式(8):

$$A' = \frac{2}{1 + \mathrm{e}^{\frac{t - T_{\max}}{T_{\max}}}} + \beta \cdot \mathrm{betarand}(p,q)。 \tag{8}$$

式中:$\mathrm{betarand}(\cdot)$ 为能产生 beta 分布的随机数生成器。

图 2 为参数 A 和 A'随迭代次数的变化曲线。在算法前期,参数 A'能够在较长时间内保持较大值且变化幅度较大,以扩大种群内粒子的搜索范围。在算法后期,参数 A'逐渐减小,强化粒子对

自身所在区域的精细勘探。

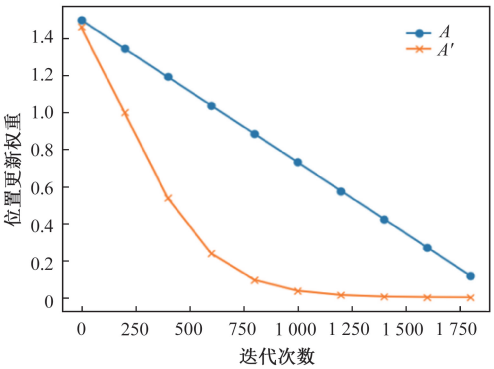

图2 参数 A 和 A' 随迭代次数的变化曲线

Figure 2 Curves of A and A' with the number of iterations

2.3 精英反向学习

群体智能算法中,适应度优的精英粒子必然包含了更多的引导种群向全局最优收敛的有益信息。令精英粒子执行反向学习,既能加快算法的收敛速度,也加强了算法的局部探索能力。

定义1 反向解。设在$[a,b]$上存在一个实数x,则x的反向数$x'=a+b-x$。基于此,假设在$\mathbf{R}$域上存在某n维点$\boldsymbol{X}=(x_1,x_2,\cdots,x_n)$,并且$x_i\in[a_i,b_i]$,则定义$\boldsymbol{X}'=(x'_1,x'_2,\cdots,x'_n)$为$\boldsymbol{X}$的反向点。其中,$x'_i=k\cdot(a_i+b_i)-x_i$,$k$为$[0,1]$上均匀分布的随机数,称作一般化系数。

定义2 基于反向解的优化。设待优化问题为最小问题,适应度函数为$f(\boldsymbol{X})$,若存在某个可行解$\boldsymbol{X}$,其反向解为$\boldsymbol{X}'$,若$f(\boldsymbol{X}')<f(\boldsymbol{X})$成立,则用$\boldsymbol{X}'$替换$\boldsymbol{X}$。

算法1 OBL算法。

输入:个体$\boldsymbol{X}_i(t)$,下限a,上限b,最大迭代次数$Iter_{\max}$;

输出:最优个体$\boldsymbol{X}_i(t)$;

Step 1 设置迭代次数为$m=0$;

Step 2 生成一般化系数$k\in[0,1]$,依据定义1生成反向解$\boldsymbol{X}'_i(m)$,并保存;

Step 3 迭代次数没有满足结束条件,则$m=m+1$,转至Step 2;

Step 4 在$\boldsymbol{X}_i(t)$和所产生的全部反向解中选择最优的个体替换$\boldsymbol{X}_i(t)$并输出,算法结束。

2.4 算法描述

算法2 具有学习机制的海鸥优化算法ISOAL。

输入:种群规模N,最大迭代次数等参数;

输出:最优粒子$\boldsymbol{X}_b(t)$。

Step 1 在解空间内随机初始化种群POP;

Step 2 计算粒子的适应度值选择最优粒子$\boldsymbol{X}_b(t)$;

Step 3 以式(7)、(8)替换式(1),执行迁移算子;

Step 4 执行捕食算子;

Step 5 精英粒子变化幅度小于0.01,则执行反向学习;

Step 6 算法不满足终止条件,则转至Step 2;

Step 7 输出最优粒子$\boldsymbol{X}_b(t)$,算法结束。

设算子最大迭代次数为$T_{\max}$,种群规模为N,分析上述算法可知,主要操作集中于Step 3~6,则时间复杂度为$O(T_{\max}\cdot N)$。

3 实验及结果分析

3.1 无约束函数测试

为了验证所提算法的性能,应用MATLAB 2019编程实现上述算法,实验环境为联想笔记本(intel i7,16 GB)。选择10个CEC2017标准测试函数来测试算法的性能,列于表1中,以近年来最新算法HPSO-TS[12]、V-DVGA[13]、DADE[14]和CMA-ES[15]作为对比算法。ISOAL的种群规模设置为30,问题维度为50,算法的迭代次数为3 000。所有算法均独立运行30次,以消除偶然因素的影响,取解均值和方差进行对比,如表2所示。

表2列出了SOA、HPSO-TS、V-DVGA、DADE、CMA-ES与本文算法在CEC2017的10个函数上的解均值和方差。由表2可知,在单峰函数f1、f2、f3上,本文算法ISOAL的解均值和方差均最小。多峰函数f4、f6的解空间并不复杂,考验的是算法跳出局部最优约束的能力,从实验结果看,ISOAL在解均值和方差上比CMA-ES算法略差,但是仍然优于其他算法。多峰函数中的f5、f7、f8的解空间较复杂,存在多个局部最优极值点,因此,对应算法不仅需要有摆脱局部极值约束的能力,而且还要有一定的勘探新解的能力。在f5函数上,HPSO-TS、V-DVGA和ISOAL的解均值精度相差不大,ISOAL的方差优于其他几个算法,表现出较好的稳定性。在f7、f8函数上,ISOAL的解均值和方差均最小,稳定性良好。在函数f9上,ISOAL的解均值和方差均略高于DADE算法,但也在同一个数量级上,优于其他算法。函数f10是一个多峰函数且局部最优值数量众多,ISOAL同样表现出色,解均值和方差均最小,优于其他对比算法。

图3为V-DVGA、HPSO-TS、ISOAL和SOA算法在f1、f2、f3、f4、f5和f6上的收敛曲线。

表 1　CEC2017 测试函数

Table 1　CEC2017 functions

测试函数	函数名称	搜索域
f1	shifted and rotated bent cigar	[-100,100]
f2	shifted and rotated sum of different power function	[-100,100]
f3	shifted and rotated Zakharov function	[-300,300]
f4	shifted and rotated rosenbrock′s function	[-100,100]
f5	shifted and rotated rastrigin′s function	[-600,600]
f6	shifted and rotated expanded scaffer′s f6 function	[-60,60]
f7	shifted and rotated lunacek bi-rastrigin′s function	[-100,100]
f8	shifted and rotated non-continuous rastrigin′s function	[-200,200]
f9	shifted and rotated levy function	[-100,100]
f10	shifted and rotated schwefel′s function	[-150,150]

表 2　与其他改进的群智能算法的对比

Table 2　Comparison with other improved swarm intelligence algorithms

测试	SOA		HPSO-TS		V-DVGA		CMA-ES		DADE		ISOAL	
函数	解均值	方差	解均值	方差	解均值	方差	解均值	方差	解均值	方差	解均值	方差
f1	2.13E-03	2.95E-03	4.32E-14	2.84E-11	3.62E-51	3.19E-53	5.69E-47	2.76E-38	6.28E-48	3.00E-40	**6.47E-53**	**3.62E-54**
f2	1.18E-01	3.49E-01	1.84E-22	5.11E-34	4.22E-25	7.64E-23	3.11E-31	4.54E-27	1.37E-22	5.88E-27	**1.56E-34**	**5.94E-28**
f3	8.05E-02	1.17E-02	8.17E-12	5.44E-14	7.56E-27	1.20E-13	3.14E-21	1.07E-27	5.05E-22	3.25E-28	**5.32E-30**	**3.06E-32**
f4	9.05E-02	2.06E-01	5.10E-06	1.53E-03	6.18E-06	2.20E-05	**5.09E-08**	**2.65E-08**	4.01E-05	2.12E-02	4.49E-07	2.71E-06
f5	6.70E-01	3.48E-01	6.12E-02	2.40E-02	**5.31E-02**	3.09E-02	7.03E-01	2.23E-01	6.94E-01	3.07E-02	6.69E-02	**1.85E-02**
f6	9.51E-02	5.42E-01	7.94E-06	3.53E-08	5.31E-02	3.09E-02	**5.91E-09**	**4.50E-09**	5.42E-06	2.46E-07	6.71E-09	1.74E-08
f7	3.78E-02	1.25E-01	5.14E-06	2.55E-04	4.28E-06	2.02E-01	3.82E-08	2.77E-04	7.21E-04	1.58E-04	**2.29E-08**	**3.46E-05**
f8	7.23E-02	3.41E-02	4.63E-03	5.51E-01	3.67E-04	2.16E-04	1.72E-05	6.43E-02	3.90E-02	2.69E-04	**3.81E-06**	**2.14E-04**
f9	2.45E-02	1.57E+02	3.15E-01	2.14E-01	3.26E+00	2.94E+00	7.64E-02	3.67E-01	**2.98E-04**	**5.67E-05**	3.66E-04	6.83E-05
f10	7.42E-01	1.03E+00	4.27E-02	7.22E-01	3.26E-02	1.38E-04	4.16E-03	7.80E-04	2.37E-02	8.09E-03	**3.08E-05**	**3.91E-05**

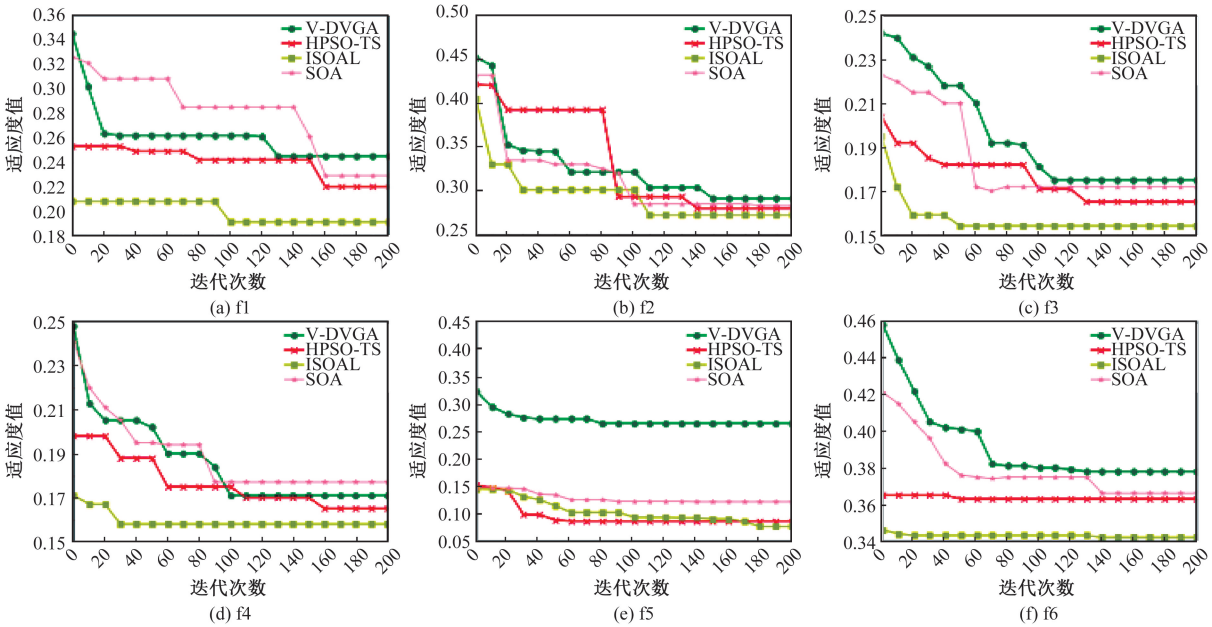

图 3　算法收敛曲线对比

Figure 3　Comparison of algorithm convergence curve

由图 3 可知,在函数 f1、f2、f3 和 f4 上,ISOAL 初始种群的最优适应度就已经优于其他算法,并且经过非常少的迭代次数就迅速收敛,收敛速度明显优于其他算法。在函数 f1 上,HPSO-TS 搜索到的最小适应度值为 0.224 5,ISOAL 搜索到的最小适应度值为 0.191 2,相比下降 14.83%。在函

数 f3 上,HPSO-TS 搜索到的最小适应度值为 0.165 7,ISOAL 搜索到的最小适应度值为 0.154 7,相比下降 6.63%。在函数 f5 上,HPSO-TS 与 ISOAL 搜索到的最小适应度值相差不大。在函数 f6 上,HPSO-TS 搜索到的最小适应度值为 0.363 3,ISOAL 搜索到的最小适应度值为 0.342 7,相比下降 5.67%。在函数 f2 上,ISOAL 算法和其他算法的收敛曲线在迭代初期交叉在一起,但是 ISOAL 算法在经过较少的迭代之后,能够迅速收敛到最优解周围,并开始进行勘探操作。结果表明,ISOAL 所获得最终解的精度优于其他算法。

3.2 约束函数测试

针对工程中常见的张力弹簧问题[16]进行测试实验,以检测本文算法在约束问题上的性能。实验目标是在满足挠度、剪应力和振动频率等约束条件下,获得最小质量的弹簧。其变量分别为弹簧的线圈直径 $W(x_1)$、弹簧的平均直径 $D(x_2)$、弹簧的有效线圈数 $L(x_3)$。该问题的目标函数(总代价)和约束函数分别为

$$\min f(x_1,x_2,x_3)=(x_3+2)x_1^2x_2; \tag{9}$$

$$\begin{cases} g_1(x)=1-\dfrac{x_2^3x_3}{71\ 785x_1^4}\leqslant 0; \\ g_2(x)=\dfrac{x_2(4x_2-x_1)}{12\ 566x_1^3(x_2-x_1)}+\dfrac{1}{5\ 108x_1^2}-1\leqslant 0; \\ g_3(x)=1-\dfrac{140.45x_1}{x_2^2x_3}\leqslant 0; \\ g_4(x)=\dfrac{2(x_1+x_2)}{3}-1\leqslant 0。 \end{cases} \tag{10}$$

式中:$0.05\leqslant x_1\leqslant 2.00$;$0.25\leqslant x_2\leqslant 1.30$;$2.0\leqslant x_3\leqslant 15.0$。

表 3 列出了对比算法在该问题上的求解结果,对比算法的数据取自文献[16]。由表 3 可知,ISOAL 在仅有 8 000 次的评价次数内,获得的总代价比 SOA 降低了 3.5%,弹簧的线圈直径和平均直径分别下降了 5.7%和 3.5%。

表 3 算法在张力弹簧问题上的结果对比

Table 3 Result comparison of design of tension/compression springs

算法	$W(x_1)$	$D(x_2)$	$L(x_3)$	总代价	评价次数
SzAPSO	0.053	0.364	11.389	0.014 1	160 000
SAMGA	0.052	0.367	11.452	0.014 5	100 000
GSDE	0.051	0.362	11.254	**0.013 2**	90 000
SOA	0.053	0.375	11.455	0.014 4	90 000
ISOAL	0.050	0.362	11.396	0.013 9	**80 000**

表 4 为对比算法约束条件函数值。上述实验结果表明,ISOAL 能够比较好地解决约束优化问题,且具有良好的性能。

表 4 弹簧设计问题的约束函数值对比

Table 4 Comparison of constrained function values in design of tension/compression springs

算法	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$
SzAPSO	−8.02E−04	−3.54E−04	−4.061	−0.760
SAMGA	−3.78E−09	−6.59E−09	−4.063	−0.778
GSDE	−1.25E−13	**−1.24E−14**	**−4.069**	−0.761
SOA	−7.63E−06	−5.32E−07	−4.054	−0.731
ISOAL	**−3.58E−14**	−7.16E−13	−4.057	**−0.799**

4 结论

为了解决标准 SOA 算法内部粒子之间缺少交流的问题,本文提出了一种具有学习机制的海鸥优化算法 ISOAL。该算法引入了基于粒子状态差异的学习机制,使当前粒子先向种群内粒子的均值状态学习,再搜索非冲突位置,执行位置迁移,并将原参数 A 由线性递减修改为非线性自适应递减方式,增强种群对解空间的全局搜索能力。为了提升算法的解精度,在后期引入精英反向学习,通过该机制加强对精英个体所在空间的勘探。在 CEC2017 和张力弹簧上的测试实验表明,ISOAL 不仅具有较快的收敛速度,还具有较好的解精度和鲁棒性。

海鸥优化算法出现时间较短,相应的研究成果较少,下一步应该加强对其收敛性能的研究。借鉴其他算法以改善 SOA 的性能,探索新的应用领域也是未来主要的研究方向。

参考文献:

[1] 徐霜, 万强, 余琍. 基于学习理论的改进粒子群优化算法[J]. 郑州大学学报(工学版), 2019, 40(2): 29-34.
XU S, WAN Q, YU L. An improved particle swarm optimization algorithm based on learning theory[J]. Journal of Zhengzhou university (engineering science), 2019, 40(2): 29-34.

[2] DHIMAN G, KUMAR V. Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems[J]. Knowledge-based systems, 2019, 165: 169-196.

[3] CHEN X, LI Y L, ZHANG Y C, et al. A novel hybrid model based on an improved seagull optimization algorithm for short-term wind speed forecasting[J].

Processes, 2021, 9(2): 387.

[4] 岳文静,孙鹏,陈志. 基于改进海鸥算法的认知无人机网络频谱分配[J]. 计算机技术与发展,2021,31(9):7-12.
YUE W J, SUN P, CHEN Z. Spectrum allocation of cognitive UAV network based on improved seagull algorithm[J]. Computer technology and development, 2021,31(9):7-12.

[5] 许乐, 莫愿斌, 卢彦越. 基于改进海鸥优化算法的 PID 控制器参数优化[J]. 机床与液压, 2021, 49(16): 17-23.
XU L, MO Y B, LU Y Y. Parameter optimization of PID controller based on improved seagull optimization algorithm[J]. Machine tool & hydraulics, 2021, 49(16): 17-23.

[6] 毛清华,王迎港. 融合改进 Logistics 混沌和正弦余弦算子的自适应 T 分布海鸥算法[EB/OL]. (2021-10-22)[2021-11-13]. http://kns. cnki. net. zzulib. vpn358. com/kns8/defaultresult/index.
MAO Q H, WANG Y G. Adaptive T distribution seagull optimization algorithm combining improved logistics chaos and sine-cosine operator [EB/OL]. (2021-10-22)[2021-11-13]. http://kns. cnki. net. zzulib. vpn358. com/kns8/defaultresult/index.

[7] 王宁, 何庆. 融合黄金正弦与 sigmoid 连续化的海鸥优化算法[J]. 计算机应用研究, 2022, 39(1): 157-162, 169.
WANG N, HE Q. Seagull optimization algorithm combining golden sine and sigmoid continuity[J]. Application research of computers, 2022, 39(1): 157-162, 169.

[8] 秦维娜, 张达敏, 尹德鑫, 等. 一种基于非线性惯性权重的海鸥优化算法[J]. 小型微型计算机系统, 2022, 43(1): 10-14.
QIN W N, ZHANG D M, YIN D X, et al. Seagull optimization algorithm based on nonlinear inertia weight [J]. Journal of Chinese computer systems, 2022, 43(1): 10-14.

[9] CHE Y H, HE D X. A hybrid whale optimization with seagull algorithm for global optimization problems [J]. Mathematical problems in engineering, 2021, 2021: 6639671.

[10] TIZHOOSH H R. Opposition-based learning: a new scheme for machine intelligence [C] // International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce. Piscataway: IEEE, 2005: 695-701.

[11] 刘琨, 赵露露, 王辉. 一种基于精英反向和纵横交叉的鲸鱼优化算法[J]. 小型微型计算机系统, 2020, 41(10): 2092-2097.
LIU K, ZHAO L L, WANG H. Whale optimization algorithm based on elite opposition-based and crisscross optimization [J]. Journal of Chinese computer systems, 2020, 41(10): 2092-2097.

[12] 周文峰, 梁晓磊, 唐可心, 等. 具有拓扑时变和搜索扰动的混合粒子群优化算法[J]. 计算机应用, 2020, 40(7): 1913-1918.
ZHOU W F, LIANG X L, TANG K X, et al. Hybrid particle swarm optimization algorithm with topological time-varying and search disturbance [J]. Journal of computer applications, 2020, 40(7): 1913-1918.

[13] 孙敏, 叶侨楠, 陈中雄. 云环境下方差定向变异遗传算法的任务调度[J]. 计算机应用, 2019, 39(11): 3328-3332.
SUN M, YE Q N, CHEN Z X. Task scheduling of variance-directional variation genetic algorithm in cloud environment [J]. Journal of computer applications, 2019, 39(11): 3328-3332.

[14] 沈鑫, 邹德旋, 张强. 采用双变异策略的自适应差分进化算法及应用[J]. 计算机工程与应用, 2020, 56(4): 146-157.
SHEN X, ZOU D X, ZHANG Q. Adaptive differential evolution algorithm using double mutation strategies and its application[J]. Computer engineering and applications, 2020, 56(4): 146-157.

[15] 赵云涛, 梅伟, 李维刚, 等. 基于改进 CMA-ES 算法的机器人轨迹规划[J]. 计算机仿真, 2019, 36(12): 317-322.
ZHAO Y T, MEI W, LI W G, et al. Robot trajectory planning based on improved CMA-ES algorithm [J]. Computer simulation, 2019, 36(12): 317-322.

[16] 李牧东, 赵辉, 翁兴伟, 等. 基于最优高斯随机游走和个体筛选策略的差分进化算法[J]. 控制与决策, 2016, 31(8): 1379-1386.
LI M D, ZHAO H, WENG X W, et al. Differential evolution based on optimal Gaussian random walk and individual selection strategies [J]. Control and decision, 2016, 31(8): 1379-1386.

An Improved Seagull Optimization Algorithm with Learning

WANG Peichong[1,2], YIN Xinjie[1,2], LI Lirong[3]

(1. School of Information Engineering, Hebei GEO University, Shijiazhuang 050031, China; 2. Laboratory of AI and Machine Learning, Hebei GEO University, Shijiazhuang 050031, China; 3. School of Art, Hebei GEO University, Shijiazhuang 050031, China)

Abstract: To overcome the weakness of slow convergence, prematureness and low accuracy of seagull optimization algorithm (SOA) in solving high-dimensional problems, an improved SOA with learning (ISAOL) was proposed. A migration operator based on the difference between the $\boldsymbol{X}_i$ and the $\boldsymbol{X}_m$ was designed, this could make $\boldsymbol{X}_i$ search wider solution spaces in early stage, and a nonlinear adaptive parameter A was introduced to ensure the algorithm suitable for the search of solution space of complex problems, which could prevent the algorithm from falling into local optimum too early. In the later stage, some elite individuals executed opposition based learning(OBL) to intensify the exploration of the space around the global optimal individual to improve the accuracy of solution. Ten unconstrained test functions in CEC2017 were selected to test the performance of the ISOA and compared with HPSO-TS, V-DVGA, DADE, CMA-ES and others. Testing results of this experiments showed that the ISOAL had higher accuracy and stability than other algorithms. Finally, experiments was carried out by using the tension spring problem. The results showed that the total cost of the spring , the coil diameter and the average diameter of the spring obtained by the ISOAL were reduced by 3.5% ,5.7% and 3.5% than SOA, respectively. ISOAL had the attributes of fast convergence, high accuracy and robustness, fitting to solve higher dimensional function optimization problem and engineering optimization problems with constraints.

Keywords: seagull optimization algorithm; learning mechanism; nonlinear parameter; opposition-based learning

---

(上接第7页)

An Efficient Approach to Creating Hand-Drawn Dataset for UI Manuscript Recognition

YANG Qi[1], LIU Mugeng[2], MA Yun[3]

(1. School of Electronic and Computer Engineering, Peking University Shenzhen Graduate School, Shenzhen 518055, China; 2. Department of Computer Science, Peking University, Beijing 100871, China; 3. Institute for Artificial Intelligence, Peking University, Beijing 100871, China)

Abstract: UI manuscript recognition is one of the important applications of image object detection in the area of software engineering. Due to the significant difference between UI manuscript images and natural images where UI manuscript images usually need to be drawn manually, it is difficult to build UI manuscript dataset for deep learning because of the dependency on tremendous manual efforts. To address the issue, in this study an approach called UIsketcher was proposed to efficiently generate UI manuscript dataset based on optimizing the current workflow. In UIsketcher, users should just draw some basic elements without labeling, and then the dataset could be automatically generated for training deep learning model. According to the experiment with UIsketcher, only 25% drawing workload of the traditional methods could get the similar training results. If the workload was 75%, the final accuracy was even better than that of traditional methods.

Keywords: intelligent software developing service; UI manuscript; recognition; object detection; dataset; data enhancement