Computer Science and Application 计算机科学与应用, 2020, 10(5), 860-867
Published Online May 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.105089

Hans汉斯

Moth-Flame Optimization Algorithm Using Elite Opposition-Based Learning

Zhiming Li[1], Jiaquan Zhou[1], Xiufang Huang[1], Yongsheng Xie[1], Liuqiang Wu[2]

[1]College of Mathematics and Computer Science, Guangxi Science & Technology Normal University, Laibin Guangxi
[2]Beijing Zhongke TeRui Technology Co., Ltd., Beijing
Email: chimingle@163.com, zhoujiaquan152468@163.com, huangxiufang2018@163.com

Received: Apr. 19th, 2020; accepted: May 4th, 2020; published: May 11th, 2020

Abstract

The Moth-flame Optimization (MFO) algorithm is a novel heuristic algorithm. The main inspiration of this algorithm is the navigation method of moths in nature called transverse orientation. Since the Moth-flame Optimization algorithm is easy to fall into the local optimum and the solution accuracy is low, elite opposition-based learning was utilized to propose the Moth-flame optimization algorithm using elite opposition-based learning (EOMFO). The proposed algorithm not only increases the diversity of the group, but also improves the overall performance of the algorithm. Through the experimental simulation of 7 sets of standard test functions, the results show that the Moth-flame optimization algorithm with elite opposition-based learning has higher convergence precision, which verifies that the new algorithm is effective and feasible.

Keywords

Transverse Orientation, Moth-Flame Optimization Algorithm, Opposition-Based Learning, Elite

应用精英反向学习的飞蛾扑火优化算法

李志明[1]，周加全[1]，黄秀芳[1]，谢永盛[1]，吴柳强[2]

[1]广西科技师范学院 数学与计算机科学学院，广西 来宾
[2]北京中科特瑞科技有限公司，北京
Email: chimingle@163.com, zhoujiaquan152468@163.com, huangxiufang2018@163.com

收稿日期：2020年4月19日；录用日期：2020年5月4日；发布日期：2020年5月11日

摘　要

飞蛾扑火优化算法(MFO)是一个新颖的启发式算法，其主要设计灵感来源于自然界中称为横向定位的导

文章引用: 李志明, 周加全, 黄秀芳, 谢永盛, 吴柳强. 应用精英反向学习的飞蛾扑火优化算法[J]. 计算机科学与应用, 2020, 10(5): 860-867. DOI: 10.12677/csa.2020.105089

航机制。由于标准飞蛾扑火优化算法容易陷入局部最优、求解精度低，所以借鉴精英反向学习策略，提出一种应用精英反向学习的飞蛾扑火优化算法(EOMFO)。新算法不仅增加了群体的多样性，并且提高了算法的综合性能。通过对7组标准测试函数的实验仿真，结果表明应用精英反向学习的飞蛾扑火优化算法具有更高的收敛精度，从而验证了新算法是有效的和可行的。

关键词

横向定位，飞蛾扑火优化算法，反向学习，精英

1. 引言

飞蛾扑火优化算法[1] [2] [3] [4] (Moth-flame optimization algorithm, MFO)是由Seyedali Mirjalili提出的一种新型启发式优化算法[5] [6]，该算法的灵感来源于飞蛾以月亮为参照来辨别方向的导航机制。由于距离月亮较远，在夜间的时候飞蛾只需要保持与月亮有个固定的角度飞行即可保证直线移动。当飞蛾迷失方向的时候，只需根据月光来自己调整方位，便能找到正确的方向。而当飞蛾误把人造光当成月光而作螺旋状飞行时，飞蛾最终便会向人造光处收敛。

飞蛾扑火优化算法因其具有诸多优点，因此常被用于求解大规模优化问题[7]、网络流量预测[8]、编码信号集设计[9]等诸多领域，同时算法在寻优过程中也存在一些不足，如由于种群多样性低导致全局寻优性能较差等。针对飞蛾扑火优化算法存在的一些不足，诸多学者尝试对其进行了改进，并成功应用到了一些工程领域。李伟琨等提出一种多目标飞蛾扑火算法，并将其用在了电力系统无功优化中[10]；高帅等利用支持向量机(SVM)提出将 SVM 和 MFO 相结合的算法(MFO-SVM)，并能够有效预测空气质量指数[11]；徐慧等提出一种融合粒子群的二进制飞蛾扑火优化算法(BPMFO)，并将其应用于网络入侵检测的特征检测中[12]；岳龙飞等通过借鉴混沌序列、模拟退火算法和遗传算法提出 Tent 混沌和模拟退火改进的飞蛾扑火优化算法，新算法增加了种群多样性，然后对当前最优解增加扰动产生新解，最终获得最优解[13]。

2. 飞蛾扑火优化算法

2.1. 标准 MFO 算法

在标准 MFO 算法中，分别用矩阵 M 和 F 来表示飞蛾和火焰集合，而 OM 和 OF 分别用来存放飞蛾和火焰相对应的适应度值。MFO 算法可以用下面的全局最优的三元组表示。

$$\mathrm{MFO}=\left(I,P,T\right) \tag{1}$$

函数 I 产生一个随机飞蛾种群及相应的适应度值。

$$I:\phi\to\left\{M,OM\right\} \tag{2}$$

函数 P 接受矩阵 M，并最终返回更新后的矩阵。

$$P:M\to M \tag{3}$$

如果满足终止准则，则函数 T 为真；如果不满足，则函数 T 为假。

$$T:M\to\{\text{true},\text{false}\} \tag{4}$$

函数 I 初始化后，P 函数迭代运行直到 T 函数为真。使用式(5)更新每只飞蛾相对于火焰的位置。

$$M_i=S\left(M_i,F_j\right) \tag{5}$$

这里的第 i 只飞蛾表示为 M_i，第 j 个火焰表示为 F_j，S 为螺旋函数。

MFO 算法中飞蛾的更新机制使用的是对数螺旋函数，其定义如下

$$S\left(M_i,F_j\right)=D_i\cdot \mathrm{e}^{bt}\cdot\cos\left(2\pi t\right)+F_j \tag{6}$$

第 i 个飞蛾与第 j 个火焰的距离在这里用 D_i 表示，b 为常数，t 为$[-1,1]$之间的随机数。

D 由式(7)计算求得

$$D_i=\left|F_j-M_i\right| \tag{7}$$

飞蛾的移动路径使用式(6)来进行模拟，并且还可以把飞蛾相的下一个位置确定下来。在式(6)中，参数 t 表示的是飞蛾在下个位置与火焰的远近程度($t=-1$ 表示为飞蛾最接近火焰，而 $t=1$ 表示为距离火焰最远)。在式(6)中需要飞蛾向火焰处移动，而这使得 MFO 算法极易陷入局部最优的停滞中。于是，使用式(6)来更新每个飞蛾的位置。在每次迭代更新火焰列表后，火焰根据其适应度值来排序。然后飞蛾更新其相对于相应火焰的位置。第一只飞蛾总是更新相对于最优火焰的位置，而最后那个飞蛾更新列表中最差火焰的位置。

对于飞蛾在位置更新的时候可能降低了对于解的开采，使用一种火焰数量的自适应机制，使用式(8)表示

$$flame\ no=round\left(N-l*\frac{N-1}{T}\right) \tag{8}$$

这里的 l 表示为当前的迭代次数，火焰数量的最大值为 N，而算法的最大迭代次数表示为 T。

式(8)表明在迭代初始步骤中存在数量为 N 的火焰。然而，在迭代的后期飞蛾仅使用最好的火焰更新它们的位置。飞蛾数量上的逐渐减少平衡了在搜索空间里的探测与开采。

标准 MFO 算法流程描述如下：

Step 1：初始化飞蛾种群 M、螺旋线等相关参数；

Step 2：随机生成飞蛾位置，由飞蛾种群排序得到火焰 F 及其适应度值 OF，迭代开始；

Step 3：由式(8)求出飞蛾数量，去掉不好的飞蛾及火焰；

Step 4：由式(7)求出飞蛾及对应火焰的距离 D_i；

Step 5：将 Step 4 求得结果与式(6)结合，求每只飞蛾更新之后的值；

Step 6：由新的飞蛾种群 M 计算出对应 OM；

Step 7：若满足结束条件，则停止迭代；若不满足，跳转到 Step2 继续执行。

2.2. 反向学习

反向学习(Opposition-Based Learning, OBL)是学者 Tizhoosh 在 2005 年提出的计算智能领域的一个新概念[14]，该方法已经被证实可以有效提高智能优化算法的搜索能力。反向学习表示为在当前个体所在区间上产生反向解个体，然后把产生的反向个体和当前群体放在一起参与择优选择，选择优秀的个体进入后序的迭代中，另外从概率论的方面也说明了反向解有高于 50%的概率远离问题的最优解[15]。下面给出反向解的定义。

定义 1：反向解(Opposite Solution, OS) [16]。如在$[a,b]$上有实数 x，x 的反向数用 $x'=a+b-x$ 表示。若有个 N 维点 $p=(x_1,x_2,\cdots,x_i,\cdots,x_N)$ 在 R 域上，$x_i\in[a_i,b_i]$，$p'=(x_1',x_2',\cdots,x_i',\cdots,x_N')$ 表示 p 的反向点，$x_i'=k*(a_i+b_i)-x_i$，k 为[0,1]区间上服从均匀分布的随机数。若适应度函数的可行解为 x，其反向解为 x'，当 $f(x)>f(x')$ 时，则用 x' 代替 x。

2.3. 精英反向学习

在标准 MFO 算法中引入反向解，使得算法的搜索范围较之以前扩大了很多。为有效提高算法的收敛速度，先求出当前解的反向解，再根据适应度值找出原解适应度值大于反向解适应度值的个体，并将其组成精英群体，然后在精英群体中生成新搜索空间，求出原解适应度值小于反向解适应度值个体的反向解。在算法找到最优解的时候，也必然会找到最优解所在的区域，然后在精英群体定义的区间上生成反向解，从而搜索到最优解处[17]。

定义 2 精英反向解(Elite Opposite Solution, EOS) [16] [18]。在 N 维空间中，当前群体的精英个体 $X_{best}=(x_1,x_2,\cdots,x_i,\cdots,x_N)$ 的反向解可表示为 $X'_{best}=(x_1',x_2',\cdots,x_i',\cdots,x_N')$，可定义成 $x_i'=k*(a_i+b_i)-x_i$，其中的 $x_i\in[a_i,b_i]$，$k\in[0,1]$ 为服从均匀分布的随机数，利用 k 可以产生精英个体的多个反向解，可以有效提高算法的开采能力。

2.4. 应用精英反向学习的飞蛾扑火优化算法

在标准 MFO 算法中，作为种群领导者的最优飞蛾如果陷入局部最优中去的话，将导致算法提前进入早熟，而使得算法搜索停滞。而对飞蛾进行反向学习之后，这种现象将在很大程度上得到规避。此外，在带有盲目性的反向学习中加入精英策略，不仅扩大了算法的搜索区域，有效减少盲目搜索带来的时间浪费，而且加快了算法的收敛速度。使得算法在每次迭代时，将一些精英个体进行反向学习，然后把生成的精英个体的反向种群放入其中参与竞争，这样做不仅可以保障算法具有较强的局部搜索能力，另外在性能上也有了较大的提升。

EOMFO 算法流程描述如下：

Step 1：使用精英反向学习策略初始化飞蛾种群 M、螺旋线等相关参数(选取策略概率 p = 0.7)；

Step 2：随机生成飞蛾位置，由飞蛾种群排序得到火焰 F 及其适应度值 OF，迭代开始；

Step 3：由式(8)求出飞蛾数量，去掉不好的飞蛾及火焰；

Step 4：由式(7)求出飞蛾及对应火焰的距离 D_i；

Step 5：将 Step 4 求得结果与式(6)结合，求每只飞蛾更新之后的值；

Step 6：由新的飞蛾种群 M 计算出对应 OM；

Step 7：若满足结束条件，则停止迭代；若不满足时，跳转到 Step2 继续执行。

3. 仿真实验

3.1. 实验环境与参数设置

为了验证新算法 EOMFO 的有效性，文章选取了 7 个测试函数。实验中算法参数设置为：种群规模为 30 个个体，最大迭代次数为 1000 次，函数维数设置为 30 维，最终测试结果采用独立运行 30 次最优值的平均值。

3.2. 测试函数

在这次仿真实验中共选取 7 个基准测试函数，其中包含有 4 个单峰函数(f_1 ~ f_4)和 3 个多峰函数

($f_5 \sim f_7$)。单峰函数因其只有一个全局最优值，可以对算法的开采能力进行测试，多峰函数具有较多的局部最优值，可以考验算法跳出局部最有停滞的能力。实验中用到的 7 个测试函数如下：

1) Sphere 函数

$f_1(x)=\sum_{i=1}^{n}x_i^2$， $-100\le x_i\le 100(i=1,2,\cdots,n;n=30)$，在$(0,0,\cdots,0)$处取得全局最小值 0。

2) Schwefel's 2.22 函数

$f_2(x)=\sum_{i=1}^{n}|x_i|+\prod_{i=1}^{n}|x_i|$， $-10\le x_i\le 10(i=1,2,\cdots,n;n=30)$，在$(0,0,\cdots,0)$处取得全局最小值 0。

3) Step 函数

$f_3(x)=\sum_{i=1}^{n}(\lfloor x_i+0.5\rfloor)^2$， $-100\le x_i\le 100(i=1,2,\cdots,n;n=30)$，在$(0,0,\cdots,0)$处取得全局最小值 0。

4) Quartic 函数

$f_4(x)=\sum_{i=1}^{n}ix_i^4+random(0,1)$， $-1.28\le x_i\le 1.28(i=1,2,\cdots,n;n=30)$，在$(0,0,\cdots,0)$处取得全局最小值 0。

5) Ackley 函数

$$f_5(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e,$$

$-32\le x_i\le 32(i=1,2,\cdots,n;n=30)$，在$(0,0,\cdots,0)$处取得全局最小值 0。

6) Griewangk 函数

$$f_6=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1,\quad -600\le x_i\le 600(i=1,2,\cdots,n;n=30)$$，在$(0,0,\cdots,0)$处取得全局最小值 0。

7) Penalized 1 函数

$$f_7(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_1-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_n-1)^2\right\}+\sum_{i=1}^{n}u(x_i,10,100,4)$$

$$-50\le x_i\le 50(i=1,2,\cdots,n;n=30),$$

$$y_i=1+\frac{x_i+1}{4};\quad u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$$

在$(0,0,\cdots,0)$处取得全局最小值 0。

3.3. 实验结果

实验结果如表 1 所示。

Table 1. Comparison of experimental results between MFO and EOMFO
表 1. MFO 与 EOMFO 实验结果比较

函数	算法	最优值	最差值	平均值	标准差
f_1	MFO	3.2261E+03	6.4131E+04	3.2377E+04	1.5163E+04
	EOMFO	1.8786E−08	2.0199E+00	1.5157E−01	4.0127E−01
f_2	MFO	7.1532E+01	2.7280E+02	1.5936E+02	4.2535E+01
	EOMFO	4.1848E−04	2.2238E−01	2.8888E−02	4.7675E−02

Continued

f_3	MFO	6.5860E+03	7.9729E+04	2.9463E+04	1.5410E+04
	EOMFO	6.3025E+00	3.1617E+01	1.5458E+01	7.6320E+00
f_4	MFO	3.0692E+01	4.8076E+02	1.5219E+02	1.1180E+02
	EOMFO	2.9033E−02	3.9697E−01	1.4490E−01	8.4964E−02
f_5	MFO	1.9145E+01	1.9967E+01	1.9850E+01	2.0687E−01
	EOMFO	3.0896E−04	1.9966E+01	5.2427E+00	7.2335E+00
f_6	MFO	2.8695E+01	6.2916E+02	2.7128E+02	1.3971E+02
	EOMFO	3.4562E−08	1.0622E+00	9.9621E−02	2.2445E−01
f_7	MFO	4.6106E+06	4.2425E+08	1.3560E+08	1.3240E+08
	EOMFO	1.0454E−01	3.0018E+01	2.7691E+00	5.7585E+00

从表 1 可知，单峰函数方面，对于函数 f_1，EOMFO 的最优值精确度达到 e^{-8}，平均值方面其求解精度比标准 MFO 提高了 5 个数量级，并且方差较小，说明新算法对于 f_1 的求解具有一定的稳定性。对于函数 f_2，EOMFO 的最优值精确度达到 e^{-4}，平均值和方差均达到了 e^{-2}，说明新算法对于 f_2 的求解具有较好的稳定性。对于函数 f_3，EOMFO 的最优值、最差值、平均值虽没有取得理想的成绩，但相较于标准 MFO 还是提高了 3 个数量级。对于函数 f_4，EOMFO 在最优值和方差方面的精确度达到 e^{-2}，说明新算法对其的寻优过程中具有良好的稳定性；另外，在平均值和最差值方面，新算法 EOMFO 的求解精度与标准 MFO 相比提高了 3 个数量级。

多峰函数方面，对于函数 f_5，EOMFO 在平均值方面比标准 MFO 只有少量的提升，但在最优值方面却达到了 e^{-4}，比标准 MFO 提高了 5 个数量级。对于函数 f_6，EOMFO 的最优值精度达到了 e^{-8}，比标准 MFO 提高了 9 个数量级，其平均求解精度也达到了 e^{-2}。对于函数 f_7，EOMFO 的最优值精度为 e^{-1}，虽没有取得很好的求解精度，但与标准 MFO 相比，也提高了 7 个数量级；另外，在最优值和方差方面，EOMFO 所求值均比原算法提高了 8 个数量级。由前面的比较结果可知，与标准 MFO 算法相比，EOMFO 算法搜索最优值的能力更好，求解的最优值也更接近理论最优值。

为了更好的对比标准 MFO 算法和 EOMFO 算法的性能，选取了 4 个函数的收敛曲线图，如图 1~4 所示。

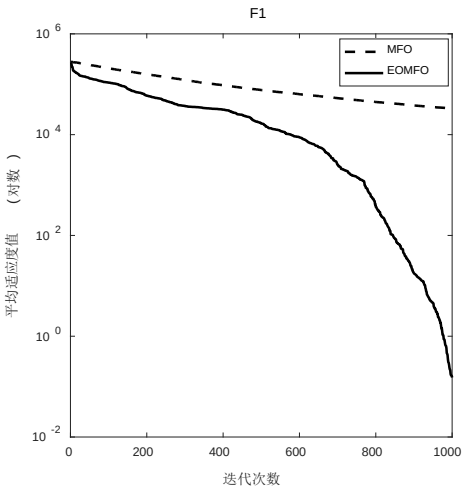

Figure 1. Curve of fitness for f_1
图 1. f_1 的适应度曲线

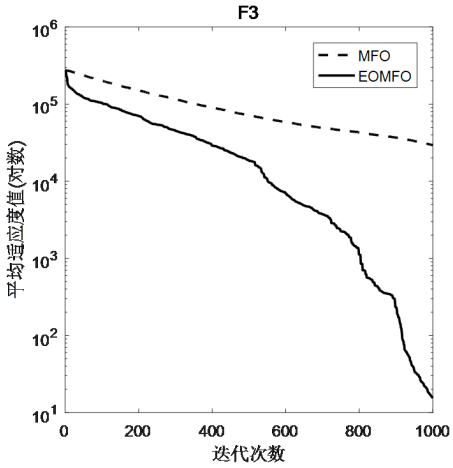

Figure 2. Curve of fitness for f_3
图 2. f_3 的适应度曲线

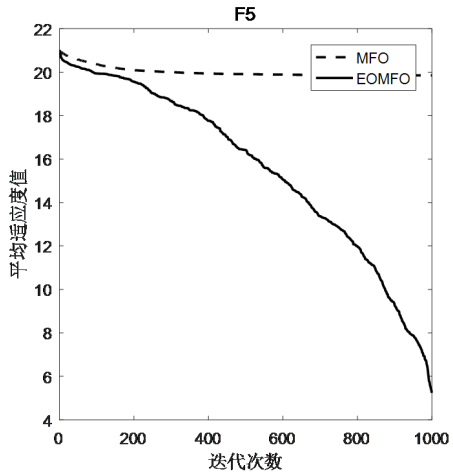

Figure 3. Curve of fitness for f_5
图 3. f_5 的适应度曲线

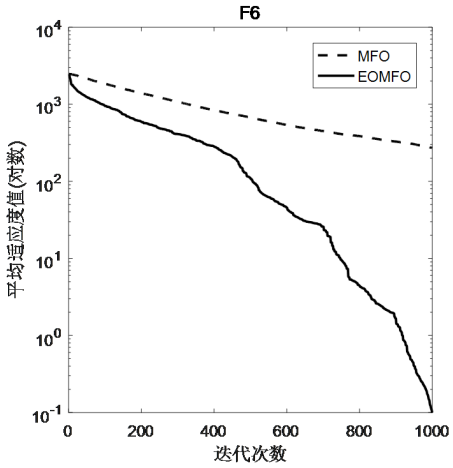

Figure 4. Curve of fitness for f_6
图 4. f_6 的适应度曲线

从以上 4 幅图可以看到 EOMFO 算法的收敛速度明显要高于标准 MFO 算法，且求解精度也有一定的提升。标准 MFO 算法在迭代一定的次数后便陷入局部最优而很难跳出，而 EOMFO 算法由于针对精英个体执行反向学习，使其具有比标准 MFO 算法更好的跳出局部最优的能力，从而能够更快的向理论最优值收敛，并且随着迭代次数的增长，所求结果的精度有进一步提升的可能。

4. 结束语

标准飞蛾扑火优化算法是一种近几年提出的新型算法，文中将精英反向学习加入到标准的飞蛾扑火优化算法当中。改进后的新算法 EOMFO 对于避免陷入局部最优具有一定的作用，可以更快地收敛到全局最优值，并且对于提高算法寻优速度和精度方面有很好的作用。从文中选取的 7 个函数优化的结果可知，EOMFO 算法不仅提高了进化速度，并且取得了更精确的函数值，说明 EOMFO 算法是可行的，并且是优于标准的 MFO 算法的。另外，EOMFO 算法在收敛速度等方面有不足，还需进一步研究。

基金项目

广西高校中青年教师基础能力提升项目(No. 2018KY0701, 2019KY0874)，广西科技厅基地和人才专项(No. AD16450003)。

参考文献

[1] Mirjalili, S. (2015) Moth-Flame Optimization Algorithm: A Novel Nature-Inspired Heuristic Paradigm. *Knowledge-Based Systems*, **89**, 228-249. https://doi.org/10.1016/j.knosys.2015.07.006

[2] Li, Z.M., Zhou, Y.Q. and Zhang, S. (2016) Lévy-Flight Moth-Flame Algorithm for Function Optimization and Engineering Design Problems. *Mathematical Problems in Engineering*, **2016**, 1-22. https://doi.org/10.1155/2016/1423930

[3] 李志明, 莫愿斌. 基于 Lévy 飞行的飞蛾扑火优化算法[J]. 计算机工程与设计, 2017, 38(3): 807-813.

[4] 李志明, 莫愿斌, 张森. 一种新颖的群智能算法:飞蛾扑火优化算法[J]. 电脑知识与技术, 2016, 12(31): 172-176.

[5] Zhang, S., Zhou, Y.Q. and Li, Z.M. (2016) Grey Wolf Optimizer for Unmanned Combat Aerial Vehicle Path Planning. *Advances in Engineering Software*, **99**, 121-136. https://doi.org/10.1016/j.advengsoft.2016.05.015

[6] Zhou, Y.q., Pan, W. and Li, Z.M. (2017) An Exponential Function Inflation Size of Multi-Verse Optimization Algorithm for Global Optimization. *International Journal of Computing Science and Mathematics*, **8**, 115-128. https://doi.org/10.1504/IJCSM.2017.083758

[7] 刘小龙. 基于统计指导的飞蛾扑火算法求解大规模优化问题[J]. 控制与决策, 2020, 35(4): 901-908.

[8] 吴伟民, 李泽熊, 林志毅. 改进飞蛾捕焰算法在网络流量预测中的应用[J]. 计算机工程, 2017, 43(10): 153-159+166.

[9] 王娇, 李琦, 高军萍. 基于改进飞蛾扑火优化算法的 MIMO 雷达相位编码信号集设计[J]. 信息与控制, 2019, 48(3): 279-284.

[10] 李伟琨, 阙波, 王万良. 基于多目标飞蛾算法的电力系统无功优化研究[J]. 计算机科学, 2017(S2): 513-519.

[11] 高帅, 胡红萍, 李洋. 基于 MFO-SVM 的空气质量指数预测[J]. 中北大学学报(自然科学版), 2018, 39(4): 373-379.

[12] 徐慧, 方策, 刘翔. 改进的飞蛾扑火优化算法在网络入侵检测系统中的应用[J]. 计算机应用, 2018,38(11): 3231-3235+3240.

[13] 岳龙飞, 杨任农, 张一杰. Tent 混沌和模拟退火改进的飞蛾扑火优化算法[J]. 哈尔滨工业大学学报, 2019, 51(5): 146-154.

[14] Tizhoosh, H.R. (2005) Opposition-Based Learning: A New Scheme for Machine Intelligence. *Proceedings of IEEE International Conference on Intelligent Agents, Web Technologies & Internet Commerce*, **1**, 695-701.

[15] 郭旭, 贺兴时, 高昂. 基于精英反向学习的混沌蝙蝠算法[J]. 计算机与数字工程, 2019, 47(4): 773-777+897.

[16] 王培崇, 高文超, 钱旭. 应用精英反向学习的混合烟花爆炸优化算法[J]. 计算机应用, 2014, 34(10): 2886-2890.

[17] 魏伟一, 文雅宏. 一种精英反向学习的萤火虫优化算法[J]. 智能系统学报, 2017, 12(5): 710-716.

[18] 周新宇, 吴志健, 王晖. 一种精英反向学习的粒子群优化算法[J]. 电子学报, 2013, 41(8): 1647-1652.