

DOI:10.11992/tis.201706014

网络出版地址: <http://kns.cnki.net/kcms/detail/23.1538.TP.20170831.1058.006.html>

# 一种精英反向学习的萤火虫优化算法

魏伟一, 文雅宏

(西北师范大学 计算机科学与工程学院, 甘肃 兰州 730070)

**摘要:** 为了提高传统萤火虫算法的收敛速度和求解精度, 提出了一种精英反向学习的萤火虫优化算法。通过反向学习策略构造精英群体, 在精英群体构成的区间上求普通群体的反向解, 增加了群体的多样性, 提高了算法的收敛速度; 同时, 为了避免最优个体陷入局部最优, 使整个群体在搜索过程中出现停滞, 提出了差分演化变异策略; 最后, 提出了一种线性递减的自适应步长来平衡算法的开发能力。实验结果表明, 算法在收敛速度和收敛精度上有更好的效果。

**关键词:** 萤火虫算法; 精英反向学习; 优化算法; 精英群体; 反向解; 反向学习策略; 差分演化变异; 自适应步长

**中图分类号:** TP309.2 **文献标志码:** A **文章编号:** 1673-4785(2017)05-0710-07

中文引用格式: 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.

英文引用格式: WEI Weiyi, WEN Yahong. Firefly optimization algorithm utilizing elite opposition-based learning[J]. CAAI transactions on intelligent systems, 2017, 12(5): 710-716.

## Firefly optimization algorithm utilizing elite opposition-based learning

WEI Weiyi, WEN Yahong

(College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China)

**Abstract:** To increase the convergence speed and solution accuracy of the traditional firefly algorithm, in this paper, we propose a firefly optimization algorithm that utilizes elite opposition-based learning. Using an opposition-based learning strategy, we constructed an elite group and, in the interval of the elite group, we solved the opposite solutions of the ordinary groups. This strategy could increase group diversity and improve the convergence speed of the algorithm. To prevent the optimal individual from falling into the local optimum, which could cause stagnation of the whole group during the search process, we introduce a differential evolutionary mutation strategy. Finally, we propose an adaptive step size with a linear decrease to balance the development ability of the algorithm. Experimental results show that the proposed algorithm can increase convergence speed and accuracy.

**Keywords:** firefly algorithm; elite opposition-based learning; optimized algorithm; elite group; opposite solutions; opposition-based learning strategy; differential evolutionary mutation; adaptive step size

萤火虫算法 (firefly algorithm, FA) 是受自然界中萤火虫发光特性的启发, 由剑桥学者 Yang 于 2008 年提出的一种群体智能随机优化算法<sup>[1-5]</sup>。在多个科学与工程领域中, 萤火虫算法已得到成功的应用<sup>[6-9]</sup>, 虽然 FA 表现出了良好的性能, 但在一些

问题的优化上, 萤火虫算法依然存在收敛速度慢、解的精度不高、容易陷入局部最优等不足。近年来, 很多学者已经进行了多角度的改进。文献[10]为了解决萤火虫算法过早地收敛和陷入局部最优的不足, 利用广义反向学习策略来优化萤火虫算法。文献[11]采用正交学习策略改进 FA 算法, 利用精英萤火虫来构造指导向量, 通过指导向量引导群体向全局最优区域移动。文献[12]提出基于蛙

收稿日期: 2017-06-07. 网络出版日期: 2017-08-31.

基金项目: 甘肃省科技计划资助项目 (1506RJZA130); 甘肃省高等学校科研项目 (2014B-018).

通信作者: 文雅宏. E-mail: wwyahong@126.com.

跳的萤火虫算法,在原始的 FA 算法中引入蛙跳算法中的分群。同时,为了加强算法的局部开发能力,引入了模拟退火的思想。该算法对于高维多模态函数的优化问题,表现得还不够理想。文献[13]提出了一种基于多种群学习机制的萤火虫优化算法,把萤火虫分为不同的子群,同时,子群建立学习机制,实现不同子群间的信息交流,完成局部和全局的寻优。文献[14]引入模式搜索思想,把 FA 算法与模式搜索相结合,FA 算法具有较强的全局搜索能力,模式搜索具有较好的局部搜索能力,利用两者的优势来提高 FA 算法的性能。文献[15]针对高维问题,提出了多维反向学习的萤火虫算法,用反向学习策略初始化萤火虫种群。同时,用基于多维的方法更新不同维度上萤火虫的位置。算法在收敛速度和精度上比原始萤火虫算法更优。

以上文献虽然对 FA 算法做了很好的改进,但是在收敛速度和精度上还不够理想,为了更好地提高 FA 算法的收敛速度和收敛精度,本文基于文献[16]利用精英反向学习策略来改进差分演化算法的思想,提出了一种精英反向学习的萤火虫算法,在文献[16]中,通过设置一个参数来选取精英个体,而本文根据原解和反向解适应度值的大小选取精英个体,这样能更充分地利用精英群体的良好信息,提高算法的收敛速度。同时,本文采用了差分演化策略(differential evolutionary mutation)来增强算法的局部搜索能力。最后,为了增强和平衡算法的开发能力,本文提出了一种线性递减的自适应步长。在5个标准测试函数上进行实验,并和多个改进的 FA 算法进行实验对比。结果表明,本文算法在收敛速度和收敛精度上更好。

## 1 萤火虫算法(FA)

FA 是受自然界中萤火虫个体通过发光来吸引同伴求偶或觅食行为的启发而提出的一种元启发式算法<sup>[1-2]</sup>,萤火虫之间相互吸引以及位置迭代更新的过程是搜索和优化的过程。寻找最亮萤火虫的问题是求解最优值的问题,不断用最好的位置替换较差的位置来完成整个搜索过程。在一定的搜索区域内所有发光的弱的萤火虫向发光的强的萤火虫移动,从而实现位置寻优<sup>[17]</sup>。每个萤火虫被看作是一个个体,个体主要有“位置、亮度、吸引度”等属性,

有两个重要的影响因子,即亮度  $I$  和吸引度  $\beta$ 。亮度高说明其所处位置好,并吸引亮度低的个体向其靠近。吸引度高则萤火虫移动的距离大。从 FA 开始,萤火虫的个体随机地分布在指定的局域内,个体的亮度由目标函数决定。

设  $I_0$  表示萤火虫个体的固有亮度,  $\gamma$  为介质的光亮度吸收系数,  $r_{ij}$  为任意两个个体  $i$  和  $j$  的相对距离(一般使用欧氏距离),  $\beta_0$  为萤火虫个体固有吸引度,随距离  $r$  变化的个体光强度  $I$  表示为

$$I = I_0 e^{-\gamma r^2} \quad (1)$$

则萤火虫  $i$  与萤火虫  $j$  之间的相互吸引力计算公式为

$$\beta = \beta_0 \times e^{-\gamma r_{ij}} \quad (2)$$

设  $x_i(t)$  和  $x_j(t)$  分别表示萤火虫  $i$  和  $j$  在  $t$  时刻的位置,则两者之间的距离计算公式为

$$r_{ij} = \| \mathbf{x}_i(t) - \mathbf{x}_j(t) \|_2 = \sqrt{\sum_{m=1}^d (x_{im} - x_{jm})^2} \quad (3)$$

萤火虫  $i$  向萤火虫  $j$  移动,其位置更新方程为

$$x_{i+1}(t+1) = x_i(t) + \beta(x_j(t) - x_i(t)) + \alpha \left( \text{rand} - \frac{1}{2} \right) \quad (4)$$

式中:  $x_{i+1}(t+1)$  表示萤火虫  $i$  在  $t+1$  时刻的位置;  $\alpha \in [0, 1]$ , 表示步长因子。

## 2 精英反向学习的萤火虫算法

### 2.1 反向学习策略

反向学习策略是近年来计算智能领域出现的新概念<sup>[18-19]</sup>,其主要思想是对一个问题的可行解,求其反向解,并对原解和反向解进行评估,从中选出较优的解作为下一代个体。其中反向点和反向解的定义如下。

**定义 1** 反向点(opposite point, OP)<sup>[18]</sup>。设  $x = (x_1, x_2, \dots, x_D)$  是  $D$  维空间中的一个点,且  $x_1, x_2, \dots, x_D \in \mathbf{R}, x_j \in [a_j, b_j]$ , 则  $x$  对应的反向点  $x^* = (x_1^*, x_2^*, \dots, x_D^*)$  定义为

$$x_j^* = a_j + b_j - x_j \quad (5)$$

**定义 2** 动态反向学习策略(dynamic opposition based learning, DOBL)<sup>[18]</sup>。设  $x_i(t) = (x_{i1}(t), x_{i2}(t), \dots, x_{iD}(t))$  为问题第  $t$  代的一个可行解,  $x_{ij}(t)$  是其  $j$  维上的分量,  $x_{ij}^*(t)$  是  $x_{ij}(t)$  对应的反向解,则  $x_i^*(t)$  是

$x_i(t)$ 对应的反向解,其中

$$x_{ij}^*(t) = k(a_j(t) + b_j(t)) - x_{ij}(t) \quad (6)$$

式中:  $a(t) = \min(x_{ij}(t))$ ,  $b(t) = \max(x_{ij}(t))$  为当前搜索区域的最小值和最大值,其随着迭代的改变,而发生变化;  $i \in [1, n]$ ,  $j \in [1, D]$ ;  $n$  是种群大小;  $D$  是解空间的维数;  $k$  是介于 0~1 的随机数。

## 2.2 精英反向学习

反向解的引入,可以扩大算法的搜索区域,但那些原解适应度值大于反向解适应度值的个体,对其进行反向区域的搜索,浪费时间,则应加强其领域搜索。而对原解适应度值小于反向解适应度值的个体,对其进行反向区域的搜索价值要高于其领域的开发价值。因此,本文将原解适应度值小于反向解适应度值的个体作为研究对象,求其反向解,既可以扩大搜索区域,也能有效避免盲目搜索带来的时间浪费。

同时,本文为了提高算法的收敛速度,首先在当前解所构造的空间中,求所有当前解的反向解;然后,通过比较适应度值,选出那些原解适应度值大于反向解适应度值的个体组成精英群体;最后,在精英群体构造的新的搜索空间上,再求原解适应度值小于反向解适应度值的个体的反向解。如果算法能收敛到全局最优解,则精英群体所形成的搜索区间必将收敛到最优解所在的区域<sup>[16]</sup>,这样充分利用了精英群体的有效信息,在精英群体所构成的动态定义区间上生成反向解,引导搜索向最优解靠近。

**定义3** 精英(elite)。设  $x_i(t) = (x_{i1}, x_{i2}, \dots, x_{iD})$  是第  $t$  次迭代的一个解,其反向解为  $x_i^*(t)$ ,  $f(x)$  为目标函数。当  $f(x_i(t)) \geq f(x_i^*(t))$  时,称  $x_i(t)$  为第  $t$  次迭代的精英个体,记为  $N_i(t)$ ; 当  $f(x_i(t)) < f(x_i^*(t))$  时,称  $x_i(t)$  为第  $t$  次迭代的普通个体,记为  $Q_i(t)$ 。若精英群体的规模为  $p$  ( $1 < p \leq n$ ,  $n$  为解的总个数) 时,则  $p$  个精英个体可表示为  $\{N_1(t), N_2(t), \dots, N_p(t)\} \subseteq \{x_1(t), x_2(t), \dots, x_n(t)\}$ 。

**定义4** 精英反向解(elite opposite solution)<sup>[18]</sup>。设  $x_{ij}$  为普通个体  $x_i$  在  $j$  维上的值,则其反向解可定义为

$$x_{ij}^*(t) = k(a_j(t) + b_j(t)) - x_{ij}(t) \quad (7)$$

式中:  $k$  是介于 0~1 的随机数;  $a_j(t) = \min(N_{1j}(t), N_{2j}(t), \dots, N_{pj}(t))$ ;  $b_j(t) = \max(N_{1j}(t), N_{2j}(t), \dots,$

$N_{pj}(t))$ 。  $[a_j(t), b_j(t)]$  为精英群体所构造的区间,当反向解越过边界  $[a_j(t), b_j(t)]$  时,可以用下列方式进行重置:

$$\begin{cases} x_{ij}^* = a_j(t), & x_{ij}^* < a_j(t) \\ x_{ij}^* = b_j(t), & x_{ij}^* > b_j(t) \end{cases} \quad (8)$$

## 2.3 差分演化变异策略

在 FA 中,群体中的最优个体  $x_{\text{best}}$  引导群体向最优方向移动,如果  $x_{\text{best}}$  陷入局部最优,群体的移动终止,即收敛到局部最优,则群体无法到达全局最优。因此,本文为了求得全局最优解,引入差分变异策略,对  $x_{\text{best}}$  进行变异操作,使其陷入局部最优的概率减小。

本文要对最优个体进行变异操作,使其跳出局部最优的概率增大,因此选择“DE/best/1”作为变异操作,公式为

$$c_{ij} = x_{\text{best},j} + F \cdot (x_{n_1,j} - x_{n_2,j}) \quad (9)$$

式中:  $x_{\text{best},j}$  为最优个体的第  $j$  维;  $F$  是缩放系数;  $n_1$ 、 $n_2$  是  $[1, n]$  上两个互不相同的随机整数,代表不同个体的下标;  $j$  是维度;  $c_{ij}$  是变异后的值。

将变异后的个体和父代个体进行如下交叉操作:

$$\bar{x}_{\text{best},j} = \begin{cases} c_{ij}, & \text{rand}[0,1] \leq \text{CR 或者 } j = \text{rand}(1,D) \\ x_{\text{best},j}, & \text{其他} \end{cases} \quad (10)$$

式中:  $\bar{x}_{\text{best},j}$  为新生成个体的第  $j$  维;  $\text{rand}[0,1]$  是  $[0,1]$  的随机数; CR 是交叉概率;  $\text{rand}(1,D)$  是  $[1, D]$  的一个随机整数;参数  $F$ 、CR 分别设为 1 和 0.1。

## 2.4 自适应步长

在原始萤火虫算法 FA 中,步长因子  $\alpha$  在每次迭代时保持不变。但是当  $\alpha$  取较大的值时,增强了算法的全局搜索能力,降低了算法的收敛速度和搜索的精度;当  $\alpha$  取较小的值时,有利于算法的局部搜索,提高了搜索精度和算法的收敛速度。在算法迭代前期,较大的  $\alpha$  有利于算法的全局搜索;在后期,较小的  $\alpha$  显得更有利。因此本文对  $\alpha$  采用动态递减的方式,计算公式为

$$\alpha_{t+1} = \alpha_t \cdot \left( \frac{T - t}{T} \right), t = 1, 2, \dots, T \quad (11)$$

式中  $t$  为迭代次数。

## 2.5 EOFA 算法描述

EOFA 算法流程如下。

输入 目标函数和搜索空间;

输出 全局最优解和最优位置。

1) 初始化参数  $m, n, T, \alpha, \beta, \gamma$ , 在  $[m, n]$  上生成初始种群  $x_i (i=1, 2, \dots, n)$ 。

2) 执行 EOFA 算法搜索。

3) 把  $x_i(t) (i=1, 2, \dots, n)$  带入目标函数, 计算函数值, 把目标函数的值作为每个个体的亮度值  $I_i(t)$ 。

4) 在  $[m, n]$  生成  $x_i(t)$  的反向解  $x_i^*(t)$ , 并计算亮度  $I_i^*(t)$ 。对  $I_i(t)$  和  $I_i^*(t)$  进行比较, if  $I_i(t) > I_i^*(t)$ , 则  $x_i(t)$  为精英个体  $N_i(t)$ , 记精英群体大小为  $p (p > 1, i=1, 2, \dots, p)$ 。设精英个体的区间范围为  $[a_j(t), b_j(t)]$  (若精英群体的规模小于 2, 则在  $x_i(t) (i=1, 2, \dots, n)$  构成的区间上求  $x_i (i=1, 2, \dots, n)$  的反向解)。else  $x_i(t)$  是普通个体, 普通群体大小为  $n-p$ 。

5) 用式(7)在精英个体构成的区间  $[a_j(t), b_j(t)]$  上计算普通群体的反向解  $x_i'(t) (i=1, \dots, n-p)$ 。

6) 精英群体和普通群体的反向解群体构成当前新种群, 计算新种群的亮度, 并进行排序, 选出最优的个体  $x_{\text{best}}(t)$ 。

7) 用式(3)计算每个个体  $i$  和最优个体  $x_{\text{best}}(t)$  之间的距离  $R_{ij}(t)$ 。

8) 用式(2)计算吸引力  $\beta_{ij}(t)$ 。个体  $i$  向最优个体  $x_{\text{best}}(t)$  移动, 用公式(6)更新位置  $x_i^{\text{new}}(t)$ 。

9) 用式(11)计算  $\alpha(t)$ , 并用式(9)、(10)对最优个体进行位置扰动。

10) 算法搜索结束, 输出全局最优解和最优位置。

若种群的规模为  $n$ , 空间维度为  $D$ , 则种群初始化的时间复杂度为  $O(nD)$ ; 从迭代开始到结束的整个过程中, 迭代的次数为  $t$ , 其中 3) 是计算种群的亮度, 复杂度为  $O(nDt)$ ; 4) ~ 9) 是建立新的种群, 并进行位置的更新, 复杂度为  $O((6n-p) \cdot Dt)$ ,  $p (p \leq n)$  是精英群体的规模。因此本文算法的时间复杂度为  $O(nDt)$ 。

### 3 实验仿真及分析

#### 3.1 测试函数

在仿真实验中, 本文采用下列 5 个常用的标准测试函数对算法进行测试。

##### 1) Sphere 函数

$$f_1(x) = \sum_{i=1}^d x_i^2, x_i \in (-5.12, 5.12)$$

Sphere 函数为多维单峰值函数, 在点  $x = (0, 0, \dots, 0)$  处取得极小值 0。

##### 2) Rosenbrock 函数

$$f_2(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2],$$

$$x_i \in (-2.048, 2.048)$$

Rosenbrock 函数为多维病态二次函数, 在点  $x = (1, 1, \dots, 1)$  处取得全局极小值 0。

##### 3) Ackley 函数

$$f_3(x) = -20e^{-0.2\sqrt{\frac{1}{d}\sum_{i=1}^d x_i^2}} - e^{\frac{1}{d}\sum_{i=1}^d \cos(2\pi x_i)} + 22.7128,$$

$$x_i \in (-32.7, 32.7)$$

Ackley 函数为多维多峰值函数, 在点  $x = (0, 0, \dots, 0)$  处取得全局极小值 0。

##### 4) Griewank 函数

$$f_4(x) = \sum_{i=1}^d \frac{x_i^2}{4000} - \prod_{i=1}^d \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$

$$x_i \in (-600, 600)$$

Griewank 函数为多维多峰值函数, 在点  $x = (0, 0, \dots, 0)$  处取得全局极小值 0。

##### 5) Rastrigin 函数

$$f_5(x) = 10d + \sum_{i=1}^d [x_i^2 - 10\cos(2\pi x_i)],$$

$$x_i \in (-5.12, 5.12)$$

Rastrigin 函数为多维多峰值函数, 在点  $x = (0, 0, \dots, 0)$  处取得全局极小值 0。

#### 3.2 EOFA 算法的测试结果

实验环境为: Inter Core(TM) i5-2450M CPU@ 2.50 GHz, 内存 4 GB, Window7 操作系统, MATLAB 7.8.0 版本。分别选取标准的 FA 算法<sup>[1]</sup>, LFA 算法<sup>[20]</sup>, MFA 算法<sup>[21]</sup>与本文提出的 EOFA 算法在 5 种标准的测试函数上进行实验比较, 种群规模  $n$  取 40, 初始  $\alpha$  取值为 0.98。维度  $D$  取 10 和 30,  $\gamma=1$ , MFA 算法中方向向量的个数  $m$  取 30, 其他参数分别取  $T=1000, \beta=1$ 。分别记录 4 种算法迭代 1000 次并在测试函数上独立运行 40 次的最优值、最差值和平均值, 结果如表 1 所示。



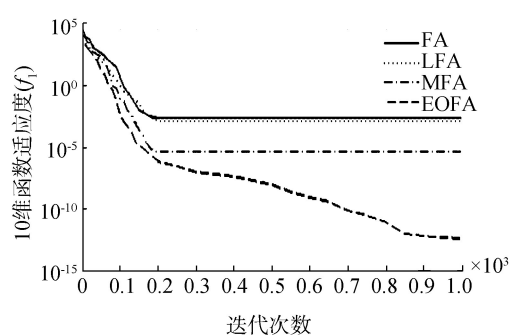
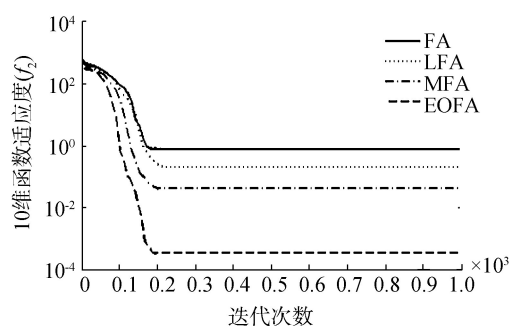
表1 4种算法的实验结果

Table 1 Experimental results of four algorithms

函数	算法	10 维			30 维		
		最优值	最差值	平均值	最优值	最差值	平均值
$f_1$	FA	$2.750\ 2\times 10^{-3}$	$6.293\ 8\times 10^{-3}$	$4.212\ 0\times 10^{-3}$	$3.786\ 1\times 10^{-1}$	$8.983\ 2\times 10^{-1}$	$5.092\ 1\times 10^{-1}$
	MFA	$3.811\times 10^{-6}$	$4.531\ 7\times 10^{-6}$	$3.671\ 4\times 10^{-6}$	$4.230\ 6\times 10^{-2}$	$7.464\ 4\times 10^{-2}$	$5.418\ 2\times 10^{-2}$
	LFA	$1.714\ 2\times 10^{-3}$	$7.932\ 5\times 10^{-3}$	$4.521\ 4\times 10^{-3}$	$1.741\ 2\times 10^{-1}$	1.290 1	$7.109\ 4\times 10^{-1}$
	EOFA	$3.794\ 1\times 10^{-13}$	$2.397\ 2\times 10^{-12}$	$2.006\ 9\times 10^{-13}$	$1.009\ 2\times 10^{-9}$	$3.143\ 4\times 10^{-9}$	$2.004\ 1\times 10^{-9}$
$f_2$	FA	$7.742\ 1\times 10^{-1}$	$8.183\ 7\times 10^{-1}$	$7.922\ 4\times 10^{-1}$	2.901 3	5.395 3	3.039 1
	MFA	$4.092\ 8\times 10^{-2}$	$1.092\ 8\times 10^{-1}$	$1\ 342\ 0\times 10^{-1}$	1.008 5	4.903 2	2.605 9
	LFA	$2.049\ 2\times 10^{-1}$	$8.495\ 2\times 10^{-1}$	$4.118\ 2\times 10^{-1}$	1.119 6	8.901 3	4.908 1
	EOFA	$3.292\ 0\times 10^{-4}$	$5.389\ 8\times 10^{-4}$	$4.253\ 9\times 10^{-4}$	$1.804\ 6\times 10^{-2}$	$2.233\ 3\times 10^{-2}$	$2.011\ 0\times 10^{-2}$
$f_3$	FA	$2.395\times 10^{-3}$	$5.091\ 9\times 10^{-3}$	$3.318\ 3\times 10^{-3}$	$4.394\ 2\times 10^{-1}$	$7.039\ 2\times 10^{-1}$	$5.309\ 2\times 10^{-1}$
	MFA	$7.347\ 8\times 10^{-6}$	$8.872\ 1\times 10^{-6}$	$7.813\ 4\times 10^{-6}$	$3.731\ 4\times 10^{-2}$	$5.093\ 0\times 10^{-2}$	$4.209\ 1\times 10^{-2}$
	LFA	$7.351\ 7\times 10^{-3}$	$2.920\ 5\times 10^{-2}$	$1.451\ 8\times 10^{-2}$	$2.397\ 1\times 10^{-1}$	$6.783\ 6\times 10^{-1}$	$3.024\ 0\times 10^{-1}$
	EOFA	$2.976\ 6\times 10^{-8}$	$5.473\ 0\times 10^{-8}$	$3.461\ 3\times 10^{-8}$	$1.341\ 0\times 10^{-6}$	$1.789\ 2\times 10^{-6}$	$1.575\ 3\times 10^{-6}$
$f_4$	FA	$4.472\ 0\times 10^{-3}$	$3.095\ 1\times 10^{-2}$	$1.977\ 1\times 10^{-2}$	$3.050\ 1\times 10^{-1}$	$5.908\ 1\times 10^{-1}$	$4.301\ 8\times 10^{-1}$
	MFA	$1.097\ 8\times 10^{-4}$	$4.092\ 8\times 10^{-4}$	$2.498\ 0\times 10^{-4}$	$3.095\ 8\times 10^{-2}$	$5.096\ 8\times 10^{-2}$	$4.410\ 3\times 10^{-2}$
	LFA	$3.119\ 2\times 10^{-3}$	$8.093\ 7\times 10^{-3}$	$5.428\ 3\times 10^{-3}$	$2.197\ 6\times 10^{-1}$	$5.892\ 1\times 10^{-1}$	$3.834\ 9\times 10^{-1}$
	EOFA	$1.041\ 1\times 10^{-9}$	$6.199\ 5\times 10^{-9}$	$3.620\ 3\times 10^{-9}$	$1.216\ 9\times 10^{-6}$	$3.900\ 1\times 10^{-6}$	$2.710\ 1\times 10^{-6}$
$f_5$	FA	$3.096\ 7\times 10^{-1}$	$6.318\ 7\times 10^{-1}$	$4.830\ 1\times 10^{-1}$	5.543 0	7.102 5	6.501 8
	MFA	$4.824\ 1\times 10^{-2}$	$2.871\ 0\times 10^{-1}$	$1.302\ 7\times 10^{-1}$	3.981 6	5.118 3	4.151 0
	LFA	$6.338\ 7\times 10^{-1}$	$7.201\ 4\times 10^{-1}$	$6.521\ 1\times 10^{-1}$	5.908 2	9.220 5	7.209 8
	EOFA	$2.068\ 9\times 10^{-10}$	$4.090\ 0\times 10^{-10}$	$3.132\ 5\times 10^{-10}$	5.879 8e-08	$8.326\ 8\times 10^{-6}$	$4.503\ 6\times 10^{-6}$

FA 是原始的萤火虫算法。LFA 是根据 Levy 分布来设置一种随机步长对传统萤火虫算法进行改进,其主要优点是算法收敛到局部最优的概率降低。MFA 算法是从随机生成的方向向量中选择使种群进化到最优的方向向量,方向向量的个数对算法的性能有很大的影响,数量越大,算法收敛性越好。由表1可知,EOFA、LFA 和 MFA 算法在10维和30维函数上都优于FA算法。本文提出的EOFA算法在5种测试函数上的函数值都小于FA、LFA、MFA 算法在测试函数上的值,即EOFA算法的收敛性更好,在每个测试函数上EOFA算法的求解精度比其他3种算法都高。

为了更好地验证EOFA算法的有效性,本文用图描述4种算法的收敛性,由于受篇幅的限制,仅给出4个代表性的函数收敛曲线图,结果如图1、2所示。

(a)  $f_1$ : Sphere 函数(b)  $f_2$ : Rosenbrock 函数

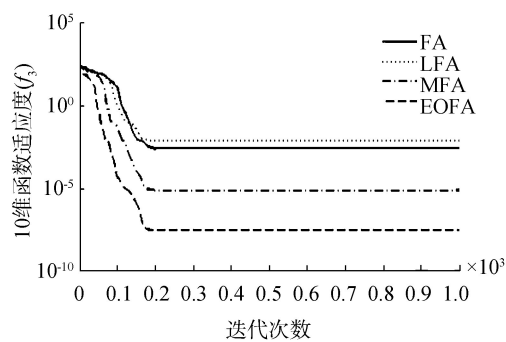
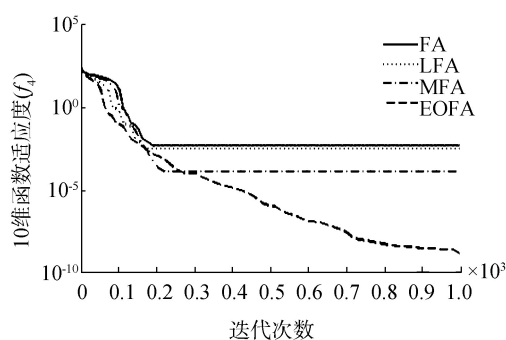
(c)  $f_3$ : Ackley 函数(d)  $f_4$ : Griewank 函数

图1 维度为10时算法收敛曲线对比  
Fig. 1 Comparison of convergent graphs for a dimensionality of 10

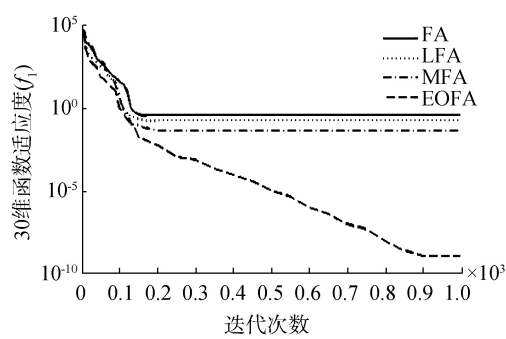
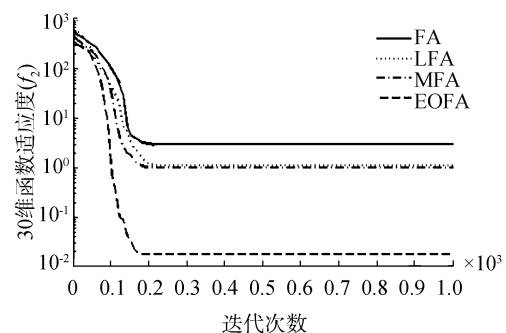
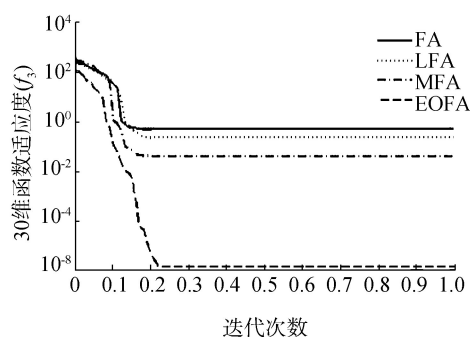
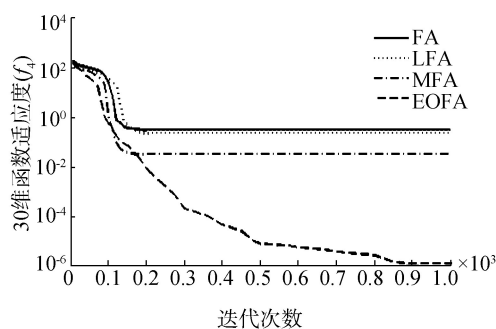
(a)  $f_1$ : Sphere 函数(b)  $f_2$ : Rosenbrock 函数(c)  $f_3$ : Ackley 函数(d)  $f_4$ : Griewank 函数

图2 维度为30时算法收敛曲线对比

Fig.2 Comparison of convergent graphs for a dimensionality of 30

由图1、2可以看出,对于每一个测试函数,EOFA算法总比其他3种算法表现出更好的收敛性,因为它构建了动态的精英反向解区间,同时,精英群体的规模自适应的改变,使普通个体向最优个体移动的速度加快。对于函数 $f_1$ 、 $f_4$ ,当函数维度为10和30时,FA、LFA、MFA这3种算法出现了早熟,而EOFA算法继续收敛,且收敛速度比其他3种算法都快。对于函数 $f_2$ 、 $f_3$ 虽然4种算法都出现了早熟,但EOFA算法解的精确度比其他算法更好。当维数从10增加到30时,4种算法的性能都有所下降,但EOFA算法的性能优于其他3种算法。EOFA算法具有较优性能的原因是:首先EOFA算法采用反向学习策略,构造精英群体和普通群体,扩大了搜索范围,通过生成每个个体的反向解,增加解的多样性来提高种群多样性。同时加入了差分演化变异策略,使其跳出局部最优的概率增大。最后,为了增强算法开发能力,采用递减的自适应步长。

## 4 结束语

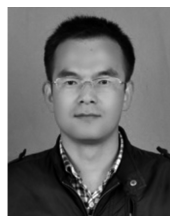
本文提出的精英反向学习萤火虫算法(EOFA),通过精英反向学习策略生成当前解的反向解,评估当前解和反向解,构建精英群体和普通群体,增加了群体的多样性;在动态的精英区间上求普通群体的反向解,提高算法的收敛速度。差分演化变异策略对最优个体进行变异操作,对其领域空间进行搜索,增强了EOFA的局部开采能力。同时,采用自适应步长,提高和平衡算法的开发能力。

通过实验结果得出,EOFA 算法在解的精度和收敛速度上都表现出更好的性能。本文只考虑了最优个体对每个个体的影响,下一步工作是将个体邻域的信息加入,进一步提高算法性能。

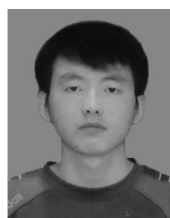
## 参考文献:

- [1] YANG X S. Firefly algorithms for multimodal optimization [C]//Stochastic Algorithms: Foundations and Applications. Sapporo, Japan, 2009, 5792: 169-178.
- [2] 朱书伟,周治平,张道文. 融合并行混沌萤火虫算法的K-调和均值聚类[J]. 智能系统学报, 2015, 10(6): 872-880.  
ZHU Shuxin, ZHOU Zhiping, ZHANG Daowen. Karm-  
onic means clustering merged with parallel chaotic firefly  
algorithm[J]. CAAI transactions on intelligent systems,  
2015, 37(2): 342-347.
- [3] 赵杰, 雷秀娟, 吴振强. 基于最优类中心扰动的萤火虫聚  
类算法[J]. 计算机工程与科学, 2015, 37(2):  
342-347.  
ZHAO Jie, LEI Xiujuan, WU Zhenqiang. A clustering al-  
gorithm for Fireflies based on optimal class center per-  
turbation[J]. Computer engineering & science, 2015, 37  
(2): 342-347.
- [4] 莫愿斌, 马彦迪, 郑巧燕. 单纯形法的改进萤火虫算法及  
其在非线性方程组求解中的应用[J]. 智能系统学报,  
2014, 9(6): 747-755.  
MO Yuanbin, MA Yanzhui, ZHENG Qiaoyan. Improved  
firefly algorithm based on simplex method and its appli-  
cation in solving non-linear equation groups[J]. CAAI tr-  
ansactions on intelligent systems, 2014, 9(6): 747-755.
- [5] HORNG M H. Vector quantization using the firefly al-  
gorithm for image compression[J]. Expert systems with  
applications, 2012, 39(1): 1078-1091.
- [6] MARICHEL VAM M K, PRABAHARAN T, YANG X S. A  
discrete firefly algorithm for the multi-objective hy-brid flow  
shop scheduling problems [J]. IEEE transactio-ns on  
evolutionary computation, 2014, 18(2): 301-305.
- [7] YANG X S, HOSSEINI S S S, GANDOMI A. Firefly  
algorithm for solving non-convex economic dispatch pr-  
oblems with valve loading effect [J]. Applied soft com-  
puting, 2012, 12(3): 1180-1186.
- [8] SENTHILNATH J, OMKAR S, MANI V. Clustering using  
firefly algorithm: performance study [J]. Swarm and  
evolutionary computation, 2011, 1(3): 164-171.
- [9] FALCON R, ALMEID M, NAYAK A. Fault identification with  
binary adaptive fireflies in parallel and distrib-uted systems  
[C]//2011 IEEE Congress on Evolutionary Computation  
(CEC). New Orleans, USA, 2011: 1359-1366.
- [10] YU S, ZHU S, MA Y, et al. Enhancing firefly algorit-  
hm using generalized opposition based learning[J]. Com-  
puting, 2015, 97(7): 741-754.
- [11] 周凌云, 丁立新, 何进荣. 精英正交学习萤火虫算法  
[J]. 计算机科学, 2015, 42(10): 211-216.  
ZHOU Lingyun, DING Lixin, HE Jinrong. Elite-orth-ogonal  
learning firefly algorithm[J]. Computer science, 2015, 42  
(10): 211-216.
- [12] 李洋. 蛙跳萤火虫算法及其在含风电场的电力系统调度  
中的应用[D]. 上海: 华东理工大学, 2013.  
LI Yang. Leapfrog firefly algorithm and application in di-  
spatch of power system containing wind farm[D]. Shanghai:  
East China University of Science and Technol ogy, 2013.
- [13] 符强, 童楠, 赵一鸣. 一种基于多种群学习机制的萤火  
虫优化算法[J]. 计算机应用研究, 2013, 30(12):  
3600-3603.  
FU Qiang, TONG Nan, ZHAO Yiming. Firefly algorithm  
based on multi-group learning mechanism[J]. Application  
research of computers, 2013, 30(12): 3600-3603.
- [14] FISTER I, JR F, YANG X S, et al. A comprehensive review of  
firefly algorithms [J]. Swarm and evolutionary computation,  
2013, 13(1): 34-46.
- [15] VERMA O P, AGGARWAL D, PATODI T. Opposition  
and dimensional based modified firefly algorithm [J]. Ex-  
pert-systems with applications an international journal,  
2016, 44(C): 168-176.
- [16] 汪慎文, 丁立新, 谢大同. 应用精英反向学习策略的混合  
差分演化算法[J]. 武汉大学学报: 理学版, 2013, 59  
(3): 111-116.  
WANG Shenwen, DING Lixin, XIE Datong. A hybrid  
differential evolution with elite opposition-based learning[J].  
Journal of Wuhan university: natural science edition, 2013,  
59(3): 111-116.
- [17] 程美英, 倪志伟, 朱旭辉. 萤火虫优化算法理论研究综  
述[J]. 计算机科学, 2015, 42(4): 19-24.  
CHENG Meiyong, NI Zhiwei, ZHU Xuhui. Over-view on  
glow-worm swarm optimization or firefly algorithm [J].  
Computer science, 2015, 42(4): 19-24.
- [18] 汪慎文, 丁立新, 谢大同. 应用反向学习策略的群搜索  
优化算法[J]. 计算机科学, 2012, 39(9): 183-187.  
WANG Shenwen, DING Lixin, XIE Datong. Group search  
optimizer applying opposition-based learning[J]. Computer  
science, 2012, 39(9): 183-187.
- [19] 周新宇, 吴志健, 王晖, 等. 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.  
ZHOU Xinyu, WU Zhijian, WANG Hui, et al. A particle  
swarm optimization algorithm based on elite reverse learning  
[J]. Sinica Acta electronica, 2013, 41(8): 1647-1652.
- [20] YANG X S. Firefly Algorithm Lévy flights and global optimization  
[C]//Research and Development in Intelligent Systems  
XXVI. London, Springer, 2010: 209-218.
- [21] TILAHUN S L, HONG C O. Modified firefly algorithm [J].  
Journal of applied mathematics, 2012, 2012(12): 2428-2439.

## 作者简介:



魏伟一,男,1976年生,博士,副教授,CCF会员,主要研究方向为智能信息处理、数字图像处理。



文雅宏,男,1993年生,硕士研究生,主要研究方向为数字图像处理、智能计算。