

基于柯西变异的果蝇优化算法

韩旭明¹, 邱 兵¹, 刘翹铭¹, 周丽媛¹, 王丽敏²

(1 长春工业大学 计算机科学与工程学院, 吉林 长春 130012;

2 吉林财经大学 管理科学与信息工程学院, 吉林 长春 130012)

摘 要: 针对果蝇优化算法在寻优过程中易陷入局部极值点的不足, 提出一种基于柯西变异的果蝇优化算法. 当算法陷入局部最优时采用柯西变异策略, 更新果蝇群体位置, 使算法继续迭代寻找全局极值. 通过六个基准函数对算法性能进行测试, 实验结果表明基于柯西变异的果蝇优化算法在收敛速度和收敛精度都有较大提高.

关键词: 果蝇优化算法; 柯西变异; 基准函数

中图分类号: TP18

文献标识码: A

文章编号: 1000-7180(2017)11-0026-05

Fruit Fly Optimization Algorithm Based on Cauchy Mutation

HAN Xu-ming¹, QIU Bing¹, LIU Qiao-ming¹, ZHOU Li-yuan¹, WANG Li-min²

(1 School of Computer Science & Engineering, Changchun University of Technology,
Changchun 130012, China; 2 School of Management Science and Information Engineering,
Jilin University of Finance and Economics, Changchun 130012, China)

Abstract: The fruit fly optimization algorithm is easy to fall into local extreme points in the optimization process. To solve this problem, a fruit fly optimization algorithm based on cauchy mutation (Fruit Fly Optimization Algorithm-based on Cauchy Mutation FOACM) is proposed. When the algorithm fall into local extreme points using cauchy mutation strategy to update flies group location and make the algorithm continue iterative and search global extremum. Through the test of six benchmark functions, The experimental results show that the fruit fly optimization algorithm based on cauchy mutation has a great improvement in the convergence speed and convergence precision.

Key words: fruit fly optimization algorithm; cauchy mutation; benchmark function

1 引言

随着计算机技术的发展, 群智能算法^[1] (Swarm Intelligence Algorithm, SIA) 作为人工智能的重要分支越来越受到国内外学者的关注, 果蝇优化算法 (Fruit Fly Optimization Algorithm, FOA) 设计简单, 设置参数少, 寻优精度高, 一经提出就广泛应用到 Z-SCORE 模型、广义回归神经网络参数的优化、灰色神经网络参数的优化和对支持向量机算法的优化, 并取得很好的效果^[2]. 但对于多峰函数, FOA 易

受到局部极值点的束缚进入早熟, 导致收敛速度减慢, 收敛精度降低. 针对上述问题, 提出基于柯西变异的果蝇优化算法 (Fruit Fly Optimization Algorithm Based on Cauchy Mutation, FOACM). 通过引入柯西变异算子, 使算法较快地跳出局部极值, 提高收敛速度和收敛精度. 采用经典函数作为测试函数对算法收敛性进行测试, 并与粒子群算法 (PSO)、布谷鸟算法 (CS)、果蝇优化算法 (FOA) 进行对比来测试改进算法的有效性^[3-4].

2 果蝇优化算法

根据果蝇觅食特性可以把 FOA 归结为以下步骤^[2]:

步骤 1: 设置参数: 设置种群规模为 sizepop, 最大的迭代次数 maxgen;

步骤 2: 初始群体的位置 x_0, y_0 ;

步骤 3: 给每个果蝇个体随机的步长与方向:

$$\begin{aligned} x_i &= x_0 + \text{step} \\ y_i &= y_0 + \text{step} \end{aligned} \quad (1)$$

步骤 4: 求出果蝇个体与原点的欧式距离 D_i , 味道浓度判定值 S_i 为其倒数:

$$\begin{aligned} D_i &= \sqrt{x_i^2 + y_i^2} \\ S_i &= 1/D_i \end{aligned} \quad (2)$$

步骤 5: 求出每个果蝇个体的味道浓度 Smell_i :

$$\text{Smell}_i = \text{function}(S_i) \quad (3)$$

步骤 6: 保留最优个体的味道浓度与位置坐标:

$$\begin{aligned} \text{Smellbest} &= \text{bestSmell} \\ x_0 &= x_{\text{bestindex}} \\ y_0 &= y_{\text{bestindex}} \end{aligned} \quad (4)$$

步骤 7: 进入迭代寻优, 重复执行步骤 3 ~ 5, 判断最优的味道浓度是否优于前一代的味道浓度, 是则执行步骤 6, 否则继续迭代直到最大迭代次数 maxgen;

3 基于柯西变异的果蝇优化算法

3.1 柯西分布

群智能算法应用高斯变异策略提高算法收敛精度已取得较好的效果^[5-6]. 高斯分布和柯西分布^[7]都是概率论与数理统计中的一种常见分布.

一维的高斯分布密度函数为:

$$f(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, -\infty < x < \infty \quad (5)$$

$\sigma = 1, \mu = 0$ 时为标准高斯分布. 一维标准柯西分布概率密度函数为:

$$f(x) = \frac{1}{\pi} \cdot \frac{t}{t^2 + x^2}, -\infty < x < \infty \quad (6)$$

图 1 所示为标准高斯分布和标准柯西分布的概率密度函数曲线. 从图中可以看出柯西分布比高斯分布平缓, 下降到 0 的速度较慢且在中心处的峰值小于高斯分布. 由随机变量生成函数定理^[7]得到柯西分布的随机变量生成函数为:

$$\eta = \tan[(\xi - 0.5)\pi] \quad (7)$$

式中, ξ 是 $[0, 1]$ 上服从均匀分布的随机变量.

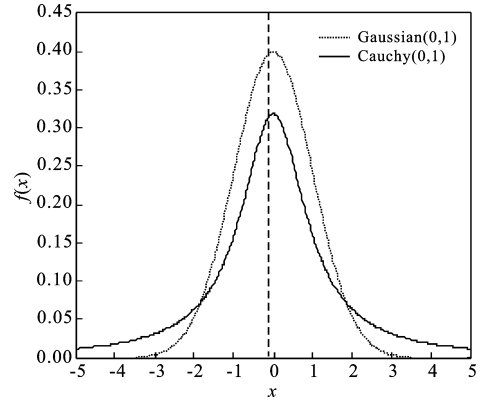


图 1 标准高斯、柯西分布密度函数曲线

3.2 基于柯西变异的果蝇优化算法

柯西分布从峰值向两侧下降平缓, 使得柯西变异后的果蝇个体受局部极值点约束力下降, 在陷入局部极值时能迅速逃离. 另一方面柯西分布的峰值较小, 变异后的果蝇会花费较少时间来搜索邻域区间, 重点搜寻全局最优点. 所以由柯西分布产生的扰动对果蝇优化算法寻找全局最优值具有很好的调节能力^[8]. 利用上述柯西分布特点对果蝇优化算法进行改进, 在原有果蝇个体的状态上引入服从柯西分布的随机向量, 如公式(8)所示:

$$M'_0 = M_0 + M_0 * \text{Cauchy}(0, 1) \quad (8)$$

式中, M_0 是原个体状态, M' 是经过扰动后的新状态.

基于柯西变异的果蝇优化算法的步骤归结为:

步骤 1: 执行基本果蝇优化算法的步骤 1 ~ 步骤 6;

步骤 2: 进入循环迭代后, 当相邻两次迭代的果蝇群体的最优味道浓度都作为历史最优值且连续两次最优值没有改变或变化极小时 ($|\text{变化量}| < \sigma$), 认为算法停滞, 则进行柯西变异;

步骤 3: 变异操作: 先将最优果蝇个体的个数复制到原先种群规模, 然后进行柯西变异, 更新群体的位置, 进行二次寻优.

$$\begin{aligned} x'_0 &= x_0 + x_0 * \text{Cauchy}(0, 1) \\ y'_0 &= y_0 + y_0 * \text{Cauchy}(0, 1) \end{aligned} \quad (9)$$

式中, x_0, y_0 是原个体位置, x'_0, y'_0 是经过扰动后的新位置.

4 实验结果与分析

为了验证改进后算法的性能, 采用 6 个经典函数作为测试函数对算法收敛性进行测试并与 PSO、CS、FOA 进行对比. 函数详细信息见表 1. 实验参数设置如下: 种群规模 sizepop=20, 除 f_4 为二维函数,

其余函数维数设置为 30,迭代次数设置为 100 次.

20 GHz,6 GB 内存,操作系统 Windows7 旗舰版 64 位操作系统,编程软件为 MATLAB R2013a.

判断是否进行柯西变异的变化量阈值 σ 取 10^{-10} . 实验环境:处理器为 Inter(R)Core(TM)i3-2330M2.

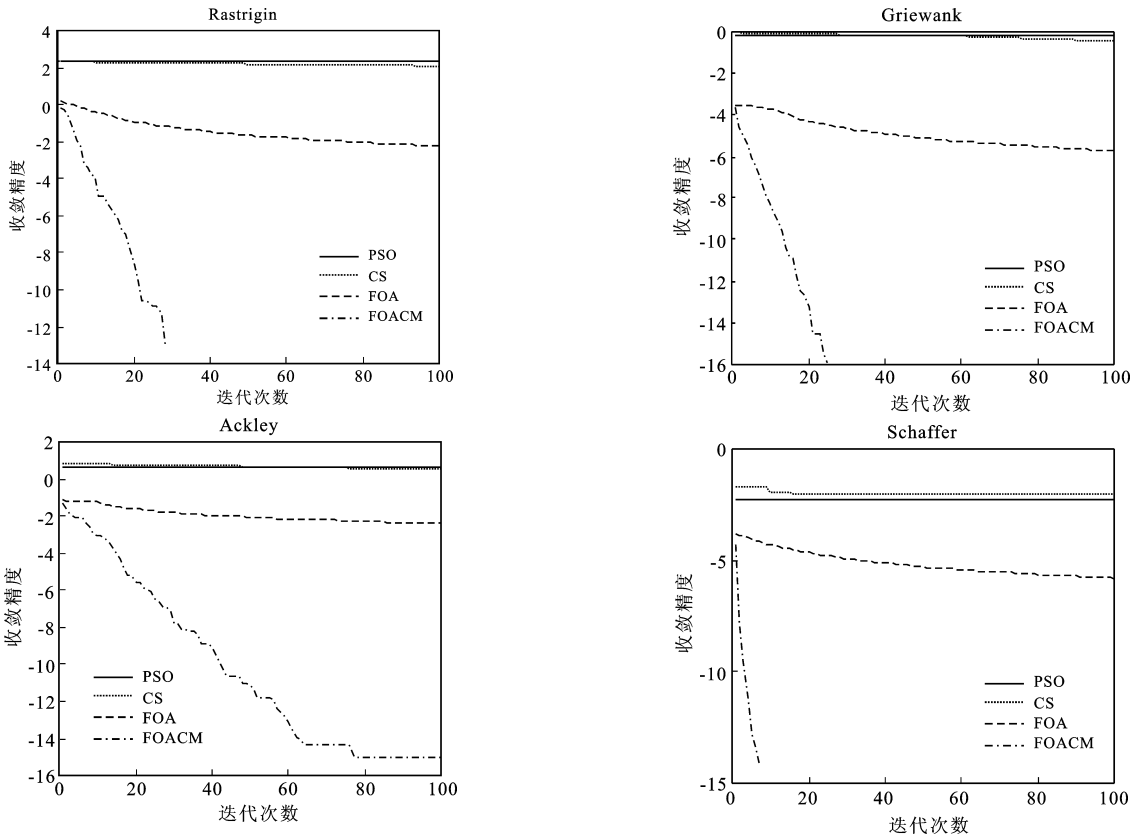
表 1 标准函数详细信息

函数名称	公式	最优值	可行域
Rastrigin	$f_1 = \sum_{i=1}^n (x_i^2 - 10\cos(2\pi x_i) + 10)$	0	$-5.12 \leq x_i \leq 5.12$
Ackley	$f_2(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^n \cos 2\pi x_i\right) + 20 + e$	0	$-32 \leq x_i \leq 32$
Griewank	$f_3 = 1/4\,000 \sum_{i=1}^n (x_i)^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0	$-600 \leq x_i \leq 600$
Schaffer	$f_4 = 0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{[1.0 + 0.001 \times (x_1^2 + x_2^2)]^2}$	0	$-100 \leq x \leq 100$
Sphere	$f_5 = \sum_{i=1}^n x_i^2$	0	$-100 \leq x_i \leq 100$
Schwefel's Problem	$f_6 = \sum_{i=1}^n \left(\sum_{j=1}^n x_j\right)^2$	0	$-100 \leq x \leq 100$

4.1 不同算法的对比

图 2 为不同算法对函数优化的实验结果图,为了方便观察,输出值取以 10 为底的对数. 通过实验

结果可以看出,PSO、CS 和 FOA 的收敛曲线平缓且近似为直线,三者中 FOA 收敛性能较好,对二维函数 f_4 进行优化时,收敛级数达到 -5 ,而 PSO、CS 收



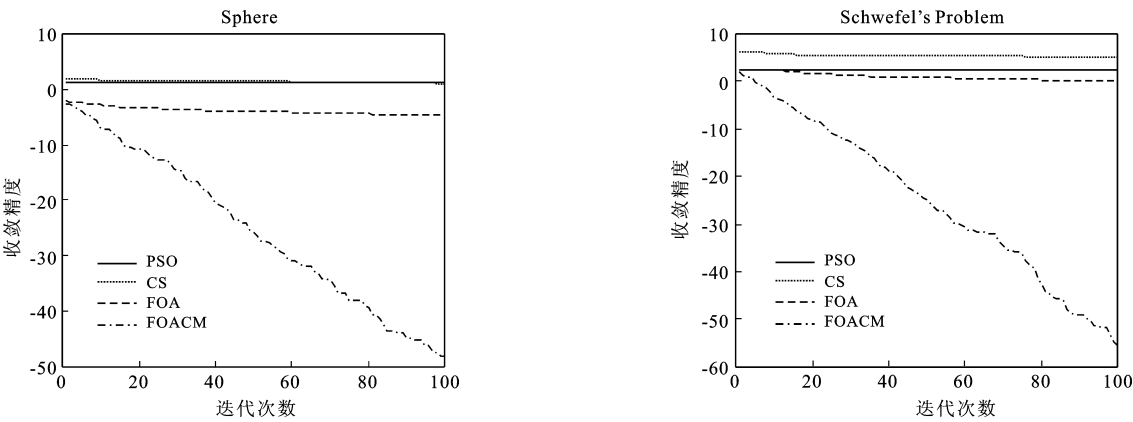


图 2 不同算法实验对比图

敛性能较差,对 f_1 、 f_2 、 f_5 和 f_6 函数进行优化时,收敛结果大于 0. FOACM 的收敛速度要优于其他算法,如从 f_1 、 f_3 、 f_4 函数的收敛曲线可以看出,FOACM 在 20 次迭代时,收敛精度可以到达 -10 以下.从 f_2 函数的收敛曲线可以看出,算法迭代到 60 次后,最优值不在变化,收敛曲线呈水平直线,但是经过变异,算法很快跳出了局部极值,继续寻找全局

最优值.所以引入柯西变异的果蝇优化算法可以有效提高收敛速度和收敛精度.

4.2 收敛性测试

为了测试每个算法的收敛率与收敛精度,将每个函数测试每个算法 20 次所得实验结果如表 2 所示.

表 2 运行 20 次实验结果的比较

函数	理论最优值	算法	最小值	最大值	平均值	收敛率
Rastrigin	0	CS	88.898 7	159.488 5	119.967 7	0/20
		PSO	224.862 7	297.244 3	258.862 2	0/20
		FOA	0.005 6	2.095 1	0.762 7	0/20
		FOACM	0	0	0	20/20
Ackley	0	CS	3.010 6	5.457 8	4.422 9	0/20
		PSO	4.054 8	5.065 8	4.629 4	0/20
		FOA	0.004 0	0.094 8	0.043 5	0/20
		FOACM	0	8.881 8e-16	5.921 2e-16	20/20
Griewank	0	CS	0.4707	0.740 7	0.547 7	0/20
		PSO	0.453 8	0.809 9	0.692 6	0/20
		FOA	1.840 5e-006	7.170 6e-004	2.023 2e-004	11/20
		FOACM	8.214 0e-007	1.426 5e-005	5.912 0e-006	14/20
Sphere	0	CS	3.376 6	26.743 7	13.066 2	0/20
		PSO	15.016 3	24.423 9	17.202 7	0/20
		FOA	2.724 2e-005	0.009 2	0.003 0	11/20
		FOACM	0	0	0	20/20
Schaffer	0	CS	0	0.009 7	0.001 9	16/20
		PSO	0.001 7	0.160 7	0.058 2	0/20
		FOA	1.612 1e-06	2.037 4e-06	1.774 8e-06	20/20
		FOACM	0	0	0	20/20
Schwefel's Problem	0	CS	6.062 9e+04	2.952 3e+05	1.659 6e+04	0/20
		PSO	291.826 4	7.132 7e+03	1.881 2e+03	0/20
		FOA	1.207 6	222.822 4	21.955 7	0/20
		FOACM	0	1.795 6e-48	6.238 8e-50	20/20

其中收敛率计算公式如下所示

$$\text{收敛率} = \frac{\text{收敛次数}}{\text{实验次数}} \quad (10)$$

当测得值与理论最优值的差小于 ϵ 时认为算法收敛, 这里取 $\epsilon = 10^{-5}$. f_1 是高维多峰函数, PSO、CS、FOA 收敛率为 0, 而 FOACM 可以收敛到理论最优值且收敛率达到 100%, 收敛精度明显优于其他算法. 对于具有许多局部极值点的 f_2 函数, FOACM 的收敛率达到 100% 且收敛精度为 -16 , 且最小值可以达到理论最优值. 对于二维多峰函数 f_4 , PSO 收敛率为 0, FOACM 和 FOA 收敛率都是 100%, 但 FOACM 全部收敛到理论最优值 0. 对 f_5 和 f_6 单峰函数进行寻优时, FOACM 全部收敛到全局最优值 0. 综上所述 FOACM 优化高维多峰函数时受局部极值点的约束减弱, 柯西变异产生的扰动使果蝇个体在进入局部极值点时可以迅速跳出, 继续搜寻全局最优值, 从而提高了算法的收敛速度和收敛精度.

5 结束语

本文提出一种基于柯西变异的果蝇优化算法, 当算法陷入局部最优时, 通过柯西变异因子对果蝇群体的位置进行扰动, 由于变异因子较强的扰动能力, 算法可以迅速跳出局部极值, 并在新位置继续搜索. 并且通过实验对比, 验证了基于柯西变异的果蝇优化算法在收敛速度和收敛精度两方面都有较大提高. 果蝇优化算法迭代过程中的步长随机且方向不定, 如何使算法进行有效的迭代将是以后的工作重点.

参考文献:

- [1] 胡中功, 李静. 群智能算法的研究进展[J]. 自动化技术与应用, 2008, 27(2):13-15.
- [2] 潘文超. 果蝇最佳化演算法[M]. 台北: 沧海书局, 2011:10-12.
- [3] 韩俊英, 刘成忠. 自适应变异的果蝇优化算法[J]. 计算机应用研究, 2013, 30(9):2641-2644.
- [4] 张勇, 夏树发, 唐冬生. 果蝇优化算法对多峰函数求解性能的仿真研究[J]. 暨南大学学报: 自然科学与医学版, 2014, 35(1):82-87.
- [5] 王晓慧, 刘雪英, 白梅花. 引入高斯变异和最速下降算子的人口迁移算法[J]. 计算机工程与应用, 2009, 45(20):57-60.
- [6] 郑云水, 岳小雪, 林俊亭. 带有高斯变异的混合蛙跳蝙蝠算法[J]. 计算机应用研究, 2015, 32(12):3629-3633.
- [7] 李贤平. 概率论基础[M]. 3 版. 北京: 高等教育出版社, 2010.
- [8] 曲良东, 何登旭. 自适应柯西变异人工鱼群算法及其应用[J]. 微电子学与计算机, 2010, 27(10):74-78.

作者简介:

韩旭明 男, (1971-), 博士, 教授, 博士生导师. 研究方向为数据挖掘、智能计算和群优化.

邱兵(通讯作者) 男, (1990-), 硕士研究生. 研究方向为智能计算. E-mail: 56460681@qq.com.

刘翹铭 男, (1991-), 硕士研究生. 研究方向为智能计算.

周丽媛 女, (1991-), 硕士研究生. 研究方向为智能计算和群优化.

王丽敏 女, (1975-), 博士, 教授, 硕士生导师. 研究方向为机器学习、智能计算和群优化.