

改进的变步长果蝇优化算法

朱富占, 邹海, 丁国绅

(安徽大学 计算机科学与技术学院, 安徽 合肥 230601)

摘 要: 为了克服基本果蝇优化算法(FOA)在求解全局优化问题时所存在的寻优精度不高, 收敛速度较慢, 易陷入局部最优等问题, 提出了改进的变步长果蝇优化算法, 在基本果蝇优化算法位置移动公式中, 该算法利用指数分布来增强算法的全局探测能力; 同时利用步长递减模式来增强算法后期的局部优化能力, 有效地权衡了算法全局与局部寻优性能. 选取 6 个基准函数将本文算法与另外两种改进的果蝇算法以及原果蝇算法进行对比, 实验结果证明, 新改进的算法能够跳出局部最优, 提高了算法的收敛速度和寻优精度.

关键词: 果蝇优化算法; 全局优化; 寻优精度; 收敛速度; 指数分布

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-7180(2018)06-0036-05

An Improved Fruit Fly Optimization with Changing Step

ZHU Fu-zhan, ZOU Hai, DING Guo-shen

(School of Computer Science and Technology, Anhui University, Hefei 230601, China)

Abstract: In order to overcome the basic Fruit Fly Optimization Algorithm(FOA) in solving global optimization problems, precision is not high, slow convergence speed and easily falling into the master problem, an improved Fruit Fly Optimization Algorithm with changing step is proposed. In this paper, Exponential Distribution can be used to enhance the exploration ability of the algorithm. At the same time, decreasing step size can be used to enhance the local optimization ability of the algorithm. This improvement achieves the equilibrium between global and local optimizations effectively. This algorithm is compared with other two improved FOA algorithms and the original FOA algorithm on 6-benchmark functions, the experimental results demonstrate that the new algorithm can jump out of local optimal, then the convergence rate and the precision of the Fruit Fly Algorithm can be improved significantly.

Key words: fruit fly optimization algorithm; global optimization; convergence precision; convergence speed; exponential distribution

1 引言

果蝇优化算法(Fruit Fly Optimization Algorithm, FOA)源于对果蝇觅食行为的模拟^[1], 该算法作为一种新的全局智能优化算法吸引了许多国内外学者的关注, 也产生了许多改进的 FOA 算法, 如基于混沌扰动策略的果蝇优化算法^[2]、递减步长果蝇优化算法^[3]、自适应步长的果蝇优化算法^[4]、基于细菌趋化的果蝇优化算法^[5]等. 同时 FOA 也被证实

可广泛应用于科学和工程领域^[6]等.

FOA 算法不同于其他智能算法, 该算法具有控制参数少、结构较为简单、运行时间短、易实现等优点, 然而, 由于该算法的提出时间相对较晚, 其理论分析还不够成熟, 尤其是对搜索半径的过度依赖, 直接影响到算法的优劣, 导致算法收敛速度慢, 收敛精度不高, 易陷入局部最优等缺点, 特别是在求解高维多极值优化问题时更难达到预期效果. 因此, 对 FOA 的改进研究仍具有重要的意义.

本文在深入分析 FOA 算法工作原理的基础上,针对其不足之处,提出了改进的变步长果蝇优化算法(CSFOA)。

2 基本果蝇算法 FOA

FOA 是一种寻求全局优化的新的启发式方法,它是根据果蝇的觅食行为推演出来的。该算法的执行步骤如下:

(1) 设定种群规模 Sizepop 和最大迭代次数 Maxgen,果蝇群体位置为 X_{axis} , Y_{axis} 。

(2) 根据如下公式随机产生果蝇的新位置:

$$\begin{cases} X_i = X_{axis} + R_{Value} \\ Y_i = Y_{axis} + R_{Value} \end{cases} \quad (1)$$

式中, R_{Value} 为果蝇的随机搜索距离。

(3) 估计果蝇个体与原点的距离 D_i 和果蝇个体的味道浓度判定值 S_i :

$$D_i = \sqrt{X_i^2 + Y_i^2} \quad (2)$$

$$S_i = 1/D_i \quad (3)$$

(4) 将味道浓度判定值 S_i 代入适应度函数 Function 中, 求出该果蝇个体位置的味道浓度 smell(i):

$$\text{smell}(i) = \text{Function}(S_i) \quad (4)$$

(5) 找出此果蝇群体中的最优个体(即味道浓度最小值) 以及其位置:

$$[\text{bestsmell}, \text{bestindex}] = \min(\text{smell})$$

(6) 保留最优味道浓度值 bestsmell 与其对应的 X 、 Y 坐标, 此时果蝇群体往该位置飞去:

$$\text{smellbest} = \text{bestsmell}$$

$$X_{axis} = X(\text{bestindex})$$

$$Y_{axis} = Y(\text{bestindex})$$

(7) 进入迭代寻优, 重复执行步骤(2)~(5), 并判断味道浓度是否优于前一迭代味道浓度, 若成立则执行步骤(6)。

3 DS-FOA

DS-FOA 在 FOA 的基础上变固定步长为递减步长, 并设步长值为:

$$L = L_0 - \frac{L_0(G-1)}{G_{\max}} \quad (5)$$

式中, L_0 为初始步长值; $G_{\max}$ 为最大迭代次数; G 为当前迭代次数。很明显, 迭代次数为 1 时, $L = L_0$, 即果蝇个体步长为最大值。果蝇迭代每增加一代, 步长减小 $L_0/G_{\max}$, 直到最后一代减小至 $L_0/G_{\max}$ 。

DS-FOA 算法有明显的缺陷, 算法的优劣与初

始步长值 L_0 密切相关, 取较大值时, 可以提高算法的全局优化性能, 同时收敛效率便会降低, 取较小值时, 收敛效率提高, 全局收缩性能下降, 极有可能错过最优解。

4 LGMS-FOA

LGMS-FOA 算法引入了权重系数 w 表示搜索半径对位置坐标的影响, 变固定步长值为自适应步长值, w 随着迭代次数的增加而按指数形式下降。果蝇个体的位置计算规则如下:

初始果蝇群体位置

$$X_{axis} = n * \text{rand}(\text{VR}) \quad (6)$$

随机飞行后的位置

$$X_i = X_{axis} + w * \text{rand}(\text{VR}) \quad (7)$$

权重系数

$$w = w_0 * \alpha^{\text{gen}} \quad (8)$$

式中, $\text{rand}(\text{VR})$ 表示待求解函数取值范围内的一个随机数; n 为搜索范围系数; w_0 为初始权重; α 为权重系数; gen 为当前迭代次数。其余和基本 FOA 原理一致。

通过自适应的改变果蝇飞行的步长值, 提高了算法的局部开发能力, 在求解复杂多峰函数时可以取得更佳的效果, 但算法容易陷入局部最优的缺点并没有得到解决。

5 CSFOA

5.1 CSFOA 算法基本思想

传统 FOA 算法的搜索半径是固定的, 限制了果蝇群体的全局搜索性能。为了提高算法的全局搜索能力, 可以通过引入随机机制来提高算法探测新的解空间的能力; 局部搜索能力指的是算法在一定范围内搜索更高精度解的能力, 很显然步长值越小则算法搜索的精度越高。

根据以上分析, 本文提出了改进的变步长果蝇优化算法, 该算法利用随机机制来调节果蝇的搜索空间, 以更大的几率跳出局部最优从而实现全局搜索, 使其能在迭代初期就能逼近目标精度, 避免陷入局部极值; 同时采用递减步长的方式, 在算法迭代初期具有较大的随机步长, 使算法能在全局范围内搜索, 随着迭代的进行, 算法在后期的搜索步长减小, 使得算法能够更加精细的搜索。

5.1.1 递减步长策略

本文采用动态递减步长的策略, 整个种群大范围的搜索, 在迭代初期具有较大的搜索步长, 算法可

以在全局范围内搜索,在一定程度上增强算法的探测能力;随着迭代的进行,搜索步长逐渐减小,算法便可以在当前最优解附近进行深度搜索,以提高算法在迭代后期的寻优精度.

主要改进策略如下:

$$\begin{cases} X_i = X_{_axis} + L * R_{Value} \\ Y_i = Y_{_axis} + L * R_{Value} \end{cases} \quad (9)$$

$$L = \frac{L_{max}}{bests_{i-1}} * (p)^c \quad (10)$$

$$c = \frac{t}{t_{max}} - \cos(\frac{t}{t_{max}}) + 1 \quad (11)$$

式中, $bests_{i-1}$ 为上一代最优味道浓度值; L_{max} 为算法中最大步长; p 为指数因子; 本文取 0.01; t 为当前迭代次数; t_{max} 为最大迭代次数; c 为过渡参数.

5.1.2 逃离局部最优策略

传统的 FOA 算法在视觉搜索过程中一旦发现最优果蝇,整个果蝇群都会往该位置飞去,而该位置不一定是最优解所处的位置,很可能是局部最优,因此整个果蝇种群的多样性就会下降,不能进行全局搜索,陷入局部最优.为了逃离局部最优,本文引入了随机机制,使算法能够以更大的几率跳出局部最优,增加解得多样性,从而实现全局搜索. 主要改进策略如下:

$$L = R_i * \frac{L_{max}}{bests_{i-1}} * (p)^c \quad (12)$$

$$R_i(x) = 1 - e^{-\gamma x} \quad (13)$$

式中, R_i 为指数分布. γ 取 0.5. 其余和基本 FOA 原理一致.

5.2 算法的时间复杂度分析

时间复杂度是指执行算法所需的计算工作量,它主要取决于问题重复执行的次数. 在基本 FOA 算法中,时间复杂度主要受到种群规模 N 和最大迭代次数 M 的影响. 设果蝇群体中个体迭代所需要的时间为 T , 则 FOA 算法的时间复杂度为 $O(N * M * T)$, 在迭代寻优过程中,每个个体迭代需要的计算量为 $O(1)$, 所以 FOA 算法的时间复杂度主要取决于种群规模和最大迭代次数,即 $O(N * M)$.

CSFOA 算法是由 FOA 算法改进而来的,在 FOA 算法的基础上增加了动态的步长值 L , L 的计算复杂度为 $O(M)$, 所以本文算法的时间复杂度为 $O(N * M + M)$.

由上述分析可知,CSFOA 算法的时间复杂度相对较高,这是追求较高的收敛速度和寻优精度所付出的代价,因此,寻求较低的时间复杂度将是日后

研究的重点工作.

6 仿真实验及结果分析

本文的实验平台是 Matlab R2014a,在内存为 4GB,CPU 速度为 3.20GHz 的 PC 机上运行. 选取 6 个基准测试函数来验证新算法的性能,如表 1 所示,并与 FOA 算法、DS-FOA 算法、LGMS-FOA 算法进行对比. 本文从两个方面对算法性能进行测试:其一,固定种群规模和迭代次数,比较算法的收敛速度和寻优精度;其二,在固定收敛精度下,比较算法的平均迭代次数和成功率.

表 1 测试函数

测试函数	搜索区间	理论极值	目标精度	函数类型
$f_1(x) = \sum_{i=1}^n x_i^2$	$[-100,100]$	0	10^{-10}	single
$f_2(x) = \sum_{i=1}^n (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-5.12,5.12]$	0	10^{-4}	multiple
$f_3(x) = \sum_{i=1}^n x_i ^{i+1}$	$[-1,1]$	0	10^{-10}	single
$f_4(x) = -\exp(-0.5 \sum_{i=1}^n x_i^2)$	$[-1,1]$	-1	10^{-4}	single
$f_5(x) = \sum_{i=1}^n ix^2 + \text{rand}()$	$[-1.28,1.28]$	0	10^{-3}	single
$f_6(x) = \sum_{i=1}^n x_i \sin(x_i) + 0.1x_i $	$[-10,10]$	0	10^{-4}	multiple

6.1 固定种群规模和迭代次数下的性能测试

具体参数设置如下:种群规模 $\text{Sizepop} = 20$, 最大迭代数 $\text{Maxgen} = 1\ 000$, DS-FOA 算法和本文算法的最大步长 L_{max} 分别取 150 和 5,初始花果蝇群体位置为定义域中的任意位置. 表 2 是在 30 维条件下各算法对 6 个测试函数独立运行 20 次后的平均优化结果. 图 2 是 6 个测试函数在 30 维条件下的适应度进化曲线,其中迭代次数为 100,其他因素保持不变.

从表 2 中可以看出,在对 6 个基准测试函数的实验中,CSFOA 算法与 FOA 算法、DS-FOA 算法和 LGMS-FOA 算法相比,平均值和方差明显低于其他算法,特别地,FOA 算法、DS-FOA 算法和 LGMS-FOA 算法对函数 $f_1(x)$ 、 $f_2(x)$ 、 $f_6(x)$ 的测试都没有达到目标精度,而 CSFOA 算法对 6 个基准函数的测试均达到了目标精度,并且除了函数 $f_5(x)$,对其余函数的测试都达到了理论极值,说明 CSFOA 算法较其他算法具有更好的稳定性和较高的收敛精度. 由图 2 可以看出,在对各测试函数的

表 2 四种算法的优化结果对比

测试函数		FOA	DS-FOA	LGMS-FOA	CSFOA
$f_1(x)$	Mean	1.277 5E-02	3.487 4E-04	3.684 9E-04	0
	Var	2.119 3E-04	9.990 7E-08	1.238 7E-06	0
$f_2(x)$	Mean	2.021 9E+02	6.495 5E-02	25.51 84	0
	Var	1.403 0E+04	3.908 1E-03	3.816 9E+03	0
$f_3(x)$	Mean	78.927 7	1.679 5E-10	8.036 4E+04	0
	Var	1.110 2E+05	4.758 3E-24	1.069 9E+11	0
$f_4(x)$	Mean	-0.470 8	-1	-0.734 5	-1
	Var	0.230 6	0	0.179 8	0
$f_5(x)$	Mean	44.605 0	2.180 0E-03	38.549 3	3.979 6E-05
	Var	8.104 4E+03	2.075 9E-07	9.234 2E+03	1.265 2E-09
$f_6(x)$	Mean	1.126 6	9.685 0E-03	0.201 9	0
	Var	1.454 4	1.454 4	9.096 5E-02	0

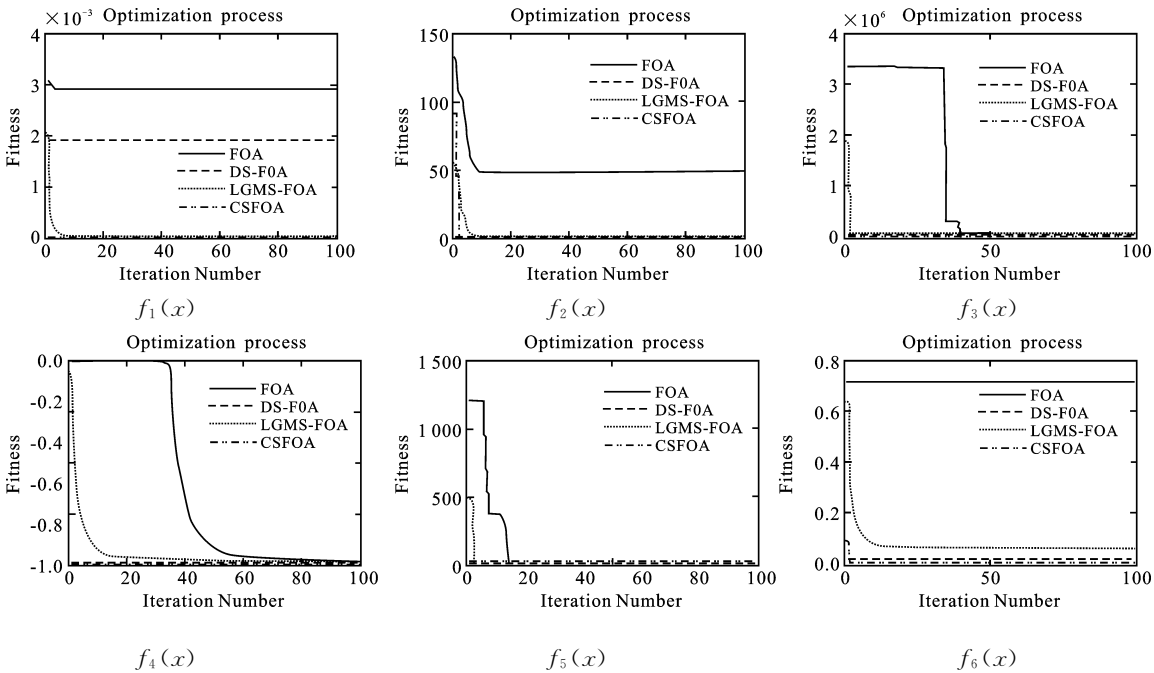


图 2 部分函数适应度进化曲线

迭代寻优过程中, FOA 算法、DS-FOA 算法和 LGMS-FOA 算法收敛速度慢,易陷入局部最优,导致寻优精度不高,甚至寻优失败,而 CSFOA 算法则表现出了较好的寻优性能和较快的收敛速度,进一步证明了本文算法的优越性.

6.2 固定收敛精度下的性能测试

6 个测试函数在表 1 中指定的目标精度下经过 20 次独立实验的平均迭代数和成功率如表 3 所示. 其中,“—”表示没有达到目标精度. 从表 3 中可以看出,在低维条件下,对于 $f_1(x)$ 、 $f_3(x)$ 、 $f_5(x)$, FOA 没有达到目标精度, $f_2(x)$ 、 $f_4(x)$ 、 $f_6(x)$ 虽然达到目标精度,但成功率不高,所需迭代次数较

大. LGMS-FOA 算法只是对 $f_6(x)$ 达到了目标精度. CSFOA 算法和 DS-FOA 算法都达到目标精度,且成功率均为 100%,尤其 CSFOA 算法所需迭代次数明显较少,说明 CSFOA 在收敛速度上更快. 在高维情况下,FOA 算法、DS-FOA 算法和 LGMS-FOA 算法的寻优效果明显变差,易陷入局部最优,即使达到目标精度,其代价也比较大,特别地,对于函数 $f_1(x)$ 、 $f_2(x)$ 、 $f_6(x)$, FOA 算法、DS-FOA 算法和 LGMS-FOA 算法都没有达到目标精度,而 CSFOA 算法对 6 个测试函数均能以 100% 的成功率达到目标精度,且所需要的迭代次数较少.

表 3 目标精度下平均迭代次数与成功率对比

测试函数	维数	FOA		DS-FOA		LGMS-FOA		CSFOA	
		平均迭代次数	成功率	平均迭代次数	成功率	平均迭代次数	成功率	平均迭代次数	成功率
$f_1(x)$	2	—	0	572	1	—	0	1	1
$f_2(x)$	2	856	0.85	5	1	—	0	1	1
$f_3(x)$	2	—	0	233	1	—	0	1	1
$f_4(x)$	2	515	1	3	1	—	0	3	1
$f_5(x)$	2	—	0	12	1	—	0	7	1
$f_6(x)$	2	279	1	1	1	15	0.95	1	1
$f_1(x)$	30	—	0	—	0	—	0	2	1
$f_2(x)$	30	—	0	—	0	—	0	3	1
$f_3(x)$	30	—	0	232	1	—	0	16	1
$f_4(x)$	30	631	0.40	4	1	—	0	1	1
$f_5(x)$	30	—	0	14	1	—	0	8	1
$f_6(x)$	30	—	0	—	0	—	0	3	1

从整体上讲,在固定收敛精度情况下,CSFOA 算法无论是对低维还是高维函数的测试中,其成功率均优于其他三种算法,且所需迭代次数少,进一步证明了本文所提算法具有较好的寻优性能和较高的成功率。

7 结束语

本文将元启发式算法中的随机机制引入基本果蝇优化算法,增加算法探测新的解空间的能力,提高算法的寻优速度,使得算法能够在迭代初期就逼近目标精度;递减步长策略使得算法在迭代初期具有较强的全局勘探能力,后期具有较强的局部搜索能力,提高了算法的收敛精度。二者的结合有效的平衡了算法的全局勘探能力和局部搜索能力。对基准测试函数的仿真实验结果证明,与基本果蝇优化算法和其他两种改进的果蝇优化算法相比,本文算法具有更高的收敛效率、更好的优化性能和鲁棒性。如何将改进的果蝇优化算法同其他智能算法相结合来解决生活中的实际问题,更加优化调节参数,将是下一步研究的重点工作。

参考文献:

[1] 潘文超. 应用果蝇优化算法优化广义回归神经网络进行企业经营绩效评估[J]. 太原理工大学学报(社会科学版),2011,29(4):1-5.

[2] 李宗岳,陈志军,李名远. 基于混沌扰动策略的果蝇优化算法[J]. 微电子学与计算机,2016,33(7):64-68.

[3] 宁剑平,王冰,李鸿儒,等. 递减步长果蝇优化算法及应用[J]. 深圳大学学报理工版,2014,31(4):367-373.

[4] 郭晓东,王丽芳,张学良. 基于自适应步长的果蝇优化算法[J]. 中北大学学报(自然科学版),2016,37(6):570-575.

[5] 韩俊英,刘成忠. 基于细菌趋化的果蝇优化算法[J]. 计算机应用,2013,33(4):964-966.

[6] Shan D, Chao G, Dong H. LGMS-FOA: an improved fruit fly optimization algorithm for solving optimization problems[J]. Math Prob Eng, 2013 (7) :1256-1271

作者简介:

朱富占 男,(1992-),硕士研究生. 研究方向为智能计算.
E-mail:2860603443@qq.com.

邹海 男,(1969-),博士,副教授. 研究方向为智能计算和数据挖掘.

丁国绅 男,(1992-),硕士研究生. 研究方向为智能计算.