

## 自适应混沌果蝇优化算法

韩俊英, 刘成忠\*

(甘肃农业大学 信息科学技术学院, 兰州 730070)

(\* 通信作者电子邮箱 liucz@gsau.edu.cn)

**摘要:**针对基本果蝇优化算法(FOA)寻优精度不高和易陷入局部最优的缺点,融入混沌算法对果蝇优化算法的进化机制进行优化,提出自适应混沌果蝇优化算法(ACFOA)。在算法处于收敛状态时,应用混沌算法进行全局寻优,从而跳出局部极值而继续优化。对几种经典测试函数的仿真结果表明,ACFOA 具有更好的全局搜索能力,在收敛速度、收敛可靠性及收敛精度上均比基本 FOA 有较大的提高。

**关键词:**自适应;混沌;果蝇优化算法;适应度

**中图分类号:** TP18 **文献标志码:** A

### Adaptive chaos fruit fly optimization algorithm

HAN Junying, LIU Chengzhong\*

(College of Information Science and Technology, Gansu Agricultural University, Lanzhou Gansu 730070, China)

**Abstract:** In order to overcome the problems of low convergence precision and easily relapsing into local extremum in basic Fruit fly Optimization Algorithm (FOA), by introducing the chaos algorithm into the evolutionary process of basic FOA, an improved FOA called Adaptive Chaos FOA (ACFOA) was proposed. In the situation of local convergence, chaos algorithm was applied to search the global optimum in the outside space of convergent area and to jump out of local extremum and continue to optimize. The experimental results show that the new algorithm has the advantages of better global searching ability, faster convergence and more precise convergence.

**Key words:** adaptive; chaos; Fruit fly Optimization Algorithm (FOA); fitness

## 0 引言

果蝇优化算法(Fruit fly Optimization Algorithm, FOA)<sup>[1-2]</sup>是一类新的全局优化进化算法,该算法源于对果蝇觅食行为的模拟,可广泛应用于科学和工程领域,也可混合其他的数据挖掘技术一起使用,现已将其成功应用于求解数学函数极值、微调 Z-SCORE 模型系数、广义回归神经网络参数优化与支持向量机参数优化等。由于 FOA 提出较晚,目前国内外的研究尚处于起步阶段,研究成果还很少,理论也不成熟,因此 FOA 的相关研究迫切需要展开。

FOA 与其他群智能算法比较,不但算法简单容易理解(如粒子群算法的优化方程是二阶微分方程<sup>[3]</sup>,而 FOA 的优化方程是一阶微分方程),程序代码易于实现,运行时间较少;而且 FOA 只需调整四个参数,其他的群智能算法至少要调整七八个参数,参数之间的相互影响和复杂关系很难研究清楚,且参数的取值不当,会严重影响算法的性能,导致分析算法复杂度变得异常困难。但同时 FOA 与其他全局优化算法(如遗传算法、粒子群算法等)一样,极易陷入局部最优,导致后期收敛速度变慢,收敛精度降低,尤其是对于高维多极值复杂优化问题。

本文针对 FOA 寻优精度不高和易陷入局部最优的缺点,提出自适应混沌果蝇优化算法(Adaptive Chaos Fruit fly Optimization Algorithm, ACFOA)。该算法通过群体适应度方差的取值来判定 FOA 是否处于局部收敛状态,若是,则利用混沌算法进行全局寻优,从而跳出局部极值,提高种群多样

性,提高 FOA 的收敛精度以及收敛速度。6 个基准测试函数的对比实验结果表明所提出的自适应混沌果蝇优化算法优于基本的果蝇优化算法。

## 1 基于 Logistic 映射的混沌技术

混沌是自然界非线性系统中一种较为普遍的现象,它充分体现了系统的复杂性,看似混乱的变化过程,实际上含有内在规律性。混沌变量是一个在 $[0,1]$ 区间波动的变量,它具有随机性、遍历性和规律性,并对初值具有敏感性<sup>[4-5]</sup>。混沌优化的基本思想是:将优化变量通过混沌映射规则映射到混沌变量空间的取值区间内,利用混沌变量的遍历性和规律性寻优搜索,最后将获得的优化解线性转化到优化空间<sup>[6]</sup>。通常所采用的 Logistic 映射是一个源于人口统计的动力学系统,其系统方程<sup>[7]</sup>为:

$$x(t+1) = \mu x(t)(1-x(t)) \quad (1)$$

其中: $t$ 为迭代次数, $x(t) \in [0,1]$ ; $\mu$ 是控制参数,当 $\mu = 4$ 时,系统处于混沌状态。Logistic 映射是一个非常简单又具有重要意义的非线性迭代方程,它具有确定的形式,并且系统不包含任何随机因素,但系统却能产生看似完全随机的、极为敏感的、依赖于参量 $\mu$ 的动态变化现象。

式(2)为混沌变量 $Cx_i$ 的一种变换算式<sup>[8]</sup>:

$$Cx(t+1)_i = 4Cx(t)_i(1-Cx(t)_i); i = 1, 2, \dots, N \quad (2)$$

$Cx(t)_i$ 为映射的第 $i$ 个混沌变量 $Cx_i$ 在第 $t$ 步混沌变换后的值,当 $Cx_i \in [0,1]$ 且 $Cx_i \neq \{0.25, 0.5, 0.75\}$ 时,将产生混沌现象,式(2)的优化变量 $x_i \in [a_i, b_i]$ ,可由式(3)、式(4)

收稿日期:2012-12-03;修回日期:2013-01-15。 基金项目:甘肃省科技支撑计划资助项目(1011NKCA058);甘肃省教育厅科研基金资助项目(1202-04);甘肃省自然科学基金资助项目(1208RJZA133)。

作者简介:韩俊英(1975-),女,甘肃兰州人,副教授,硕士,主要研究方向:优化计算、农业信息化; 刘成忠(1969-),男,甘肃天祝人,副教授,博士研究生,主要研究方向:智能决策支持系统。

与混沌变量  $Cx_i \in [0,1]$  进行往返映射<sup>[9]</sup>。

$$Cx_i = (x_i - a_i) / (b_i - a_i) \quad (3)$$

$$x_i' = a_i + Cx_i(b_i - a_i) \quad (4)$$

式(4)中  $x_i'$  为经混沌映射后的第  $i$  个混沌变量  $Cx_i$  转化为常规优化变量而获得的值,由于在 Logistic 映射空间  $[0,1]$  内有 3 个断点,因此在进行混沌映射优化时应跳过 0.25, 0.50, 0.75 这 3 个断点。

## 2 果蝇优化算法

果蝇优化算法是一种基于果蝇觅食行为推演出的寻求全局优化的新方法。果蝇本身在感官知觉上优于其他物种,尤其是在嗅觉与视觉上。果蝇的嗅觉器官能很好地搜集飘浮在空气中的各种气味,然后飞近食物位置后亦可使用敏锐的视觉发现食物与同伴聚集的位置,并且往该方向飞去。

依据果蝇搜索食物特性,将果蝇优化算法归纳为以下几个必要的步骤<sup>[2]</sup>:

1) 给定群体规模  $Sizepop$ , 最大迭代数  $Maxgen$ , 随机初始化果蝇群体位置  $X\_axis, Y\_axis$ 。

2) 赋予果蝇个体利用嗅觉搜寻食物之随机方向与距离,  $RandomValue$  为搜索距离:

$$\begin{cases} X_i = X\_axis + RandomValue \\ Y_i = Y\_axis + RandomValue \end{cases} \quad (5)$$

3) 由于无法得知食物位置,因此先估计与原点之距离  $Dist_i$ ,再计算新位置的味道浓度判定值  $S_i$ ,此值为距离之倒数:

$$Dist_i = \sqrt{X_i^2 + Y_i^2} \quad (6)$$

$$S_i = 1/Dist_i \quad (7)$$

4) 将味道浓度判定值  $S_i$  代入味道浓度判定函数(或称为适应度函数),用来求出果蝇个体位置的味道浓度  $Smell_i$ :

$$Smell_i = Function(S_i) \quad (8)$$

5) 找出该果蝇群体中味道浓度最佳的果蝇(最优个体);  $[bestSmell \quad bestindex] = \min(Smell_i)$  (9)

6) 记录并保留最佳味道浓度值  $bestSmell$  与其  $X, Y$  坐标,这时候果蝇群体利用视觉向该位置飞去:

$$\begin{cases} Smellbest = bestSmell \\ X\_axis = X(bestindex) \\ Y\_axis = Y(bestindex) \end{cases} \quad (10)$$

7) 进入迭代寻优,重复执行步骤2)~5),并判断最佳味道浓度是否优于前一迭代最佳味道浓度,并且当前迭代次数小于最大迭代数  $Maxgen$ ,若是则执行步骤6)。

## 3 自适应混沌果蝇优化算法 ACFOA

FOA 在整个迭代寻优进化过程中只向当前最优果蝇个体学习,一旦发现本次迭代的最优个体,所有个体都聚集到该个体位置,降低了种群多样性,若该个体并不是全局最优,极易使算法陷入局部最优,降低收敛速度和收敛精度,带来早熟收敛的问题。

本文提出的自适应混沌果蝇优化算法以基本果蝇优化算法运算流程为主体流程,通过群体适应度(味道浓度)方差取值来判定 FOA 是否处于局部收敛状态,若是,则利用式(3)将果蝇个体位置通过 Logistic 映射映射到  $[0,1]$  混沌区间内,然后利用式(4)将混沌变量转化为寻优搜索空间内果蝇个体位置,由于混沌映射使得果蝇个体新位置呈现遍历性、随机性和

多样性,可在收敛区域以外空间进行全局搜索,当映射产生的新个体位置的适应度小于当前全局最优位置的适应度时,完成全局最优位置的更新,以此提高种群多样性,解决基本 FOA 中只向最优个体靠近,而导致种群多样性丢失引起的早熟收敛问题,从而提高整个算法的求解精度及求解效率。

ACFOA 具体流程如下:

1) 初始化参数:群体规模  $Sizepop$ , 最大迭代数  $Maxgen$ , 果蝇群体位置  $X\_axis, Y\_axis$ , 适应度(味道浓度) 方差阈值  $\delta$ , 混沌遍历次数  $M$ 。

2) 执行 FOA 算法流程2)~6)。

3) 根据式(11)计算该果蝇群体的平均味道浓度  $Smell_{avg}$ (平均适应度),根据式(12)计算该果蝇群体味道浓度方差  $\sigma^2$ (适应度方差)。

$$Smell_{avg} = \sum_{i=1}^{SizePop} Smell_i / SizePop \quad (11)$$

$$\sigma^2 = \sum_{i=1}^{SizePop} (Smell_i - Smell_{avg})^2 \quad (12)$$

4) 若  $\sigma^2 < \delta$  且  $M > 0$ ,则利用式(3)将果蝇个体位置  $X_i, Y_i$  通过 Logistic 映射映射为混沌变量  $CX_i, CY_i$ ,然后利用式(2)对混沌变量  $CX_i, CY_i$  进行变换,最后利用式(4)将混沌变量  $CX_i, CY_i$  转化为搜索空间内果蝇个体新位置  $X_i', Y_i', M = M - 1$ ;否则,转8)。

5) 先估计新位置  $X_i', Y_i'$  与原点之距离  $Dist_i'$ ,再计算味道浓度判定值  $S_i'$ :

$$Dist_i' = \sqrt{X_i'^2 + Y_i'^2} \quad (13)$$

$$S_i' = 1/Dist_i' \quad (14)$$

6) 将味道浓度判定值  $S_i'$  代入味道浓度判定函数,求出果蝇个体位置的味道浓度  $Smell_i'$ :

$$Smell_i' = Function(S_i') \quad (15)$$

7) 若  $Smell_i' < Smellbest$ ,则  $Smellbest = Smell_i', X\_axis = X_i', Y\_axis = Y_i'$ ,然后转4);否则,直接转4);

8) 进入迭代寻优,重复执行步骤2)~7),直至当前迭代次数等于最大迭代数  $Maxgen$  或已达到精度目标要求。

## 4 实验及结果分析

### 4.1 实验设计

为了验证本文提出的 ACFOA 的性能,本文设计了两类测试实验:1) FOA 优化实验;2) ACFOA 优化实验。实验选用 6 个常用于优化算法比较的基准函数(求解最小值),函数形式、搜索区间、理论极值和收敛精度见表 1。

待优化函数的维数越高、自变量范围越大、目标精度越高,优化过程的难度就越大<sup>[10]</sup>。为了便于比较和突出 ACFOA 的性能,本文均选用比较苛刻的实验参数,具体参数设置为:群体规模  $Sizepop = 30$ ,最大迭代数  $Maxgen = 2000$ ,混沌遍历次数  $M = 5$ ,适应度(味道浓度) 方差阈值  $\delta = 0.00001$ 。随机初始化果蝇群体位置为表 1 中各函数的搜索区间。

性能评估采用的方法:1) 固定进化迭代次数,评估算法收敛速度和收敛精度,并与参考文献算法进行比较;2) 固定收敛精度目标值,评估算法达到该精度目标所需的平均迭代次数和成功率,并与参考文献算法进行比较;3) 评估算法在高维、多峰函数上的性能,并与参考文献算法进行比较;4) 评估算法的时间复杂度,并与参考文献算法进行比较。

表1 用于测试改进算法的优化函数

函数名	维数	公式	取值范围	最优值	目标值
Sphere	30	$f_1(x) = \sum_{i=1}^n x_i^2$	$[-100, 100]$	0	$10^{-5}$
Griewank	30	$f_2(x) = \frac{1}{4000} \sum_{i=1}^n (x_i)^2 - \prod_{i=1}^n \cos(x_i/\sqrt{i}) + 1$	$[-600, 600]$	0	$10^{-6}$
Rosenbrock	30	$f_3(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$[-100, 100]$	10	30
Rastrigin	30	$f_4(x) = \sum_{i=1}^n (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-100, 100]$	0	$10^{-4}$
Ackley	30	$f_5(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^n \cos 2\pi x_i\right) + 20 + e$	$[-100, 100]$	0	$10^{-1}$
Schaffer	2	$f_6(x) = \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2} - 0.5$	$[-100, 100]$	-1	$10^{-5} - 1$

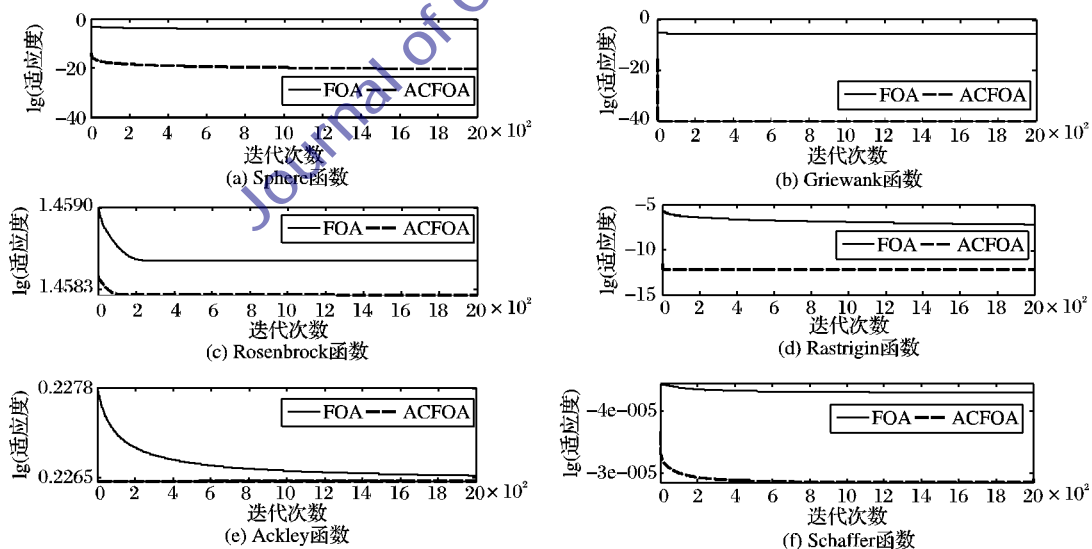
## 4.2 实验结果与分析

### 4.2.1 固定进化迭代次数的收敛速度和精度

6个测试函数固定进化次数2000次,分别采用FOA和ACFOA经过20次独立运行后的实验结果如表2所示,图1是6个测试函数适应度对数值进化曲线(注:为了方便进化曲线的显示和观察,本文对函数的适应度取以10为底的对数;同时,为了避免真数为0和纵坐标范围过大,对函数 $f_5$ 的适应度加上 $10^{-40}$ 作为截止值)。从中可看出,对单峰值函数Sphere( $f_1$ )和Rosenbrock( $f_3$ ),ACFOA最优值、最差值、优化均值、标准差均优于FOA,进化曲线也表明ACFOA的收敛速度明显优于FOA;对多峰值函数Griewank( $f_2$ )和Rastrigin( $f_4$ ),ACFOA更易实现全局收敛,进化曲线也表明ACFOA收敛速度显著优于FOA;对强烈振荡的多峰值函数Schaffer( $f_6$ ),ACFOA能很快达到理论极小点-1,且避免了FOA陷入局部最优的缺点;对多峰值函数Ackley( $f_5$ ),FOA

和ACFOA的性能持平。因此,总体来说ACFOA收敛精度、收敛速度和收敛稳定性均优于FOA。

与参考文献算法的性能比较如表3所示(注:文献[11]中 $f_1, f_2, f_3, f_4, f_5$ 对应的搜索范围分别为 $\pm 100, \pm 600, \pm 30, \pm 5.12, \pm 32$ ,  $Sizepop = 30, Maxgen = 3000$ ;文献[12]中 $f_1, f_2, f_3, f_4$ 对应的搜索范围分别为 $\pm 100, \pm 600, \pm 100, \pm 10$ ,  $Sizepop = 20, Maxgen = 2000$ ;文献[13]中 $f_1, f_3, f_4, f_5$ 对应的搜索范围分别为 $\pm 100, \pm 30, \pm 5.12, \pm 32$ ,  $Sizepop = 40, Maxgen = 2000$ ),从中可以看出,ACFOA在实验条件(如种群数或进化迭代次数)和函数约束条件(如搜索区间)比较苛刻的情况下,基本上在所有测试函数上均比参考文献的算法具有更高的收敛精度,尤其对于多峰值函数Griewank( $f_2$ )和Rastrigin( $f_4$ )来说,ACFOA的优化均值显著优于参考文献,相差最少达16个数量级。

图1  $f_1 \sim f_6$  在两个实验中的适应度进化曲线

### 4.2.2 固定收敛精度下的平均迭代次数与成功率

6个测试函数在表1中指定的收敛精度下经过20次独立运行后的平均迭代次数和成功率如表4所示(注:文献[11]中 $f_1, f_2, f_3, f_4, f_5$ 对应的搜索范围分别为 $\pm 100, \pm 600, \pm 30, \pm 5.12, \pm 32$ , 对应的精度目标分别为0.01, 0.1, 100, 100, 0.1,  $Maxgen = 3000$ ;文献[12]中 $f_1, f_2, f_3, f_4$ 对应的搜索范围分别为 $\pm 100, \pm 600, \pm 100, \pm 10$ , 对应的精度目标分别为0.0001, 0.0001, 100, 100,  $Maxgen = 2000$ )。其中,成功率

(Success Rate) = 达到精度的运行次数/总实验次数;平均迭代次数(Mean Iterations)是达到精度的迭代次数平均值。表中括号中的数据是本文实验达到参考文献[11]的精度目标所需的平均迭代次数和成功率。

实验结果表明,对于 $f_1, f_2, f_3, f_4$ 来说,FOA和ACFOA均以100%的成功率达到了文献[11]的目标精度,并且平均迭代次数最多只需文献[11]中SPSO的0.5%和APSO的0.2%;对于 $f_5$ 来说,FOA和ACFOA性能与文献[11]中SPSO

相同,但达不到 APSO 的性能。

同时实验结果也表明,ACFOA 算法在实验条件(如精度目标、进化迭代次数)和函数约束条件(如搜索区间)比较苛刻的情况下,对于  $f_1$ 、 $f_2$ 、 $f_3$ 、 $f_4$  和  $f_6$  来说,ACFOA 以 100% 的成功率,绝对少的平均迭代次数达到了本文的目标精度;对于  $f_5$  来说,FOA 和 ACFOA 性能持平。

总体来说,ACFOA 比 FOA、文献[11]和文献[12]的算法具有更快的收敛速度和更高的收敛可靠性。

#### 4.2.3 算法在高维、多峰函数上的性能

全局优化算法普遍存在易陷入局部最优,导致后期收敛速度变慢、收敛精度降低的问题,尤其对于高维、多峰复杂优化问题。FOA、ACFOA、文献[14]的 GA 和 BFO 以及文献[15]的 PSO 和 BFA-PSO 在 50 维多峰函数 Griewank ( $f_2$ )、Rastrigin ( $f_4$ ) 和 Ackley ( $f_5$ ) 上的性能比较如表 5 所示(注:为了便于比较,本文实验  $f_2$ 、 $f_4$ 、 $f_5$  对应的搜索范围分别为

$\pm 600$ ,  $\pm 100$ ,  $\pm 100$ ,  $Sizepop = 15$ ,  $Maxgen = 800$ ;文献[14]中  $f_2$ 、 $f_4$ 、 $f_5$  对应的搜索范围分别为  $\pm 600$ ,  $\pm 5.12$ ,  $\pm 32$ ,  $Sizepop = 15$ ,  $Maxgen = 800$ ;文献[15]中  $f_2$ 、 $f_4$ 、 $f_5$  对应的搜索范围分别为  $\pm 600$ ,  $\pm 5.12$ ,  $\pm 32$ ,  $Sizepop = 30$ ,  $Maxgen = 3\ 000$ )。从中可知,对于多峰函数 Griewank ( $f_2$ ) 和 Rastrigin ( $f_4$ ),ACFOA 的优化均值和标准差显著优于 FOA、文献[14]的 GA、BFO 以及文献[15]的 PSO 和 BFA-PSO,至少相差达 34 个数量级;尤其对于多峰函数 Rastrigin ( $f_4$ ),ACFOA 达到了理论极小值 0 (注:该函数 50 维的优化效果比 30 维要好)。对于多峰函数 Ackley ( $f_5$ ),ACFOA 的优化均值与 FOA 持平;与文献[14]的 GA 在同一个数量级,也仅仅只是 GA 的 39.5%,标准差相差达 15 个数量级;比文献[15]的 PSO 更优;比文献[14]的 BFO 和文献[15]的 BFA-PSO 略差。通过上述分析,总体来看 ACFOA 在高维、多峰函数上比文献[14-15]具有更高的收敛精度和收敛可靠性。

表2 算法性能测试

函数	最优值		最差值		优化均值		标准差	
	FOA	ACFOA	FOA	ACFOA	FOA	ACFOA	FOA	ACFOA
$f_1$	4.4966e-6	3.7868e-21	0.0034	3.8281e-21	8.3065e-4	3.8126e-21	0.0012	9.3732e-24
$f_2$	9.4703e-8	0	1.5672e-5	0	2.9592e-6	0	4.8959e-6	0
$f_3$	28.7070	28.7070	29.1637	28.8763	28.7605	28.7327	0.1090	0.0490
$f_4$	3.0705e-5	1.7764e-15	3.9307e-5	3.5527e-15	3.6222e-5	3.1086e-15	2.6126e-6	7.8917e-16
$f_5$	1.6847	1.6844	1.6847	1.6844	1.6847	1.6844	9.8621e-6	2.3180e-14
$f_6$	-1.0000	-1.0000	-0.9987	-0.9999	-0.9999	-1.0000	3.2514e-4	4.0305e-5

表3 ACFOA 与参考文献算法优化均值比较

函数	ACFOA	SPSO <sup>[11]</sup>	APSO <sup>[11]</sup>	AFSA <sup>[12]</sup>	GAFA <sup>[12]</sup>	CMQPSO <sup>[13]</sup>
$f_1$	3.8126e-21	1.1e-24	1.2e-32	1.012182e+4	1.1e-146	5.651e-16
$f_2$	0	0.13	0.0039	1.065802e+2	1.4e-5	—
$f_3$	28.7327	34.1	33.6	1.09e+10	2.473051e+1	3.1982e+1
$f_4$	3.1086e-15	101.7	50.1	3.068264e+2	6.674231e+1	21.025
$f_5$	1.6844	4.76	1.45	—	—	12.890
$f_6$	-1.0000	—	—	—	—	—

表4 在目标精度下的平均进化代数与成功率比较

函数	目标值	FOA 平均迭代次数/成功率	ACFOA 平均迭代次数/成功率	SPSO <sup>[11]</sup> 平均迭代次数/成功率	APSO <sup>[11]</sup> 平均迭代次数/成功率	AFSA <sup>[12]</sup> 平均迭代次数/成功率	GAFA <sup>[12]</sup> 平均迭代次数/成功率
$f_1$	$10^{-5}$	1118/0.8(1/1)	1/1(1/1)	528/1	484/1	—/0	575/1
$f_2$	$10^{-6}$	917.3/0.75(1/1)	1/1(1/1)	413/0.74	454/1	—/0	970/1
$f_3$	30	1/1(1/1)	1/1(1/1)	911/1	994/1	—/0	1076/1
$f_4$	$10^{-4}$	1117/1(1/1)	2.1/1(1/1)	196/0.58	420/1	—/0	944/1
$f_5$	$10^{-1}$	—/0	—/0	—/0	412/1	—	—
$f_6$	$10^{-5} - 1$	81.8/0.35	138.65/1	—/0	—/—	—	—

表5 高维、多峰函数上的算法性能比较

函数	指标	FOA	ACFOA	GA <sup>[14]</sup>	BFO <sup>[14]</sup>	PSO <sup>[15]</sup>	BFA-PSO <sup>[15]</sup>
$f_2$	优化均值	1.1203e-6	2.8374e-55	1.505	1.028	4.2353e+1	4.58e-3
	标准差	2.3361e-6	4.1843e-56	1.15	0.8e-2	1.18556e+1	3.4372e-3
$f_4$	优化均值	2.8996e-6	0	66.791	3.73	2.4382e+2	4.9337e+1
	标准差	3.5669e-5	0	11.57	3.6e-1	35.1271	8.8406
$f_5$	优化均值	1.6847	1.6844	4.262	3.97e-1	1.35e+1	1.128e-1
	标准差	1.2661e-5	3.7847e-14	8.77e-1	3.4e-2	9.68e-1	1.11e-2

#### 4.2.4 算法的时间复杂度

好的优化算法除了要求计算精度高、收敛速度快、收敛稳定性好之外,还要求算法时间复杂度低。以优化函数  $f_5$  为例,FOA、ACFOA、文献[14]的 PSO、GA 和 BFO 共 5 种优化算

法在函数为 30 维和 50 维时的耗时如表 6 所示。从中可知,GA 的耗时特别长,FOA 的耗时最短,ACFOA 的耗时比 BFO 要短,但比 PSO 略长一些。

(下转第 1333 页)



户数量为0时无法进行预测,且在评分用户数量相同时其MAE值也明显高于GC-PLSA方法;3)通过表1可知,GC-PLSA方法的平均MAE值明显低于其他两种方法,GC-PLSA方法具有良好的项目推荐品质。

#### 4 结语

针对PLSA模型中存在的新项目难以推荐及推荐准确性较低等问题,引入了用户组和项目组两个潜在变量以及项目的特征,提出一种基于广义内容的推荐方法GC-PLSA,并通过实验与G-PLSA、IBCF等方法进行了对比,分别从新项目 and 准确性两个方面说明该方法具有良好的物品推荐品质。

#### 参考文献:

- [1] KIM H-N, JI A-T, HA I, *et al.* Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation[J]. *Electronic Commerce Research and Applications*, 2010, 9(1): 73–83.
- [2] LEE J-S, OLAFSSON S. Two-way cooperative prediction for collaborative filtering recommendations[J]. *Expert Systems with Applications*, 2009, 36(3): 5353–5361.
- [3] WANG K B, TAN Y. A new collaborative filtering recommendation approach based on naive Bayesian method[C]// *Proceedings of the Second International Conference on Advances in Swarm Intelligence*. Berlin: Springer, 2011: 218–227.
- [4] 杨阳, 向阳, 熊磊. 基于矩阵分解与用户近邻模型的协同过滤推荐算法[J]. *计算机应用*, 2012, 32(2): 395–398.
- [5] 宋晓雷, 王素格, 李红霞. 基于概率潜在语义分析的词汇情感倾向判别[J]. *中文信息学报*, 2011, 25(2): 89–93.
- [6] HOFMANN T. Collaborative filtering via Gaussian probabilistic latent semantic analysis [C] // *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York: ACM, 2003: 259–266.

(上接第1316页)

综上所述,本文提出的ACFOA总体来说比FOA以及权威文献中算法具有更快的收敛速度、更高的收敛精度和收敛可靠性。

表6 基于 $f_3$ 的算法时间比较

$f_3$ 维数	FOA	ACFOA	PSO <sup>[4]</sup>	GA <sup>[14]</sup>	BFO <sup>[14]</sup>
30	0.348	2.024	1.203	21.563	2.484
50	0.423	2.516	1.360	32.938	2.719

#### 5 结语

本文将混沌算法融入基本果蝇优化算法,提出自适应混沌果蝇优化算法。通过群体适应度方差,判定果蝇优化算法处于局部收敛状态时,利用混沌算法进行全局寻优,从而跳出局部极值,提高了果蝇优化算法的收敛精度和收敛速度。6个基准测试函数的对比实验结果表明新算法具有更好的全局搜索能力,在收敛速度、收敛可靠性及收敛精度上均比基本果蝇优化算法有很大的提高。

#### 参考文献:

- [1] PAN W T. A new fruit fly optimization algorithm: Taking the financial distress model as an example[J]. *Knowledge-Based Systems*, 2012, 26: 69–74.
- [2] 潘文超. 果蝇最佳化演算法[M]. 台北: 沧海书局, 2011.
- [3] 胡旺, 李志蜀. 一种更简化而高效的粒子群优化算法[J]. *软件学报*, 2007, 18(4): 862–863.

- [7] 张玉芳, 朱俊, 熊忠阳. 改进的概率潜在语义分析下的文本聚类算法[J]. *计算机应用*, 2011, 31(3): 674–676.
- [8] MONAY F, GATICA-PEREZ D. Modeling semantic aspects for cross-media image indexing [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007, 29(10): 1802–1817.
- [9] WANG X, JIN X M. Understanding and enhancing the folding-in method in latent semantic indexing[C] // *Proceedings of the 17th International Conference on Database and Expert Systems Applications*, LNCS 4080. Berlin: Springer, 2006: 104–113.
- [10] BLEI D M, LAFFERTY J D. Dynamic topic models [C] // *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*. Washington, DC: IEEE Computer Society, 2006: 113–120.
- [11] FENG S L, MANMATHA R, LAVRENKO V. Multiple Bernoulli relevance models for image and video annotation[C] // *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Washington, DC: IEEE Computer Society, 2004: 1002–1009.
- [12] HUANG J, KUMAR S R, MITRA M. Spatial color indexing and applications[J]. *International Journal of Computer Vision*, 1999, 35(3): 245–268.
- [13] MANJUNATH B S, MA W Y. Texture features for browsing and retrieval of image data[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996, 18(8): 837–842.
- [14] LOWE D G. Distinctive image features from scale-invariant keypoints[J]. *International Journal of Computer Vision*, 2004, 60(2): 91–110.
- [15] MIRANDA C, JORGE A M. Item-based and user-based incremental collaborative filtering for Web recommendations[C]// *Proceedings of the 14th Portuguese Conference on Artificial Intelligence*. Berlin: Springer-Verlag, 2009: 673–684.

- [4] 相征, 张太猛, 孙建成. 基于混沌吸引子的快衰落信道预测算法[J]. *西安电子科技大学学报*, 2006, 33(1): 145–149.
- [5] 吕晓明, 黄考利, 连光耀. 基于混沌粒子群优化的系统级故障诊断策略优化[J]. *系统工程与电子技术*, 2010, 32(1): 217–220.
- [6] 杨俊杰, 周建中, 喻菁, 等. 混合混沌优化方法及其在非线性规划问题中的应用[J]. *计算机应用*, 2004, 24(10): 119–120.
- [7] 刘道华, 原思聪, 兰洋, 等. 混沌映射的粒子群优化方法[J]. *西安电子科技大学学报*, 2010, 37(4): 764–769.
- [8] 张劲松, 李歧强, 王朝霞. 基于混沌搜索的混和粒子群优化算法[J]. *山东大学学报*, 2007, 37(1): 48–50.
- [9] 莫恩斌, 陈德钊, 胡上序. 混沌粒子群算法及其在生化过程动态优化中的应用[J]. *化工学报*, 2006, 57(9): 2123–2127.
- [10] 王凌. 智能优化算法及其应用[M]. 北京: 清华大学出版社, 2001: 148–149.
- [11] 林川, 冯全源. 一种新的自适应粒子群优化算法[J]. *计算机工程*, 2008, 34(7): 181–183.
- [12] 王联国, 洪毅, 施秋红. 全局版人工鱼群算法[J]. *系统仿真学报*, 2009, 21(23): 7483–7486.
- [13] 逢珊, 杨欣毅, 张小峰. 混沌映射的多种群量子粒子群优化算法[J/OL]. [2011–11–14]. <http://cnki.net/kcms/detail/11.2127.TP.20111114.0947.044.html>.
- [14] 胡洁. 细菌觅食优化算法的改进及应用研究[D]. 武汉: 武汉大学, 2012.
- [15] 杨萍, 孙延明, 刘小龙, 等. 基于细菌觅食趋化算子的PSO算法[J]. *计算机应用研究*, 2011, 28(10): 3640–3642.