

自适应变异的果蝇优化算法^{*}

韩俊英, 刘成忠[†]

(甘肃农业大学 信息科学技术学院, 兰州 730070)

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,提出自适应变异的果蝇优化算法(FOAAM)。该算法在运行过程中根据群体适应度方差和当前最优解的大小判断算法陷入局部最优时,首先将最优果蝇个体复制 M 个;然后对复制的最优果蝇个体进行扰动,按一定的概率 P 执行高斯变异操作;最后对变异后的最优果蝇个体进行二次寻优,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,FOAAM 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 FOA 算法有较大的提高。

关键词: 果蝇优化; 自适应; 变异; 早熟收敛

中图分类号: TP18; TP301

文献标志码: A

文章编号: 1001-3695(2013)09-2641-04

doi:10.3969/j.issn.1001-3695.2013.09.021

Fruit fly optimization algorithm with adaptive mutation

HAN Jun-ying, LIU Cheng-zhong[†]

(College of Information Science & Technology, Gansu Agricultural University, Lanzhou 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), this paper presented an adaptive mutation fruit fly optimization algorithm(FOAAM). During the evolution, in the condition of basic FOA's trapping in local extremum judging from the population's fitness variance and the current optimal, first, it generated M current optimal replicates. Then, it disturbed replicates by a certain probability P Gauss mutation operator. Finally, it optimized mutated replicates again to jump out of local extremum and continue to optimize. Experimental results show that the new algorithm has the advantages of better global searching ability, speedier convergence and more precise convergence.

Key words: fruit fly optimization; adaptive; mutation; premature convergence

果蝇优化算法(FOA)是由台湾潘文超^[1-3]博士在2011年6月提出,它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 算法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^[4],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 的寻优精度不高和易陷入局部最优的缺点,提出自适应变异的果蝇优化算法(FOAAM)。新算法采用群体适应度方差 σ^2 作为触发条件,当 $\sigma^2 < \delta$ (群体适应度方差

阈值)并且最优适应值 $\text{Smellbest} > \text{理论最优值或目标精度}$ (这里考虑的是最小化情况)时,首先将最优果蝇个体复制 M 个,其次对复制的最优果蝇个体按一定的概率 P 执行变异操作,然后对变异后的最优果蝇个体进行二次寻优,完成全局最优个体的更新,以此提高算法跳出局部极值的能力。六个基准测试函数的对比实验结果说明,本文算法的全局收敛性和寻优精度得到了显著提高。

1 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。

依据果蝇搜索食物的特性,将果蝇优化算法归纳为以下几个必要的步骤^[1-3]:

a) 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 。

b) 赋予果蝇个体利用嗅觉搜寻食物的随机方向与距离, RandomValue 为搜索距离。

收稿日期: 2012-12-26; **修回日期:** 2013-02-11 **基金项目:** 甘肃省科技支撑计划资助项目(1011NKCA058); 甘肃省自然科学基金资助项目(1208RJZA133); 甘肃省教育厅科研基金资助项目(1202-04)

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

$$X_i = X_axis + \text{RandomValue}$$

$$Y_i = Y_axis + \text{RandomValue} \quad (1)$$

c) 由于无法得知食物位置, 因此先估计与原点的距离 Dist_i , 再计算味道浓度判定值 S_i , 此值为距离倒数:

$$\text{Dist}_i = \sqrt{X_i^2 + Y_i^2} \quad (2)$$

$$S_i = 1/\text{Dist}_i \quad (3)$$

d) 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 fitness function), 用来求果蝇个体位置的味道浓度 Smell_i :

$$\text{Smell}_i = \text{function}(S_i) \quad (4)$$

e) 找出该果蝇群体中味道浓度最低的果蝇(最优个体);

$$[\text{bestSmell} \text{ bestindex}] = \min(\text{Smell}_i) \quad (5)$$

f) 记录并保留最佳味道浓度值 bestSmell 与其 X 、 Y 坐标, 此时果蝇群体利用视觉向该位置飞去:

$$\text{Smellbest} = \text{bestSmell}$$

$$X_axis = X(\text{bestindex})$$

$$Y_axis = Y(\text{bestindex}) \quad (6)$$

g) 进入迭代寻优, 重复执行步骤 b) ~ e), 并判断最佳味道浓度是否优于前一迭代最佳味道浓度。若当前迭代次数小于最大迭代数 Maxgen , 则执行步骤 f)。

2 自适应变异的果蝇优化算法 FOAAM

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。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。要克服早熟收敛问题, 就必须提供一种机制, 让算法在发生早熟收敛时, 能够跳出局部最优, 进入解空间的其他区域继续进行搜索, 直到找到全局最优解^[5]。

本文提出的自适应变异的果蝇优化算法 FOAAM 以基本果蝇优化算法 FOA 运算流程为主体流程, 根据群体适应度方差 σ^2 (σ^2 反映了果蝇群体中所有个体适应度的整体变化情况, σ^2 越小, 反映群体的聚集程度越高, 种群多样性越低) 及当前全局最优值 Smellbest 的大小来判定 FOA 算法是否陷入局部最优。若 $\sigma^2 \leq \delta$ (适应度方差阈值) 并且 $\text{Smellbest} >$ 理论最优值或目标精度(这里考虑的是最小化情况), 则判定算法陷入了局部最优。此时, 首先将最优果蝇个体复制 M 个, 其次对复制的最优果蝇个体进行扰动, 按一定的概率 P 执行高斯变异操作, 然后对变异后的最优果蝇个体进行二次寻优, 完成全局最优解的更新, 以此克服基本 FOA 算法寻优精度不高和易陷入局部最优的缺点, 从而提高整个算法的求解精度及求解效率。

FOAAM 算法具体流程如下:

a) 初始化参数: 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 , 适应度(味道浓度)方差阈值 δ , 最优果蝇个体复制个数 M , 变异概率 P 。

b) 执行 FOA 算法流程 b) ~ e)。

c) 记录并保留最佳味道浓度值 bestSmell 与最优果蝇个体位置 X_axisb 、 Y_axisb 。

$$\text{Smellbest} = \text{bestSmell}$$

$$X_axisb = X(\text{bestindex})$$

$$Y_axisb = Y(\text{bestindex}) \quad (7)$$

d) 按照式(8)计算该果蝇群体的平均味道浓度 $\text{Smell}_{\text{avg}}$

(平均适应度), 根据式(9)计算该果蝇群体适应度(味道浓度)方差 σ^2 :

$$\text{Smell}_{\text{avg}} = \sum_{i=1}^{\text{Sizepop}} \text{Smell}_i / \text{Sizepop} \quad (8)$$

$$\sigma^2 = \sum_{i=1}^{\text{Sizepop}} (\text{Smell}_i - \text{Smell}_{\text{avg}})^2 \quad (9)$$

e) 若 $\sigma^2 \leq \delta$ 且 $\text{Smellbest} >$ 理论最优值或目标精度 分布于 $[0, 1]$ 之间的随机数 $r < P$, 则首先按照式(10)复制 M 个最优果蝇个体 X_axisb_j 、 Y_axisb_j ($j = 1, 2, \dots, M$); 其次按照式(11)对复制的最优果蝇个体进行高斯变异, 将复制的最优果蝇个体位置由 X_axisb_j 、 Y_axisb_j , 更新为新位置 $X_axisb'_j$ 、 $Y_axisb'_j$ ($j = 1, 2, \dots, M$)。

$$X_axisb_j = X_axisb \quad j = 1, 2, \dots, M$$

$$Y_axisb_j = Y_axisb \quad j = 1, 2, \dots, M \quad (10)$$

$$X_axisb'_j = X_axisb_j \cdot \times (1 + 0.5 \times \text{normrnd}(0, 1, [1, \text{dim}])) \quad j = 1, 2, \dots, M$$

$$Y_axisb'_j = Y_axisb_j \cdot \times (1 + 0.5 \times \text{normrnd}(0, 1, [1, \text{dim}])) \quad j = 1, 2, \dots, M \quad (11)$$

f) 按照式(12)先估计新位置 $X_axisb'_j$ 、 $Y_axisb'_j$ 与原点之距离 Dist'_j , 再按照式(13)计算新位置味道浓度判定值 S'_j ($j = 1, 2, \dots, M$)。

$$\text{Dist}'_j = \sqrt{X_axisb'^2_j + Y_axisb'^2_j} \quad (12)$$

$$S'_j = 1/\text{Dist}'_j \quad (13)$$

g) 将味道浓度判定值 S'_j 代入味道浓度判定函数, 求出新位置的味道浓度 Smell'_j ($j = 1, 2, \dots, M$)。

$$\text{Smell}'_j = \text{function}(S'_j) \quad (14)$$

h) 若 $\text{Smell}'_j < \text{Smellbest}$, 则 $\text{Smellbest} = \text{Smell}'_j$, $X_axis = X_axisb'_j$, $Y_axis = Y_axisb'_j$ ($j = 1, 2, \dots, M$)。

i) 进入迭代寻优, 重复执行步骤 b) ~ h), 直至当前迭代次数等于最大迭代数 Maxgen 或已达到目标精度要求或理论最优值。

3 实验及结果分析

3.1 实验设计

为了验证本文提出的 FOAAM 算法的性能, 设计了两类测试实验, 即 FOA 优化实验和 FOAAM 优化实验。实验选用六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如表 1 所示。

表 1 用于测试改进算法的优化函数

Name and Code	Formula	Range	Optimal f	Goal for f
Sphere f_1	$f_1(x) = \sum_{i=1}^n x_i^2$	$[-100, 100]$	0	10^{-5}
Griewank f_2	$f_2(x) = 1/4000 \sum_{i=1}^n (x_i)^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]$	0	10^{-6}
Rosenbrock f_3	$f_3(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i)^2 + (x_i - 1)^2)$	$[-100, 100]$	10	28.8
Rastrigin f_4	$f_4(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-100, 100]$	0	0
Ackley f_5	$f_5(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{30} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{30} \sum_{i=1}^n \cos 2\pi x_i\right) + 20 + e$	$[-100, 100]$	0	10^{-1}
Schaffer f_6	$f_6(x) = \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2} - 0.5$	$[-100, 100]$	-1	-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^[6]。为了便于比较和突出 FOAAM 算

法的性能,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 $\text{Sizepop} = 30$,最大迭代数 $\text{Maxgen} = 2000$, f_1, f_2, f_3 适应度(味道浓度)方差阈值 $\delta = 0$, f_4, f_5, f_6 适应度(味道浓度)方差阈值 $\delta = 1e-8$;随机初始化果蝇群体位置为表1中各函数的搜索区间,变异概率 $P = 0.3$,最优果蝇个体复制个数 $M = 5$ 。

性能评估采用的方法如下:a)固定进化迭代次数,评估算法收敛速度和收敛精度,并与参考文献进行比较;b)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,并与参考文献进行比较;c)评估算法在高维、多峰函数上的性能,并与参考文献进行比较。

3.2 实验结果与分析

3.2.1 固定进化迭代次数的收敛速度和精度

六个测试函数固定进化次数2000次,分别采用FOA和FOAAM经过20次独立运行后的实验结果比较如表2所示。图1是六个测试函数适应度对数值的进化曲线(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),图中实线是FOA的进化曲线,虚线是FOAAM的进化曲线。从中可看出,对所有函数,FOAAM的优化均值和标准差均优于FOA,进化曲线也表明FOAAM的收敛速度明显优于FOA。对多峰值Rastrigin f_4 和强烈振荡的多峰值Schaffer f_6 ,FOAAM能很快达到理论极小点0和-1,且避免了FOA陷入局部最优的缺点。因此,总体来说FOAAM的全局搜索能力、收敛精度和收敛速度均优于FOA。

与文献[7]的SPSO和LDWPSO的性能比较结果表明(注:文献[7]中 $f_1 \sim f_3$ 对应的搜索范围分别为 $\pm 100, \pm 600, \pm 30, \pm 5.12, \pm 32$; $\text{Maxgen} = 3000$),在参数比较苛刻的情况下,除去单峰值Sphere f_1 函数之外,FOAAM的优化均值、标准差均优于文献[7]的SPSO和LDWPSO,尤其对于多峰值Rastrigin f_4 来说,FOAAM的优化均值、标准差显著优于SPSO和LDWPSO,达到了理论极小值0。总体来说,FOAAM比SPSO和LDWPSO具有更高的收敛精度和收敛可靠性。

表2 算法性能与参考文献比较

f	维数	FOA		FOAAM		SPSO		LDWPSO	
		优化均值	标准差	优化均值	标准差	优化均值	标准差	优化均值	标准差
f_1	30	$9.7151e-4$	0.0017	$5.5406e-6$	$3.4165e-7$	$1.1e-24$	$7.6e-22$	$4.3e-7$	$1.6e-6$
f_2	30	$6.6521e-7$	$1.3355e-6$	$1.8896e-7$	$6.7275e-8$	0.13	0.27	0.0242	0.0243
f_3	30	28.7343	0.0434	28.7070	$2.2232e-14$	34.1	37.9	60.8	48.1
f_4	30	$3.5597e-5$	$2.3011e-6$	0	0	101.7	23	71.6	14.6
f_5	30	1.6847	$1.1565e-5$	1.6844	$1.8786e-15$	4.76	2.39	1.45	0.98
f_6	2	-0.9998	$3.4434e-4$	-1	0	-	-	-	-

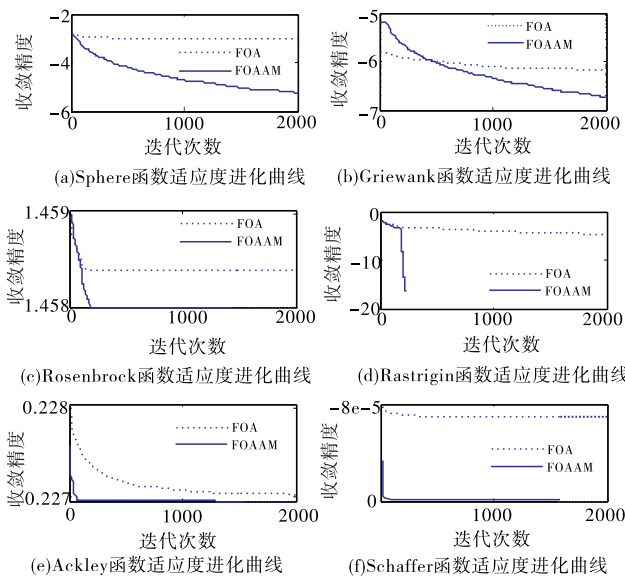


图1 $f_1 \sim f_6$ 在两个实验中的适应度进化曲线

3.2.2 固定收敛精度下的平均迭代次数与成功率

六个测试函数,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3所示(注:文献[7]中 $f_1 \sim f_3$ 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; $\text{Maxgen} = 3000$)。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[7]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 $f_1 \sim f_4$,在参数比较苛刻的情况下,FOA和FOAAM均以100%的成功率、仅仅只需1次的平均迭代次数达到了文献[7]的目标精度;对于 f_5 ,FOA和FOAAM性

能与文献[7]中SPSO相同,但达不到LDWPSO和APSO的性能。

同时实验结果也表明,对于 $f_1 \sim f_4$ 和 f_6 ,FOAAM以100%的成功率、较少的平均迭代次数达到了本文的目标精度。相比之下,FOA在所有函数上的成功率最高仅为75%,尤其在 f_4 和 f_6 上的成功率为0。对于 f_5 ,FOA和FOAAM性能持平。

总体来说,FOAAM比FOA以及文献[7]的SPSO、LDWPSO和APSO具有更快的收敛速度和更高的收敛可靠性及实用性。

3.2.3 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^[8-10]。FOA、FOAAM、文献[11]的GA和BFO以及文献[12]的PSO和BFA-PSO在50维多峰函数 f_2, f_4 和 f_5 上的性能比较如表4所示(注:为了便于比较,本文实验中参数作了一些调整, f_2, f_4, f_5 对应的搜索范围分别为 $\pm 600, \pm 100, \pm 100$, $\text{Sizepop} = 15$, $\text{Maxgen} = 800$;文献[11]中 f_2, f_4, f_5 对应的搜索范围分别为 $\pm 600, \pm 5.12, \pm 32$, $\text{Sizepop} = 15$, $\text{Maxgen} = 800$;文献[12]中 f_2, f_4, f_5 对应的搜索范围分别为 $\pm 600, \pm 5.12, \pm 32$, $\text{Sizepop} = 30$, $\text{Maxgen} = 3000$)。从中可知,对于多峰函数Griewank f_2 和Rastrigin f_4 ,FOAAM的优化均值和标准差显著优于FOA、文献[11]的GA和BFO以及文献[12]的PSO和BFA-PSO;尤其对于多峰函数Rastrigin f_4 ,FOAAM达到了理论极小值0。对于多峰函数Ackley f_5 ,FOAAM的优化均值与FOA持平;与文献[11]的GA在同一个数量级,也仅仅是GA的39.5%,标准差相差达14个数量级;比文献[12]的PSO更优;比文献[11]的BFO和文献[12]的BFA-PSO略差。通过上述分析,总体来看FOAAM在高维、多峰函数上比FOA、文献[11,12]中的算法具有更高的收敛精度和收敛可靠性。

表 3 在目标精度下的平均迭代次数与成功率比较

函数	维数	目标精度	测试项目	FOA	FOAAM	SPSO	LDWPSO	APSO
f_1	30	10^{-5}	平均迭代次数	1402(1)	1416(1)	528	2183	484
			成功率	0.75(1)	1(1)	1	1	1
f_2	30	10^{-6}	平均迭代次数	134(1)	381(1)	413	2133	454
			成功率	0.7(1)	1(1)	0.74	1	1
f_3	30	28.8	平均迭代次数	25(1)	26(1)	911	2477	994
			成功率	0.85(1)	1(1)	1	1	1
f_4	30	0	平均迭代次数	$\infty(1)$	82(1)	196	1710	420
			成功率	0(1)	1(1)	0.58	1	1
f_5	30	10^{-1}	平均迭代次数	$\infty(\infty)$	$\infty(\infty)$	∞	2197	412
			成功率	0(0)	0(0)	0	0.72	1
f_6	2	-1	平均迭代次数	∞	71	-	-	-
			成功率	0	1	-	-	-

表 4 高维、多峰函数上的算法性能比较

Function	Test Item	FOA	FOAAM	GA	BFO	PSO	BFA-PSO
f_2	优化均值	2.4827e-6	7.7156e-7	1.505	1.028	4.2353e+1	4.58e-3
	(标准差)	(3.6032e-6)	(5.0379e-7)	(1.15)	(0.8e-2)	(1.18556e+1)	(3.4372e-3)
f_4	优化均值	2.2352e-4	0	66.791	3.73	2.4382e+2	4.9337e+1
	(标准差)	(4.3320e-5)	(0)	(11.57)	(3.6e-1)	(35.1271)	(8.8406)
f_5	优化均值	1.6852	1.6844	4.262	3.97e-1	1.35e+1	1.128e-1
	(标准差)	(7.8881e-5)	(1.7200e-15)	(8.77e-1)	(3.4e-2)	(9.68e-1)	0.0111

由上可知,本文提出的 FOAAM 总体来说比 FOA、文献[7, 11, 12]中算法具有更好的全局搜索能力,更快的收敛速度和更高的收敛精度。

4 结束语

本文针对果蝇优化算法寻优精度不高和易陷入局部最优的缺点,提出了一种自适应变异的果蝇优化算法。六个经典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、收敛精度及收敛可靠性上均比基本果蝇优化算法有较大的提高。

参考文献:

- [1] PAN Wen-tao. 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: 10-12.
- [3] 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.
- [4] 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): 861-868.
- [5] 吕振肃, 侯志荣. 自适应变异的粒子群优化算法[J]. 电子学报, 2004, 32(3): 416-420.
- [6] 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.
- [7] 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.
- [8] LI M S, JI T Y, TANG W J, et al. Bacterial foraging algorithm with varying population[J]. Biosystems, 2010, 100(3): 185-197.
- [9] 靳其兵, 赵振兴, 苏晓静, 等. 基于粒子健康度的快速收敛粒子群优化算法[J]. 化工学报, 2011, 62(8): 2328-2333.
- [10] 阳春华, 谷丽珊, 桂卫华. 自适应变异的粒子群优化算法[J]. 计算机工程, 2008, 34(16): 188-190.
- [11] 胡洁. 细菌觅食优化算法的改进及应用研究[D]. 武汉: 武汉理工大学, 2012.
- [12] 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.
- [13] PAN Rong, SCHOLZ M. Mind the gaps; weighting the unknown in large-scale one-class collaborative filtering[C]//Proc of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2009: 667-676.
- [14] ADOMAVICIUS G, TUZILIN A. Toward the next generation of recommender systems; a survey of the state-of-the-art and possible extensions[J]. IEEE Trans on Knowledge and Data Engineering, 2005, 17(6): 734-749.
- [15] ZHANG Yu, CAO Bin, YEUNG D Y. Multi-domain collaborative filtering[C]//Proc of the 26th Conference on Uncertainty in Artificial Intelligence. 2010: 725-732.
- [16] MA Hao, ZHOU Deng-yong, LIU Chao, et al. Recommender systems with social regularization[C]//Proc of the 4th International Conference on WSDM. New York: ACM Press, 2011: 287-296.
- [17] MA Hao, YANG Hai-xuan, LYU M R, et al. SoRec: social recommendation using probabilistic matrix factorization[C]//Proc of the 17th ACM Conference on Information and Knowledge Management. New York: ACM Press, 2008: 931-940.
- [18] MA Hao, KING I, LYU M R. Learning to recommend with social trust ensemble[C]//Proc of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM Press, 2009: 203-210.
- [19] JAMALI M, ESTER M. Matrix factorization technique with trust propagation for recommendation in social networks[C]//Proc of the 4th ACM Conference on Recommender Systems. New York: ACM Press, 2010: 135-142.
- [20] MASSA P, AVESANI P. Trust metrics in recommender systems[M]//Computing with Social Trust. London: Springer, 2009: 259-285.
- [21] SALAKHUTDINOV R, MNH A. Probabilistic matrix factorization[C]//Proc of NIPS. 2008: 1257-1264.

(上接第 2618 页)