文章编号:1001-506X(2011)12-2755-07

加速收敛的人工蜂群算法

毕晓君,王艳娇
(哈尔滨工程大学信息与通信工程学院,黑龙江 哈尔滨 150001)

摘 要:针对人工蜂群算法(artificial bee colony algorithm, ABC)存在的收敛速度慢、易陷入局部最优的缺点,提出了一种改进算法。首先,设计新的选择策略和交叉策略,使群体快速向最优解靠近;然后,鉴于控制侦查蜂行为的参数难于确定,且对算法性能影响较大,提出了基于反向学习的变异策略代替侦查蜂行为,同样达到避免陷入局部最优的效果。通过对 10 个标准测试函数的仿真表明,改进算法几乎都可以得到各测试函数的全局最优解,而且收敛速度快、鲁棒性好。改进性能明显优于现有人工蜂群算法。

关键词:人工蜂群算法;自由搜索算法;反向学习;函数优化
中图分类号:TP 18 **文献标志码**:A **DOI**:10.3969/j.issn.1001-506X.2011.12.34

Artificial bee colony algorithm with fast convergence

BI Xiao-jun, WANG Yan-jiao
(*College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China*)

Abstract: Aiming at the shortcoming of artificial bee colony algorithms, such as the low convergence rate and easy to be trapped into the local optimums, an improved algorithm is proposed. First, a new crossover strategy is designed to make the group close to the optimal solution as soon as possible. Then, considering that the parameter of controlling the behavior of the scouts to avoid falling into local optimal setting is difficult and of a greater impact on the performance of the algorithm, a mutation strategy based on opposition-based learning is proposed to replace the scouts' behavior. The simulation results on 10 standard test functions show that this new improved algorithm can obtain the global optimal solutions for almost all the functions, with fast convergence and good robustness. The performance of this algorithm is significantly better than the existing artificial bee colony algorithms.

Keywords: artificial bee colony algorithm; free search algorithm; opposition-based learning; function optimization

0 引 言

人工蜂群算法[1]是受蜜蜂采蜜机制启发的群集智能优化算法,于 2005 年由 Karaboga 提出,文献[2]中指出与差分进化算法[3]、粒子群算法[4]等相比可获得更佳的测试结果,是目前最为优秀的函数优化方法之一,具有设置参数少、计算简单等优点,特别适合工程应用。然而,这种新的智能优化算法仍然存在易陷入局部最优、收敛速度慢的缺陷,为此,一些学者对其进行了改进。联赛选择和 boltzmann 选择方式被用来代替 ABC 算法中跟随蜂利用轮盘赌选择方式选择较优蜜源[5-6],减低算法陷入局部最优的可能;文献[7]引入遗传交叉因子增加种群多样性;文献[8]与粒子群算法结合加快收敛,但收敛速度和精度并不高。为此,本文通过深入研究,提出一种快速收敛于全局最优解的改进人工蜂群算法。首先,利用自由搜索算法[9]中的信息素-灵敏度模型代替轮盘赌方法进行蜜源的选择,避免陷入局部最优;其次,提出一种新的交叉方式使种群快速收敛于全局最优解;最后,考虑到控制侦查蜂行为的参数对算法性能影响大并且难于确定,提出一种基于反向学习[10]的变异策略代替其行为,同样达到避免算法陷入局部最优的效果。优化 10 个被广泛选用的标准测试函数,结果表明,与现有的两种人工蜂群算法相比,本文算法寻优精度高、收敛速度快、鲁棒性好。

1 人工蜂群算法的描述

ABC 算法模拟实际蜜蜂采蜜机制处理函数优化问题,

收稿日期:2011-02-23;**修回日期**:2011-10-27。
作者简介:毕晓君(1964-),女,教授,博士研究生导师,主要研究方向为人工智能、智能信号处理及图像处理。
E-mail:bixiaojun@hrbeu.edu.cn

将人工蜂群分为三类：引领蜂、跟随蜂和侦查蜂[11]。引领蜂、跟随蜂用于蜜源的开采，侦查蜂避免蜜源种类过少。优化问题的解及相应的函数值抽象为蜜源的位置和花蜜的数量。寻找最优蜜源的过程如下：引领蜂发现蜜源并记忆，在各蜜源附近搜索新蜜源，根据前后蜜源的花蜜数量选择较优蜜源并作标记；引领蜂释放与标记蜜源质量成正比的信息，用来招募跟随蜂，跟随蜂在某种机制下选取合适的标记蜜源并在其附近搜索新蜜源，与标记蜜源进行比较，选取较优异的蜜源作为本次循环的最终标记蜜源，反复循环寻找最佳蜜源。但是如果在采蜜过程中，蜜源经若干次搜索不变，相应的引领蜂变成侦查蜂，随机搜索新蜜源。

一个函数优化问题可以表示如下：

$\min f=f(\boldsymbol{x})$，　$\boldsymbol{x}=(x_1,x_2,\cdots,x_m)\in S$，　$S=[x_{iL},x_{iH}]$

式中，f 表示目标函数；$\boldsymbol{x}$ 为 m 维变量；$[x_{iL},x_{iH}]$ 为第 i 维变量对应的上下界。

设算法运行中蜜源、引领蜂、跟随蜂数量都为 N。ABC 算法的具体步骤如下：

步骤 1　按照式(1)随机产生 $2N$ 个位置。

$$V_{ij}=x_{jL}+rand\times(x_{jH}-x_{jL}) \tag{1}$$

式中，V_{ij} 为第 i 个蜜蜂第 j 维对应的搜索后的位置；x_{jL}、x_{jH} 代表第 j 维变量的上下界。选取适应度值低的 N 个作为蜜源位置。

步骤 2　引领蜂在蜜源附近按式(2)搜索新蜜源。

$$V_{ij}=x_{ij}+R_{ij}\times(x_{ij}-x_{kj}) \tag{2}$$

式中，V_{ij} 为新的蜜源位置；x_{ij} 为蜜源 i 的第 j 维位置；x_{kj} 为随机选择的不等于 i 的蜜源 k 的第 j 维位置；R_{ij} 为$[-1,1]$间的随机数，控制搜索范围，随着搜索进行，各蜜源逐渐靠近，邻域范围逐渐减小。

步骤 3　比较前后蜜源优劣，若搜索后蜜源优于搜索前，则代替先前蜜源。

步骤 4　跟随蜂根据处于蜜源处的引领蜂释放的花蜜信息，按轮盘赌方式选择蜜源，并在其附近按式(2)搜索新蜜源。

步骤 5　比较跟随蜂及引领蜂搜索的蜜源的花蜜数量，将花蜜数量较优的 N 个作为引领蜂、蜜源位置，其余为跟随蜂位置。

步骤 6　判断是否出现侦查蜂，设置参数 $lim\ it$，若某些蜜源经 $lim\ it$ 次循环不变，放弃该蜜源，相应引领蜂变成侦查蜂，按式(1)随机产生新蜜源。

步骤 7　用新确定的引领蜂、跟随蜂、蜜源位置，从步骤 2 开始重新搜索，直到满足终止条件。

2　改进的人工蜂群算法

为了提高 ABC 算法的收敛速度及寻优精度，本文在以下三方面进行改进。

2.1　跟随蜂选择蜜源方式的改进

ABC 算法中的跟随蜂依靠轮盘赌方式选择较优蜜源进行交叉操作产生新的蜜源，轮盘赌选择本身是一种较为贪婪的选择方式，使得算法极易陷入局部最优。自由搜索算法中通过“信息素-灵敏度”配合的模型选择合适区域，在一定程度上避免了算法陷入局部最优，也保证了算法快速进化的方向。这种灵敏度与信息素配合进行区域选择的方式可看做是一种新的选择方法。在本文第 3 部分与现在较为常用的 4 种选择方法进行比较，效果更好，是一种行之有效的选择策略。因此，本文所提算法中的跟随蜂按式(3)选择较优蜜源。

$$\begin{cases} nf(i)=\begin{cases}\dfrac{f(i)-f_{\min}}{f_{\max}-f_{\min}},\ \text{if } f_{\max}\neq f_{\min}\\ 0,\ \text{else}\end{cases}\\ nf(k)\leqslant S(i)\end{cases} \tag{3}$$

式中，$f(i)$为个体 i 的适应度值；$f_{\max}$、$f_{\min}$为 N 个蜜源中的最大和最小适应度值；$nf(k)$为第 $k(k\neq i)$个蜜源的信息素；第 k 个跟随蜂的灵敏度 $S(k)\sim U(0,1)$。

2.2　新的交叉方式

在 ABC 算法中，引领蜂和跟随蜂主要依靠交叉操作(即式(2))指导种群的进化，这种方式存在许多不足，下面详细分析引领蜂和跟随蜂交叉方式的不足并提出解决方案。

2.2.1　引领蜂的交叉方式

引领蜂按照式(2)在自身位置附近随机选择一个蜜源进行交叉，虽然在一定程度上避免了陷入局部最优，但却使种群进化具有盲目性，大大降低了算法的收敛速度。

希望引领蜂较快的向较优蜜源靠近，即选取适应度值较优的蜜源与自身蜜源交叉，但是这样又会忽视较差蜜源对最优蜜源的贡献而使算法易陷入局部最优。正基于此，对种群进行有针对性的分类：让那些较为优秀的蜜源随机选择蜜源交叉，增大产生新解的概率，避免陷入局部最优；而较差蜜源选择较为优秀的蜜源进行交叉操作，用以加快算法的收敛速度。具体方式为：

if $rand<$交叉概率 CR

选取较优蜜源与自身蜜源进行交叉

else

随机选取蜜源与自身蜜源进行交叉

end

其中，$CR(i)=\dfrac{f(i)-\min(f_g)}{\max(f_g)-\min(f_g)}$，$f(i)$为引领蜂所在蜜源对应的适应度值；$\max(f_g)$、$\min(f_g)$分别代表本次迭代的最差和最优蜜源(最小化问题)；$rand$ 为随机产生的$[0,1]$的数。

由式(2)可知，新产生的蜜源可以理解为是向自身和其他个体学习的结果。在算法进化的初期，希望更多更快的向较优个体学习，而在进化后期，由于自身个体相对已经较为优秀，只有在自身左右进行小范围的迈进，才能精确搜索到最优解。基于这种思想，将式(2)更改为式(4)。

$$V_{ij}=F_1\times x_{ij}+(-1+2\times F_2)(x_{ij}-x_{kj}) \tag{4}$$

式中，学习因子 $F_1=2-\mathrm{e}^{\frac{t}{G\ln 2}}$；$F_2=\mathrm{e}^{\frac{t}{G\ln 2}}-1$。

综上所述,本文为引领蜂勘探蜜源设计的交叉方式为:

if *rand*<交叉概率 *CR*

利用信息素-灵敏度模型选取较优蜜源 k 与自身蜜源 i 按式(4)进行交叉

else

随机选取蜜源 k 与自身蜜源 i 按式(4)进行交叉

end

2.2.2　跟随蜂的交叉方式

跟随蜂依据轮盘赌方式选取较优蜜源并在其附近开采,是一种较为贪婪的进化方式,尽管可以加速种群收敛,但也增大了算法陷入局部最优的可能,究其原因有以下两点:其一,轮盘赌选择方式本身;其二,交叉方式:在进化初期,可以较快的向最优解靠近,但随着进化的进行,各个解的差别逐渐减少,再采取类似于式(4)这样的交叉方式就会降低产生新解的可能,从而使算法陷入局部最优。

于是考虑在进化后期在较优蜜源处进行混沌搜索以增强其跳出局部最优的能力。具体实现方式如下:

if *rand*<分类概率 *GCR*

按信息素-灵敏度模型选择较优蜜源 i,随机选择另一蜜源 k 按式(4)交叉

else

按信息素-灵敏度模型选择较优蜜源 i 进行混沌搜索

end

其中,$GCR=0.4+0.5\times(2-e^{\frac{g}{G}\ln 2})$。设对较优蜜源 $k(X_k=(x_{k1},x_{k2},\cdots,x_{kn}),x_{ki}\in[a_i,b_i])$ 进行混沌搜索[12],方法如下:

步骤 1　将 X_k 映射到 Logistic 方程的定义域[0,1]内,即

$$Z_{ki}^0=\frac{x_{ki}-a_i}{b_i-a_i},\ k=1,2,\cdots,m;\ i=1,2,\cdots,d \quad (5)$$

步骤 2　用 Logistic 方程进行迭代产生混沌变量序列 Z_k^m $(m=1,2,\cdots,C_{\max})$。其中 $C_{\max}$ 是混沌搜索的最大迭代次数。

步骤 3　把产生的混沌序列通过逆映射 $x_{ki}=a_i+(b_i-a_i)Z_{ki}^m$ 返回到原解空间,计算其适应度值,并将其与原来的解比较,保留最好解。

步骤 4　若达到最大迭代次数,则优化过程结束,否则返回步骤 2。

由新设计的跟随蜂交叉方案可以看出,即便是在进化初期进行混沌搜索,由于是选在较优蜜源附近开采,也是一种较为贪婪的方式,不会过多的降低算法的收敛速度。

2.3　反向学习变异策略

在 ABC 算法中,除必要的种群数目外只有控制侦查蜂行为的参数 *lim it* 需要确定,而且对算法性能影响较大,参数调试麻烦,但侦查蜂行为的本质是为避免算法陷入局部最优,完全可以采取其他操作达到同样效果。为此,本文提出反向学习变异策略,具体描述为:

for $i=1:n$(n 为种群数目)

if *rand*<变异概率 *PM*

对该蜜源进行反向学习

end

end

其中,$PM=0.01+0.1\times(2-e^{\frac{g}{G}\ln 2})$。

本文提出的方案是鉴于以下考虑:在进化末期,种群相对成熟和优秀,希望变异概率较大从而增强其跳出局部最优的能力。而文献[13]已从数学上证明:依靠产生候选解的相对点取代原候选解的反向学习策略是对原候选解的一种较好估计,与产生随机点代替原候选解的方式相比,通常会取得更佳的优化效果,不仅大大提高了收敛速度,而且在一定程度上避免了陷入局部最优。所以变异策略采用反向学习的方法,而不选择传统变异中产生随机点的方式。

反向学习[10]方法如下:

蜜源所在位置为 X_b。反向学习后的新蜜源位置为 X'_b,则新位置的第 j 维 X'_{bj} 如式(6)所示。

$$X'_{bj}=X_{jL}+X_{jH}+rand\times X_{bj} \quad (6)$$

如果新位置对应的蜜源更佳,则用其代替原蜜源位置。

另外,每代的最差蜜源通常不会对最优解有所贡献,而且大大影响了算法的收敛速度。因此,对每代最差蜜源也进行反向学习。

2.4　改进人工蜂群算法的算法步骤

经过上述修改的算法称为高效人工蜂群算法(fast artificial bee colony algorithm, FBC 算法),具体操作步骤如下:

步骤 1　算法的初始化,设置引领蜂、跟随蜂数目 YL、GS;算法运行的最大迭代次数 G;混沌搜索的最大迭代次数 $C_{\max}$;产生初始种群。

步骤 2　每个引领蜂按照新的交叉方式产生新的蜜源。

步骤 3　新产生的蜜源与原蜜源做相应对比取优,确定标记蜜源。

步骤 4　每个跟随蜂都在标记蜜源中选取合适蜜源依据新的交叉方式产生新蜜源。

步骤 5　选取标记蜜源和跟随蜂所产生蜜源中较优个体作为本次迭代的蜜源,也即引领蜂的位置。

步骤 6　进行反向学习的变异策略。

步骤 7　判断是否满足算法终止条件,若满足,输出最优结果;否则转至步骤 2。

2.5　本文算法的计算量分析

与原有的 ABC 算法相比,本文算法在选择方式、搜索方式、变异上都有所变化,有必要对算法改进前后的计算量进行分析。算法的计算量通常以迭代次数再乘以每次迭代所消耗的乘法和加法次数来衡量。设引领蜂、跟随蜂数目都为 N,以一次迭代分析改进前后算法的计算量。

ABC 算法的计算量如下:

(1) 跟随蜂选择较优蜜源时依靠轮盘赌选择方式,主要消耗在累加概率的计算中,累加适应值需计算 $N-1$ 次加法,累加概率需计算 N 次乘法;

(2) 引领蜂、跟随蜂搜索需计算 $3N$ 次加法,$2N$ 次

乘法；

(3) 侦查蜂操作需计算 $2N$ 次加法，N 次乘法。

本文算法的计算量如下：

(1) 信息素-灵敏度配合的选择方式需计算 $N+1$ 次加法，N 次乘法；

(2) 引领蜂搜索需计算

$5\times CR\times N+(1-CR)\times 3\times N$ 次加法，$3\times CR\times N+(1-CR)\times 2\times N$ 次乘法；

(3) 跟随蜂搜索需计算

$5\times GCR\times N+(1-GCR)\times 2\times N\times C_{\max}$ 次加法，$3\times GCR\times N+(1-GCR)\times C_{\max}\times N$ 次乘法；

(4) 反向学习变异策略需分别计算 $PM\times 2N+1$、$PM\times N+1$ 次加法和乘法。

通过上述分析表明本文算法的计算量略有增加。

3 实验仿真与结果分析

为验证本文算法——快速人工蜂群(fast artificial bee colony, FABC)算法的有效性，进行了大量的实验仿真，实验仿真是在 Intel Centrino Duo, CPU:T7250、1 GB 内存、2.0 GHz 主频的计算机上实现，程序采用 Matlab7.5 语言实现。

3.1 测试函数

本文实验采用函数优化领域被广泛采用的 10 个标准测试函数[14-16]进行测试，表 1 给出了这些测试函数的定义、取值范围，函数 10 的理论最优值为 8.881 8e−016，其余函数的理论最优值都为 0。

表 1 测试函数

函数定义	变量范围
$f_1(x)=\sum_{i=1}^{D}x_i^2$	$[-100,100]^D$
$f_2(x)=\sum_{i=1}^{D}(\sum_{i=1}^{j}x_j)^2$	$[-100,100]^D$
$f_3(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-100,100]^D$
$f_4(x)=\max_i\{\lvert x_i\rvert\}$	$[-100,100]^D$
$f_5(x)=\sum_{i=1}^{D}\lfloor x_i+0.5\rfloor^2$	$[-100,100]^D$
$f_6(x)=\sum_{i=1}^{D}ix_i^4+\mathrm{rand}[0,1)$	$[-1.28,1.28]^D$
$f_7(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	$[-5.12,5.12]^D$
$f_8(x)=0.5+\dfrac{\sin(\sqrt{\sum_{i=1}^{D}x_i})^2-0.5}{(1+0.001\sum_{i=1}^{D}x_i)^2}$	$[-5.12,5.12]^D$
$f_9(x)=\dfrac{1}{4\,000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\dfrac{x_i}{\sqrt{i}})+1$	$[-600,600]^D$
$f_{10}=-20\exp(-0.2\sqrt{\dfrac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp(\dfrac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i)+20+\mathrm{e}$	$[-32,32]^D$

函数 1～4 是连续型单模态函数，函数 5 为非连续的单模态函数，它们主要用来测试算法的寻优精度，考察执行性能；函数 6 为噪音函数，很难收敛到全局最优解，其余函数都为复杂的多模态函数，有许多局部极值点，一般算法难于找到全局最优值，因此可用来检测算法的全局搜索性能和避免早熟的能力。

3.2 验证选择方式的有效性

选择方式的好坏会直接影响算法的性能，为验证本文提出的选择方式的效果好坏，更改跟随蜂选择较优蜜源的方式为本文的选择方式、排序选择[17]、分裂选择[18]和联赛选择[19]并与 ABC 算法中的轮盘赌方式进行对比。

鉴于公平性原则，设置相同的参数：引领蜂、跟随蜂和蜜源的数量都为 25；$limit$ 为 500；各函数迭代 1 000 次。针对测试函数 1，各算法均随机运行 30 次，选取测试函数(30 维)的平均值和方差来考察各算法的寻优性能。五种方法的对比结果如表 2 所示。

表 2 选择方式对算法性能的影响

方法	平均值	方差
轮盘赌选择	1.829 4e−09	2.639 2e−09
分裂选择	9.798 1e−12	1.063 7e−11
排序选择	1.978 1e−12	2.033 1e−12
联赛选择	1.514 1e−12	1.380 4e−12
本文方法	2.623 8e−028	6.174 9e−055

显然，与其他几种选择方式相比，本文方法大大提高了算法的寻优精度并且比较稳定，是一种更为有效的选择策略。

3.3 交叉方式的有效性

为验证本文提出的交叉方式的有效性，在基本的 ABC 算法上做出相应的改进，分别称单独改变引领蜂、跟随蜂交叉方式的算法为引领交叉蜂群法(lead crossover artificial bee colony algorithm, LCABC)、跟随交叉蜂群法(follow crossover artificial bee colony algorithm, FCABC)。

参数设置为：引领蜂、跟随蜂的数量都为 50；其他参数及条件同第 3.2 节。测试函数 1、2 都取 50 维以平均值、方差、平均迭代次数及运行时间作为评价标准来考察各算法的寻优性能。性能比较结果如表 3 所示。

表 3 交叉方式对算法性能的影响

函数	方法	均值	方差	迭代次数	运行时间/s
F1	ABC	5.5e−05	2.3e−04	1 000	2.78
	LCABC	0	0	727.8	3.43
	FCABC	0	0	272.2	4.51
F2	ABC	7.9e−04	4.3e−19	1 000	4.16
	LCABC	5.4e−211	0	1 000	4.46
	FCABC	4.0e−10	2.1e−19	1 000	19.18

通过表 3 数据可以明显地看出更改引领蜂或跟随蜂的交叉方式都大大提高了算法的运行精度和收敛速度，而且运行时间并未增加多少。足以见证本文提出的交叉方式的有效性。

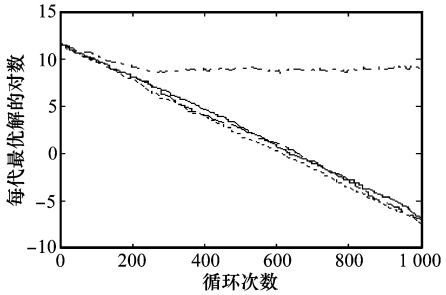
3.4　反向学习变异策略的有效性

为验证本文提出的反向学习变异策略的有效性,仅将基本 ABC 算法中的侦查峰行为修改为反向学习变异策略,并与 ABC 算法(最优控制参数 *lim it* 下)进行比较,其他条件及参数设置与第 3.2 中相同,对函数 1 进行测试,性能比较结果如图 1 所示。

从图 1 可以看出:① 参数 *lim it* 对算法性能的影响较大;② 本文提出的反向学习变异策略不仅有效的代替了跟随蜂的作用,而且取得了比侦查蜂行为更佳的效果。最重要的是免除了算法参数设置的麻烦。

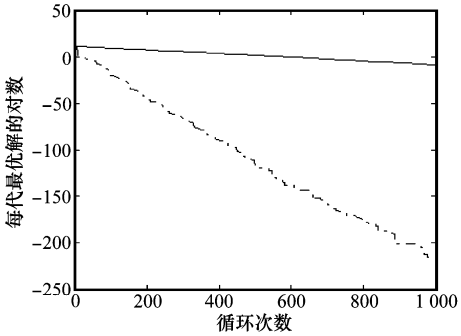
3.5　本文算法的性能分析

为验证本文提出算法的性能优劣,从以下几方面做比较:算法的寻优精度,将其与 ABC 算法、目前改进效果较好的 ABCP 算法[20] 在最优值、平均值、方差方面进行比较;收敛速度和复杂度,分别以平均迭代次数、运行时间作为评价准则;为验证本文提出算法的鲁棒性优劣对 10 个主要的标准测试函数进行测试。测试函数都取 50 维,实验条件及参数设置同第 3.2 节。各函数的性能比较结果如表 4 所示。为了直观的表现各算法的收敛速度,随机选取一次实验的函数进化过程曲线进行比较,比较结果如图 2 所示,其中,横、纵坐标分别为进化代数、每代最优适应度值的对数。



(a) 改变参数对算法性能的影响

-·-·-: *lim it*=10; — —: *lim it*=50; ----: *lim it*=100; ——: *lim it*=500。



(b) 本文算法与最优参数设置下的ABC算法的比较

——: ABC最优; -·-·-: 反向学习变异策略。

图 1　反向学习变异策略的效果

表 4　三种算法的性能比较

函数	方法	最优值	平均值	方差	平均迭代次数	运行时间/s
1	FABC	0	0	0	28.4	1.483
	ABCP	2.720 2e−180	1.007 6e−173	0	1 000	209.844
	ABC	7.895 4e−006	5.051 0e−005	2.339 4e−004	1 000	2.719
2	FABC	0	0	0	27.8	43.493
	ABCP	1.088 8e−175	1.005 2e−172	0	1 000	429.906
	ABC	1.838 4e−004	7.954 9e−004	4.336 8e−019	1 000	47.453
3	FABC	0	0	0	765.3	18.248
	ABCP	4.054 7e−099	2.745 3e−094	0	1 000	197.984
	ABC	0.015 5	10.074 1	323.956 4	1 000	3.36
4	FABC	0	0	0	689.6	14.872
	ABCP	5.036 6e−072	7.732 3e−071	0	1 000	198.453
	ABC	32.414 4	57.502 3	76.912 9	1 000	3.152
5	FABC	0	0	0	8.3	1.047
	ABCP	0	0	0	37.8	213.672
	ABC	0	0.400 0	4.690 4	1 000	0.812
6	FABC	0	0	0	187.6	75.107
	ABCP	9.268 1e−016	1.839 5e−016	0	1 000	445.547
	ABC	0.101 0	0.162 6	0.145 4	1 000	30.078
7	FABC	0	0	0	15.5	2.223
	ABCP	0	0	0	1 000	193.891
	ABC	0.056 4	0.095 0	0.182 0	1 000	10.281
8	FABC	0	0	0	12.5	1.261
	ABCP	0.009 715 9	0.009 715 9	0	1 000	176.391
	ABC	0.152 6	0.218 2	0.252 6	1 000	2.86
9	FABC	0	1.332 3e−016	3.648 5e−032	938	27.778
	ABCP	0.437 12	0.679 6	0.190 3	1 000	254.625
	ABC	4.743 7e−012	0.001 5	0.023 6	1 000	6.031
10	FABC	8.881 8e−016	8.881 8e−016	0	40.7	3.507
	ABCP	8.881 8e−016	8.881 8e−016	0	196.5	140.449
	ABC	0.001 1	0.002 6	1.078 4e−006	1 000	10.002 4

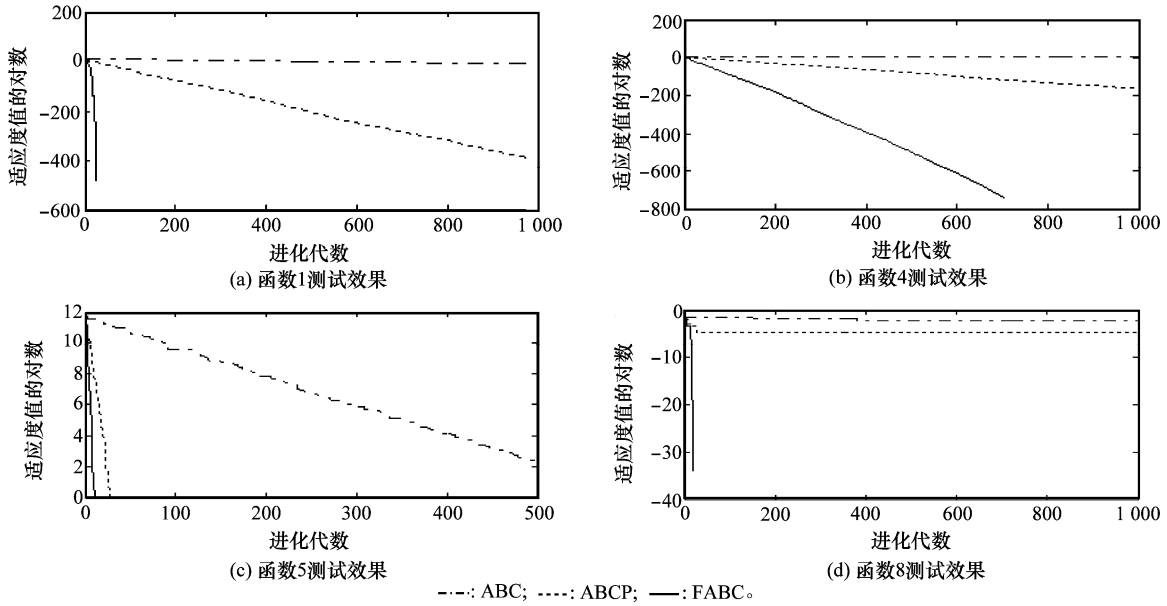


图 2 各函数的进化过程比较

从以上数据很容易得到以下结论：

(1)寻优精度：ABCP 算法与 ABC 算法相比，寻优精度有所改善，但也仅在函数 5、7、10 上收敛到全局最优解，而本文算法对各函数(除函数 9 外)寻优，每次都可以得到全局最优解，而函数 9 也可以找到全局最优解，寻优精度明显得到大幅度提高；

(2)收敛速度：ABC 算法和 ABCP 算法在 1 000 代以内几乎都没有收敛，但本文提出的算法在 1 000 代以内都已收敛，某些函数甚至在十几代就可以精确收敛到理论最优解。另外，通过图 2 也可以直观地看出，与另两种人工蜂群算法相比，本文算法的收敛速度有显著提高；

(3)鲁棒性：与其他人工蜂群算法相比，本文提出算法在各测试函数上都取得了非常满意的效果，且算法稳定，是一种鲁棒性较好的算法；

(4)复杂度：在相同的截止代数内，ABC 算法比本文算法的运行时间略短，但都比 ABCP 算法的运行时间短得多，也就是说，与基本算法相比，本文算法的复杂度略有增加。这与第 2.5 节中的算法计算量分析也是吻合的。而根据“没有免费午餐的理论”，一个算法不可能在各性能指标上都有提高，所以在其他性能指标都有大幅度提高的前提下，复杂度略有增加也是正常的。

由此可以得出结论：与其他蜂群算法相比，本文算法的寻优精度高、鲁棒性好、收敛速度快、复杂度不高，是一种很好的函数优化方法。

4 结束语

针对 ABC 算法易陷入局部最优、收敛速度慢的缺点，本文提出一种改进的人工蜂群算法——FABC 算法。提出一种新的交叉方式使种群快速收敛于全局最优解；以一种新的基于反向学习的变异策略代替跟随蜂行为，解决算法参数设置的难题，并且避免算法陷入局部最优。与传统的 ABC 算法及 ABCP 算法相比，本文算法获得的最优解精度明显提高很多，而且只需要经过较少次的迭代，大大提高了寻优能力；在运行时间上，FABC 算法与 ABC 算法运行时间相差不大，且都比 ABCP 算法短得多，即本文算法复杂度不高。简言之，本文算法寻优精度高、鲁棒性好、收敛速度快、复杂度不高。由此可见，本文提出的 FABC 算法在解决函数优化问题上优势明显，是一种较为优秀的函数优化方法。

参考文献：

[1] Karaboga D. A idea based on bee swarm for numerical optimization[R]. Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[2] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm[J]. *Applied Soft Computing*, 2008, 8(1): 687 - 697.

[3] Lou Y, Li J L. A differential evolution algorithm based on ordering of individuals[C]// *Proc. of the 2nd International Conference on Industrial Mechatronics and Automation*, 2010: 105 - 108.

[4] He L H, Yao N, Wu J H, et al. Application of modified PSO in the optimization of reactive power[C]// *Proc. of the Chinese Control and Decision Conference*, 2009: 3493 - 3496.

[5] 暴励，曾建潮. 自适应搜索空间的混沌蜂群算法[J]. 计算机应用研究，2010，27(4)：1331 - 1335. (Bao L, Zeng J C. Self-adapting search space chaos-artificial bee colony algorithm[J]. *Application Research of Computers*, 2010, 27(4): 1331 - 1335.)

[6] 丁海军,冯庆娴. 基于 Boltzmann 选择策略的人工蜂群算法[J]. 计算机工程与应用,2009,45(31):53 - 55. (Ding H J, Feng Q X. Artificial bee colony algorithm based on Boltzmann selection policy[J]. *Computer Engineering and Applications*,2009,45(31):53 - 55.)

[7] 罗钧,樊鹏程. 基于遗传交叉因子的改进蜂群优化算法[J]. 计算机应用研究,2009,26(10):3751 - 3753. (Luo J, Fan P C. Improved particle swarm optimization based on genetic hybrid genes[J]. *Application Research of Computer*,2009,26(10):3751 - 3753.)

[8] Zhu G P, Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization[J]. *Applied Mathematics and Computation*,2010,217(7):3166 - 3173.

[9] Penev K, Littlefair G. Free Search—a comparative analysis[J]. *Information Sciences*,2005,172(1):173 - 193.

[10] Tizhoosh H. Opposition-based learning: a new scheme for machine intelligence[C]// *Proc. of the International Conference on Computational Intelligence for Modelling, Control and Automation*, 2005:695 - 701.

[11] Nurhan K. A new design method based on artificial bee colony algorithm for digital IIR filters[J]. *Journal of the Franklin Institute*,2009,346(4):328 - 348.

[12] 曲良东,何登旭. 一种混沌人工鱼群优化算法[J]. 计算机工程与应用,2010,46(22):40 - 42. (Qu L D, He D X. Novel artificial fish-school algorithm based on chaos search[J]. *Computer Engineering and Application*,2010,46(22):40 - 42.)

[13] Rahnamayan S, Tizhoosh H R, Salama M M A. Opposition versus randomness in soft computing techniques[J]. *Applied Soft Computing*,2008,8(2):906 - 918.

[14] Mahamed G H O, Andries P E, Ayed S. Self-adaptive barebones differential evolution[C]// *Proc. of the IEEE Congress on Evolutionary Computation*,2007:2858 - 2865.

[15] Hao Z F, Guo G H, Huang H. A particle swarm optimization algorithm with differential evolution[C]// *Proc. of the Sixth International Conference on Machine Learning and Cybernetics*,2007:1031 - 1035.

[16] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm[J]. *Applied Mathematics and Computation*, 2009,214(1):108 - 132.

[17] 王斌,施朝健. 多边形近似曲线的机遇排序选择的拆分合并算法[J]. 计算机辅助设计与图形学学报,2006,18(8):1149 - 1153. (Wang B, Shi C J. Ploygonal approximation of curves using split-and-merge method with ranking selection[J]. *Journal of Computer-Aided Design & Computer Graphics*,2006, 18(8):1149 - 1153.)

[18] 田东平. 改进的 AGA 及其在约束函数优化中的应用[J]. 计算机工程与应用,2010,46(17):30 - 32. (Tian D P. Improved adaptive genetic algorithm and its application in constrained function optimization[J]. *Computer Engineering and Applications*,2010,46(17):30 - 32.)

[19] 薛富强,葛临东. 用于调制信号特征选择的改进遗传算法[J]. 计算机工程,2008,34(3):213 - 214. (Xue F Q, Ge L D. Improved genetic algorithm for feature selection of modulation signal[J]. *Computer Engineering*,2008,34(3):213 - 214.)

[20] Liu X B, Cai Z X. Artificial bee colony programming made faster[C]// *Proc. of the Fifth International Conference on Natural Computation*,2009:154 - 159.