基于 Memetic 框架的混沌人工蜂群算法*

李志勇[a], 李玲玲[b], 王　翔[c], 王　艳[b]
(郑州航空工业管理学院 a. 数理系; b. 计算机科学与应用系; c. 土木建筑工程学院, 郑州 450015)

摘　要: 针对无约束优化问题,提出一种新颖的混沌人工蜂群算法。新算法在 Memetic 算法框架的基础上,采用人工蜂群算法作为全局搜索算法,采用混沌搜索算子作为局部搜索算法。为了进一步提升算法的开采能力,新算法的侦察蜂抛弃了随机生成新食物源的方法,采用针对陷入局部极值食物源进行混沌局部搜索生成候选食物源的方式。针对五个标准 Benchmark 函数的仿真实验结果显示,与标准人工蜂群算法相比,新算法求解精度具有一定优势。

关键词: Memetic 框架; 混沌; 人工蜂群算法; 优化
中图分类号: TP391;TP301.6　　**文献标志码**: A　　**文章编号**: 1001-3695(2012)11-4045-05
doi:10.3969/j.issn.1001-3695.2012.11.011

Chaos artificial bee colony based on Memetic framework

LI Zhi-yong[a], LI Ling-ling[b], WANG Xiang[c], WANG Yan[b]
(*a. Dept. of Mathematics & Physics, b. Dept. of Computer Science & Technology, c. School of Civil Engineering, Zhengzhou Institute of Aeronautical Industry Management, Zhengzhou* 450015, *China*)

Abstract: This paper proposed novel chaos-artificial bee colony algorithm for continuous function optimization problems. Based on the Memetic algorithm framework, the new algorithm took artificial bee colony algorithm as the global search algorithm and the chaos local search operator as the local search operator algorithm. Furthermore, for the food source which was into local minimum, the scout didn't randomly generate the new food source, but used the chaos local operator to generate the candidate food source in order to enhance the exploit of bee colony. The simulation results for five Benchmark functions show that, compared with those of artificial bee colony algorithm, the new algorithm has the advantages to the solutions' quality.
Key words: Memetic framework; chaos; artificial bee colony; optimization

0　引言

人工蜂群算法(artificial bee colony,ABC)[1]是一种模拟蜜蜂采蜜行为的新型群智能算法,它于 2005 年由 Karaboga 以技术报告的形式提出。由于 ABC 算法具有参数少、计算简单、收敛速度快等优点,被成功应用于无约束优化问题[2]、约束优化问题[3]和离散组合优化问题[4]。然而,文献[5~7]都指出 ABC 算法存在局部开采能力不足以及收敛速度慢的缺陷。

为了提升 ABC 算法的局部搜索能力,许多学者进行了不同的尝试。其中,利用混沌序列的随机性、遍历性和规律性改进 ABC 算法寻优能力是一种重要的解决方案。此类算法的改进策略可以分成两种类型:a)混沌初始化[8],其基本思想是利用混沌序列生成初始食物源种群,从而使得初始种群在定义域内的分布更加均匀;b)混沌侦察蜂[8~10],其思想是在侦察蜂阶段,针对陷入局部极值的食物源 X_i,利用混沌序列进行局部搜索,进而产生较优的食物源取代 X_i。虽然混沌初始化和混沌侦察蜂在一定程度上能够提升算法寻优能力,但是它们针对 Rosenbrock 函数的实验结果并不理想。

与以上改进算法思路不同,本文提出了一种新型的混沌人工蜂群算法(chaos artificial bee colony,CABC)。新算法基于 Memetic 框架[11],一方面将 ABC 作为全局搜索算法,另一方面设计了一种混沌搜索算子作为局部搜索算法。此外,为了进一步提升求解质量,新算法还引入了混沌侦察蜂,同时利用目标函数进行选择操作。针对五个标准 Benchmark 函数的实验结果显示,新算法的求解质量显著优于标准 ABC 算法。

1　基本人工蜂群算法

基本人工蜂群算法包含五个阶段:食物源初始化、雇佣蜂阶段、观察蜂选择食物源概率的计算、观察蜂阶段、侦察蜂阶段。以下是这五个阶段的详细解释,其中假定求解最小化问题,问题规模为 D,种群大小为 sn,食物源数目、雇佣蜂数目和观察蜂数目都为 $sn/2$。

1.1　食物源初始化

在 ABC 算法中,每个食物源代表一个解决方案,食物源初始化包含两个基本操作:

a)按照式(1)随机产生 $sn/2$ 个 D 维食物源向量 $\{X_i=(x_i,$

收稿日期: 2012-02-27; **修回日期**: 2012-05-03　**基金项目**: 国家自然科学基金资助项目(41171341);国家教育部新世纪优秀人才支持计划资助项目(NCET-09-0126);河南省科技创新人才计划杰出青年基金资助项目(114100510006);中国航空科学基金资助项目(20095155008);河南省重点科技攻关计划资助项目(112102310082);河南省软科学研究计划资助项目(122400450306)

作者简介:李志勇(1976-),男,河南临颍人,讲师,主要研究方向为演化算法、数据挖掘(lizhiyong1976@126.com);李玲玲(1973-),女,河南开封人,副教授,博士,主要研究方向为模式识别;王翔(1976-),男,河南洛阳人,博士,主要研究方向为进化算法;王艳(1978-),女,河南周口人,副教授,硕士,主要研究方向为数据挖掘.

$x_i^2,\cdots,x_i^D)\mid i=1,\cdots,sn/2\}$。

$$x_i^j = x_{\min}^j + \mathrm{rand}(0,1)(x_{\max}^j - x_{\min}^j) \tag{1}$$

其中：$x_{\min}^j$ 和 $x_{\max}^j$ 分别是 X_i 中第 j 维的下限和上限。

b）按照式（2）初始每个食物源的状态标志。

$$\mathrm{failure}_i = 0 \tag{2}$$

其中，$i=1,\cdots,sn/2$；$\mathrm{failure}_i$ 表示食物源 X_i 没有得到提升的连续次数，它作为侦察蜂随机搜索的判断指标。

1.2 雇佣蜂阶段

该阶段针对每个食物源，利用雇佣蜂进行局部搜索操作。其思路为：a）针对食物源 $\{X_i=(x_i^1,x_i^2,\cdots,x_i^D)\mid i=1,\cdots,sn/2\}$ 中的一个随机位置 j，按照式（3）进行变异操作，从而生成新的候选食物源 $\{V_i=(x_i^1,x_i^2,\cdots,v_i^j,\cdots,x_i^D)\mid i=1,\cdots,sn/2\}$；b）根据适应度函数，在 X_i 与 V_i 之间进行选择操作，进而生成 X_i 的新食物源位置：

$$v_i^j = x_i^j + \mathrm{rand}(-1,1)(x_i^j - x_k^j) \tag{3}$$

其中：$j\in\{1,2,\cdots,D\}$；$k\in\{1,2,\cdots,sn/2\}$，且 $k\neq i$。

1.3 观察蜂选择食物源概率的计算

与雇佣蜂遍历每个食物源不同，观察蜂根据概率选择食物源，食物源 X_i 的选择概率 p_i 与其适应度函数 $\mathrm{fitness}_i$ 相关。其实现步骤为首先根据每个食物源 X_i 的目标函数 f_i，利用式（4）计算其适应度函数 $\mathrm{fitness}_i$；然后根据适应度函数，利用式（5）确定 X_i 的选择概率 p_i。

$$\mathrm{fitness}_i = \begin{cases} 1/(1+f_i) & \text{if} \quad f_i \geqslant 0 \\ 1+\mathrm{abs}(f_i) & \text{if} \quad f_i < 0 \end{cases} \tag{4}$$

$$p_i = \mathrm{fitness}_i / \sum_{i=1}^{sn/2} \mathrm{fitness}_i \tag{5}$$

由式（4）（5）可知，拥有较优目标函数的食物源被选中的概率也较大。

1.4 观察蜂阶段

该阶段根据选择概率，在较优的食物源周围进行局部搜索操作。一只观察蜂包含两个操作步骤：a）依据每个食物源的选择概率 $\{p_j\mid j=1,\cdots,sn/2\}$，利用轮盘赌法选出一个食物源 X_i；b）针对 X_i 执行与 1.2 节中雇佣蜂相同的变异和选择操作。

1.5 侦察蜂阶段

侦察蜂阶段中将陷入局部极值的食物源位置放弃。实现思路是，如果一个食物源 X_i 在指定的迭代次数 limit 内没有提升，那么放弃其位置，进而利用式（1）随机生成新位置。

2 基于 Memetic 框架的混沌人工蜂群算法

为了提升局部开采能力，提出了一种基于 Memetic 框架的人工蜂群算法。新算法在基本 ABC 算法基础上引入了四种创新机制：Memetic 框架、混沌局部搜索算子、混沌侦察蜂和基于目标函数的选择算子。

2.1 Memetic 框架

Memetic 是一种基于种群全局搜索和个体局部搜索的混合算法框架，它于 2000 年开始被学术界认可，近期成为一个研究热点。文献[12]设计的新算法基于 Memetic 框架，其中 ABC 作为全局搜索算法，混沌局部搜索算子作为局部搜索算法。当 ABC 中的雇佣蜂和侦察蜂完成搜索之后，针对种群中的最优食物源 X_{best}，调用混沌局部搜索算子，进一步提升 X_{best} 的质量。

2.2 混沌局部搜索算子

混沌局部搜索算子在观察蜂阶段结束之后执行，用于增强新算法局部搜索能力。该算子利用混沌序列的随机性、遍历性和规律性，针对种群中最优食物源 X_{best} 进行局部搜索操作，同时，该算子还引入了一种收缩策略，具有开始阶段搜索范围较大、进化后期搜索范围较小的特点。其步骤如下：

a）在食物源 $\{X_j=(x_j^1,x_j^2,\cdots,x_j^D)\mid j=1,\cdots,sn/2\}$ 中，选出当前种群最优食物源 X_{best}，同时设置 $i=1$。

b）利用式（6）生成 Logistic 混沌变量 ch_i。

$$ch_{i+1} = 4\times ch_i \times (1-ch_i) \quad i=\{1,\cdots,K\} \tag{6}$$

其中：$ch_i\in(0,1)$ 且 $ch_i\neq 0.25, 0.5$ 和 0.75；同时 ch_1 是 $(0,1)$ 之间均匀分布的随机数；K 表示混沌序列的长度。

c）利用式（7），将混沌变量 ch_i 映射成定义域 $[LB,UB]$ 内的混沌向量 CH_i。

$$CH_i = LB + ch_i \times (UB-LB) \quad i=\{1,\cdots,K\} \tag{7}$$

其中：LB 和 UB 分别表示定义域的下限向量和上限向量。

d）利用式（8），将混沌向量 CH_i 与最优食物源 X_{best} 线性组合，生成候选向量食物源 V_i。

$$V_i = (1-\lambda)X_{\mathrm{best}} + \lambda \times CH_i \quad i=\{1,\cdots,K\} \tag{8}$$

其中，λ 是收缩因子，由式（9）确定。

$$\lambda = \frac{\mathrm{maxFes} - \mathrm{fes} + 1}{\mathrm{maxFes}} \tag{9}$$

其中：maxFes 表示最大适应度函数评价次数；fes 表示当前适应度函数评价次数。

由式（9）可知，λ 随进化代数增加逐渐变小，又由式（8）可知，λ 越小则混沌搜索的范围越小。综上可知随着进化的进行，局部搜索范围越来越小。

d）如果 V_i 的目标函数优于 X_{best}，那么 $X_{\mathrm{best}}=V_i$，局部搜索结束；否则，如果混沌序列长度达到 K，那么局部搜索也结束。否则，$i=i+1$，跳到第 b）步继续执行。

这里需要特别指出的是，本文选用了 Logistic 混沌序列，是基于以下两点考虑：它是最常用的混沌序列；它作为局部搜索算子，被文献[13]证明是有效的。

2.3 混沌侦察蜂

在标准 ABC 算法中的侦察蜂阶段，对陷入局部极值的食物源，即连续迭代 limit 次没有提升的食物源，利用随机生成的位置进行替换操作，进而增强种群的分散性。这种侦察蜂机制存在明显缺陷，它缺少方向性，虽然增加了种群的分散性，但是也降低了算法的收敛速度。

为了克服以上缺陷，本文设计了一种混沌侦察蜂机制，它的基本思想是，针对陷入局部极值的食物源 X_j 进行替换操作时，不是使用式（1）随机生成新位置，而是利用 2.2 节的混沌局部搜索算子在 X_j 周围进一步寻优。

混沌侦察蜂与 2.2 节的混沌局部搜索算子主要的不同之处在于操作对象不同，混沌侦察蜂的操作对象是陷入局部极值的食物源 X_j，而 2.2 节的混沌局部搜索算子的对象是当前种群中的最优食物源 X_{best}。

2.4 雇佣蜂、观察蜂和侦察蜂的选择算子

新算法还对选择算子进行了改进。标准 ABC 算法中，雇佣蜂、观察蜂和侦察蜂的选择算子都利用适应度函数的函数值进行选择操作。事实上，基于适应度函数的选择操作存在一个明显缺陷：由式（4）可知，当目标函数 $f(X)>0$ 且数值较小时，

适应度函数 fitness(X)就不具有区分度,如 $f(X_1)=1e-20$、$f(X_2)=1e-120$,此时 fitness(X_1)与 fitness(X_2)都等于 1;然而,基于目标函数进行比较操作就不存在这个缺陷。故在新算法中,三种蜜蜂的选择算子都采用目标函数进行选择操作。

2.5　混沌人工蜂群算法伪码

混沌人工蜂群算法的伪码如下所示,其中黑体部分表示 CABC 算法与 ABC 算法的区别之处。

```
//初始化阶段
1   按照式(1)初始化食物源种群{X_i = (x_i^1, x_i^2, …, x_i^D) | i = 1, …, sn/2}
2   初始化每个食物源的状态标志{failure_i = 0 | i = 1, …, sn/2}
3   while(fes≤maxFes) do
//雇佣蜂阶段
4   for i = 1 to sn/2 do
5     利用式(3),针对食物源 X_i 生成候选食物源 V_i
6     根据目标函数,V_i 与 X_i 之间选出最优
7   end for
//每个食物源被观察蜂选中概率的计算
8   for i = 1 to sn/2 do
9     由式(5)计算食物源 X_i 被观察蜂选中的概率 p_i
10  end for
//观察蜂阶段
11  t = 0, i = 1
12  repeat
13  if rand(0,1) < p_i then
14    t = t + 1
15    由式(3),针对食物源 X_i 生成候选食物源 V_i
16    根据目标函数,在 V_i 与 X_i 之间选出最优
17  end if
18  i = i + 1;
19  i = i mod ((sn/2) + 1)
20  until t = sn/2
21  按照 2.3 节调用混沌局部搜索算子
//侦察蜂阶段
22  按照 2.4 节调用混沌侦察蜂机制
23  end if
24  end while
```

其中:第 6 和 16 行对应 2.4 节,表示利用目标函数进行选择操作;第 21 行对应 2.2 节,表示针对种群中最优食物源 X_{best},调用混沌局部搜索算子,增强 X_{best} 的质量;第 22 行对应 2.3 节,表示针对陷入局部极值的食物源 X_j,调用混沌局部搜索算子,帮助其跳出局部极值,增强种群的分散性。

2.6　时间复杂度分析

每次循环时,基本 ABC 算法的时间复杂度为 $O(sn)$,其中 sn 表示蜂群规模。

与之相比,CABC 算法在 ABC 的基础上,增加了混沌局部搜索算子和混沌侦察蜂机制,它们的时间复杂度都是 $O(K)$,其中 K 表示混沌序列长度。此外,由混沌侦察蜂机制可知,其调用次数不确定,但是远远小于混沌局部搜索算子调用次数,故其时间复杂度可以忽略不计。综上所述,每次循环时,CABC 算法时间的复杂度为 $O(sn+K)$。事实上,本文算法假定 K 与 sn 存在线性关系,如 $K=sn/4$,经过约减后,CABC 的时间复杂度也为 $O(sn)$,即两个算法时间复杂度属于同一数量级。

需要特别指出的是:为了比较的公平性,本文在第 3 章进行实验结果比较时,采用国际通用的适应度函数评价次数 maxFes 作为循环结束的标准,即两个算法的实验结果都是在相同的适应度函数评价次数条件下获得的。

3　仿真实验及讨论

针对五个基本的单模和多模函数,利用 CABC 算法和 ABC 算法分别进行 30 次独立实验,进而比较了两个算法的求解效果和收敛速度。

3.1　测试函数选择

五个函数的基本特征如表 1 所示。其中,Sphere 和 Rosenbrock 函数,是单模函数分别用于测试算法的求解精度和收敛速度;Arckly、Griewank 和 Rastrigin 函数都是多模函数,拥有多个局部极值点,用于测试算法的全局寻优能力。

表 1　基础的单模和多模测试函数

function	$f(X^*)$	搜索范围	函数形式
Sphere	$f(0)=0$	$[-100,100]^D$	$f(x)=\sum_{i=1}^{D}x_i^2$
Rosenbrock	$f(1)=0$	$[-2.048,2.048]^D$	$f(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$
Ackley	$f(0)=0$	$[-32.768,32.768]^D$	$f(x)=-20\times\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$
Griewank	$f(0)=0$	$[-600,600]^D$	$f(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$
Rastrigin	$f(0)=0$	$[-5.12,5.12]^D$	$f(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$

3.2　算法参数设置

为了比较的公平性,两个算法参数设置基本相同,具体设置如表 2 所示。由表 2 可知,CABC 算法增加了一个参数 K,表示混沌序列长度。这里设置 K 为种群大小 sn 的四分之一,这样设置基于以下原因:若 K 过大,一方面会增大算法的适应度函数评价次数,另一方面会造成算法的早熟收敛。

表 2　两算法参数设置

算法参数	变量	ABC	CABC
种群大小	sn	40	40
雇佣蜂数目	$sn/2$	20	20
观察蜂数目	$sn/2$	20	20
侦察蜂参数	limit	100	100
最大适应度函数评价次数	maxFes	150 000	150 000
混沌序列长度	K	—	10

3.3　实验结果

在最大适应度函数评价次数 150 000 次的条件下,两个算法对 30 维的五个测试函数进行了 30 次独立实验。表 3 展示了相应的统计结果,其中括号外是均值,括号内是标准差。

表 3　30 次独立运行实验的均值和标准差

function	CABC	ABC	Wilcoxon's test
Sphere	2.75e-44(3.61e-44)	7.24e-16(9.25e-17)	0.00
Rosenbrock	1.82e-05(3.02e-05)	10.22(5.82)	0.00
Ackley	3.82e-14(4.35e-15)	6.07e-14(1.01e-14)	0.00
Griewank	0(0)	6.79e-09(3.72e-08)	0.00
Rastrigin	1.89e-15(1.04e-14)	2.88e-13(3.64e-13)	0.00

由表 3 可知,无论是均值还是标准差,CABC 算法都优于 ABC 算法,且都存在数量级上的提升,这说明了改进算法的有效性。此外,由于 CABC 算法在 Griewank 函数上 30 次实验结果都为 0,显然不满足正态分布,故不能采用 t 检验比较两种算法的实验结果。根据文献[14]的建议,本文采用非参数的

Wilcoxon 检验,测试在每个函数上两个算法实验结果是否存在显著性差异。由表 3 的 Wilcoxon 检验结果可知,CABC 算法的实验结果全部显著优于 ABC 算法。

针对 Rastrigin 函数,CABC 算法 30 次实验的分布情况如图 1 所示。显然,实验结果非常理想,30 次实验中 29 次结果为 0,只有一次陷入局部极值。

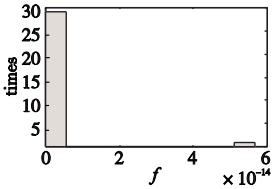

图1　Rastrigin函数30次实验最优解的分布

3.4　收敛性分析

图 2 展示了两个算法针对五个函数的收敛曲线,其中纵坐标表示 30 次实验目标函数的均值。需要特别指出的是,为了更清晰地显示趋势,纵坐标使用了实验中最优解的对数形式 $\log_{10}$(最优解)。

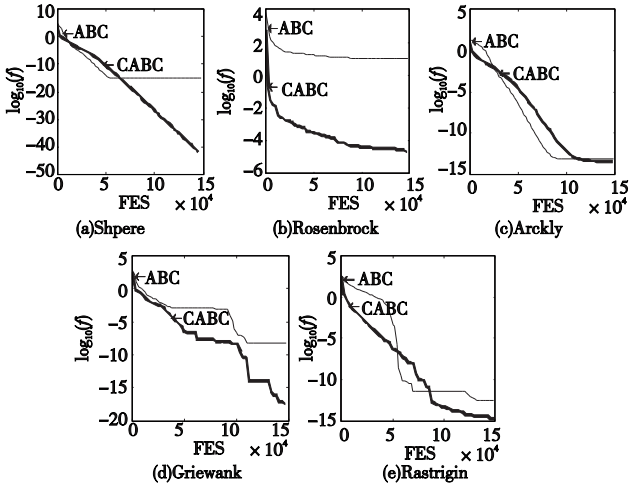

图2　五个函数的收敛曲线

根据图 2 中 CABC 算法的收敛曲线特征,可以将其分为三个类别:

a)Sphere 函数的收敛曲线。CABC 算法中该函数的收敛曲线呈线性下降状态。这是合理的,因为 Sphere 函数是单模函数,比较简单;另外 CABC 算法采用了基于目标函数的选择操作。

b)Rosenbrock 和 Arckly 函数的收敛曲线。CABC 算法中这两个函数的收敛曲线可以分为两个阶段,在第一个阶段收敛速度较快,第二个阶段收敛基本停滞,陷入局部极值。这是因为,第一个阶段中,在混沌局部搜索算子的作用下,算法收敛速度较快;第二个阶段中,算法陷入局部最优,混沌侦察蜂起到了一定的作用,但是有限。

c)Rastrigin 与 Griewank 函数的收敛曲线。CACB 算法中这两个函数的收敛曲线分成三个阶段。初始时,CABC 算法收敛速度较快;此后陷入局部极值;最后加速收敛到 0。这种现象也是合理的,它是混沌局部搜索算子与混沌侦察蜂共同作用的结果。在收敛曲线的三个阶段中,前两个阶段与类别 b)的两个阶段完全相同;在第三个阶段中,针对种群中那些陷入局部极值的较差食物源,混沌侦察蜂进行了局部搜索操作,从而再次启动种群的进化过程,最终快速进入全局最优解 0。

3.5　参数敏感性分析

本节利用方差分析法(ANOVA)探讨算法的参数敏感性,其基本原理是比较不同参数设置条件下实验结果的均值是否存在显著性差异,原假设是所有参数设置下实验结果均值相等,备择假设是至少存在一个参数设置其实验结果均值与其他不同。在 ANOVA 分析中,显著性水平设置为 0.05。此外,本文还利用了均值分析法(ANOM)寻找算法的最优参数。

与 ABC 算法相比,CABC 算法只增加了参数 K;其他参数的敏感性分析在文献[2]中已有讨论,故只讨论 K 的敏感性。由于 K 值越大,每次迭代时适应度函数评价次数越高,故它不宜设置过大,这里将其设置成一个与种群大小 sn 线性相关的变量,具体设置成 $K=\{sn/4, sn/2, sn, 2\times sn\}=\{10,20,40,80\}$ 四个不同水平,同时保持其他参数与 3.2 节完全相同,此外每组实验独立运行 30 次。需要特别强调的是每组实验的适应度函数评价次数同样设置为 maxFes = 150 000。

表 4 展示了参数 K 的 ANOVA 结果,SS 表示组间离差平方和;DF 表示 SS 的自由度;MS 表示平均组间离差平方和;F 表示 ANOVA 的 F 统计量;Sig 表示与 F 统计量相对应的显著性水平。由表 4 可知,除了 Griewank 函数,K 的不同设置会影响算法在其余四个问题上的实验结果。

表 4　不同 K(0,10,20,40,80,160)值的 ANOVA 分析

function	SS	DF	MS	F	Sig
Sphere	1.19e − 27	3	3.96e − 28	26.27	0.000
Rosenbrock	4.26e − 9	3	1.42e − 9	3.35	0.021
Arckly	1.81e − 12	3	6.04e − 13	128.13	0.000
Griewank	6.86e − 16	3	2.30e − 16	1.29	0.283
Rastrigin	2.53e − 16	3	8.44e − 17	7.54	0.000

ANOM 通过比较不同参数下实验结果的均值与所有实验结果的均值,最终确定算法参数的等级。表 5 展示了在不同 K 参数设置下 ANOM 分析的结果。由表 5 实验结果可知,$K=80$ 时实验结果存在不理想的情况,故推荐 $K=10$、20 或 40 作为比较合理的参数设置。

表 5　不同 K(0,10,20,40,80,160)值的 ANOM 分析

function	above level	below level	no significance
Sphere	10,20,40	80	—
Rosenbrock	—	—	10,20,40,80
Arckly	10,20,40	80	—
Griewank	—	—	10,20,40,80
Rastrigin	—	80	10,20,40

3.6　讨论

当比较 CABC 算法与 ABC 算法的收敛曲线(图 2)时,有两个需要特别注意的现象:

a)针对所有函数,在进化过程的初始阶段,CABC 算法的收敛速度都快于 ABC 算法。这种现象是合理的,它主要基于以下三个原因:(a)每次迭代时,CABC 算法针对 ABC 算法得出的当前最优解,利用混沌局部搜索算子继续寻优,故最优解肯定会有所改进;(b)由进化算法的特点可知,初始阶段时,当前最优解的可提升空间较大,故混沌局部搜索算子对于算法的改进是明显的;(c)由式(8)(9)可知,混沌局部搜索算子初始阶段搜索范围较大,后期搜索范围较小,故初始阶段改进是明

显的。

b)对于大部分函数,在进化过程的中期,ABC 算法的收敛速度反超 CABC 算法,如 Sphere、Arckly 和 Rastrigin 函数都存在这种现象。这主要由两个原因造成:(a)由 3.4 节收敛曲线类别 b)c)的讨论可知,在进化过程中期,种群中最优食物源已经比较优秀,混沌局部搜索算子对其没有改进;(b)每次迭代时混沌局部搜索算子都会执行一次,从而花费 K 次适应度函数评价。

综合以上两种现象的讨论以及 3.4 节关于收敛曲线三个类别的分析,可以得到以下结论:

a)在进化过程初期,混沌局部搜索算子对于种群中最优解改进较大,加快了算法收敛。

b)在进化过程中期,混沌搜索算子对于种群中最优食物源基本没有改进。

c)在进化过程后期,针对种群中已陷入局部极值的较差食物源,混沌侦察蜂有一定改进,从而重新启动了整个种群的进化过程。

4　结束语

本文提出了一种新颖的基于 Memetic 框架的混沌蜂群算法。新算法在人工蜂群算法基础上进行了以下三方面的创新:a)设计了一种混沌局部搜索算子,针对雇佣蜂和观察蜂操作完成后的最优食物源进行局部增强搜索操作,进一步增强了算法的求解效果和收敛速度;b)针对已经停滞进化的食物源,侦察蜂利用混沌序列生成候选食物源,从而进一步提升了求解质量;c)雇佣蜂、观察蜂和侦察蜂都利用目标函数进行选择操作。仿真实验结果证明了新算法的求解质量在统计意义上显著优于标准人工蜂群算法。

参考文献:

[1] KARABOGA D. An idea based on honey bee swarm for numerical optimization, TR06 [R]. Kayseri, Turkey: Erciyes University Press, 2005.

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

[3] KARABOGA D, AKAY B. A modified artificial bee colony (ABC) algorithm for constrained optimization problems[J]. Applied Soft Computing, 2011, 11(3):3021-3031.

[4] PAN Quan-ke, TASGETIREN M F, SUGANTHAN P N, *et al*. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem[J]. Information Sciences, 2011, 181(12):2455-2468.

[5] GAO Wei-feng, LIU San-yang. A modified artificial bee colony algorithm[J]. Computers and Operations Research, 2012, 39(2):687-697.

[6] GAO Wei-feng, LIU San-yang. Improved artificial bee colony algorithm for global optimization[J]. Information Processing Letters, 2011, 111(17):871-882.

[7] BANHARNSAKUN A, ACHALAKUL T, SIRINAOVAKUL B. The best-so-far selection in artificial bee colony algorithm[J]. Applied Soft Computing, 2011, 11(2):2888-2901.

[8] ALATAS B. Chaotic bee colony algorithms for global numerical optimization[J]. Expert Systems with Applictaions, 2010, 37(8):5682-5687.

[9] 罗钧,李研. 具有混沌搜索策略的蜂群优化算法[J]. 控制与决策, 2010, 25(12):1913-1916.

[10] 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):1330-1334

[11] ONG Y S, KEANE A J. Meta-Lamarckian learning in memetic algorithms[J]. IEEE Trans on Evolutionary Computation, 2004, 8(2):99-110.

[12] 公茂果,焦李成,刘芳,等. 基于神经系统与免疫系统调节机理的 Memetic 计算[J]. 中国科学:信息科学, 2010, 40(11):1428-1436.

[13] 孟红记,郑鹏,梅国辉,等. 基于混沌序列的粒子群优化算法[J]. 控制与决策, 2006, 21(3):263-266.

[14] S GARCÍA, MOLINA D, LOZANO M, *et al*. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC' 2005 special session on real parameter optimization[J]. Journal of Heuristics, 2009, 15(6):617-644.

(上接第 4044 页)

参考文献:

[1] 程代展,李志强. 非线性系统线性化综述[J]. 山东大学学报:工学版, 2009, 39(2):26-36.

[2] 胡德文,王正志. 非线性系统 Wiener 模型辨识[J]. 自动化学报, 1991, l7(2):151-159.

[3] 罗秋滨,李秀英,韩志刚,等. 一种非线性系统多项式逼近的建模方法[J]. 系统仿真学报, 2008, 20(4):829-832.

[4] 吴德会. 非线性动态系统的 Wiener 神经网络辨识法[J]. 控制理论与应用, 2009, 26(11):1192-1196.

[5] KENNEDY J, EBERHART R C. Particle swarm optimization[C]// Proc of IEEE International Conference on Neural Networks. Piscataway: IEEE Service Center, 1995:1942-1948.

[6] 毛鹏轩,肖扬. 粒子群算法在稳定时滞系统设计中的应用[J]. 计算机应用研究, 2012, 29(1):214-216.

[7] 张艳,李少远,王笑波,等. 基于粒子群优化的 Wiener 模型辨识与实例研究[J]. 控制理论与应用, 2006, 23(6):991-995.

[8] 刘利斌,欧阳艾嘉,乐光学,等. 基于混合粒子群的土壤水分特征曲线参数优化[J]. 计算机工程与应用, 2011, 47(35):218-221.

[9] 张小红,宁红梅. 基于混沌粒子群和模糊聚类的图像分割算法[J]. 计算机应用研究, 2011, 28(12):4786-4789

[10] 郑春颖,王晓丹,郑全弟,等. 自逃逸云简化粒子群优化算法[J]. 小型微型计算机系统, 2010, 31(7):1457-1460.

[11] 刘衍民,赵庆祯,邵增珍. 基于正态云的粒子群优化算法及其应用[J]. 计算机工程与应用, 2011, 47(17):161-162, 166.

[12] 李德毅,孟海军,史学梅. 隶属云和隶属云发生器[J]. 计算机研究与发展, 1995, 32(6):15-20.

[13] 张光卫,何锐,刘禹,等. 基于云模型的进化算法[J]. 计算机学报, 2008, 31(7):1082-1090.

[14] CLERC M, KENNEDY J. The particle swarm: explosion, stability, and convergence in multi-dimension complex space[J]. IEEE Trans on Evolutionary Computation, 2002, 16(1):58-73.

[15] 方伟,孙俊,谢振平,等. 量子粒子群优化算法的收敛性分析及控制参数研究[J]. 物理学报, 2010, 59(6):3686-3694.

[16] 万百五. 工业大系统优化与产品质量控制[M]. 北京:科学出版社, 2003.

敛状态时,优化算法与 PSO 算法融合,ACMPSO 算法在整个进化过程表现出了较强的持续寻优能力,未陷入局部最优,搜索到的全局最优位置较为理想,寻优结果均优于其他三种算法。

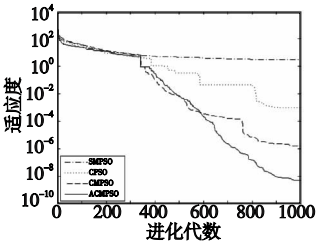

图4 Griewank函数适应度—进化代数曲线对比

表2 适应度比较表

函数	算法	平均值	最优值
Sphere	SMPSO	4.040 3	2.884 1
	CPSO	5.3104e - 06	9.7831e - 09
	CMPSO	3.6942e - 11	1.9306e - 12
	ACMPSO	3.9629e - 16	2.3041e - 17
Ackley	SMPSO	3.451 9	2.338 5
	CPSO	4.4623e - 04	2.7812e - 06
	CMPSO	8.8013e - 06	7.9732e - 07
	ACMPSO	8.0244e - 08	3.6271e - 09
Griewank	SMPSO	3.176 2	2.473 1
	CPSO	8.7421e - 04	6.9694e - 07
	CMPSO	4.5286e - 06	7.5760e - 07
	ACMPSO	4.6510e - 09	5.4381e - 10

从表 2 可以看出,ACMPSO 实验得到的适应度平均值、最优值在四种算法中都是最低,寻优效果最佳。

3 Wiener 模型系统辨识

传统的 Wiener 模型系统辨识方法有人工神经网络 ANN、量子粒子群 QPSO 算法[15]等,本文应用 ACMPSO 算法对 Wiener 模型进行辨识并与常用方法进行对比测试。

3.1 Wiener 模型描述

Wiener 模型如图 5 所示,是单输入/输出的离散时间模型,由线性子系统和无记忆的非线性增益级联构成,其差分方程描述如下:

$$\begin{cases} A(q^{-1})u(t) = q^{-d}B(q^{-1})x(t) \\ y(t) = f[u(t)] + e(t) \end{cases} \tag{16}$$

其中:$A(q^{-1}) = 1 + a_1q^{-1} + \cdots + a_nq^{-n}$,$B(q^{-1}) = 1 + b_1q^{-1} + \cdots + b_mq^{-m}$分别为 n 和 m 阶后移算子多项式;d 为系统时延;$x(t)$和 $y(t)$分别是系统的输入和输出;$e(t)$为系统噪声干扰输入;$u(t)$是系统的中间环节信号;$f[t]$为无记忆非线性静态增益函数。假设系统模型参数 n、m、d 和$f[t]$已知,设定参数$\{a_i\}$和$\{b_j\}$的变化区间,根据给定的系统输入 $x(t)$、输出 $y(t)$估计参数$\{a_i\}$和$\{b_j\}$的值。算法中控制辨识误差的适应度函数定义为

$$J = \sum_{i=1}^{L} r^{i-1}[y(t-i) - \tilde{y}(t-i)]^2 \tag{17}$$

其中:L 为辨识窗口长度;$y(t-i)$和$\tilde{y}(t-i)$是 $t-i$ 时刻输出的测量值和估计值;r^{i-1}是第 i 个数据的权重。系统辨识的目标是在窗口 L 长度内输出的误差和 J 最小,该非线性系统的辨识问题即可等价为在相应的限定条件下求极小值问题。

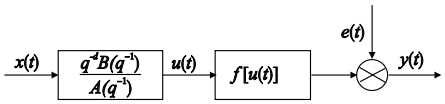

图5 Wiener模型

3.2 Wiener 模型系统辨识测试

作为辨识目标的 Wiener 模型[16]如式(18)所示:

$$\begin{cases} u(t) = 1.5u(t-1) - 0.7u(t-2) + x(t-1) + 0.5x(t-2) \\ f[u(t)] = \begin{cases} \sqrt{u(t)/2} & u(t) \geqslant 0 \\ -\sqrt{-u(t)/2} & u(t) < 0 \end{cases} \\ y(t) = f[u(t)] + e(t) \end{cases} \tag{18}$$

输入信号 $x(t)$是零均值、方差为 1 的高斯白噪声;噪声 $e(t)$是零均值、方差为 0.1 的高斯白噪声。辨识窗口宽度 $L = 500$,权重 r^{i-1}设置为 1,后移算子 n 和 m 取值为 2,系统时延 d 取值为 0。需要辨识的参数真值为$\{-1.5, 0.7, 1, 0.5\}$,参数的选择范围限制在$[-2, 2]$。PSO 算法种群规模均为 20,进化代数为 200,c_1 和 c_2 等于 2;ACMPSO 算法中 N 取值为 5,L 的值为 10,K 取 $\sqrt{10}$;QPSO 的学习系数取值方法为从 0.8 线性下降到 0.3。BP 神经网络隐层神经元个数为 10,隐层传递函数采用 tansig,输出层传递函数采用 purelin,最大训练次数为 200 次。系统辨识的对比测试重复进行 50 次,表 3 是测试得到的系统辨识结果。

表3 系统辨识结果

参数	真值	ANN		QPSO		ACMPSO	
		平均值	标准差	平均值	标准差	平均值	标准差
a_1	-1.5	-1.4976	0.0105	-1.5008	0.0032	-1.5002	0.0014
a_2	0.7	0.6982	0.0086	0.7010	0.0045	0.6999	0.0028
b_0	1.0	1.0014	0.0073	1.0013	0.0035	1.0000	0.0007
b_1	0.5	0.4982	0.0122	0.4989	0.0021	0.4998	0.0010

从表 3 结果可以看出,三种算法均可得到真值的近似值,说明三种方法均可用于 Wiener 模型系统辨识。其中应用 ACMPSO 算法得到的估计值的平均值更加逼近真实值,且标准偏差均小于其他两种方法,说明该算法多次实验得到的估计值精确度最高。

图 6 为应用 ACMPSO 算法进行 Wiener 模型辨识时参数估计值随着进化代数的选择曲线。可以看出,在 150 个进化代后,参数$\{a_i\}$和$\{b_j\}$估计值基本上已经收敛于真值附近。

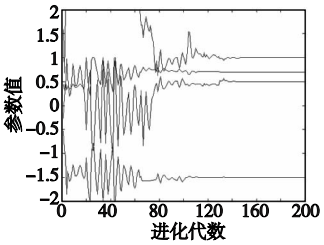

图6 参数进化曲线

4 结束语

本文提出了一种自适应云模型粒子群算法 ACMPSO,算法的寻优性能优于常见的 PSO 算法。将该算法应用于 Wiener 模型的辨识,相比人工神经网络及其他改进 PSO 算法等常用方法,该方法有更好的辨识效果,对目标参数的辨识较为精确。下一步研究的目标是将该算法应用于其他非线性系统模型的辨识,最终应用于实际的工业生产过程。 (下转第 4049 页)