DOI: 10.7641/CTA.2014.31114

自适应Tent混沌搜索的人工蜂群算法

匡芳君[1,2†], 徐蔚鸿[1,3], 金 忠[1]

(1. 南京理工大学 计算机科学与工程学院, 江苏 南京 210094; 2. 湖南安全技术职业学院 电气与信息工程系, 湖南 长沙 410151;
3. 长沙理工大学 计算机与通信工程学院, 湖南 长沙 410114)

摘要: 为了有效改善人工蜂群算法(artificial bee colony algorithm, ABC)的性能, 结合Tent混沌优化算法, 提出自适应Tent混沌搜索的人工蜂群算法. 该算法使用Tent混沌以改善ABC的收敛性能, 避免陷入局部最优解, 首先应用Tent映射初始化种群, 使得初始个体尽可能均匀分布, 其次自适应调整混沌搜索空间, 并以迄今为止搜索到的最优解产生Tent混沌序列, 从而获得最优解. 通过对六个复杂高维的基准函数寻优测试, 仿真结果表明, 该算法不仅加快了收敛速度, 提高了寻优精度, 与其他最近改进人工蜂群算法相比, 其性能整体较优, 尤其适合复杂的高维函数寻优.

关键词: 人工蜂群算法; 混沌理论; Tent映射; 自适应搜索; 锦标赛选择策略

中图分类号: TP18 **文献标识码**: A

Artificial bee colony algorithm based on self-adaptive Tent chaos search

KUANG Fang-jun[1,2†], XU Wei-hong[1,3], JIN Zhong[1]

(1. School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing Jiangsu 210094, China;
2. Department of Electronic and Information Engineering, Hunan Vocational Institute of Safety & Technology, Changsha Hunan 410151, China;
3. College of Computer and Communications Engineering, Changsha University of Science and Technology, Changsha Hunan 410114, China)

Abstract: In order to improve the performance of artificial bee colony (ABC) algorithm, a novel ABC algorithm based on self-adaptive Tent chaos search which is combined with Tent chaos algorithm is proposed. The algorithm uses Tent Chaos mapping to improve the convergence characteristics and prevent the ABC to get stuck on local solutions. In this algorithm, Tent mapping is applied to diversify the initial individuals in the search space. Tent chaotic sequence based an optimal location is produced, and the self-adaptive adjustment of chaos search scopes can obtain the global optima. Experiments on six complex benchmark functions with high-dimension, simulation results further demonstrate that, the improved algorithm not only accelerates the convergence rate and improves solution precision. Compared with other latest improved artificial colony algorithm, it has a better overall performance, especially for complex high-dimensional functions optimization.

Key words: artificial bee colony; chaos theory; tent mapping; self-adapting search; tournament selection strategy

1 引言(Introduction)

人工蜂群算法(artificial bee colony algorithm, ABC)是2005年由Karaboga提出的一种群智能优化算法[1]. 它具有控制参数少、全局寻优能力强、收敛速度快、鲁棒性强等优点[2], 已被越来越多的学者关注. 如: Karaboga等用ABC训练神经网络[3]、聚类分析[4–5]和解决约束优化问题[6]; Akay等[7]提出了改进人工蜂群算法解决实参优化问题; 暴励等[8]提出了双种群差分蜂群算法等. 与其他智能优化算法进行性能比较, 研究表明ABC具有良好的性能[9].

为改善ABC算法在接近全局最优时, 搜索速度变慢, 种群的多样性也减少, 甚至易于陷入局部最优等缺陷, 很多学者利用混沌的遍历性、随机性等特性, 将混沌映射和混沌搜索引入人工蜂群中. 如: 罗钧等[10]提出了基于Logistic混沌搜索的蜂群优化算法, 以提高解的多样性和搜索的遍历性; 暴励等[11]提出了自适应搜索空间的混沌蜂群算法, 利用Logistic混沌搜索对搜索停滞的解产生混沌序列, 以提高算法收敛速度和精度; Alatas[12]提出混沌人工蜂群算法, 利用混沌搜索自适应调整以提高ABC算法的收敛性; Gao等[13]

收稿日期: 2013−10−25; 录用日期: 2014−08−22.

†通信作者. E-mail: kfjztb@126.com; Tel.: +86 18073101198.

基金项目: 国家自然科学基金资助项目(61373063, 61233011, 61125305); 湖南省科技计划项目(2013FJ4217); 湖南省教育厅资助科研项目(13C086).

利用Logistic映射和反向学习方法改进ABC算法搜索等式; Liao等[14]利用自适应混沌人工蜂群算法解决短期水火电力系统调度问题; Xu等[15]利用混沌人工蜂群算法解决无人驾驶战斗机的路径规划问题等. 但这些算法的混沌搜索大部分是基于Logistic映射, 寻优速度受Logistic遍历不均匀的影响. 因此, 本文考虑到Tent映射具有遍历均匀性和迭代速度快的优势, 提出了自适应Tent混沌搜索的人工蜂群算法(selfadaptive Tent chaos search artificial bee colony algorithm, SATC-ABC). 该算法应用Tent映射初始化种群, 使初始个体尽可能分布均匀, 并自适应调整混沌搜索空间, 最后以迄今为止搜索到的最优位置产生Tent混沌序列. 实验仿真结果表明, 该算法有利于提高粒子搜索的遍历性和种群的多样性, 能提升ABC算法的收敛速度, 提高寻优精度和后期跳出局部最优的能力, 在收敛速度与寻优精度方面比较新的ABC改进算法有明显改善.

2 ABC算法(ABC algorithm)

在ABC算法中蜂群分为引领蜂(employed bee)、跟随蜂(onlooked bee)和侦察蜂(scout bee). 蜜源的初始位置代表优化问题的可行解通过式(1)随机产生, 且被分派给引领峰. 蜜源的丰富程度表示可行解的质量或适应度, 最优适应度的蜜源表示问题最优解. 蜜源数量等于引领蜂或跟随蜂的数量, 占蜂群总数的一半.

$$X_i^j(0) = X_{\min}^j + R \times (X_{\max}^j - X_{\min}^j), \tag{1}$$

其中: $X_i^j(0)$是初始化时第i个蜜源的第j维向量; $X_{\max}^j$, $X_{\min}^j$分别为第j维向量的上界和下界; R为$[0,1]$间的随机数.

所有蜜源的适应度通过式(2)计算:

$$F_i(t) = \begin{cases} \dfrac{1}{1+f_i(t)}, & \text{如果} f_i(t) \geqslant 0, \\ 1 + \mathrm{abs}(f_i(t)), & \text{其他}, \end{cases} \tag{2}$$

其中$F_i(t)$为t时刻第i个蜜源的适应度值, $f_i(t)$为具体优化问题的目标函数值.

跟随蜂根据蜜源的适应度选择蜜源的概率P_i为

$$P_i(t) = \frac{F_i(t)}{\sum\limits_{n=1}^{N} F_n(t)}. \tag{3}$$

为了能根据旧的蜜源位置X_i^j产生新的蜜源位置V_i^j, ABC算法采用如下表达式:

$$V_i^j(t) = X_i^j(t) + \Phi(X_i^j(t) - X_k^j(t)), \tag{4}$$

其中Φ是[-1,1]之间的随机数; $k = 1, 2, \cdots, N$是随机选择的下标, 且$k \neq i$. 以上各式中$i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, D$, N是蜜源或引领蜂的数量, D是优化参数的个数或问题空间的维数.

另外, 如果一个蜜源位置经过限定次数Limit循环后仍不能被改进, 那么该蜜源处的引领蜂成为侦察蜂, 则由该侦察蜂按式(1)在解空间中随机产生新蜜源位置并替换.

3 自适应Tent混沌搜索的人工蜂群算法(SATC-ABC algorithm)

3.1 Tent混沌序列(Tent chaos sequence)

利用混沌变量的随机性、遍历性和规律性特征进行优化搜索, 使算法跳出局部最优, 保持群体多样性, 并使全局搜索能力得到改善, 但不同的混沌映射对混沌优化过程的影响很大. 很多学者将混沌映射及混沌搜索思想引入到人工蜂群算法中, 提出了基于Logistic混沌的ABC算法(chaotic ABC, CABC), 改善了其后期易陷入局部最优的能力, 提高了算法的收敛速度和精度[9–14], 目前文献中引用较多的混沌映射是Logistic, 但它在$[0, 0.1]$和$[0.9, 1]$两个范围的取值概率较高, 且寻优速度受Logistic遍历不均匀性的影响, 算法效率会降低. 单梁等[16]指出Tent映射比Logistic映射具有更好的遍历均匀性和更快的收敛速度, 并通过严格的数学推理, 证明了Tent映射可以作为产生优化算法的混沌序列. Tent映射表达式如下:

$$x_{t+1} = \begin{cases} 2x_t, & 0 \leqslant x_t \leqslant \dfrac{1}{2}, \\ 2(1-x_t), & \dfrac{1}{2} \leqslant x_t \leqslant 1. \end{cases} \tag{5}$$

Tent映射经贝努利移位变换后表示如下[16]:

$$x_{t+1} = (2x_t) \mod 1, \tag{6}$$

根据Tent映射的特性, 在可行域中产生Tent混沌序列的步骤描述如下:

步骤 1 随机产生初值x_0(避免x_0落入小周期内$(0.2, 0.4, 0.6, 0.8)$), 记为标志组z, $z(1) = x_0$, $i = j = 1$.

步骤 2 按式(6)迭代, 迭代每次自增1, 产生一个x序列.

步骤 3 若达到最大迭代次数, 则转向步骤5; 否则, 若$x(i) = \{0, 0.25, 0.5, 0, 75\}$或$x(i) = x(i-k)$, $k = \{0, 1, 2, 3, 4\}$, 则转向步骤2.

步骤 4 按式$x(i) = z(j+1) = z(j) + \varepsilon$改变迭代初值, $j = j + 1$, 转向步骤2.

步骤 5 终止运行, 保存产生的x序列.

3.2 自适应Tent混沌搜索(Self-adaptive Tent chaos search)

混沌人工蜂群算法(CABC)的混沌搜索是以当前整个蜂群迄今为止搜索到的最优蜜源为基础产生混

沌序列, 因此每次搜索的范围比较大, 很难搜索到更优蜜源, 而且其采用Logistic映射, 寻优速度受Logistic遍历不均匀的影响, 算法效率会降低. 本文利用Tent映射比Logistic映射具有更快的收敛速度和更好的遍历均匀性, 提出了自适应Tent混沌搜索人工蜂群算法(SATC-ABC). 其基本思想主要体现如下:

1) 采用Tent混沌序列初始化种群.

采用Tent混沌产生分布均匀的初始种群, 既能保留初始化个体的随机性, 又能提高种群的多样性. 在引领蜂放弃蜜源成为侦察蜂时, 利用Tent混沌搜索产生新解, 以提高算法的收敛速度.

2) 采用自适应动态调整混沌搜索空间.

对当前进化第k代的所有蜜源按适应度从大到小排列, 取前面n个(占总引领蜂群数的80%)引领蜂, 分别求出这n只引领蜂第j维的最小值$X^j_{\min}$和最大值$X^j_{\max}$作为混沌搜索空间. 以迄今为止搜索到的最优解为基础产生Tent混沌序列, 并将序列中的最优解作为新蜜源位置, 使其跳出局部最优.

本文假设搜索停滞的解是$X_k=(x_{k1},\cdots,x_{kD})$, $x_{kj}\in[X^j_{\min},X^j_{\max}]$, Tent混沌搜索的主要步骤如下:

步骤 1　利用$z^0_{kj}=(x_{kj}-X^j_{\min})/(X^j_{\max}-X^j_{\min})$将$X_k$映射到$(0,1)$, 其中$k=1,\cdots,n$, $j=1,\cdots,D$.

步骤 2　将上式代入到式(6)Tent映射进行迭代产生混沌变量序列$z^m_{kj}(m=1,2,\cdots,C_{\max})$, $C_{\max}$是混沌搜索的最大迭代次数.

步骤 3　利用式(7)将z^m_{kj}载波到原解空间的邻域内产生新解V_k.

$$v_{kj}=x_{kj}+\frac{(X^j_{\max}-X^j_{\min})}{2}\times(2z^m_{kj}-1),\quad(7)$$

步骤 4　计算V_k的适应度值$F(V_k)$, 并与原来解的适应度值$F(X_k)$比较, 保留最好解.

步骤 5　判断是否达到最大混沌搜索次数, 若达到, 则混沌搜索结束, 否则转向步骤2.

3.3 锦标赛选择策略(Tournament selection strategy)

基本ABC算法的跟随蜂采用比例选择策略选择蜜源, 使其在进化初期收敛速度较快, 但在算法后期算法易陷入局部最优.锦标赛选择策略[17]是基于局部竞争机制, 即从种群中随机选取q个个体进行比较, 适应度大的个体作为最优个体. 本文算法跟随蜂采用锦标赛选择策略选择蜜源, 且取$q=2$, 对适应度大的个体加1分, 对所有个体重复这一过程, 最后得分最高者其权重也最大. 锦标赛选择策略只将适应度值的相对值作为选择标准, 对适应度值的正负未做要求, 从而避免了超级个体对算法的影响. 适应度的选择概率为

$$P_i(t)=\frac{c_i(t)}{\sum\limits_{i=1}^{N}c_i(t)},\quad(8)$$

其中: c_i为每个个体的得分.

3.4 自适应Tent混沌搜索的人工蜂群算法流程(Main steps of SATC-ABC)

SATC-ABC算法引入Tent混沌搜索和锦标赛选择策略, 避免算法早熟收敛和陷入局部最优值, 其算法流程如下:

步骤 1　设置相关参数: 种群规模(初始可行解的个数)SN、蜜源个数$N=(SN/2)$, 蜜源维数D、最大进化次数T或求解精度ε、同一蜜源被限定的采蜜次数Limit和混沌最大迭代次数$C_{\max}$.

步骤 2　设置初始迭代次数$\mathrm{iter}=0$, 在搜索区域内利用Tent混沌序列初始化种群生成D维SN个向量X_i, D为目标函数中的维数.

步骤 2.1　随机产生D维每个分量数值在$(0,1)$之间的向量$Z^D_i(t)$.

步骤 2.2　根据3.1节的Tent混沌映射步骤, 可得到SN个不同轨迹的混沌序列$Z^D_i(t)$.

步骤 2.3　将$Z^D_i(t)$各个分量通过式(9)载波到对应变量的取值范围内, 即

$$X^j_i(t)=X^j_{\min}+(X^j_{\max}-X^j_{\min})z^j_i(t),\quad(9)$$

其中: $i=1,2,\cdots,SN$; $j=1,2,\cdots,D$.

步骤 3　计算X_i的适应度函数值$F(X_i)$, 适应度值大的前$N=(SN/2)$个向量作为蜜源位置, 对应N个引领蜂, 初始标志向量$\mathrm{trial}(i)=0$, 记录引领蜂停留同一蜜源的循环次数.

步骤 4　每只引领蜂i按照式(4)在附近蜜源搜索产生新解V_i, 并计算适应度值$F(V_i)$.

步骤 5　如果$F(V_i)>F(X_i)$, 则$X_i=V_i$, $\mathrm{trial}(i)=0$; 否则保持原解X_i不变, $\mathrm{trial}(i)=\mathrm{trial}(i)+1$.

步骤 6　利用锦标赛选择策略, 根据式(8)计算选择概率$P_i(t)$.

步骤 7　跟随蜂根据概率$P_i(t)$选择蜜源, 由式(4)进行邻域搜索产生新解V_i, 计算适应度值$F(V_i)$.

步骤 8　按步骤5执行.

步骤 9　若$\mathrm{trial}(i)>\mathrm{Limit}$, 则第$i$个引领蜂放弃当前蜜源而成为侦察蜂, 侦察蜂根据3.2节的自适应Tent混沌搜索产生一个新解V_i替代蜜源.

步骤 10　记录当前所有蜜蜂找到的最优蜜源, 保存迄今为止的最优解.

步骤 11 更新iter = iter + 1, 判断是否达到最大迭代次数T或满足求解精度ε要求, 如满足, 则输出最优解, 否则返回步骤4.

4 仿真实验与结果分析(Experiment of simulation and the analysis of results)

4.1 基准测试函数(The benchmark functions)

本文选取6个Benchmark基准函数对算法进行寻优测试. 这些函数具有不同特性, 可充分考察算法的寻优能力, 如表1所示. 其中: Sphere是单峰函数, 主要测试算法寻优精度和收敛速度; Rosenbrock是非凸、病态单峰函数, 有局部极小值, 主要测试算法的收敛速度和执行效率; Rastrigin, Griewank, Ackley和Schwefel2.26是复杂的非线性多峰函数, 有许多局部极值点, 主要测试算法的全局搜索能力、跳出局部极值并避免早熟的收敛能力.

表 1 6个标准测试函数

Table 1 Six benchmark functions

函数名	表达式及取值范围	全局最小值	特性
Sphere	$f(X)=\sum_{i=1}^{D} x_i^2,\ x_i\in[-100,100]$	$x_i=0, f(X)=0$	单峰
Rosenbrock	$f(X)=\sum_{i=1}^{D-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2), x_i\in[-30,30]$	$x_i=1, f(X)=0$	单峰
Rastrigin	$f(X)=\sum_{i=1}^{D}(100(x_i^2-10\cos(2\pi x_i)+10)), x_i\in[-5.12,5.12]$	$x_i=0, f(X)=0$	多峰
Griewank	$f(X)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1, x_i\in[-600,600]$	$x_i=0, f(X)=0$	多峰
Ackley	$f(X)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{D}\cos(2\pi x_i)+20+\mathrm{e}),\ x_i\in[-32,32]$	$x_i=0, f(X)=0$	多峰
Schwefel	$f(X)=-\sum_{i=1}^{D}(x_i\sin(\sqrt{\lvert x_i\rvert})), x_i\in[-500,500]$	$x_i=420.9687, f(X)=-418.9829D$	多峰

4.2 SATC-ABC算法的性能分析(Performance analysis of SATC-ABC)

为衡量本文算法性能的有效性, 分别采用基本ABC, CABC(基于Logistic混沌映射的ABC)和SATC-ABC算法进行寻优测试.在实验中, 所有算法的种群规模为100; 蜜源个数为50; 测试函数分别取50维和100维时, 最大迭代次数分别取3000和5000; Limit= 100; $X_{\max}^j$和$X_{\min}^j$分别为函数输入向量第j维的上界和下界. ABC和CABC算法中跟随蜂采用比例选择策略选择蜜源, SATC-ABC算法中跟随蜂则采用锦标赛选择策略选择蜜源; 对于CABC和SATC-ABC 算法, 混沌搜索次数$C\max = 30$. 算法在Matlab7.10平台实现, 通过对每个测试函数运行30次所得适应度的最优值、最差值、均值和标准方差来考察算法的性能, 50维的平均适应度值的进化过程曲线如图1−6所示. 50维和100维的测试结果比较如表2所示.

由表2和图1−6可看出, 所有算法对大部分函数都能达到寻优精度, 说明人工蜂群算法具有较强的全局搜索能力. 对于单峰函数Sphere, SATC-ABC的寻优效果和收敛速度最好, 能够持续、有效搜索函数全局最小值. 对于非凸、病态单峰函数Rosenbrock而言, 由于在其取值范围内走势平坦, 要收敛到全局最优点的机会很少, 但SATC-ABC算法依然优于其他两种算法, 且鲁棒性也较好. 对多峰函数Rastrigin, Griewank和Ackley, SATC-ABC的求解精度和收敛速度均高于其他算法, 主要因为SATC-ABC采用自适应Tent混沌搜索, 能动态调整混沌搜索空间, 既保证种群多样性, 提高寻优精度和收敛速度, 又使其尽可能跳出局部最优. 而多峰函数Schwefel的取值范围为$[500,500]$, 该函数的最优值位于搜索空间边界附近, 即维数是50时, 函数在点$(420.9687,\cdots,420.9687)$处取得理论最优值$-20949.145$, 而SATC-ABC算法的方差明显优于ABC算法, 且得到接近理论最优值-20949.144, 即SATC-ABC算法对于最优解位于搜索空间边缘时, 也能较好获得最优解. 说明SATC-ABC算法不仅收敛速度较快, 而且全局搜索能力较强, 在收敛速度和寻优精度方面比其他两种算法有明显提高.

为进一步验证SATC-ABC算法的有效性, 与最近改进算法IABC[18], HABC[19], LRABC[20]以及ABC算法和CABC算法对上述函数30维进行寻优测试. 通过对参数Limit, 从$\mathrm{Limit}=0.1*SN*D$到$\mathrm{Limit}=SN*D$之间反复测试SATC-ABC算法的

性能, 发现当Limit $= (SN * D)/2$ 时, SATC-ABC算法的性能整体较优. 因此, 在本实验中, 所有算法种群规模为$SN = 100$, Limit $= (SN * D)/2$, 最大迭代次数$G_{\max} = (D * 10000)/SN$; ABC, CABC和SATC-ABC算法的跟随蜂采用锦标赛策略选择蜜源; 对于涉及混沌搜索的算法, 其混沌搜索次数为$C_{\max} = 30$. 通过对每个测试函数运行30次, 当求解精度达到10^{-20}就假定结果为0, 各算法适应度的均值(mean)和标准方差(std)以及算法的平均运行时间(avgtime)如表3所示.

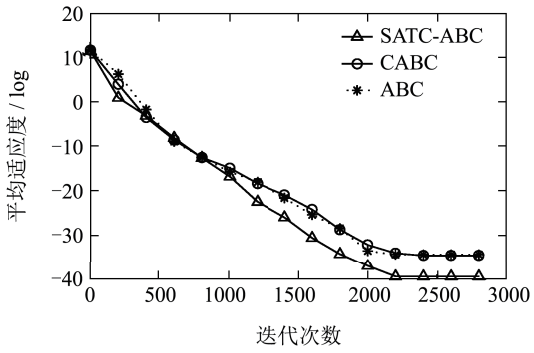

图 1 Sphere函数动态寻优过程(50维)

Fig. 1 The progress curve of Sphere Function (50 dimensions)

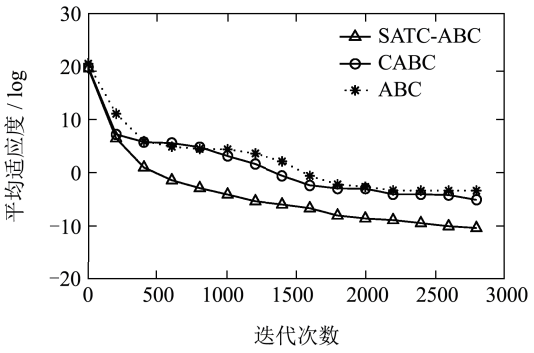

图 2 Rosenbrock函数动态寻优过程(50维)

Fig. 2 The progress curve of Rosenbrock Function (50 dimensions)

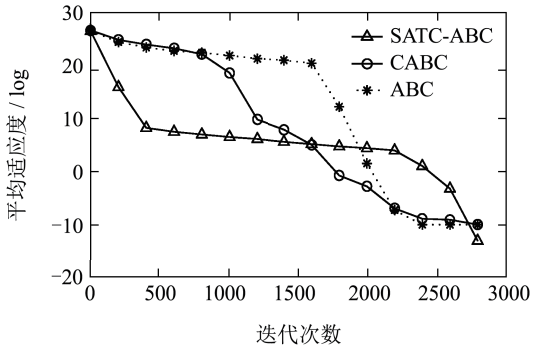

图 3 Rastrigin函数动态寻优过程(50维)

Fig. 3 The progress curve of Rosenbrock Function (50 dimensions)

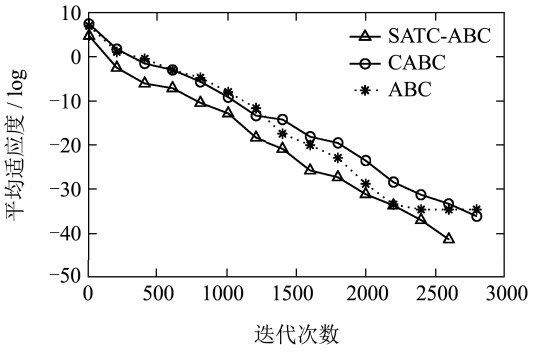

图 4 Griewank函数动态寻优过程(50维)

Fig. 4 The progress curve of Griewank Function (50 dimensions)

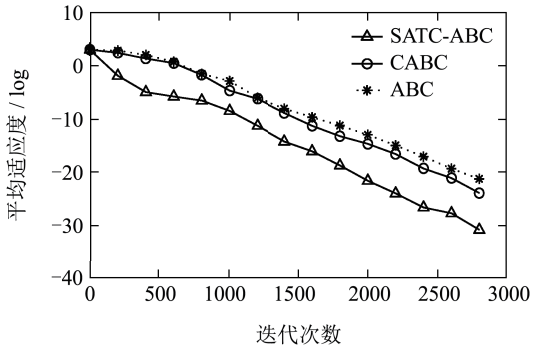

图 5 Ackley函数动态寻优过程(50维)

Fig. 5 The progress curve of Ackley Function (50 dimensions)

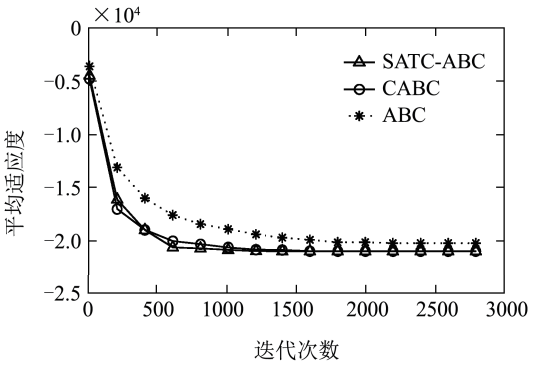

图 6 Schwefel函数动态寻优过程(50维)

Fig. 6 The progress curve of Schwefel Function (50 dimensions)

从表3可以看出, 对于函数sphere和Rastrigin, IABC, HABC, LRABC和SATC-ABC算法都达到了最优值; 对于函数Griewank, IABC, HABC和LRABC算法对函数的均值和方差略优于SATC-ABC算法; 对于其他函数, SATC-ABC算法的均值和标准方差都要优于其他算法, 特别是对于函数Schwefel, SATC-ABC算法不仅获得了理论最优值, 而且标准方差也最优, 这表明SATC-ABC算法整体来说具有较好的稳定性和鲁棒性.

另外, 除LRABC算法对函数Griewank的执行时间稍小于SATC-ABC算法外, SATC-ABC算法的执行时间相对而言用时稍短, 主要原因是SATC-ABC算法引入的Tent映射是通过贝努利移位变换得到, 即将Tent小数部分的二进制数进行无符号左移, 它能充分地利用计算机的特性, 能更快更有效地处理大数量级数据序列的运算.

综上分析, SATC-ABC不仅具有较强的全局搜索能力, 而且具有较强的局部寻优能力, 在收敛速度和求解精度上均有明显的提高, 并随着维数增加, 也能保持较好的有效性和鲁棒性.

表 2 基准函数的优化结果比较

Table 2 Optimization results comparison of benchmark functions

函数	维数	算法	最优值	最差值	均值	标准方差
Sphere	50	ABC	1.16921E−015	2.30472E−015	1.59341E−015	2.43195E−016
		CABC	9.68606E−016	1.86614E−015	1.50491E−015	2.09925E−016
		SATC-ABC	**7.64861E−018**	**1.17095E−017**	**9.73869E−018**	**7.48267E−019**
	100	ABC	3.13080E−015	9.42898E−015	5.23869E−015	1.45330E−015
		CABC	3.17306E−016	6.50622E−016	4.82773E−016	7.81467E−017
		SATC-ABC	**2.08129E−018**	**2.89175E−017**	**2.54345E−017**	**1.87021E−018**
Rosenbrock	50	ABC	3.48953E−002	2.44179E+000	4.98511E−001	5.92527E−001
		CABC	4.83917E−003	7.16931E−001	1.69646E−001	2.00503E−001
		SATC-ABC	**1.06604E−005**	**2.71035E−004**	**6.53865E−005**	**5.71185E−005**
	100	ABC	6.84431E−002	2.18751E+000	6.09265E−001	5.25902E−001
		CABC	2.47457E−002	4.31986E+000	8.49603E−001	1.01245E+000
		SATC-ABC	**7.00425E−004**	**1.65020E−002**	**3.08147E−003**	**3.53669E−003**
Rastrigin	50	ABC	1.13687E− 013	9.48717E−011	7.40859E−012	2.04839E−011
		CABC	**0**	2.27374E− 013	7.57912E−014	4.56052E−014
		SATC-ABC	**0**	**4.04480E−015**	**1.89802E−016**	**7.44311E−016**
	100	ABC	1.13687E−012	9.65354E−004	3.78494E−005	1.75991E−004
		CABC	1.47793E−012	4.48139E−010	1.62183E−011	8.16551E−010
		SATC-ABC	**4.54747E−015**	**9.95129E−011**	**3.39372E−014**	**1.81566E−011**
Griewank	50	ABC	9.99201E− 016	8.53762E−014	1.19978E−014	2.07400E−014
		CABC	**0**	**5.55112E−016**	1.25825E−016	**1.22778E−016**
		SATC-ABC	**0**	6.96476E−016	**3.15714E−017**	1.29540E−016
	100	ABC	3.21965E−015	1.70675E−012	1.71522E−013	3.67265E−013
		CABC	1.11022E−016	4.62963E−014	2.76446E−015	8.24211E−015
		SATC-ABC	**2.77556E−019**	**6.13398E−016**	**5.90565E−017**	**1.19541E−016**
Ackley	50	ABC	3.00249E−011	1.94881E−010	7.55055E−011	3.34588E−011
		CABC	2.29594E−012	1.24105E−011	7.04130E−012	2.66576E−012
		SATC-ABC	**4.83258E−015**	**1.59135E−014**	**1.00407E−014**	**2.82226E−015**
	100	ABC	1.47565E−008	8.05324E−008	3.38329E−008	1.35035E−008
		CABC	1.73741E−009	6.13972E−009	3.56879E−009	1.13386E−009
		SATC-ABC	**1.52748E−012**	**8.01208E−012**	**3.93873E−012**	**1.40618E−012**
Schwefel	50	ABC	−2.02027E+004	−2.07123E+004	−2.08738E+004	7.28559E+001
		CABC	**−2.09491E+004**	**−2.09491E+004**	**−2.09491E+004**	1.88551E−007
		SATC-ABC	**−2.09491E+004**	**−2.09487E+004**	**−2.09491E+004**	**9.90947E−008**
	100	ABC	−4.15385E+004	−4.09245E+004	−4.12141E+004	1.52428E+002
		CABC	**−4.18983E+004**	**−4.15427E+004**	**−4.17359E+004**	7.71880E+001
		SATC-ABC	**−4.18983E+004**	−4.16606E+004	−4.18912E+004	**7.89936E−001**

表 3 算法对标准函数在30维的优化结果比较

Table 3 Optimization results comparison of 30 dimensions benchmark functions

函数		算法					
		ABC	**CABC**	**IABC**	**HABC**	**LRABC**	**SATC-ABC**
Sphere	Mean	4.76561E−016	4.74064E−016	**0**	**0**	**0**	**0**
	Std	8.35788E−017	6.41359E−017	**0**	**0**	**0**	**0**
	AvgTime/s	19.3311	23.4615	23.9839	23.8946	20.3872	**18.7634**
Rosenbrock	Mean	2.80954E−002	4.91687E−001	1.56479E+002	2.81635E+001	1.36418E−001	**1.54578E−002**
	Std	5.04724E−002	7.66717E−001	8.19357E+001	5.44279E−001	8.35317E−002	**1.28435 E−002**
	AvgTime/s	23.6253	24.1755	24.0182	26.2538	23.1843	**19.0072**
Rastrigin	Mean	**0**	**0**	**0**	**0**	**0**	**0**
	Std	**0**	**0**	**0**	**0**	**0**	**0**
	AvgTime/s	25.5212	22.6794	24.2919	26.8853	20.5127	**19.0087**
Griewank	Mean	6.29126E−017	4.44089E−017	**0**	**0**	**0**	5.18104E−019
	Std	9.07274E−017	5.53194E−017	**0**	**0**	**0**	5.63345E−019
	AvgTime/s	32.3926	29.4093	30.0128	29.7429	**26.8934**	28.3381
Ackley	Mean	3.32179E−014	3.33363E−014	3.87532E−014	8.45531E−016	9.64575E−015	**2.93099E−016**
	Std	3.00156E−015	3.45752E−015	2.53489E−015	5.53722E−017	8.32527E−016	**4.20708E−017**
	AvgTime/s	30.8024	24.4604	27.7914	26.9675	23.8491	**22.3862**
Schwefel	Mean	**−1.25695E+004**	**−1.25695E+004**	**−1.25695E+004**	**−1.25695E+004**	**−1.25695E+004**	**−1.25695E+004**
	Std	2.74412E−012	2.18905E−012	3.95626E−012	3.64387E−011	1.96538E−011	**6.94721E−016**
	AvgTime/s	28.9028	23.7075	26.9823	22.7092	26.6732	**22.6274**

5 结束语(Conclusions)

针对ABC算法特性, 结合Tent混沌优化思想和自适应调整搜索空间, 提出了一种自适应Tent混沌搜索的人工蜂群算法(SATC-ABC). 该算法主要有如下特点: 1) 引入Tent混沌序列初始化种群, 提高种群多样性; 2) 引入自适应Tent混沌映射在迭代中产生局部最优解的邻域点, 使算法不仅能跳出局部最优, 提高种群多样性和搜索的遍历性, 还能加快算法收敛速度, 快速搜寻最优解; 3) 跟随蜂采用锦标赛选择策略选择蜜源, 在一定程度上避免算法早熟收敛和超级个体对算法的影响; 4) SATC-ABC对易陷入局部最优或有局部极小的非线性函数也具有较高的寻优精度和各类测试函数都具有较好的稳定性和鲁棒性, 确保整个群体的全局优化能力.

通过对六个标准函数的寻优测试, 实验结果表明, 该算法在保持种群多样性的同时, 避免了早熟收敛, 对于高维、多极值点的非线性函数, 具有良好的全局搜索能力和寻优精度, 且算法相对稳定性, 鲁棒性较好. 下一步工作考虑将SATC-ABC算法应用到求解复杂组合优化问题和具体应用, 并结合其他各种智能优化算法, 提出性能更好的全局优化算法.

参考文献(References):

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

[2] AKAY B. *Performance analysis of artificial bee colony algorithm on numerical optimization problems* [D]. Kayseri: Dissertation in Turkish, Erciyes University, Graduate School of Natural and Applied Sciences, 2009.

[3] KARABOGA D, OZTURK C. Neural networks training by artificial bee colony algorithm on pattern classification [J]. *Neural Network World*, 2009, 19(3): 279 – 292.

[4] KARABOGA D, OZTURK C A. novel clustering approach: artificial bee colony (ABC) algorithm [J]. *Applied Soft Computing*, 2011, 11(1): 652 – 657.

[5] ZHANG C, OUYANG D, NING J. An artificial bee colony approach for clustering [J]. *Expert Systems with Applications*, 2010, 37(7): 4761 – 4767.

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

[7] AKAY B, KARABOGA D. A modified artificial bee colony algorithm for real-parameter optimization [J]. *Information Sciences*, 2012, 192(6): 120 – 142.

[8] 暴励, 曾建潮. 一种双种群差分蜂群算法 [J]. 控制理论与应用, 2011, 28(2): 266 – 272.
(BAO Li, ZENG Jianchao. A bi-group differential artificial bee colony algorithm [J]. *Control Theory and Applications*, 2011, 28(2): 266 – 272.)

[9] KARABOGA D, BASTURK B. A comparative study of artificial bee colony algorithm [J]. *Applied Mathematics and Computation*, 2009, 214(1): 108 – 132.

[10] 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
(LUO Jun, LI Yan. Artificial bee colony algorithm with chaotic-search strategy [J]. *Control and Decision*, 2010, 25(12): 1913 – 1916.)

[11] 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.
(BAO Li, ZENG Jianchao. Selfadapting search space chaos-artificial bee colony algorithm [J]. *Application Research of Computers*, 2010, 27(4): 1330 – 1334.)

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

[13] GAO W, LIU S. A modified artificial bee colony algorithm [J]. *Computers & Operations Research*, 2012, 39(3): 687 – 697.

[14] LIAO X, ZHOU J, OUYANG S, et al. An adaptive chaotic artificial bee colony algorithm for short-term hydrothermal generation scheduling [J]. *Electrical Power and Energy Systems*, 2013, 53(12): 34 – 42.

[15] XU C, DUAN H, LIU F. Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning [J]. *Aerospace Science and Technology*, 2010, 14(8): 535 – 541.

[16] 单梁, 强浩, 李军, 等. 基于Tent映射的混沌优化算法 [J]. 控制与决策, 2005, 20(2): 179 – 182.
(SHAN Liang, QIANG Hao, LI Jun, et al. Chaotic optimization algorithm based on Tent map [J]. *Control and Decision*, 2005, 20(2): 179 – 182.)

[17] BAO L, ZENG J. Comparison and analysis of the selection mechanism in the artificial bee colony algorithm [C] //*2009 9th International Conference on Hybrid Intelligent Systems*. Los Alamitos, CA: IEEE, 2009: 411 – 416.

[18] GAO W, LIU S. Improved artificial bee colony algorithm for global optimization [J]. *Information Processing Letters*, 2011, 111(17): 871 – 882.

[19] YAN X, ZHU Y, ZOU W, et al. A new approach for data clustering using hybrid artificial bee colony algorithm [J]. *Neurocomputing*, 2012, 97(11): 241 – 250.

[20] 刘三阳, 张平, 朱明敏. 基于局部搜索的人工蜂群算法 [J]. 控制与决策, 2014, 29(1): 123 – 128.
(LIU Sanyang, ZHANG Ping, ZHU Mingmin. Artificial bee colony algorithm based on local search [J]. *Control and Decision*, 2014, 29(1): 123 – 128.)

作者简介:

匡芳君 (1976—), 女, 副教授, 博士, CCF会员(E200014005G), 主要研究方向为模式识别、智能优化、信息安全等, E-mail: kfjztb@126.com;

徐蔚鸿 (1963—), 男, 教授, 博士, 博导, 主要研究方向为人工智能、模式识别、软件工程等, E-mail: xwhxdcs@126.com;

金 忠 (1961—), 男, 教授, 博士, 博导, 主要研究方向为模式识别、机器学习、计算机视觉等, E-mail: zhongjin@njust.edu.cn.