ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 38,No. 3,May 2023,pp. 690－703
DOI:10. 16337/j. 1004-9037. 2023. 03. 017

http://sjcj. nuaa. edu. cn
E-mail:sjcj@nuaa. edu. cn
Tel/Fax:＋86-025-84892742

基于正余弦优化算子和Levy飞行机制的和声搜索算法

程翠娜，奉松绿，莫礼平

（吉首大学计算机科学与工程学院，吉首 416000）

摘　要：针对基本和声搜索（Harmony search，HS）算法收敛速度较慢、易陷入局部最优和计算精度不高的缺点，结合正余弦优化算子、Levy飞行机制和参数动态调整策略，提出一种改进的和声搜索算法。该算法在即兴创作阶段，首先引入正余弦优化算子和微调带宽相结合的方式对和声向量进行微调操作，充分利用最优个体和当前个体的位置信息，提高算法的计算精度和收敛速度；再采用Levy飞行机制对微调带宽进行更新，避免算法陷入局部最优，提高全局搜索能力；在算法迭代过程中，对和声记忆库存储概率、基音微调概率和搜索域进行自适应动态调整，以进一步提高算法收敛性能。在10个基准函数上进行性能对比试验的结果表明，本文提出的算法具有较强的全局搜索能力，较快的收敛速度和较高的计算精度。

关键词：群智能优化算法；和声搜索算法；正余弦优化算子；Levy飞行机制

中图分类号：TP181　　**文献标志码：**A

Adaptive Harmony Search Algorithm Based on Sine Cosine Optimization Operator and Levy Flight Mechanism

CHENG Cuina，FENG Songlyu，MO Liping

（College of Computer Science and Engineering，Jishou University，Jishou 416000，China）

Abstract：Aiming at the shortcomings of slow convergence speed，easy to fall into local optimum and low convergence accuracy of basic harmony search（HS）algorithm，an improved HS（IHS）algorithm is proposed by combining sine cosine optimization operator，Levy flight mechanism and parameter dynamic adjustment strategy. In the improvisation stage，the algorithm first introduces a combination of sine cosine optimization operator and fine-tuning bandwidth to fine-tune the harmony vectors，makes full use of the position information of the optimal individual and the current individual，and improves the calculation accuracy and convergence speed of the algorithm.The Levy flight mechanism is then used to update the fine-tuned bandwidth to avoid the algorithm falling into local optimization and improve the global search capability. During the algorithm iteration process，adaptive dynamic adjustments are made to the storage probability，base tone fine-tuning probability and search domain of the harmony memory to further improve the convergence performance of the algorithm. The results of the performance test comparison experiment on ten reference functions show that the proposed algorithm has the stronger global search ability，the faster convergence speed and the better calculation accuracy.

基金项目：国家自然科学基金(62266019)；吉首大学校级科研创新项目(JGY2022070)；吉首大学校级科研创新项目(Jdy22027)。
收稿日期：2022-08-30；**修订日期：**2022-12-30

Key words: swarm intelligent optimization algorithm; harmony search algorithm; sine cosine algorithm; Levy flight mechanism

引 言

和声搜索(Harmony search, HS)算法是韩国学者Geem等[1]于2001年提出的一种元启发式群智能优化算法。与经典的遗传算法[2]、粒子群优化算法[3]和蚁群算法[4]等群智能优化算法相比,HS算法具有对问题依赖性小、易于实现、易与其他算法结合等优点。目前,HS算法已经成功应用到车辆调度[5]、目标检测[6]和缓冲区分配问题[7]等实际优化问题求解过程。但基本HS算法在处理复杂问题时还存在收敛速度慢、易陷入局部最优、计算精度不够等问题。针对HS算法存在的上述不足,国内外学者基于参数动态调整和算法混合等思路提出了多种改进的HS算法。针对前者,Mahdavi等[8]以提高算法的收敛速度为目标,提出了一种对HS算法参数进行动态调整的改进和声搜索(Improved harmony search, IHS)算法;Mahdavi等[9]通过调整HS算法的搜索机制来提高全局搜索能力,提出了一种全局最优和声搜索(Global-best harmony search, GHS)算法;Khalili等[10]将HS算法中涉及到的参数和搜索域进行自适应调整,提出了一种全局动态和声搜索(Global dynamic harmony search, GDHS)算法。针对后者,Chakraborty等[11]将差分进化(Differential evolution, DE)算法中的变异算子[12]引入到HS算法中,提出了一种改进的基于差分变异算子的和声搜索算法;Taherinejad等[13]使用模拟退火(Simulated annealing, SA)算法的冷却策略[14]来改进HS算法,提出了一种性能高度可靠的改进HS算法;Zhu等[15]在利用差分进化策略微调参数的同时,采用自适应调整机制来更新搜索域,提出了一种具有线性动态域的基于差分进化的改进HS算法(Improved differential-based harmony search algorithm with linear dynamic domain, ID-HS-LDD)。实验结果表明,将上述改进HS算法应用于解决函数优化问题和复杂问题时,在收敛速度和计算精度等方面的性能均优于基本HS算法。

本文尝试将参数动态调整和算法混合两种思路相结合,通过引入正余弦优化算子、Levy飞行策略以及重要参数和搜索域的自适应动态调整等方法来对HS算法进行改进研究,提出一种结合正余弦优化算子和Levy飞行策略的自适应HS算法(Adaptive harmony search algorithm combining sine cosine optimization operator and levy flight strategy, AHS-SC-LF),并使用10个基准测试函数对AHS-SC-LF同基本HS、IHS[8]、GDHS[10]、ID-HS-LDD[15]4种算法进行性能对比试验,以验证AHS-SC-LF的算法性能。

1 基本HS算法

基本HS算法通过模拟乐队即兴创作时反复调整各乐器的音调以达到一个最美和声状态的过程实现对优化问题最优解的寻找过程。算法涉及到的重要参数如表1所示。

表1 参数名称及含义

Table 1 Parameter and corresponding meaning

参数名称	含义
HMS	和声记忆库的大小
HMCR	和声记忆库存储概率
PAR	基音微调概率
BW	微调带宽
$T_{\max}$	最大迭代次数
U	搜索域的上边界
L	搜索域的下边界

基本HS算法以初始化和声记忆库、即兴创作和更新和声库3个操作为核心,算法步骤可描述如下:

步骤1:定义优化问题,并初始化参数。假设求目标函数的最小值问题,即$\min f(x)$, $x=\{x_1,x_2,x_3,\cdots,x_n\}\in\mathbf{R}^n$,其中$x_1,x_2,x_3,\cdots,x_n$为目

标函数的解;初始化表1所示的重要参数。

步骤2:初始化和声记忆库。利用式(1)在搜索域内随机生成HMS个和声,并将其加入到式(2)所示的矩阵表示的和声记忆库HM中。

$$X_{i,j}=L+(U-L)\times \text{rand} \tag{1}$$

$$\text{HM}=\begin{bmatrix} X^1 & f(X^1) \\ X^2 & f(X^2) \\ \vdots & \vdots \\ X^{\text{HMS}} & f(X^{\text{HMS}}) \end{bmatrix}=\begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_n^1 & f(X^1) \\ x_1^2 & x_2^2 & \cdots & x_n^2 & f(X^2) \\ \vdots & \vdots & & \vdots & \vdots \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \cdots & x_n^{\text{HMS}} & f(X^{\text{HMS}}) \end{bmatrix} \tag{2}$$

式中:rand为(0,1)之间的随机数;$f(x)$为适应度函数。

步骤3:即兴创作。生成一个(0,1)之间的随机数r_1,如果$r_1\geqslant$HMCR,在解空间中随机生成一个和声向量$\boldsymbol{X}^{\text{new}}$;反之,生成一个(0,1)之间的随机数$r_2$,如果$r_2<$PAR,则利用式(3)对得到的和声向量进行微调,以生成新和声向量$\boldsymbol{X}^{\text{new}}$。

$$X_{\text{new},j}=X_{\text{new},j}\pm \text{rand}\times \text{BW} \tag{3}$$

步骤4:更新和声记忆库。将即兴创作过程生成的$\boldsymbol{X}^{\text{new}}$和初始HM中的最差和声向量$\boldsymbol{X}^{\text{worst}}$的适应度函数进行比较,如果$f\left(\boldsymbol{X}^{\text{new}}\right)<f\left(\boldsymbol{X}^{\text{worst}}\right)$,则用$\boldsymbol{X}^{\text{new}}$替换$\boldsymbol{X}^{\text{worst}}$;否则,HM中的和声保持不变。

步骤5:检查是否达到算法终止条件。若达到,则输出全局最优解,算法结束;否则,转步骤3。

2 AHS-SC-LF改进策略

本文提出的AHS-SC-LF主要从3个方面对基本HS算法进行改进:(1)采用引入正余弦优化算子和微调带宽相结合的方式对即兴创作阶段的和声向量进行微调,以提高算法的收敛速度和计算精度;(2)采用Levy飞行机制对BW进行更新,以避免算法陷入局部最优;(3)对HMCR和PAR以及搜索域进行自适应动态调整,以提高算法的全局搜索能力。

2.1 正余弦优化算子

正余弦优化算法(Sine cosine algorithm, SCA)是澳大利亚学者Mirjalili[16]于2016年提出的一种利用正弦函数和余弦函数的震荡特性来实现问题解的寻优操作的算法。

SCA求解优化问题时,首先在搜索域中产生由个体组成的一个候选解种群,通过计算个体的适应度值确定最优个体,并利用式(4)进行所有个体位置的更新操作。

$$X_i^{t+1}=\begin{cases} X_i^t+r_1\cdot\sin(r_2)\cdot\left|r_3\cdot P_i^t-X_i^t\right| & r_4<0.5 \\ X_i^t+r_1\cdot\cos(r_2)\cdot\left|r_3\cdot P_i^t-X_i^t\right| & r_4>0.5 \end{cases} \tag{4}$$

式中:X_i^t为当前个体;X_i^{t+1}为新个体;P_i^t为当前最优个体;r_2、r_3分别为(0,2π)和(0,2)区间服从均匀分布的随机数;r_4为正弦和余弦函数的选择控制因子,是(0,1)区间的随机数;r_1决定算法的局部探索和全局搜索能力,当$\left|r_1\sin(r_2)\right|<1$或$\left|r_1\cos(r_2)\right|<1$时,有利于下一代个体向最优个体范围靠近,从而提高算法的局部搜索能力;当$\left|r_1\sin(r_2)\right|\geqslant 1$或者$\left|r_1\cos(r_2)\right|\geqslant 1$时,算法注重进行全部搜索域的搜索,加强算法的全局搜索能力。

r_1随着迭代次数动态变化,其计算公式为

$$r_1=a-a\frac{t}{T_{\max}} \tag{5}$$

式中a为大于1的常数,常取经验值2。

本文将SCA中的个体更新方式引入到HS算法的即兴创作阶段的和声向量微调操作中,采用式(6)代替式(3)进行和声向量微调,有

$$X_i^{t+1}=\begin{cases}X_i^t+r_1\cos(r_2)\left|r_3X_i^{\text{best}}-X_i^t\right|\times \text{BW} & r_4>0.5\\ X_i^t+r_1\sin(r_2)\left|r_3X_i^{\text{best}}-X_i^t\right|\times \text{BW} & r_4\leqslant 0.5\end{cases} \tag{6}$$

式中r_1、r_2、r_4的说明同前。为了加快算法的收敛速度,将r_3设为在(0,1)区间服从均匀分布的随机数。

2.2 Levy飞行机制

法国数学家Paul Levy观察发现自然界中多种生物活动轨迹具有长时间以较小步长随机游走,偶尔以较大步长进行跳跃变化的特点,据此提出了Levy飞行的概念[17]。Levy飞行这种长短距离交替进行的飞行特性,对于平衡优化算法的局部探索和全局搜索能力具有重要作用:较小步长的随机游走有利于算法进行局部探索,而偶尔的较大步长的跳跃则有利于算法跳出局部最优,提高全局搜索能力。

为了便于表示,Mantegna[18]使用式(7)表示Levy飞行随机步长s的生成方法。

$$s=\frac{\mu}{|\nu|^{\frac{1}{\beta}}} \tag{7}$$

式中:μ、ν为服从正态分布的参数,即$\mu\sim N\left(0,\sigma_\mu^2\right)$、$\nu\sim N\left(0,\sigma_\nu^2\right)$。$\sigma_\mu$和$\sigma_\nu$为

$$\sigma_\mu=\left[\frac{\Gamma\left(1+\beta\right)\cdot\sin\left(\pi\cdot\beta/2\right)}{\Gamma\left(\left(1+\beta\right)/2\right)\cdot\beta\cdot2^{(\beta-1)/2}}\right]^{\frac{1}{\beta}},\qquad \sigma_\nu=1 \tag{8}$$

式中:$\Gamma(x)$为Gamma函数;$\beta\in(0,2)$,这里取经验值1.5。

本文利用Levy飞行机制来对参数BW进行更新。具体做法是,在迭代过程中,对最优个体和随机个体按式(7)中的Levy飞行步长s进行调整,再利用式(9)实现BW的更新。这样,当s变化较小时,能够尽可能保证步长的变化方向在最优个体附近,有利于提高算法的计算精度;当s变化较大时,能够避免算法陷入局部最优,有利于提高算法的全局搜索能力,有

$$\text{BW}=(X_i^{\text{best}}-X_i^t)\cdot\alpha\cdot s \tag{9}$$

式中α取经验值0.001。

2.3 参数自适应调整

文献[19]将基本HS算法中重要参数变化对优化结果的影响进行了理论分析和数学证明,得出如下结论:较小的HMCR和较小的搜索域有利于提高算法的局部搜索能力;较大的HMCR和较大的搜索域有利于增加和声库的多样性;较大的PAR有利于提高算法的计算精度,而较小的PAR有利于保持算法稳定性。由此可见,算法迭代前期,侧重于进行局部探索,宜将HMCR设为一个较小值,PAR设为一个较大值,为了避免算法陷入局部最优,搜索域应较大;算法迭代后期,适当地增大HMCR有利于增强算法的全局搜索能力,适当地减小PAR可以保持较好的算法稳定性,利于让*HM*中的和声逐渐接近最优和声。

为达到上述目标,本文首先使用式(10,11)对HMCR和PAR进行自适应调整。

$$\text{HMCR}=\begin{cases}0.4+0.5\times\dfrac{t}{T_{\max}} & t<\dfrac{T_{\max}}{4}\\ 0.9 & t\geqslant\dfrac{T_{\max}}{4}\end{cases} \tag{10}$$

$$\mathrm{PAR}=\begin{cases}0.99 & t<\dfrac{T_{\max}}{4}\\ 0.99-0.05\times\dfrac{t}{T_{\max}} & t\geqslant\dfrac{T_{\max}}{4}\end{cases}\tag{11}$$

然后,再参考文献[15]中的策略对搜索域进行线性动态调整。具体方法是,分别利用式(12)和式(13)对搜索域的上、下界进行初始化,在算法迭代过程中再利用式(14~17)对搜索域进行线性动态更新。

$$X_{U_{\mathrm{new},j}}=U\tag{12}$$

$$X_{L_{\mathrm{new},j}}=L\tag{13}$$

$$X_{\max\mathrm{bound},j}=\max\left(\mathrm{HM}(:,j)\right)\tag{14}$$

$$X_{\min\mathrm{bound},j}=\min\left(\mathrm{HM}(:,j)\right)\tag{15}$$

$$X_{U_{\mathrm{new},j}}=\left(1-\frac{t}{T_{\max}}\right)\times X_{U_{\mathrm{new},j}}+\frac{t}{T_{\max}}\times X_{\max\mathrm{bound},j}\tag{16}$$

$$X_{L_{\mathrm{new},j}}=\left(1-\frac{t}{T_{\max}}\right)\times X_{L_{\mathrm{new},j}}+\frac{t}{T_{\max}}\times X_{\min\mathrm{bound},j}\tag{17}$$

3 AHS-SC-LF描述与分析

3.1 算法描述

基于上述3种改进策略,可用图1表示AHS-SC-LF的算法流程。AHS-SC-LF的具体步骤可描述如下:

步骤1 定义优化问题,并初始化参数。

步骤2 初始化和声记忆库HM。

步骤3 即兴创作。生成随机数r_1,若$r_1\geqslant$ HMCR,利用式(1)从HM外的搜索域内随机选择一个和声向量作为$\boldsymbol{X}^{\mathrm{new}}$;若$r_1<$ HMCR,再生成一个随机数r_2,此时若$r_2<$ PAR,则利用式(6)对HM中所有和声向量进行微调,并从中选择最优者作为$\boldsymbol{X}^{\mathrm{new}}$。

步骤4 更新和声库。若$\boldsymbol{X}^{\mathrm{new}}$的适应度值优于HM中的$\boldsymbol{X}^{\mathrm{worst}}$适应度值,则用$\boldsymbol{X}^{\mathrm{new}}$取代$\boldsymbol{X}^{\mathrm{worst}}$,加入HM中;否则,则利用式(9)更新BW,用式(14~17)更新搜索域,并转步骤3。

步骤5 检查是否达到算法终止条件。若达到,则输出全局最优解,算法结束;否则,分别利用式(10)和式(11)更新HMCR和PAR,并转步骤3。

3.2 算法复杂度分析

以迭代1次为例,对AHS-SC-LF算法的最坏时间复杂度进行分析。假设测试函数的维度为N,Levy飞行操作的时间复杂度为$O(\mathrm{Levy})$,则:

(1)在搜索域内初始化HMS个和声向量的时间复杂度为$O(N\times\mathrm{HMS})$;

(2)即兴创作阶段的时间复杂度为$O\left(N\times\mathrm{HMS}\times O(\mathrm{Levy})\right)$。通常,Levy飞行操作经过降噪后的时间复杂度为$O(1)$,故此阶段的时间复杂度为$O(N\times\mathrm{HMS})$;

(3)计算适应度函数的时间复杂度为$O(N\times\mathrm{HMS})$;

(4)更新搜索域和参数需要的时间复杂度为$O(N\times\mathrm{HMS})$;

(5)其他阶段的时间复杂度均为$O(1)$。

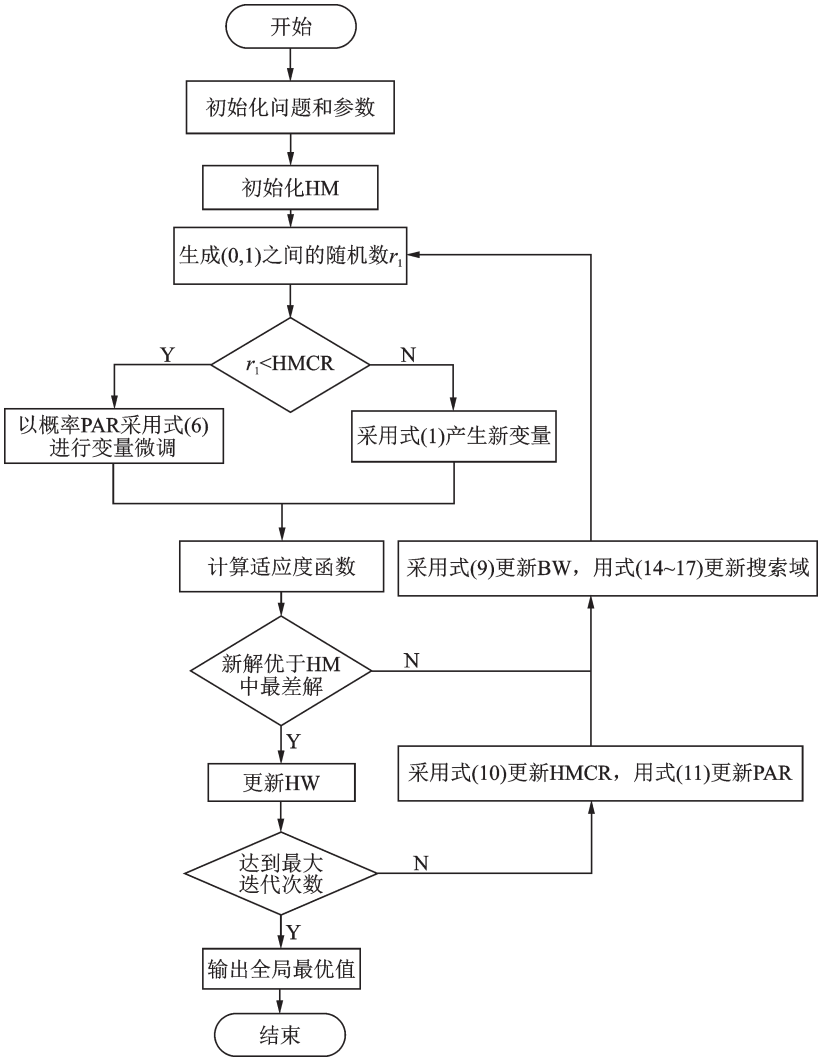

图1 AHS-SC-LF算法流程图

Fig.1 Flowchart of AHS-SC-LF algorithm

综上,算法迭代一次的时间复杂度为$O(N\times \mathrm{HMS})$。因此,AHS-SC-LF经过T次迭代的时间复杂度为$O(T\times N\times \mathrm{HMS})$。

4 性能测试实验

4.1 实验环境及参数说明

为了验证算法性能,本文将AHS-SC-LF与基本HS算法及IHS[8]、GDHS[10]、ID-HS-LDD[15]这3种性能比较优越的改进HS算法在10个基准测试函数上进行优化求解的性能对比实验。5种算法的参数设置如表2所示。实验在配置为Intel(R)Core(TM)R5-5600 CPU @ 4.20 GHz、16 GB内存、Window10操作系统的笔记本电脑上进行,采用的编程语言为Python 3.8。

实验使用的10个基准函数如表3所示。其中,F_1—F_3为可设置维度的单峰函数,F_4—F_7为可设置维度的多峰函数,F_8—F_{10}为固定维度的多峰值函数。实验时,将F_1—F_7的维度设置为10维、30维两种,

将F_8—F_{10}设置为固定2维。5种算法均单独执行30次，每次执行的最大迭代次数设为7 000。

表2 算法参数设置

Table 2 Parameter setting for algorithms

算法	相关参数
HS	HMS=5，HMCR=0.9，PAR=0.99，BW=0.008
IHS[8]	HMS=5，HMCR=0.9，$PAR_{min}=0.05$，$PAR_{max}=0.99$，$BW_{min}=0.001$，$BW_{max}=0.008$
GDHS[10]	HMS=5
ID-HS-LDD[15]	HMS=30，$HMCR_{min}=0.4$，$HMCR_{max}=0.9$，$PAR_{min}=0.05$，$PAR_{max}=0.99$
AHS-SC-LF	HMS=5

表3 基准测试函数

Table 3 Benchmark functions

函数符号	函数名	表达式	函数维度	搜索域	全局最优解
F_1	Sphere	$f(x)=\sum_{i=1}^{d}x_i^2$	10，30	[−5.12，5.12]	0
F_2	Schwefel 2.22	$f(x)=\sum_{i=1}^{d}\lvert x_i\rvert+\prod_{i=1}^{d}\lvert x_i\rvert$	10，30	[−10，10]	0
F_3	Sum Squares	$f(x)=\sum_{i=1}^{d}ix_i^2$	10，30	[−10，10]	0
F_4	Ackley	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{d}\cos(2\pi x_i)\right)+20+\exp(1)$	10，30	[−32，32]	0
F_5	Griewank	$f(x)=\frac{1}{4\,000}\sum_{i=1}^{d}x_i^2-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	10，30	[−600，600]	0
F_6	Rastrigin	$f(x)=\sum_{i=1}^{d}(x_i^2-10\cos(2\pi x_i)+10)$	10，30	[−5.12，5.12]	0
F_7	Alpine	$f(x)=\sum_{i=1}^{d}\lvert x_i\sin(x_i)+0.1x_i\rvert$	10，30	[−10，10]	0
F_8	Three Hump Camel	$f(x)=2x_1^2-1.05x_1^4+\frac{x_1^6}{6}+x_1x_2+x_2^2$	2	[−5，5]	0
F_9	Schaffer N.2	$f(x)=0.5+\frac{\sin^2(x_1^2-x_2^2-0.5)}{(1+0.001(x_1^2+x_2^2))^2}$	2	[−100，100]	0
F_{10}	Drop Wave	$f(x)=-\frac{1+\cos(12\sqrt{x_1^2+x_2^2})}{0.5(x_1^2+x_2^2)+2}$	2	[−5.12，5.12]	−1

4.2 实验结果及分析

表4记录了5种算法在10个基准函数上运行30次的实验对比结果。表中Dim表示维度，Mean表示30次最优目标函数值的均值(体现算法的计算精度)，Std表示30次最优目标函数值的标准差(体现算法的稳定性)。

表4 5种HS算法运行30次结果

Table 4 Results of the five HS algorithms to run 30 times

函数	Dim	HS[1]	IHS[8]	GDHS[10]	ID-HS-LDD[15]	AHS-SC-LF
		Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std	Mean ± Std
F_1	10	1.37E+01±5.55	5.63E−05± 2.97E−04	1.38E−24± 7.41E−24	2.81E−90± 1.51E−89	0.00±0.00
	30	1.17E+02± 1.31E+01	8.74E−01± 2.72E−01	2.33E−18± 7.77E−18	7.32E−21± 2.49E−20	1.51E−69±4.95E−69
F_2	10	1.86E+01±4.12	8.47E−02± 6.22E−02	3.22E−22± 1.10E−21	4.32E−45± 2.00E−44	0.00±0.00
	30	2.87E+08± 1.27E+08	6.22±1.16	2.00E−08± 4.04E−08	1.63E−07± 2.39E−07	2.48E−34±6.68E−34
F_3	10	1.12E+03± 5.13E+02	5.37E−02± 1.23E−01	1.72E−27± 8.63E−27	3.94E−78± 1.72E−77	0.00±0.00
	30	1.07E+05± 2.05E+04	5.69E+02± 1.92E+02	1.01E−06± 2.98E−06	2.52E−05± 5.84E−05	1.05E−22±3.78E−22
F_4	10	1.67E+01±1.30	1.06±6.29E−01	7.70E−15± 3.32E−15	4.03E−15± 1.21E−15	8.29E−16±1.50E−15
	30	1.98E+01± 2.89E−01	5.47±6.23E−01	1.53E−08± 2.42E−08	1.18E−09± 9.24E−10	4.62E−15±1.63E−15
F_5	10	4.61E+01± 1.73E+01	8.91E−01± 1.72E−01	4.93E−02± 2.99E−02	3.19E−02± 1.94E−02	0.00±0.00
	30	4.05E+02± 4.37E+01	4.35±9.11E−01	1.17E−02± 1.15E−02	3.33E−02± 3.99E−02	0.00±0.00
F_6	10	7.17E+01± 1.29E+01	4.28E−01± 6.56E−01	2.29E−01± 4.60E−01	0.00±0.00	0.00±0.00
	30	3.58E+02± 3.01E+01	2.72E+01±2.84	6.79±3.41	1.98E+01±7.10	0.00±0.00
F_7	10	7.59±2.54	1.11E−02± 7.50E−03	4.12E−04± 1.07E−03	4.66E−04± 5.56E−04	0.00±0.00
	30	4.74E+01±4.48	1.29±3.57E−01	6.87E−02± 5.06E−02	8.04E−01± 5.53E−01	1.16E−20±6.22E−20
F_8	2	3.26E−02± 8.95E−02	5.97E−02± 1.19E−01	8.59E−02± 3.55E−01	9.78E−281± 0.00	0.00±0.00
F_9	2	3.06E−02± 4.03E−02	6.26E−03± 7.31E−03	5.40E−02± 1.07E−01	0.00±0.00	0.00±0.00
F_{10}	2	−0.92± 5.84E−02	−0.93± 5.54E−02	−0.94± 7.85E−02	−1.00±0.00	−1.00±0.00

从表4中的数据可以看出:

(1) 在单峰函数F_1、F_2、F_3上10维和30维的情况下,AHS-SC-LF的计算精度和稳定性明显优于其它4种算法,且在10维时,AHS-SC-LF在F_1、F_2、F_3上均收敛到理论最优解。

(2) 在多峰函数F_4上10维的情况下,AHS-SC-LF稳定性与ID-HS-LDD接近,但计算精度略优于ID-HS-LDD;30维情况下,AHS-SC-LF的计算精度和稳定性显著优于其他4种HS算法。

(3) 在多峰函数F_5、F_6和F_7上10维的情况下,AHS-SC-LF能直接收敛到理论最优解,计算精度和稳定性均达到最优;30维的情况下,AHS-SC-LF没有收敛到理论最优解,但计算精度和稳定性均远优于其他4种HS算法。

(4) 在维度为2的多峰函数F_9、F_{10}上,AHS-SC-LF和ID-HS-LDD的计算精度和稳定性都达到理论最优解,均优于其他3种算法;在维度为2的多峰函数F_8上,AHS-SC-LF的稳定性与ID-HS-LDD相同,但具有更高的计算精度。

为了进一步验证AHS-SC-LF的收敛性能,根据式(18)所示的平均绝对误差(Mean absolute error, MAE)[20]对AHS-SC-LF和其他4个HS算法进行排序。排序结果如表5所示。

$$\mathrm{MAE}=\frac{\sum_{i=1}^{N}\left|m_i-f_i\right|}{N} \tag{18}$$

式中:m_i表示算法最优解的平均值;f_i表示对应测试函数的理论最优解;N表示测试函数的个数。

表5 算法MAE排名

Table 5 MAE ranking of algorithms

算法	MAE(Dim=30)	rank(Dim=30)	MAE(Dim=10)	rank(Dim=10)	MAE(Dim=2)	rank(Dim=2)
AHS-SC-LF	6.60E−16	1	1.18E−16	1	0.00	1
ID-HS-LDD[15]	2.95	2	4.62E−03	2	3.26E−281	2
GDHS[10]	9.81E−01	3	3.98E−02	3	6.66E−02	5
IHS[8]	8.78E+01	4	3.61E−01	4	4.53E−02	3
HS[1]	4.10E+07	5	4.09E+01	5	4.77E−02	4

由表5可知,在30维、10维和2维3种情况下,在5种算法中,AHS-SC-LF的MAE值均最小,排名均最前,有效地验证了AHS-SC-LF具有较好的收敛性能。

为了分析实验结果,本文采用显著性水平为0.05的Wilcoxon rank-sum test秩和检验方法[21]判断AHS-SC-LF与HS、IHS、GDHS、ID-HS-LDD算法是否具有显著性差异。表6记录了AHS-SC-LF与其他4种HS算法在10个测试函数上进行秩和检验的结果。表中P列的值小于0.05,表示AHS-SC-LF与所比较的算法具有显著性差异(即其性能显著优于所比较的算法);P值为N/A,表示两种算法均收敛到理论最优解;表中D列的符号“+”“=”“−”分别表示AHS-SC-LF优于、同于和差于所比较的算法。

由表6可知,在单峰函数F_1−F_3上的10维和30维的情况下,AHS-SC-LF和与之比较的其他4中HS算法表现出显著性差异;在多峰函数F_6上的10维情况下,AHS-SC-LF与ID-HS-LDD均收敛到理论最优解,但在多峰函数F_4上的10维情况下,相比于ID-HS-LDD,AHS-SC-LF的显著性差异效果有所下降,在多峰函数F_5上的30维情况下,相比于GDHS,AHS-SC-LF的显著性差异效果同样有所下降;对于固定维度为2的多峰函数F_8,与其他4种HS算法相比,AHS-SC-LF表现出显著性差异;对于固定维度为2的多峰函数F_9、F_{10},AHS-SC-LF与ID-HS-LDD的收敛性能相同,且均优于其他3种HS算法。

图2~11给出了使用10个基准函数对5种算法进行性能测试得到的收敛曲线。由图可见:

(1) 在单峰函数F_1、F_2和F_3上,无论是在10维还是在30维的情况下,AHS-SC-LF都是5种HS算法中收敛性能最好的算法。

(2) 在多峰函数F_5、F_6上的10维和30维,以及F_7上10维的情况下,迭代3 000次或4 000次左右,

表6 Wilcoxon秩和检验的*P*值和检验结果

Table 6 *P* value of Wilcoxon rank-sum test and its results

函数	维度	HS[1]		IHS[8]		GDHS[10]		ID-HS-LDD[15]	
		P	*D*	*P*	*D*	*P*	*D*	*P*	*D*
F_1	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
	30	3.02E−11	+	3.02E−11	+	3.02E−11	+	3.02E−11	+
F_2	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
	30	3.02E−11	+	3.02E−11	+	3.02E−11	+	3.02E−11	+
F_3	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
	30	3.02E−11	+	3.02E−11	+	3.02E−11	+	3.02E−11	+
F_4	10	4.08E−12	+	2.36E−12	+	1.37E−11	+	1.58E−10	−
	30	7.57E−12	+	6.39E−12	+	1.01E−11	+	5.14E−12	+
F_5	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
	30	4.11E−12	+	2.37E−12	+	3.77E−09	−	1.57E−11	+
F_6	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	N/A	=
	30	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
F_7	10	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
	30	3.02E−11	+	3.02E−11	+	3.02E−11	+	3.02E−11	+
F_8	2	1.21E−12	+	1.21E−12	+	1.21E−12	+	1.21E−12	+
F_9	2	1.21E−12	+	1.21E−12	+	1.21E−12	+	N/A	=
F_{10}	2	1.21E−12	+	1.21E−12	+	1.13E−12	+	N/A	=
+/=/−		10/0/0		10/0/0		9/0/1		6/3/1	

AHS-SC-LF收敛到理论最优解;F_6上10维的情况下,虽然ID-HS-LDD也能收敛到理论最优解,但速度远慢于AHS-SC-LF。

(3) 在多峰函数F_4、F_7上的30维情况下,AHS-SC-LF在最大迭代次数7 000次内没能收敛到理论最优解,但其收敛性能在5种HS算法中最佳。

(4) 在维度固定为2的多峰函数F_9、F_{10}上,AHS-SC-LF和ID-HS-LDD都能收敛到理论最优解,但AHS-SC-LF在迭代不到1 000次时就收敛了,收敛速度远快于ID-HS-LDD;在F_8上,AHS-SC-LF迭代2000多次就收敛到理论最优解,其收敛性能显著优于其他4种HS算法。

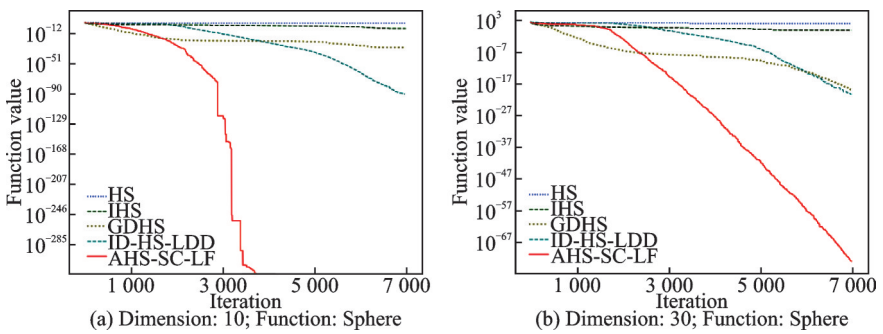

图2 F_1收敛曲线

Fig.2 F_1 convergence curves

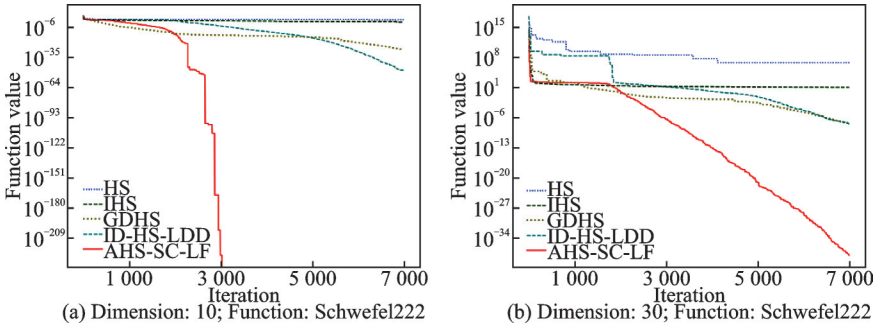

图3 F_2收敛曲线

Fig.3 F_2 convergence curves

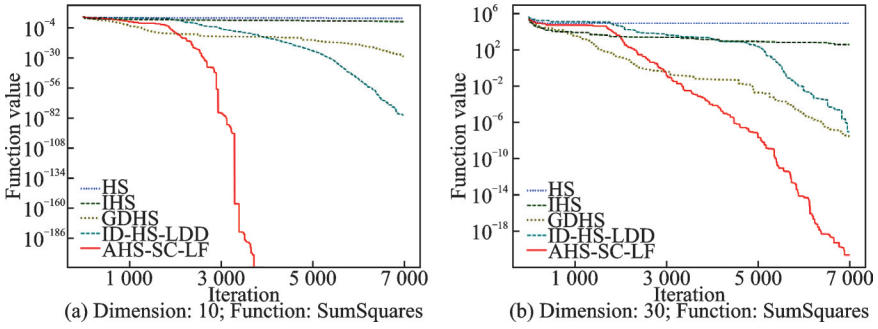

图4 F_3收敛曲线

Fig.4 F_3 convergence curves

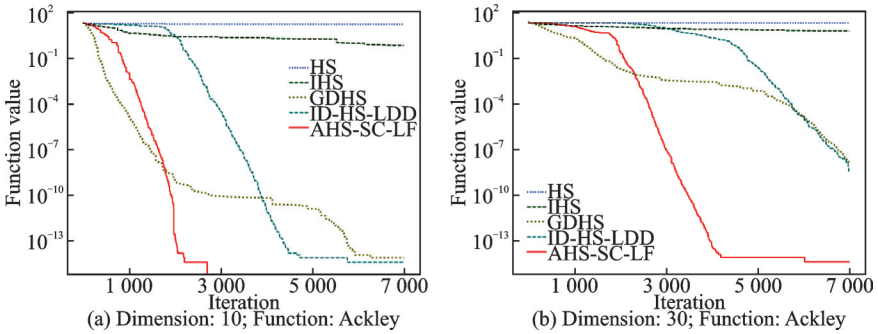

图5 F_4收敛曲线

Fig.5 F_4 convergence curves

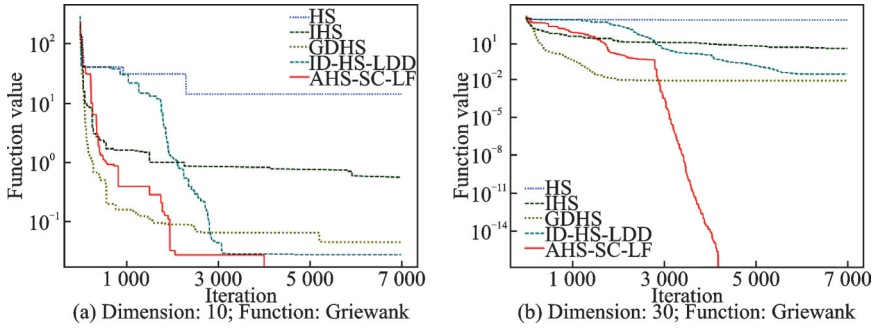

图6 F_5收敛曲线

Fig.6 F_5 convergence curves

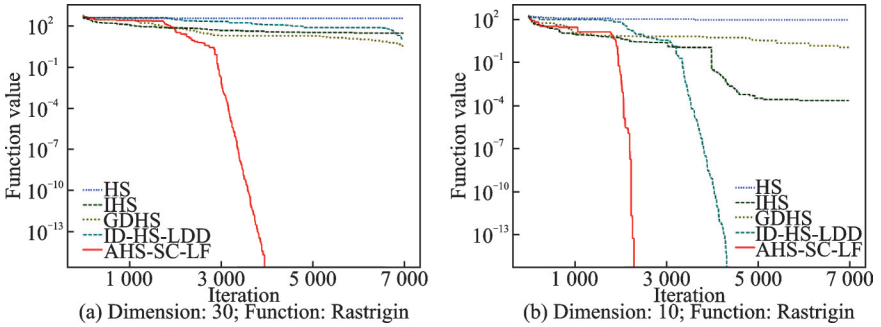

图7 F_6收敛曲线

Fig.7 F_6 convergence curves

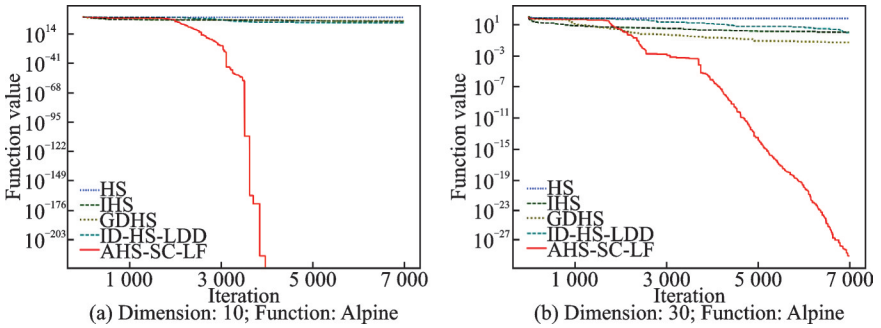

图8 F_7收敛曲线

Fig.8 F_7 convergence curves

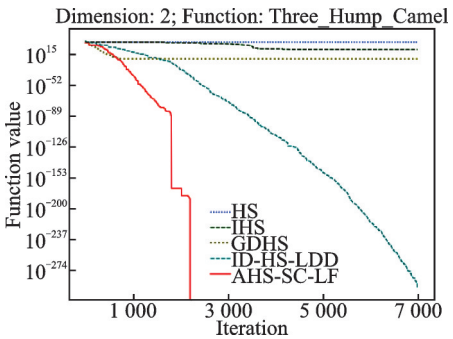

图9 F_8收敛曲线

Fig.9 F_8 convergence curves

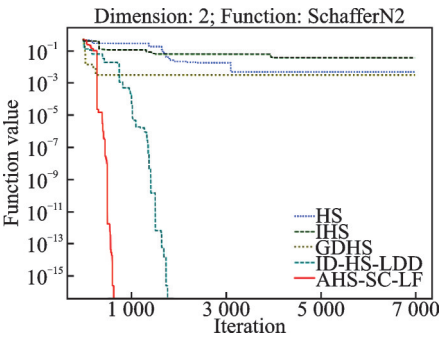

图10 F_9收敛曲线

Fig.10 F_9 convergence curves

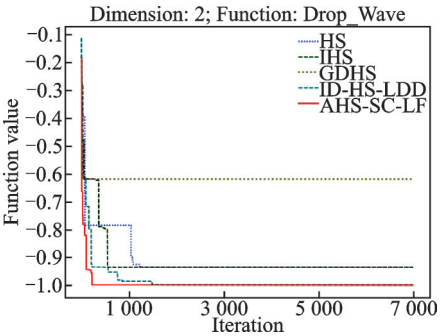

图11 F_{10}收敛曲线图

Fig.11 F_{10} convergence curves

5 结束语

本文针对基本HS算法的不足,通过引入正余弦优化算子、Levy飞行机制和参数自适应动态调整策略,提出了一种改进的HS算法,即AHS-SC-LF。该算法首先在即兴创作阶段,将正余弦优化算子引入到和声向量的微调操作中,通过平衡局部探索能力和全局搜索能力,提高了算法的收敛性能;其次,利用Levy飞行机制来更新BW,避免算法陷入局部最优,进一步提高了算法的全局搜索能力;然后,对HMCR和PAR进行自适应调整,对搜索域进行动态线性调整,提高了算法的收敛速度。与基本HS算法以及其他3种性能较优的改进HS算法在包括单峰函数和多峰函数的10个基准函数的不同维度上进行性能对比测试实验的结果证明,AHS-SC-LF在处理函数优化问题时具有比其他4种HS算法更强的全局搜索能力、更快的收敛速度和更高的计算精度。

参 考 文 献:

[1] GEEM Z W, KIM J H, LOGANATHAN G V. A new heuristic optimization algorithm: harmony search[J]. Simulation, 2001, 76(2): 60-68.

[2] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multi-objective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197.

[3] CUI Yingying, MENG Xi, QIAO Junfei. A multi-objective particle swarm optimization algorithm based on two-archive mechanism[J]. Applied Soft Computing, 2022, 119: 108532.

[4] 薛婷, 贝绍轶, 李波. 基于蚁群算法的智能小车路径规划[J]. 计算机仿真, 2021, 38(12): 362-366.
XUE Ting, BEI Shaoyi, LI Bo. Path planning of intelligent vehicle based on ant colony algorithm[J]. Computer Simulation, 2021, 38(12): 362-366.

[5] 何胜学. 公交车辆调度的超级时空网络模型及改进和声搜索算法[J]. 计算机应用研究, 2021, 38(10): 3078-3084.
HE Shengxue. Super time-space network-based transit vehicle scheduling model and modified harmony search algorithm[J]. Application Research of Computers, 2021, 38(10): 3078-3084.

[6] SONG Y N, PAN Q K, GAO L, et al. Improved non-maximum suppression for object detection using harmony search algorithm[J]. Applied Soft Computing Journal, 2019, 81(5): 105478.

[7] MISTARIHI M Z, OKOUR R A, et al. Integrating advanced harmony search with fuzzy logic for solving buffer allocation problems[J]. Arabian Journal Science and Engineering, 2020, 45: 3233-3244.

[8] MAHDAVI M, FESANGHARY M, DAMMANGIR E. An improved harmony search algorithm for solving optimization problems[J]. Applied Mathematics and Computation, 2007, 188 (2): 1567-1579.

[9] MAHDAVI M. Global-best harmony search[J]. Applied Mathematics and Computation, 2008, 198(2): 643-656.

[10] KHALILI M, KHARRAT R, SALAHSHOOR K, et al. Global dynamic harmony search algorithm: GDHS[J]. Applied Mathematics and Computation, 2014, 228: 195-219.

[11] CHAKRABORTY P, ROY G G, DAS S. An improved harmony search algorithm with differential mutation operator[J]. Fundamenta Informaticae, 2004, 95: 401-426.

[12] STORN R, PRICE K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of Global Optimization, 1997, 11: 341-359.

[13] TAHERINEJAO N. Highly reliable harmony search algorithm[C]//Proceedings of 2009 European Conference on Circuit Theory and Design, IEEE Transactions on Evolutionary Computation. Piscataway, NJ: IEEE, 2009: 818-822.

[14] 卢宇婷, 林禹攸, 彭乔姿, 等. 模拟退火算法改进综述及参数探究[J]. 大学数学, 2015, 31(6): 96-103.

LU Yuting, LIN Yuyou, PENG Qiaozi, et al. A review of improvement and research on parameters of simulated annealing algorithm[J]. College Mathematics, 2015, 31(6): 96-103.

[15] ZHU Qidan, TANG Xiangmeng. An improved differential based harmony search algorithm with liner dynamic domain[J]. Knowledge-Based Systems, 2020, 188: 104809.

[16] MIRJALILI S. SCA: A sine cosine algorithm for solving optimization problems[J]. Knowledge-Based Systems, 2016, 96: 120-133.

[17] 雍欣,高岳林,赫亚华, 等. 多策略融合的改进萤火虫优化算法[J]. 计算机应用, 2022(12): 1-10.
YONG Xin, GAO Yuelin, HE Yahua' et al. Improved firefly optimization algorithm based on multi-strategy fusion[J]. Journal of Computer Applications, 2022(12): 1-10.

[18] MANTEGNA R N. Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes[J]. Physical review E Statistical Physics Plasmas Fluids and Related Interdisciplinary Topics, 1994, 49(5): 4677-4683.

[19] KANG Diwen, MO Liping, Wang Fangling, et al. Adaptive harmony search algorithm utilizing differential evolution and opposition-based learning[J]. Mathematical Biosciences and Engineering, 2021, 18(4): 4226-4246.

[20] NABIL E. A modified flower pollination algorithm for global optimization[J]. Expert Systems with Applications, 2016, 57: 192-2.

[21] DERRAC J, GARCIA S, MOLiNA D, et al. A practical tutorial on the use of nonparametric statistical tests as a method logy for comparing evolutionary and swarm intelligence algorithms[J]. Swarm and Evolutionary Computation, 2011, 1(1): 3-18.

作者简介:

程翠娜(1999-),女,硕士研究生,研究方向:智能计算,自然语言处理,E-mail:2879835115@qq.com。

奉松绿(1997-),男,硕士研究生,研究方向:智能计算,自然语言处理,E-mail:979675358@qq.com。

莫礼平(1972-),**通信作者**,女,教授,硕士生导师,研究方向:自然语言处理、智能计算、Petri网及其应用,E-mail:zmx89@jsu.edu.cn。

(编辑:刘彦东)