基于等级制度和布朗运动的混沌麻雀搜索算法

汤安迪[1]，韩　统[1]，徐登武[2]，谢　磊[1]
(1. 空军工程大学航空工程学院，西安，710038；2. 94855 部队，浙江衢州，324000)

摘要　针对麻雀搜索算法在迭代后期种群多样性减弱、易于陷入局部最优等问题，提出了一种基于等级制度和布朗运动的混沌麻雀搜索算法(CSSA-HB)。首先引入混沌映射调整麻雀搜索算法关键参数；其次引入等级制度，利用父代种群中 3 个最优个体对警戒者进行位置更新，加强个体间交流，增强种群多样性；然后利用布朗运动可控均匀步长，增强算法的探索能力；当算法陷入停滞时，利用布朗运动策略对个体施加扰动，促使算法跳出局部最优；最后利用贪婪策略保留优势个体，有效加快收敛速度。对 12 个测试函数进行仿真实验，结果表明混沌映射能有效增强算法性能，迭代映射表现最佳；改进算法具有较强的局部最优规避能力、更快的收敛速度和更高的收敛精度。

关键词　麻雀搜索算法；混沌映射；等级制度；贪婪策略
DOI　10.3969/j.issn.1009-3516.2021.03.015
中图分类号　TP301.6　　**文献标志码**　A　　**文章编号**　1009-3516(2021)03-0096-08

A Chaos Sparrow Search Algorithm Based on Hierarchy and Brownian Motion

TANG Andi[1], HAN Tong[1], XU Dengwu[2], XIE Lei[1]
(1. Aeronautics Engineering College, Air Force Engineering University, Xi'an 710038, China;
2. Unit 94855, Quzhou 324000, Zhejiang, China)

Abstract　Aimed at the problems that population diversity is weakening and easy to fall into local optimization in the late iteration by sparrow search algorithm (SSA), a chaos sparrow search algorithm is proposed based on hierarchy and Brownian motion (CSSA-HB). Firstly, chaotic map is introduced for adjusting the key parameters of SSA. Secondly, a hierarchy is introduced. The three best individuals of the parent population are used to update the position of the vigilantes, enhancing communication among individuals and increasing population diversity. The uniform step controlled by Brownian motion is used to enhance the exploration ability of the algorithm. When the algorithm is stagnant, the Brownian motion strategy is used to disturb the individual to urge the algorithm to get rid of the local optimum. Finally, the greedy strategy is used to retain the dominant individuals and effectively accelerate the convergence speed. 12 test functions are made by the simulation experiments. The results show that the chaotic maps can enhance effectively the performance of the algorithm, and make the iterative maps perform the best. The improved algorithm is stronger in local optimum avoidance ability, faster at convergence speed, and higher in convergence accuracy.

收稿日期：2020-11-09
基金项目：陕西省自然科学基金(2020JQ-481，2021JM224)；航空科学基金(201951096002)
作者简介：汤安迪(1996—)，男，重庆永川人，硕士生，研究方向：作战任务规划、智能优化算法。E-mail：418932433@qq.com

引用格式：汤安迪，韩统，徐登武，等. 基于等级制度和布朗运动的混沌麻雀搜索算法[J]. 空军工程大学学报(自然科学版)，2021，22(3)：96-103. TANG Andi, HAN Tong, XU Dengwu, et al. A Chaos Sparrow Search Algorithm Based on Hierarchy and Brownian Motion[J]. Journal of Air Force Engineering University (Natural Science Edition), 2021, 22(3): 96-103.

Key words　sparrow search algorithm (SSA)；chaotic map；hierarchy；greedy strategy

群智能优化算法是一类模拟自然界生物行为和自然现象的元启发式优化算法，具有良好的并行性和自主探索性。自 1975 年美国教授 Holland 根据达尔文进化论以及自然界优胜劣汰机制提出了遗传算法[1]以后，越来越多的学者通过对不同生物种群和物理现象进行分析，从中获取灵感，提出多种群智能优化算法。包括粒子群算法[2](particle swarm optimization，PSO)、鲸鱼优化算法[3](whale optimization algorithm，WOA)、灰狼优化算法[4](grey wolf optimization，GWO)等。

麻雀搜索算法(sparrow search algorithm，SSA)是薛建凯等[5]于 2020 年提出的群智能优化算法，具有搜索精度高、收敛快等特点，但其在接近全局最优时，仍旧会出现种群多样性减小、易于陷入局部最优等缺陷。

为了改善群体的多样性，防止算法陷入局部最优，杨万里等[6]利用 Logistic 映射调整 PSO 算法惯性权重，通过混沌映射的随机遍历性，使算法在迭代过程中随机选择开发或探索行为；IBRAHIM 等[7]利用 Logistic 映射初始化 GWO 算法种群，增加初始种群个体多样性，以此增加算法收敛效率；吕鑫等[8]利用 Tent 映射对 SSA 算法个体进行扰动，防止算法陷入局部最优。

上述文献主要利用某一种混沌映射对算法进行改进，但没有讨论不同混沌映射对于算法性能改进的影响。本文为解决麻雀搜索算法在迭代后期多样性减弱、易于陷入局部最优的问题，提出利用混沌映射调整麻雀搜索算法关键参数，通过函数测试确定使用哪种混沌映射，并引入 GWO 算法的等级制度，增强种群多样性，利用布朗运动扩大搜索范围，增强算法探索能力，当算法陷入停滞时，使用布朗运动策略对个体施加扰动，帮助算法跳出局部最优，最后利用贪婪策略有效保留优势个体，加快算法收敛速度。

1　麻雀搜索算法

麻雀搜索算法是一种新兴的群智能优化算法。SSA 主要模拟了麻雀觅食的过程。麻雀觅食过程是发现者-跟随者模型的一种，同时还叠加了侦查预警机制。种群中找到食物较好的个体作为发现者，其他个体作为跟随者，同时种群中选取一定比例的个体进行侦查预警，如果发现危险则放弃食物，安全第一。

SSA 算法中有发现者、追随者以及警戒者。分别按照各自规则进行位置更新，更新规则如下：

$$X_{i,j}^{t+1}=\begin{cases}X_{i,j}^{t}\exp\left(\dfrac{-i}{\partial\times I_{\max}}\right), & \text{if}\quad R_2<S_T\\ X_{i,j}^{t}+QL, & \text{if}\quad R_2\geqslant S_T\end{cases}\tag{1}$$

式中：t 为当前代数；$X_{i,j}^{t+1}$ 表示在 t 代第 i 只麻雀 j 维的位置；$I_{\max}$ 为最大迭代数 ∂ 是一个 0 到 1 的随机数；R_2 表示警戒值；S_T 为安全阈值；Q 是一个服从正态分布的随机数；$\boldsymbol{L}$ 是一个一行多维的全一矩阵。

$$X_{i,j}^{t+1}=\begin{cases}Q\exp\left(\dfrac{X_{\text{worst}}^{t}-X_{i,j}^{t}}{i^2}\right), & \text{if}\quad i>\dfrac{n}{2}\\ X_{p}^{t+1}+\left|X_{i,j}^{t}-X_{p}^{t+1}\right|\boldsymbol{A}^{+}\boldsymbol{L}, & \text{if}\quad \text{others}\end{cases}\tag{2}$$

式中：X_p 表示被发现者占据的最佳位置；X_{worst} 表示当前最差位置，$\boldsymbol{A}$ 是一个一行多维的元素为 1 或−1 的矩阵。

$$X_{i,j}^{t+1}=\begin{cases}X_{\text{best}}^{t}+\beta\left|X_{i,j}^{t}-X_{\text{best}}^{t}\right|, & \text{if}\quad f_i>f_g\\ X_{i,j}^{t}+K\left(\dfrac{\left|X_{i,j}^{t}-X_{\text{worst}}^{t}\right|}{(f_i-f_w)+\varepsilon}\right), & \text{if}\quad f_i=f_g\end{cases}\tag{3}$$

式中：X_{best} 是当前全局最佳位置；β 是步长控制参数；f_i 是当前麻雀的适应度；f_g 和 f_w 是当前最佳适应度和最差适应度；ε 是一个常数，用于避免分母为零。

2　基于等级制度和布朗运动的混沌麻雀搜索算法

2.1　混沌映射

混沌是一种在非线性动力系统中产生的随机现象，具有规律性、随机性，对初始条件和遍历性敏感。根据这些特征，构造了不同方程表示的混沌图，用以更新优化算法中的随机变量[9]。

在本文中，使用一维且不可逆的混沌映射来生成一组混沌值来改善基本 SSA 算法的参数，如表 1，这 10 种混沌映射在生成数值时效果不同，且已在文献[10～12]中证明其有效性。

2.2　等级制度策略

麻雀搜索算法在对警戒者更新时，仅考虑当前状态最优解，没有考虑其他次优解，缺少群体间交流，降低种群个体多样性，易使算法陷入局部最优，因此学习灰狼优化算法中的等级制度策略，选取前 3 个最优解对警戒者进行位置更新，更新公式如下：

$$X_{i,j}^{t+1}=\sum_{j=\alpha,\beta,\gamma}\frac{f(X_{j\text{best}}^{t})}{\sum\limits_{z=\alpha,\beta,\gamma}f(X_{j\text{best}}^{t})}X_{i,j}^{t}\tag{4}$$

式中：$X_{j\text{best}}^{t}$ 为当前种群优势个体；$f(X_{j\text{best}}^{t})$ 为当前种群优势个体适应度值。

表 1 混沌映射

序号	映射类型	函数
1	Chebyshev map	$x_{i+1}=\cos(i\cos^{-1}(x_i))$
2	Circle map	$x_{i+1}=\mathrm{mod}(x_i+b-(a/2\pi)\sin(2\pi x_k)),1)$, $a=0.5$ and $b=0.2$
3	Gauss map	$x_{i+1}=\begin{cases}1, & x_i=0\\ 1/\mathrm{mod}(x_i,1), & \text{otherwaise}\end{cases}$
4	Iterative map	$x_{i+1}=D\sin(a\pi/x_i)$, $a=0.7$
5	Logistic map	$x_{i+1}=ax_i(1-x_i)$, $a=4$
6	Precewise map	$x_{i+1}=\begin{cases}x_i/p, & 0\leqslant x_i<p\\ (x_i-p)/(0.5-p), & p\leqslant x_i<0.5\\ (1-x_i-p)/(0.5-p), & 0.5\leqslant x_i<1-p\\ (1-x_i)/p, & 1-p\leqslant x_i<1\end{cases}$
7	Sine map	$x_{i+1}=a/4\sin(\pi x_i)$, $a=4$
8	Singer map	$x_{i+1}=\mu(7.86x_i-23.32x_i^2+28.75x_i^3-13.301\,875x_i^4)$, $\mu=1.07$
9	Sinusoidal map	$x_{i+1}=ax_i^2\sin(\pi x_i)$, $a=2.3$
10	Tent map	$x_{i+1}=\begin{cases}x_i/0.7, & x_i<0.7\\ 10/3\times(1-x_i), & x_i\geqslant 0.7\end{cases}$

2.3 布朗运动

布朗运动是一个随机过程,其步长取决于一个均值为 0、方差为 1 的高斯分布概率函数,能够更均匀、更可控的步长覆盖搜索空间,布朗运动在点 x 处的概率密度函数如下:

$$f_B(x;\mu,\sigma)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{5}$$

麻雀位置更新公式为:

$$X_{i,j}^{t+1}=X_{i,j}^{t}+PR(R_B(X_{\mathrm{best}}^{t}-R_BX_{i,j}^{t})) \tag{6}$$

式中:$P=0.5$;R 为 0 到 1 均匀分布的随机数;RB 为布朗运动步长。

2.4 改进算法描述

本文针对麻雀搜索算法种群多样性减少、易于陷入局部最优的不足,提出了基于等级制度和布朗运动的混沌麻雀搜索算法(chaos sparrow search algorithm based on hierarchy and Brownian motion, CSSA-HB)。首先利用混沌映射调整麻雀搜索算法的警戒值参数,然后利用等级制度策略和布朗运动策略对种群中优势个体和劣势个体分别进行更新,当算法陷入停滞时,使用布朗运动策略,帮助算法跳出停滞,最后利用贪婪策略保留优势个体,加快算法收敛效率。改进算法流程见图 1。

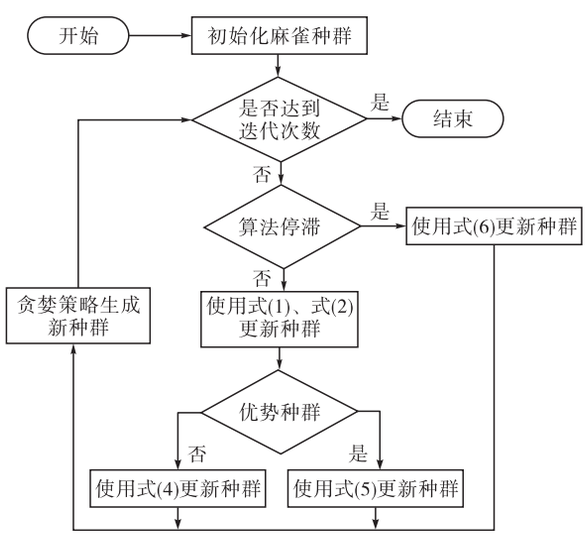

图 1 CSSA-HB 流程图

3 仿真实验

为确定使用哪种混沌映射调整 SSA 参数,将 SSA 与各混沌映射结合,与第 1 种混沌映射结合的算法命名为 SSA-1,与第 2 种混沌映射结合的算法命名为 SSA-2,以此类推,将 SSA 与上述 10 种混沌映射分别与 SSA 结合的算法在 12 种测试函数中进行比较,测试函数如表 2 所示。为公平比较,在相同实验平台上,设置种群数为 50,最大迭代数为 300,算法参数与原文献保持一致。所有算法均使用 MATLAB R2018b 编程,计算机操作系统为 Windows10,处理器为 AMD R7 4700 U 16 GB。表 3 为

各算法独立运行 30 次的统计结果。

表 2　测试函数

测试函数	维度	范围	最优值
$F_1(x)=\sum_{i=1}^{D}x_i^2$	30	$[-100,100]$	0
$F_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	30	$[-10,10]$	0
$F_3(x)=\sum_{i=1}^{D}\left(\sum_{j-1}^{D}x_i\right)^2$	30	$[-100,100]$	0
$F_4(x)=\max_i\{\lvert x_i\rvert,1\leqslant i\leqslant D\}$	30	$[-100,100]$	0
$F_5(x)=\sum_{i=1}^{D}100(x_{i+1}^2-x_i^2)^2+(x_i-1)^2$	30	$[-30,30]$	0
$F_6(x)=20+e-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$	30	$[-32,32]$	8.881 8E−16
$F_7(x)=\frac{1}{4\ 000}\sum_{i=1}^{D}(x_i^2)-\left(\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)\right)+1$	30	$[-600,600]$	0
$F_8(x)=\frac{\pi}{D}\left\{10\sin^2(\pi y_i)+\sum_{i-1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)\right\}+\sum_{i=1}^{D}u(x_i,10,100,4)\quad y_i=1+\frac{x_i+1}{4}u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a<x_i<a\\ k(-x_i-a)^m, & x_i<a\end{cases}$	30	$[-50,50]$	0
$F_9(x)=0.1\left\{\begin{matrix}\sin^2(3\pi x_i)+\sum_{i=1}^{D}(x_i-1)^2[1+\sin^2(3\pi x_i)]+\\ (x_D-1)^2[1+\sin^2(2\pi x_D)]\end{matrix}\right\}+\sum_{i-1}^{D}u(x_i,5,100,4)$	30	$[-50,50]$	0
$F_{10}(x)=\sum_{i=1}^{11}\left(a_i-\frac{x_1(b_i^2+b_ix_2)^{-1}}{b_i^2+b_ix_3+x_4}\right.$	4	$[-5,5]$	0.000 307 5
$F_{11}(x)=4x_i^2-2.1x_1^4+1/3x_1^6+x_1x_2-4x_2^2+x_2^4$	2	$[-5,5]$	−1.031 63
$F_{12}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^{\mathrm{T}}+c_i]^{-1}$	4	$[0,10]$	−10.402 9

表 3　11 种混沌映射组合算法计算平均值比较

函数	SSA	SSA-1	SSA-2	SSA-3	SSA-4	SSA-5	SSA-6	SSA-7	SSA-8	SSA-9	SSA-10
F_1	4.53E−144	3.02E−119	5.02E−160	7.70E−156	7.44E−170	3.34E−103	9.18E−191	1.78E−94	1.92E−191	3.69E−130	8.88E−129
F_2	9.49E−68	1.01E−53	1.24E−82	8.11E−83	1.45E−94	3.68E−63	3.10E−73	1.86E−47	1.09E−90	2.88E−58	8.26E−61
F_3	3.47E−99	3.50E−85	3.58E−98	1.05E−110	4.29E−128	5.14E−83	6.94E−107	2.86E−80	1.95E−135	9.82E−91	9.53E−89
F_4	4.47E−68	2.01E−47	7.39E−80	8.66E−79	3.27E−82	5.26E−52	1.49E−88	1.23E−52	2.51E−88	1.30E−68	2.36E−64
F_5	1.25E−04	7.54E−05	2.00E−04	1.35E−04	2.72E−04	9.10E−05	1.45E−04	1.83E−04	2.61E−04	1.14E−04	1.44E−04
F_6	8.88E−16	5.92E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16	8.88E−16
F_7	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_8	2.98E−09	1.63E−09	4.28E−09	1.17E−09	2.44E−09	4.34E−09	6.79E−09	2.93E−09	9.30E−09	4.71E−09	3.00E−09
F_9	4.44E−08	1.01E−08	1.88E−08	4.68E−08	2.68E−08	1.44E−08	7.26E−08	7.60E−08	3.38E−08	7.01E−08	4.08E−08
F_{10}	3.08E−04	2.36E−04	3.08E−04	3.08E−04	3.08E−04	3.08E−04	3.09E−04	3.08E−04	3.09E−04	3.08E−04	3.08E−04
F_{11}	−1.03E+00	−6.88E−01	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00
F_{12}	−9.45E+00	−6.69E+00	−9.09E+00	−7.72E+00	−8.55E+00	−9.45E+00	−8.92E+00	−1.02E+01	−8.91E+00	−9.09E+00	−9.34E+00

从表 3 可以得知,对于 F_1、F_2,有 5 种映射结合算法优于 SSA 算法;对于 F_3、F_8,有 4 种映射结合算法优于 SSA 算法;对于 F_4、F_9,有 6 种映射结合算法优于 SSA 算法;对于 F_5,有 3 种映射结合算法优于 SSA 算法;对于 F_6、F_7、F_{11},所有算法性能相近;对于 F_{10},则有 7 种映射结合算法优于 SSA 算法;而对于 F_{12},仅有 2 种映射结合算法优于 SSA 算法。其中,SSA-4 算法在 7 个测试函数中优于原算法,在 3 个测试函数中与原算法性能差异不大,仅在 1 个测试函数中表现劣于 SSA 算法;SSA-3 在 6 个测试函数中优于原算法,在 3 个测试函数中与原算法性能相近,同样仅在 1 个测试函数中表现劣于 SSA 算法。

为了进一步分析混沌映射对于 SSA 算法的改进能力,根据表 3 的平均值对各算法进行比较排序,结果如表 4 所示,最后一栏为各算法平均排序结果。可以得知,其中 4 种映射结合算法表现优于 SSA 算法,SSA-4 排序第 1,寻优性能在 11 种算法中最强,SSA-3 次之,其余算法排名为:SSA-2 和 SSA-8 并列,SSA、SSA-5 和 SSA-6 并列,SSA-10、SSA-9、SSA-1。结合以上分析,混沌映射对于 SSA 算法的性能具有促进作用,且第 4 种混沌映射表现最佳,因此在后文对改进算法进行性能测试时,使用该映射进行参数调整。

为了充分验证 CSSA-HB 算法的有效性与优越性,选择 WOA[3]、GWO[4]、BSO[15]、PSO[14]、FPA[15]以及传统 SSA 算法进行对比分析,参数同前,表 5 为各算法独立运行 30 次的统计结果。最优值加粗体表示。

表 4 11 种混沌映射组合算法排序结果

测试函数	SSA	SSA-1	SSA-2	SSA-3	SSA-4	SSA-5	SSA-6	SSA-7	SSA-8	SSA-9	SSA-10
F_1	6	9	4	5	3	10	2	11	1	7	8
F_2	6	10	4	3	1	7	5	11	2	9	8
F_3	5	9	6	3	2	10	4	11	1	7	8
F_4	7	11	4	5	3	10	1	9	2	6	8
F_5	4	1	8	5	11	2	6	7	9	3	5
F_6	1	11	1	1	1	1	1	1	1	1	1
F_7	1	1	1	1	1	1	1	1	1	1	1
F_8	5	2	7	1	3	8	10	4	11	9	6
F_9	7	1	3	8	4	2	10	11	5	9	6
F_{10}	8	11	7	1	2	4	10	3	9	6	5
F_{11}	1	11	1	1	1	1	1	1	1	1	1
F_{12}	3	11	5	10	9	2	7	1	8	6	4
平均排名	4.5	7.33	4.25	3.66	3.41	4.83	4.83	5.91	4.25	5.41	5.08

表 5 7 种算法平均值比较

测试函数	WOA	BSO	PSO	SSA	GWO	FPA	CSSA-HB
F_1	5.67E−43	7.24E+02	2.07E−02	1.64E−120	2.41E−18	4.31E+03	**1.00E−163**
F_2	2.04E−29	6.97E+00	1.96E−01	1.01E−62	2.03E−11	5.48E+01	**5.03E−91**
F_3	4.58E+04	5.81E+05	7.88E+02	3.39E−89	1.56E−03	4.19E+03	**2.92E−133**
F_4	5.59E+01	1.14E+01	4.44E+00	9.00E−88	1.07E−04	3.44E+01	**1.55E−93**
F_5	2.85E+01	2.42E+03	1.18E+02	1.95E−04	2.70E+01	1.51E+06	**6.02E−05**
F_6	6.45E−15	4.92E+00	1.62E+00	**8.88E−16**	1.88E−10	1.26E+01	**8.88E−16**
F_7	3.70E−18	2.22E+02	6.89E−02	**0.00E+00**	7.41E−03	3.93E+01	**0.00E+00**
F_8	9.82E−02	1.79E+00	7.85E−01	6.73E−08	3.14E−02	6.22E+04	**4.79E−08**
F_9	1.23E+00	2.23E+01	3.27E−01	6.01E−07	4.73E−01	1.44E+06	**2.86E−07**
F_{10}	1.30E−03	2.44E−03	6.09E−04	**3.08E−04**	3.10E−03	8.32E−04	3.48E−04
F_{11}	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**
F_{12}	−6.42E+00	−1.02E+01	−7.93E+00	−9.52E+00	−9.79E+00	−9.97E+00	**−1.00E+01**

分析表 5 可知,对于单峰测试函数 $F_1 \sim F_5$,CSSA-HB 在 7 种算法中表现最佳,寻优精度和寻优稳定性较其他算法有较大提升,对于多峰测试函数 $F_6 \sim F_9$,CSSA-HB 在 F_6 和 F_7 中表现与原算法相近,但优于其他对比算法,在 F_8 和 F_9 中,CSSA-HB 性能优于所有对比算法,对于固定维度测试函数 $F_{10} \sim F_{12}$,CSSA-HB 在 F_{10} 中寻优效果弱于 SSA,但差异不大,在 F_{11} 中,各算法性能相近,在 F_{12} 中,CS-

SA-HB 优于所有对比算法。因此，本文提出的 CSSA-HB 算法在其中 8 个测试函数中寻优性能最佳，在 3 个测试函数中与 SSA 性能相近，优于其余对比算法，仅在 1 个测试函数中表现差于 SSA，证明 CSSA-HB 算法改进的有效性。

为了进一步验证 CSSA-HB 算法的寻优性能，根据表 5 的均值对各算法进行排序，结果如表 6 所示。

表 6　7 种算法性能排序结果

测试函数	WOA	BSO	PSO	SSA	GWO	FPA	CSSA-HB
F_1	3	6	5	2	4	7	1
F_2	3	6	5	2	4	7	1
F_3	7	5	4	2	3	6	1
F_4	7	5	4	1	3	6	2
F_5	3	6	5	2	4	7	1
F_6	3	6	5	2	4	7	1
F_7	3	7	5	1	4	6	1
F_8	4	6	5	2	3	7	1
F_9	5	6	3	2	4	7	1
F_{10}	5	4	6	1	2	7	3
F_{11}	1	1	1	1	1	1	1
F_{12}	7	2	6	5	4	3	1
平均排序	4.25	5	4.5	1.92	3.33	5.9	1.25

CSSA-HB 排序第一，说明 CSSA-HB 在求解测试函数时寻优性能最强，其余算法排名为：SSA、GWO、WOA、PSO、BSO、FPA。为更加直观显示 7 种算法在不同测试函数上的排序结果，采用雷达图将表 6 的排序结果进行绘图，如图 2 所示。图中算法性能曲线所围面积越小，表明算法的排序越小，其性能就越好。黑色加粗曲线即为 CSSA-HB 算法排序结果曲线，可以直观地看到，CSSA-HB 在 F_{10} 和 F_{11} 上性能相对较差，在其他测试函数中均表现较好，且其所围面积最小，说明 CSSA-HB 整体上具有最好的寻优性能。

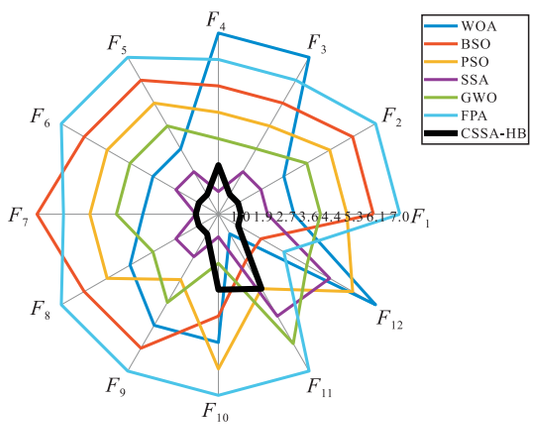

图 2　7 种算法性能雷达图

图 3 为 7 种算法独立求解 12 个基准测试函数 30 次所得结果的箱式图，从图中可以得知，在进行求解时，CSSA-HB 求得的异常点均少于对比算法，且在求解所有测试函数时，收敛值的分布相比其它对比算法整体上更为集中，明显优于其他对比算法，说明改进的 CSSA-HB 算法具有较强的鲁棒性。

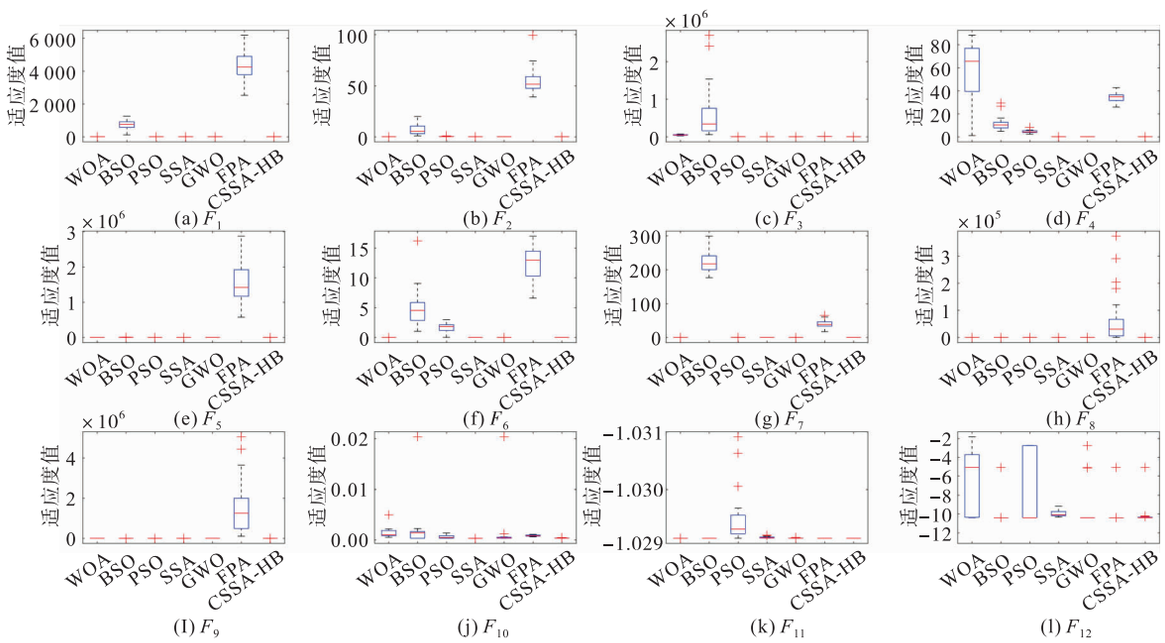

图 3　7 种算法收敛箱式图

为了进一步阐述 CSSA-HB 的收敛性能，7 种算法独立运行 30 次求解 12 个基准测试函数收敛曲线如图 4 所示。在求解 F_1～F_5、F_7、F_9 和 F_{10} 时，CSSA-HB 有更快的收敛速度和收敛精度，在求解 F_6 和 F_8 时，CSSA-HB 收敛速度在前期弱于 SSA，但在牺牲一定的收敛速度的情况下，能够在后期更

快收敛到全局最优值,且收敛精度优于所有对比算法,在求解 F_{12} 时,PSO 算法性能最佳,但 CSSA-HB 能在后期收敛到全局最优值。因此 CSSA-HB 相比 SSA,其寻优性能具有明显提升,具有较强局部最优规避能力和更高的收敛精度与收敛速度。

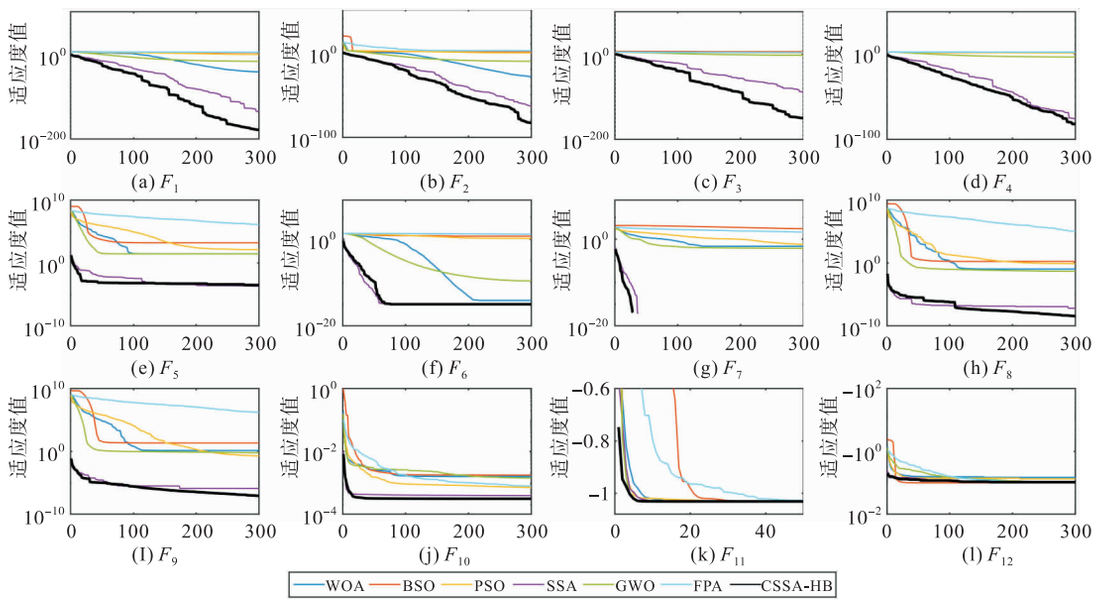

图 4 7 种算法收敛曲线图

算法运行时间也是衡量算法性能的重要指标,表 7 列出了各算法求解测试函数的平均计算耗时,由表 7 可知,CSSA-HB 的计算耗时平均排名排在最后,这是由于每一次迭代中都会使用混沌映射,以及在更新警戒者位置时,使用了高斯概率密度函数,同时 SSA 排在第 4,而改进策略的加入进一步加大了计算耗时。另一方面,由前文的收敛性分析可知,相较对比算法,CSSA-HB 的收敛速度更快,收敛精度更高,因此增加的计算耗时换来了更好的巡游精度,这是可以接受的。

表 7 7 种算法计算耗时比较

测试函数	WOA	BSO	PSO	SSA	GWO	FPA	CSSA-HB
F_1	3.85E−02	1.35E−01	9.78E−02	1.18E−01	8.41E−02	1.46E−01	2.50E−01
F_2	4.26E−02	1.54E−01	1.04E−01	1.27E−01	9.16E−02	1.59E−01	2.62E−01
F_3	3.36E−01	1.05E+00	4.13E−01	4.77E−01	3.90E−01	4.57E−01	5.79E−01
F_4	3.31E−02	1.27E−01	9.62E−02	1.15E−01	8.35E−02	1.56E−01	2.44E−01
F_5	4.66E−02	1.71E−01	1.12E−01	1.30E−01	9.72E−02	1.69E−01	2.58E−01
F_6	4.81E−02	1.79E−01	1.17E−01	1.33E−01	9.74E−02	1.84E−01	2.62E−01
F_7	5.79E−02	2.26E−01	1.24E−01	1.45E−01	1.08E−01	1.85E−01	2.76E−01
F_8	1.97E−01	6.17E−01	2.61E−01	3.16E−01	2.47E−01	3.22E−01	4.18E−01
F_9	1.96E−01	6.21E−01	2.63E−01	3.14E−01	2.47E−01	3.23E−01	4.31E−01
F_{10}	2.90E−02	1.35E−01	9.13E−02	1.15E−01	5.15E−02	1.46E−01	1.64E−01
F_{11}	1.81E−02	9.46E−02	7.99E−02	9.80E−02	3.54E−02	1.26E−01	1.34E−01
F_{12}	1.41E−01	4.72E−01	2.06E−01	2.46E−01	1.66E−01	2.50E−01	2.77E−01
平均排名	1.00	5.75	3.00	4.17	2.00	5.42	6.67

为进一步体现本文提出的 CSSA-HB 的有效性,选取几个具有代表的测试函数与文献[8]提出的改进算法在同一条件下进行对比。仿真结果如表 8 所示。由表 8 可知,对于单峰测试函数 F_1 和 F_3,CSSA-HB 在平均值和标准差上相较于 CSSA 至少提升了 20 个数量级,提升效果明显,对于 F_2 和 F_4,CSSA-HB 至少提升了 10 个数量级,对于 F_5,相较于 CSSA,CSSA-HB 能够更稳定地求解最优值。对于多峰测试函数 F_6～F_{12},两种算法各有优劣,对于 F_7-F_9,两种算法性能相似,CSSA-HB 在 F_{11} 上效果

好于 CSSA，CSSA 则在 F_6、F_{10}、F_{12} 上效果更好。总体来说，CSSA-HB 在 12 个测试函数中的 9 个测试函数上的效果不差于 CSSA，表明 CSSA-HB 的性能更好，再一次验证了本文改进算法的有效性。

表 8　CSSA-HB 与 CSSA 算法性能比较

类型	函数	CSSA-HB		CSSA[8]	
		平均值	标准差	平均值	标准差
单峰函数	F_1	1.75E−98	5.53E−98	6.19E−78	3.20E−77
	F_2	2.50E−55	7.91E−55	1.75E−40	4.16E−40
	F_3	1.17E−99	3.58E−99	3.29E−65	1.28E−64
	F_4	1.04E−49	3.27E−49	5.19E−39	2.25E−38
	F_5	4.33E−04	2.30E−04	7.23E−04	6.38E−04
多峰函数	F_6	−7.33E+03	3.82E+03	−1.11E+04	7.13E+02
	F_7	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	F_8	8.88E−16	0.00E+00	8.88E−16	0.00E+00
	F_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	F_{10}	4.32E+00	4.56E+00	1.16E+00	5.27E−01
	F_{11}	−3.31E+00	3.33E−02	−3.31E+00	4.11E−02
	F_{12}	−8.91E+00	2.61E+00	−1.05E+01	1.28E−05

4　结语

本文首先探讨混沌映射调整麻雀搜索算法参数对于麻雀搜索算法性能的影响，然后引入等级制度和布朗运动策略，提出了基于等级制度和布朗运动的混沌麻雀搜索算法，通过 12 个测试函数验证，结果表明使用迭代映射调整麻雀搜索算法参数效果最佳；利用 6 个对比算法和改进算法以及原始算法进行比较，证明了本文提出的改进算法寻优性能具有明显提升，具有较强局部最优规避能力和更高的收敛精度与收敛速度。

本文对于麻雀搜索算法的改进主要集中在搜索算子的改进，这些改进策略不仅可以应用于麻雀搜索算法，也可以应用于其他智能优化算法的研究，但改进策略的适用性需进一步验证。同时，机器学习是近些年优化领域的研究热点，可以将麻雀搜索算法与机器学习方法进行结合。此外，随着工业生产需求增大，复杂的现实优化问题对于智能优化算法的要求增多，对于麻雀搜索算法的改进需适应不同问题的特性。尤其是对于实时性要求较高的工程优化问题，需要在算法的精度和速度做更多考虑。

参考文献

[1] HOLLAND J H. Genetic Algorithms[J]. Scientific American, 1992,267:66-73.

[2] POLI R, KENNEDY J, BLACKWELL T. Particle Swarm Optimization[J]. Swarm Intelligence, 2007, 1(1):33-57

[3] MIRJALILI S, LEWIS A. The Whale Optimization Algorithm[J]. Advances in Engineering Software, 2016, 95(5):51-67.

[4] MIRJALILI S, MIRJALILI S M, LEWIS A. Grey Wolf Optimizer[J]. Advances in Engineering Software, 2014, 69:46-61.

[5] XUE J, SHEN B. A Novel Swarm Intelligence Optimization Approach: Sparrow Search Algorithm[J]. Systems Science & Control Engineering, 2020, 8(1): 22-34.

[6] 杨万里，周雪婷，陈孟娜. 基于 Logistic 映射的新型混沌简化 PSO 算法[J]. 计算机与现代化，2019(12):15-20+26.

[7] IBRAHIM R A, ELAZIZ M A, LU S F. Chaotic Opposition-Based Grey-Wolf Optimization Algorithm Based on Differential Evolution and Disruption Operator for Global Optimization[J]. Expert Systems with Application, 2018, 108:1-27.

[8] 吕鑫，慕晓冬，张钧，等. 混沌麻雀搜索优化算法[J/OL]. 北京航空航天大学学报:1-10.[2020-10-14]. https://doi.org/10.13700/j.bh.1001-5965.2020.0298.

[9] WANG G G, GUO L, GANDOMI A H, et al. Chaotic Krill Herd Algorithm[J]. Information science, 2014, 274:17-34.

[10] 黄辉先，张广炎，陈思溢，等. 基于混沌权重和精英引导的鲸鱼优化算法[J]. 传感器与微系统，2020，39(5):113-116.

[11] 匡芳君，徐蔚鸿，金忠. 自适应 Tent 混沌搜索的人工蜂群算法[J]. 控制理论与应用，2014，31(11):1502-1509.

[12] 张达敏，徐航，王依柔，等. 嵌入 Circle 映射和逐维小孔成像反向学习的鲸鱼优化算法[J/OL]. 控制与决策:1-8[2020-10-14]. https://doi.org/10.13195/j.kzyjc.2019.1362.

[13] WOOLARD E W, EINSTEIN A, FURTH R, et al. Investigations on the Theory of the Brownian Movement[J]. 1957, 41(337):231.

[14] WANG T T, YANG L, LIU Q. Beetle Swarm Optimization Algorithm: Theory and Application[J]. Arxiv, 2018:1808.00206.

[15] YANG X S. Flower Pollination Algorithm[J]. Unconventional Concputation and Natural. Camputation, 2012, 7445:240-249.

（编辑：徐敏）