第 13 卷第 5 期
2018 年 10 月
智 能 系 统 学 报
CAAI Transactions on Intelligent Systems
Vol.13 No.5
Oct. 2018

DOI: 10.11992/tis.201705017
网络出版地址: http://kns.cnki.net/kcms/detail/23.1538.TP.20170728.1901.006.html

基于混沌搜索和权重学习的教与学优化算法及其应用

柳缔西子[1], 范勤勤[1,2], 胡志华[1]
（1. 上海海事大学 物流研究中心，上海 201306; 2. 华东理工大学 化工过程先进控制和优化技术教育部重点实验室，上海 200237）

摘　要: 针对教与学优化算法容易陷入早熟收敛的问题，本研究提出了一种基于混沌搜索和权重学习的教与学优化 (teaching-learning-based optimization algorithm based on chaotic search and weighted learning, TLBO-CSWL) 算法。在 TLBO-CSWL 算法的教学阶段，不仅利用权重学习得到的个体来指引种群的进化，而且还使用正态分布随机数来替代原有的均匀随机数。另外，TLBO-CSWL 还使用 Logistics 混沌搜索策略来提高其全局搜索能力。仿真结果表明，TLBO-CSWL 的整体优化性能要好于其他所比较的算法。最后，将 TLBO-CSWL 用于求解非合作博弈纳什均衡问题，获得满意的结果。
关键词: 教与学优化; 权重学习; 启发式算法; 混沌搜索; 全局优化; 进化计算; 非合作博弈; 纳什均衡
中图分类号: TP301.6　　**文献标志码**: A　　**文章编号**: 1673−4785(2018)05−0818−11

中文引用格式: 柳缔西子, 范勤勤, 胡志华. 基于混沌搜索和权重学习的教与学优化算法及其应用[J]. 智能系统学报, 2018, 13(5): 818–828.
英文引用格式: LIU Dixizi, FAN Qinqin, HU Zhihua. Teaching-learning-based optimization algorithm based on chaotic search and weighted learning and its application[J]. CAAI transactions on intelligent systems, 2018, 13(5): 818–828.

Teaching-learning-based optimization algorithm based on chaotic search and weighted learning and its application

LIU Dixizi[1], FAN Qinqin[1,2], HU Zhihua[1]
(1. Logistics Research Center, Shanghai Maritime University, Shanghai 201306, China; 2. MOE Key Laboratory of Advanced Control and Optimization for Chemical Processes, East China University of Science and Technology, Shanghai 200237, China)

Abstract: To avoid premature convergence, a teaching-learning-based optimization algorithm based on chaotic search and weighted learning (TLBO-CSWL) is introduced in this study. In the teaching phase, TLBO-CSWL does not only use the individuals obtained by weight learning to guide the population evolution, it also utilizes a normal random number to replace the original uniform random number. In addition, TLBO-CSWL uses a logistics chaotic search strategy to improve its global search ability. Simulation results showed that TLBO-CSWL outperformed other compared algorithms in terms of overall performance. Finally, the proposed algorithm was employed to solve two Nash equilibrium problems of non-cooperative game, and satisfactory results were obtained.
Keywords: teaching-learning-based optimization; weight learning; heuristic algorithm; chaotic search; global optimization; evolutionary computation; non-cooperative game; Nash equilibrium

近几十年来，博弈论得到了许多研究人员的关注，逐渐成为了经济学中的标准分析工具之一，并且在经济学、军事科学和其他社会科学中得到了广泛的应用。在博弈问题中，非合作博弈纳什均衡是其最核心的研究内容。1951 年，Nash[1] 在提出的“纳什定理”中揭示并证明了纳什均衡

收稿日期: 2017−05−15.　**网络出版日期**: 2017−07−28.
基金项目: 国家自然科学基金项目 (611603244); 中央高校基本科研业务费重点科研基地创新基金项目 (222201717006); 上海海事大学研究生创新基金资助项目 (2017YCX020).
通信作者: 范勤勤. E-mail: forever123fan@163.com.

解的存在, 但是 Nash 并没有给出求解纳什均衡的一般性方法。传统的求解方法如 Lemke-Howson 算法[2]、牛顿算法[3]、同伦算法[4]等均存在一定的局限性; 特别是对于高维的博弈模型 (如 3 维及以上的矩阵策略), 传统算法的计算复杂求解成本较高。除了传统的优化方法, 许多学者还利用遗传算法[5]、免疫算法[6]、粒子群算法[7]、蚁群算法[8]等启发式算法来求解博弈纳什均衡问题, 这为求解非合作博弈问题提供了一种新的有效途径和方法。

教与学优化算法 (teaching-learning-based optimization, TLBO) 是 Rao 等[9-10]于 2011 年提出, 它是一种模拟教师教学与学生学习的群体智能优化算法。由于该算法参数设置少、操作简便、寻优性能好, 引起了国内外学者的重视。目前它已被成功应用于各个领域, 如 LQR 控制器优化设计[11]、热交换器优化设计[12]、人工神经网络优化[13]等。

一般来说, 原始的 TLBO 算法容易出现"早熟收敛"的现象, 从而易于陷入局部最优。为了提高 TLBO 算法的寻优性能, 很多学者提出了改进算法。比如, Rao 等[14]提出了 ETLBO 算法, 该算法将精英策略引入 TLBO 算法中, 保留每代中的最优解, 并随机对精英个体进行变异操作, 其主要目的是提高算法的收敛速度和寻优精度; Yu 等[15]提出了 ITLBO 算法, 该算法在 TLBO 算法中引入教学反馈阶段和差分算法中的交叉变异策略, 并在算法后期加入混沌扰动机制; Zou 等[16]提出了 DGSTLBO 算法, 该算法在教学阶段引入动态分组教学策略, 在学习阶段加入量子行为策略, 以增加种群的多样性和避免算法早熟收敛; Chen 等[17]提出了 VTTLBO 算法, 该算法将种群的数量以先增后减的方式来进行动态调整。在种群数量增加阶段利用高斯分布生成个体, 并在减少阶段进行相同个体的去重操作, 其主要目的是减少计算成本以及增强算法的收敛精度和速度。Wu 等[18]提出了 NIWTLBO 算法, 在该算法中, 利用非线性惯性权重因子来控制个体的学习速率, 同时使用动态惯性权重因子代替原有随机数, 仿真结果表明此改进策略提高了算法的收敛速率和寻优性能。Shahbeig 等[19]提出了 TLBO-PSO 算法, 该算法将改进的变异模糊自适应的 PSO 算法与 TLBO 算法进行结合, 目的在于提高算法的寻优精度从而解决多目标优化问题。上述研究结果均表明, 将改进策略引入 TLBO 算法中可以提高算法的寻优性能。

为了能更进一步提高 TLBO 的搜索性能, 本文提出一种基于混沌搜索和权重学习的教与学优化 (TLBO-CSWL) 算法。仿真结果表明, 所提算法的整体优化性能明显优于其他所比较的算法。最后, 将改进的算法应用于非合作博弈纳什均衡问题的求解。

1　预备知识

1.1　标准的教与学优化算法

标准 TLBO 算法[9-10]主要包括两个阶段, 教学阶段与学习阶段。

1.1.1　教学阶段

$\boldsymbol{X}_i=(x_{i,1},x_{i,2},\cdots,x_{i,D}), i=1,2,\cdots,\mathrm{NP}$, 表示第 i 个个体, 其中, D 为优化问题的维数, NP 表示种群规模。种群的平均值表示为$\boldsymbol{X}_{\rm mean}=\sum\limits_{i=1}^{\rm NP}\boldsymbol{X}_i/\mathrm{NP}$, 其中当前适应度值最好的个体被选为教师, 即 $\boldsymbol{X}_{\rm teacher}$。在教学阶段, 每个个体基于教师与种群的平均值之差进行学习。新的个体更新公式为

$$\boldsymbol{X}_{i,\rm new}=\boldsymbol{X}_{i,\rm old}+r\cdot(\boldsymbol{X}_{\rm teacher}-T_F\cdot\boldsymbol{X}_{\rm mean}) \tag{1}$$

式中: $\boldsymbol{X}_{i,\ \rm old}$ 表示第 i 个个体学习之前的个体; r 为[0, 1]之间的均匀随机数, 表示学习步长; T_F= round[1+rand(0, 1)], 取值 1 或 2, 表示教学因子。若 $\boldsymbol{X}_{i,\ \rm new}$ 的适应度值优于 $\boldsymbol{X}_{i,\ \rm old}$, 则更新个体; 否则, 不更新。

1.1.2　学习阶段

在 TLBO 算法的学习阶段中, 个体的更新公式为

$$\begin{cases}\boldsymbol{X}_{p,\rm new}=\boldsymbol{X}_{p,\rm old}+r\cdot(\boldsymbol{X}_p-\boldsymbol{X}_l), & f(\boldsymbol{X}_p)<f(\boldsymbol{X}_l)\\ \boldsymbol{X}_{p,\rm new}=\boldsymbol{X}_{p,\rm old}+r\cdot(\boldsymbol{X}_l-\boldsymbol{X}_p), & f(\boldsymbol{X}_l)<f(\boldsymbol{X}_p)\end{cases} \tag{2}$$

式中: $p\neq l$, p, $l\in(1,2,\cdots,\mathrm{NP})$; $f(\boldsymbol{X})$ 为适应度函数。若 $\boldsymbol{X}_{p,\ \rm new}$ 的适应度值优于 $\boldsymbol{X}_{p,\ \rm old}$, 则更新个体; 否则, 不更新。

1.2　混沌映射

混沌是指确定性动力学系统产生的一种不可预测的、类似随机性的运动, 最显著的特点是初值敏感性。混沌模型不仅具有随机性、初值敏感性, 同时还有一个很重要的性质是遍历性。由于混沌序列是遍历的, 它已被越来越多地应用到智能优化算法中。其主要被用来初始化种群[20], 以及能够对个体进行随机次数的扰动使其跳出局部最优[21], 从而在个体周围进行遍历搜索。借鉴文献[21]中的混沌扰动策略, 所提算法利用 Logistics 搜索策略来对个体进行更新, 公式为

$$x_{\text{new}} = x_{\text{old}} \cdot \mu \cdot (1 - x_{\text{old}}) \tag{3}$$

式中：x_{old} 和 x_{new} 分别表示混沌映射之前和混沌映射之后的变量，$x \in [0, 1]$；$\mu \in [0, 4]$为控制参数，当 μ=4 时，Logistics 映射将处于完全混沌状态。

2 基于混沌搜索和权重学习的教与学优化算法

2.1 权重学习

在原始的 TLBO 算法中，教学阶段主要是使用当前最佳个体来指导种群进化，这将会造成算法陷入局部最优。因此，本文提出一种权重学习的策略，基于个体适应度值产生一个可以代表种群适应度水平的综合个体 $\boldsymbol{X}_{\text{weight}}$，并且引导其他个体向其学习。这可以缓解算法“早熟”现象的发生。

1) 计算种群的最大适应度值及每个个体的权重

$$f_{\max} = \max(f(\boldsymbol{X}_i)) \tag{4}$$

$$\boldsymbol{w}_i = \frac{|f(\boldsymbol{X}_i) - f_{\max}|}{\sum_{i=1}^{\text{NP}} |f(\boldsymbol{X}_i) - f_{\max}|} \tag{5}$$

2) 计算加权平均个体

$$\boldsymbol{x}_{\text{weight},j} = \sum_{i=1}^{\text{NP}} \boldsymbol{w}_i \times \boldsymbol{x}_{i,j}, \quad j = 1, 2, \cdots, D \tag{6}$$

3) 改进后的教学阶段更新公式为

$$\begin{aligned} \boldsymbol{X}_{i,\text{new}} = & \boldsymbol{X}_{i,\text{old}} + r \cdot (\boldsymbol{X}_{\text{teacher}} - T_F \cdot \boldsymbol{X}_{\text{mean}}) + \\ & r \cdot (\boldsymbol{X}_{\text{weight}} - \boldsymbol{X}_{i,\text{old}}) \end{aligned} \tag{7}$$

式中 $r = N(0.5, 0.2)$。若 $\boldsymbol{X}_{i,\text{ new}}$ 的适应度值优于 $\boldsymbol{X}_{i,\text{ old}}$，则更新个体；否则，不更新。

2.2 混沌搜索

为了提高 TLBO 算法的全局搜索能力，将混沌搜索策略加入到该算法中。混沌搜索的执行步骤如下：

1) 对种群中的所有个体进行适应度值降序排列 (最小化问题)；

2) 随机取出一个排名前 10 的个体；

3) 利用式 (3) 对选择的个体进行混沌扰动，产生混沌个体 $\boldsymbol{X}_{\text{chaos}}$；

4) 若 $\boldsymbol{X}_{\text{chaos}}$ 的适应度值优于 $\boldsymbol{X}_i$，则更新个体；否则，不更新。

2.3 TLBO-CSWL 算法的实现步骤

1) 初始化：设定种群大小 NP，维数 D，最大评价次数，初始化种群。

2) 教学阶段：根据式 (7) 对个体进行更新。

3) 学习阶段：利用正态分布随机数代替式 (2) 的均匀随机数，然后根据式 (2) 对个体进行更新。

4) 混沌搜索：利用 2.2 部分随机对个体进行混沌扰动操作，并更新个体。

5) 判定程序是否达到最大评价次数，若没有达到，则转至 2)；如达到，则执行 6)。

6) 输出最优解。

3 仿真测试

为验证 TLBO-CSWL 算法的有效性，本文选取了文献[17]中的 18 个测试函数，其中 f_1~f_5 为单峰函数，f_6~f_{10} 为多峰函数，f_{11}~f_{18} 为旋转函数。改进算法分别与 jDE[22]、SaDE[23]、PSOwFIPS[24]、CLPSO[25]、TLBO[9-10]、ETLBO[14]、VTTLBO[17]等算法进行对比。根据文献[17]的设定，最大的函数评价次数均为 50 000。对于每个测试函数，所有算法均独立运行 30 次。为了保证结论的可靠性，采用 Friedman[26]、Dunn[27]、Holm 和 Hochberg[28]检验来对结果进行统计分析，其中，显著水平设定为 5%。

3.1 TLBO-CSWL 算法与其他算法的比较

3.1.1 与其他算法在 10 维测试上的比较

在该实验中，所有算法的种群规模设定为 30，仿真结果如表 1 所示。由表 1 可知，TLBO-CSWL 在函数 f_1、f_2、f_3、f_4、f_6、f_{11}、f_{12}、f_{14}、f_{15}、f_{17} 上均有很好的寻优效果，性能明显优于其他所比较算法。但在函数 f_5、f_{10}、f_{13}、f_{18} 上，所有其他算法的寻优性能都略优于 TLBO-CSWL 算法，其主要原因有两个方面：1) 虽然所提算法使用正态分布和权重学习来提高原始 TLBO 的搜索效率，但在某种程度上却降低了算法的全局搜索能力；2) 每种算法都有自身的寻优特性，到目前为止，没有一种算法能够在所有的测试函数上都能表现得最好，因此，所提算法在某些测试函数上表现的差，也符合没有免费午餐定理[29]。而对于其余测试函数，TLBO-CSWL 所获得的结果与所比较算法中获得的最好结果相同。同时，利用非参数的统计方法来对实验结果进行分析，所得结果见表 2、3 所示。从表 2 可知，TLBO-CSWL 算法的整体性能是最好的。从表 3 可以看出，所提算法的整体性能要显著好于 PSOwFIPS 算法和 CLPSO 算法。另外，虽然 TLBO-CSWL 的整体性能在统计意义上没有显著好于 SaDE、ETLBO、TLBO、jDE、VTTLBO，但是从结果来看，所提算法的整体性能优于其他算法。

表 1　10 维仿真测试结果

Table 1　Experimental results on 10D

函数	结果	jDE	SaDE	PSOwFIPS	CLPSO	TLBO	ETLBO	VTTLBO	TLBO-CSWL
f_1	平均值	1.31×10^{-76}	1.35×10^{-71}	3.98×10^{-16}	1.09×10^{-18}	3.29×10^{-184}	2.84×10^{-1661}	3.56×10^{-269}	0.00
	标准差	1.58×10^{-76}	2.02×10^{-71}	6.09×10^{-16}	1.50×10^{-18}	3.08×10^{-185}	4.27×10^{-167}	0.00	0.00
f_2	平均值	1.14×10^{-21}	1.89×10^{-19}	6.19×10^{-6}	5.37×10^{-1}	2.56×10^{-82}	3.22×10^{-79}	3.50×10^{-130}	0.00
	标准差	1.52×10^{-21}	3.54×10^{-19}	2.15×10^{-6}	1.38×10^{-1}	5.58×10^{-82}	5.07×10^{-79}	5.05×10^{-130}	0.00
f_3	平均值	6.97×10^{-78}	1.28×10^{-74}	2.18×10^{-17}	2.59×10^{-20}	9.94×10^{-187}	6.50×10^{-169}	4.63×10^{-1}	0.00
	标准差	1.39×10^{-31}	2.52×10^{-74}	1.39×10^{-17}	1.76×10^{-20}	1×10^{-187}	5.49×10^{-170}	0.00	0.00
f_4	平均值	1.31×10^{-31}	6.65×10^{-31}	3.23×10^{-9}	2.66×10^{-3}	1.51×10^{-89}	2.94×10^{-87}	1.02×10^{-139}	0.00
	标准差	1.30×10^{-31}	1.48×10^{-31}	2.23×10^{-9}	2.37×10^{-3}	1.62×10^{-89}	3.10×10^{-87}	3.21×10^{-139}	0.00
f_5	平均值	5.14×10^{-7}	2.62	4.51	2.45	4.96×10^{-1}	1.46×10^{-1}	1.13	8.06
	标准差	9.47×10^{-7}	1.50	7.17×10^{-2}	1.00	4.21×10^{-1}	1.38×10^{-1}	5.06×10^{-1}	0.24
f_6	平均值	3.36×10^{-15}	3.28×10^{-15}	8.04×10^{-9}	4.28×10^{-10}	3.43×10^{-15}	3.37×10^{-15}	1.78×10^{-15}	0.00
	标准差	4.27×10^{-16}	2.51×10^{-16}	4.33×10^{-9}	2.89×10^{-10}	3.18×10^{-15}	1.05×10^{-15}	1.87×10^{-15}	0.00
f_7	平均值	0.00	0.00	1.89	2.76×10^{-9}	3.06	3.02	1.09	0.00
	标准差	0.00	0.00	1.03	3.94×10^{-9}	1.52	1.86	1.52	0.00
f_8	平均值	0.00	0.00	4.24×10^{-4}	6.33×10^{-12}	0.00	0.00	0.00	0.00
	标准差	0.00	0.00	6.60×10^{-4}	6.08×10^{-12}	0.00	0.00	0.00	0.00
f_9	平均值	0.00	1.48×10^{-3}	7.59×10^{-2}	4.15×10^{-3}	6.48×10^{-3}	2.42×10^{-2}	3.82×10^{-8}	0.00
	标准差	0.00	3.31×10^{-3}	5.19×10^{-2}	5.68×10^{-3}	9.71×10^{-3}	3.69×10^{-2}	1.21×10^{-7}	0.00
f_{10}	平均值	1.27×10^{-4}	1.27×10^{-4}	3.69×10^{-2}	1.27×10^{-4}	6.68×10^{-3}	7.03×10^{2}	7.92×10^{2}	9.79×10^{2}
	标准差	0.00	0.00	3.41×10^{-2}	1.22×10^{-9}	1.51×10^{2}	1.81×10^{2}	1.90×10^{2}	3.82×10^{2}
f_{11}	平均值	5.63×10^{-58}	3.32×10^{-46}	8.42×10^{-15}	7.05×10^{-11}	1.23×10^{-173}	2.21×10^{-161}	3.09×10^{-279}	0.00
	标准差	1.26×10^{-57}	7.42×10^{-46}	8.03×10^{-15}	1.55×10^{-10}	2.37×10^{-174}	3.04×10^{-161}	0.00	0.00
f_{12}	平均值	7.13×10^{-31}	6.80×10^{-46}	2.88×10^{-9}	1.78×10^{-3}	4.18×10^{-89}	2.18×10^{-86}	1.50×10^{-138}	0.00
	标准差	1.31×10^{-30}	5.81×10^{-36}	9.64×10^{-10}	1.31×10^{-3}	6.95×10^{-89}	4.82×10^{-86}	4.75×10^{-138}	0.00
f_{13}	平均值	3.61×10^{-2}	2.08	4.55	7	2.50	2.63	5.02	8.71
	标准差	7.94×10^{-2}	1.82	5.29×10^{-1}	6.39	2.33	2.87	2.53	0.16
f_{14}	平均值	2.84×10^{-15}	3.21×10^{-15}	9.98×10^{-9}	1.68×10^{-7}	3.52×10^{-15}	3.49×10^{-15}	2.13×10^{-15}	0.00
	标准差	1.59×10^{-15}	2.38×10^{-16}	2.82×10^{-9}	2.04×10^{-7}	96.05×10^{-16}	3.27×10^{-16}	1.83×10^{-15}	0.00
f_{15}	平均值	3.59	5.78	9.31	6.11	4.38	3.00	2.55	0.00
	标准差	1.84	1.85	1.96	3.00	8.91×10^{-1}	1.22	2.14	0.00
f_{16}	平均值	9.27×10^{-2}	0.00	2.55×10^{-2}	1.44×10^{-1}	0.00	0.00	0.00	0.00
	标准差	2.07×10^{-1}	0.00	2.95×10^{-2}	1.21×10^{-1}	0.00	0.00	0.00	0.00
f_{17}	平均值	1.40×10^{-2}	6.90×10^{-3}	1.02×10^{-1}	2.73×10^{-2}	2.81×10^{-3}	6.77×10^{-3}	5.40×10^{-10}	0.00
	标准差	9.09×10^{-3}	6.83×10^{-3}	3.85×10^{-2}	2.35×10^{-2}	6.29×10^{-3}	5.15×10^{-3}	1.70×10^{-9}	0.00
f_{18}	平均值	1.27×10^{-4}	3.59×10^{2}	2.81×10^{2}	2.32×10^{2}	5.58×10^{2}	5.69×10^{2}	7.77×10^{2}	1.37×10^{3}
	标准差	0.00	3.54×10^{2}	1.91×10^{2}	1.04×10^{2}	2.76×10^{2}	2.63×10^{2}	3.05×10^{2}	2.26×10^{3}

表 2　Friedman 测试在 10 维函数上得到的排序

Table 2　Ranking obtained by Friedman's test on 10D

算法	排序
TLBO-CSWL	2.888 9
VTTLBO	3.250 0
jDE	3.722 2
TLBO	4.083 3
ETLBO	4.416 7
SaDE	4.416 7
CLPSO	6.333 3
PSOwFIPS	6.888 9

3.1.2　与其他算法在 30 维测试上的比较

在该实验中，所有算法的种群规模设定为 40。仿真结果见表 4，从表 4 可以看出本文算法获得的平均结果在 f_1、f_2、f_3、f_4、f_6、f_7、f_{11}、f_{12}、f_{14}、f_{15} 上均明显优于其他所比较算法。对于函数 f_5，除了 CLPSO 外，其他比较算法的整体性能都要好于 TLBO-CSWL。对于函数 f_{10}，从结果来看，TLBO 及其改进算法的性能比改进的差分进化算法差，这主要是由算法的本身搜索特性所决定的。对于函数 f_{13}，虽然 TLBO-CSWL 的性能比 jDE、SaDE 和 PSOwFIPS 差，但比 CLPSO 和 TLBO 及其改进算法要好。对于函数 f_{18}，TLBO-CSWL 的性能表现要比其他所比较算法差，其主要原因可能是所提方法虽然可以加快 TLBO 的收敛速度，但也损失了算法一部分全局搜索的能力。在其余测试函数中，TLBO-CSWL 的寻优结果与所比较算法中获得的最好的结果相同。统计分析结果见表 5 和表 6。由表 5 可知，本文所提出的 TLBO-CSWL 算法与其他算法相比具有优越的整体性能。由表 6 可知，TLBO-CSWL 算法的性能要显著优于 PSOwFIPS 算法和 CLPSO 算法。此外，虽然 TLBO-CSWL 的寻优性能在统计学意义上没有显著性地优于其他比较算法，但是结合以上分析可知，TLBO-CSWL 算法在解决 18 个测试函数问题上整体表现得最好。

表 3　10 维测试结果 Bonferroni-Dunn、Holm 以及 Hochberg 检验的 *p*-Values

Table 3　*p*-Values obtained by Bonferroni-Dunn's, Holm's, and Hochberg's procedures on experimental results with 10D

算法	z	未调整 p	Bonferroni-Dunn p	Holm p	Hochberg p
PSOwFIPS	4.899 0	9.63×10^{-6}	6.73×10^{-6}	6.74×10^{-6}	6.74×10^{-6}
CLPSO	4.218 6	2.46×10^{-5}	1.72×10^{-4}	1.48×10^{-4}	1.48×10^{-4}
SaDE	1.871 1	6.13×10^{-2}	0.429 3	0.306 6	0.245 3
ETLBO	1.871 1	6.13×10^{-2}	0.429 3	0.306 6	0.245 3
TLBO	1.462 9	0.143 5	1.000 0	0.430 5	0.430 5
jDE	1.020 6	0.307 4	1.000 0	0.614 9	0.614 9
VTTLBO	0.442 3	0.658 3	1.000 0	0.658 3	0.658 3

表 4　30 维仿真测试结果

Table 4　Experimental results on 30D

函数	结果	jDE	SaDE	PSOwFIPS	CLPSO	TLBO	ETLBO	VTTLBO	TLBO-CSWL
f_1	平均值	1.95×10^{-22}	3.84×10^{-23}	1.43	1.94×10^{-1}	4.04×10^{-111}	2.66×10^{-95}	4.85×10^{-158}	0.00
	标准差	2.76×10^{-22}	2.15×10^{-23}	2.78×10^{-1}	7.79×10^{-2}	3.20×10^{-111}	1.84×10^{-95}	1.06×10^{-157}	0.00
f_2	平均值	2.06×10	1.06×10	3.82×10^{3}	1.16×10^{4}	1.08×10^{-22}	3.42×10^{-22}	1.24×10^{-32}	1.27×10^{-251}
	标准差	6.71	6.53	1.01×10^{3}	2.72×10^{3}	1.43×10^{-22}	4.72×10^{-22}	1.79×10^{-32}	0.00
f_3	平均值	3.92×10^{-23}	3.00×10^{-24}	2.17×10^{-1}	2.25×10^{-2}	5.38×10^{-111}	8.21×10^{-96}	2.50×10^{-158}	0.00
	标准差	3.86×10^{-23}	2.47×10^{-24}	7.69×10^{-2}	6.43×10^{-3}	3.43×10^{-111}	1.11×10^{-95}	2.48×10^{-158}	0.00
f_4	平均值	1.35	1.50×10^{-1}	1.11×10^{2}	2.30×10^{2}	5.11×10^{-11}	1.92×10^{-11}	8.06×10^{-18}	0.00
	标准差	1.68	1.31×10^{-1}	1.89×10	5.45×10	8.59×10^{-11}	1.93×10^{-11}	1.11×10^{-17}	0.00
f_5	平均值	2.18×10	2.52×10	2.73×10	6.84×10	2.38×10	2.38×10	2.28×10	2.83×10
	标准差	2.59×10^{-1}	1.36	2.99×10^{-1}	2.84×10	7.01×10^{-1}	8.57×10^{-1}	4.23×10^{-1}	0.44

续表 4

函数	结果	jDE	SaDE	PSOwFIPS	CLPSO	TLBO	ETLBO	VTTLBO	TLBO-CSWL
f_6	平均值	2.82×10^{-12}	1.30×10^{-12}	5.21×10^{-1}	7.13×10^{-1}	3.55×10^{-15}	3.55×10^{-15}	3.55×10^{-15}	9.47×10^{-16}
	标准差	1.77×10^{-12}	8.26×10^{-13}	1.78×10^{-1}	5.74×10^{-1}	0.00	0.00	0.00	1.60×10^{-15}
f_7	平均值	2.24×10^{-9}	5.53×10^{-1}	1.23×10^{2}	2.91×10	1.17×10	1.22×10	1.25×10	0.00
	标准差	4.15×10^{-9}	7.55×10^{-1}	1.57×10	4.77	3.71	9.07	7.27	0.00
f_8	平均值	3.16×10^{-1}	6.56×10^{-11}	2.46	5.18×10^{-1}	0.00	0.00	0.00	0.00
	标准差	5.07×10^{-1}	8.87×10^{-11}	7.11×10^{-1}	6.83×10^{-2}	0.00	0.00	0.00	0.00
f_9	平均值	0.00	0.00	8.92×10^{-1}	2.68×10^{-1}	0.00	0.00	0.00	0.00
	标准差	0.00	0.00	5.53×10^{-2}	4.38×10^{-2}	0.00	0.00	0.00	0.00
f_{10}	平均值	2.37×10	3.82×10^{-4}	4.99×10^{3}	1.72×10^{3}	4.29×10^{3}	4.55×10^{3}	4.70×10^{3}	4.56×10^{3}
	标准差	5.30×10	1.99×10^{-10}	5.50×10^{2}	3.21×10^{2}	9.40×10^{2}	7.81×10^{2}	4.22×10^{2}	1.03×10^{3}
f_{11}	平均值	2.57×10^{-6}	1.21×10^{-4}	7.91×10^{-1}	1.37	1.56×10^{-106}	1.25×10^{-92}	2.58×10^{-153}	0.00
	标准差	5.75×10^{-6}	2.54×10^{-4}	6.74×10^{-1}	5.16×10^{-1}	3.33×10^{-106}	1.37×10^{-92}	5.31×10^{-153}	0.00
f_{12}	平均值	6.79×10^{-1}	1.51×10^{-1}	1.13×10^{2}	2.27×10^{2}	3.24×10^{-12}	6.28×10^{-11}	1.99×10^{-17}	0.00
	标准差	6.36×10^{-1}	6.29×10^{-2}	1.65×10	1.60×10	3.24×10^{-12}	6.35×10^{-11}	4.29×10^{-17}	0.00
f_{13}	平均值	2.42×10	2.74×10	2.83×10	5.61×10	5.10×10	5.50×10	7.05×10	2.88×10
	标准差	1.03	2.04	5.57×10^{-1}	1.63×10	2.82×10	2.95×10	2.56×10	0.11
f_{14}	平均值	7.24×10^{-12}	1.36×10^{-12}	5.96×10^{-1}	2.68	3.55×10^{-15}	3.55×10^{-15}	3.55×10^{-15}	8.29×10^{-16}
	标准差	3.21×10^{-12}	9.25×10^{-13}	1.55×10^{-1}	1.40×10^{-1}	0.00	0.00	0.00	1.53×10^{-15}
f_{15}	平均值	6.18×10	9.63×10	1.52×10^{2}	1.10×10^{2}	2.17×10	1.36×10	1.41×10	0.00
	标准差	1.29×10	1.50×10	1.53×10	1.14×10	1.64×10	7.85	5.55	0.00
f_{16}	平均值	2.20	9.96×10^{-3}	5.03	1.38×10	0.00	0.00	0.00	0.00
	标准差	4.19	2.18×10^{-2}	6.56×10^{-1}	1.21	0.00	0.00	0.00	0.00
f_{17}	平均值	2.46×10^{-3}	7.98×10^{-13}	9.34×10^{-1}	3.64×10^{-1}	1.15×10^{-3}	0.00	0.00	0.00
	标准差	5.51×10^{-3}	1.26×10^{-12}	6.89×10^{-2}	1.05×10^{-1}	2.57×10^{-3}	0.00	0.00	0.00
f_{18}	平均值	3.49×10^{3}	4.66×10^{3}	5.86×10^{3}	4.34×10^{3}	4.53×10^{3}	4.78×10^{3}	4.70×10^{3}	7.59×10^{3}
	标准差	2.98×10^{2}	6.74×10^{2}	4.95×10^{2}	2.33×10^{2}	6.01×10^{2}	1.09×10^{3}	1.04×10^{3}	4.68×10^{2}

表 5　Friedman 测试在 30 维函数上得到的排序

Table 5　Ranking obtained by Friedman's test on 30D

算法	排序
TLBO-CSWL	2.527 8
VTTLBO	3.305 6
TLBO	3.444 4
ETLBO	3.722 2
SaDE	4.416 7
jDE	4.472 2
CLPSO	6.944 4
PSOwFIPS	7.166 7

3.2　TLBO-CSWL 算法分析

为了验证所提算法的有效性，利用 18 个 30 维测试函数来对 TLBO-CSWL 和它的变种算法 TLBO-CSWL-1(使用均匀随机数)、TLBO-CSWL-2(没有混沌搜索) 进行测试。其中，种群规模设定为 40，最大评价次数 50 000。针对每个测试函数，每个算法均独立运行 30 次。同时，利用 Friedman、Bonfeeroni-Dunn、Holm 以及 Hochberg 检验等非参数的统计方法对结果进行统计分析，其中显著性水平设定为 5%。

3.2.1　与变种算法 TLBO-CSWL-1 的比较

为了验证策略正态随机数的有效性，将对 TLBO-CSWL-1 与 TLBO-CSWL 进行仿真测试。结果见表 7。由表 7 可知，TLBO-CSWL 算法所得

到的平均结果在 f_2、f_6、f_{10}、f_{14} 和 f_{18} 上要比 TLBO-CSWL-1 好，而对于其余的测试函数，两者结果相同；这说明 TLBO-CSWL 算法具有更好的寻优性能。统计分析结果见表 8，由表 8 可知 TLBO-CSWL 和 TLBO-CSWL-1 在统计学意义上虽然不存在显著性差异，但是从 Friedman 测试所得到的排序结果来看 (见图 1)，TLBO-CSWL 的整体性能要好于 TLBO-CSWL-1。以上统计分析结果表明，利用正态分布产生随机数替代原有均匀随机数的策略对于提升 TLBO 算法的性能是有效的。

表 6 30 维测试结果 Bonferroni-Dunn、Holm 以及 Hochberg 检验的 *p*-Values

Table 6 *p*-Values obtained by Bonferroni-Dunn's, Holm's, and Hochberg's procedures on experimental results with 30D

算法	z	未调整 *p*	Bonferroni-Dunn *p*	Holm *p*	Hochberg *p*
PSOwFIPS	5.681 5	1.34×10^{-8}	9.35×10^{-8}	9.35×10^{-8}	9.35×10^{-8}
CLPSO	5.409 3	6.33×10^{-8}	4.43×10^{-7}	3.80×10^{-7}	3.79×10^{-7}
jDE	2.381 4	1.72×10^{-2}	0.120 7	8.62×10^{-2}	8.28×10^{-2}
SaDE	2.313 4	2.07×10^{-2}	0.144 9	8.62×10^{-2}	8.28×10^{-2}
ETLBO	1.462 9	0.143 5	1.000 0	0.430 5	0.340 8
TLBO	1.122 7	0.261 6	1.000 0	0.523 1	0.340 8
VTTLBO	0.952 6	0.340 8	1.000 0	0.523 1	0.340 8

表 7 TLBO-CSWL-1、TLBO-CSWL-2 与 TLBO-CSWL 算法的 30 维仿真测试结果

Table 7 Comparison of TLBO-CSWL-1, TLBO-CSWL-2, and TLBO-CSWL with experimental results on 30D

函数	结果	TLBO-CSWL-1	TLBO-CSWL-2	TLBO-CSWL	函数	结果	TLBO-CSWL-1	TLBO-CSWL-2	TLBO-CSWL
f_1	平均值	0.00	0.00	0.00	f_{10}	平均值	4.77×10^{3}	8.60×10^{3}	4.56×10^{3}
	标准差	0.00	0.00	0.00		标准差	7.01×10^{2}	8.35×10^{2}	1.03×10^{3}
f_2	平均值	1.93×10^{-157}	4.52×10^{-220}	1.27×10^{-251}	f_{11}	平均值	0.00	0.00	0.00
	标准差	1.06×10^{-156}	0.00	0.00		标准差	0.00	0.00	0.00
f_3	平均值	0.00	0.00	0.00	f_{12}	平均值	0.00	0.00	0.00
	标准差	0.00	0.00	0.00		标准差	0.00	0.00	0.00
f_4	平均值	0.00	0.00	0.00	f_{13}	平均值	2.88×10	2.89×10	2.88×10
	标准差	0.00	0.00	0.00		标准差	0.18	0.03	0.11
f_5	平均值	2.73×10	2.88×10	2.83×10	f_{14}	平均值	1.53×10^{-15}	1.78×10^{-15}	8.29×10^{-16}
	标准差	0.69	0.09	0.44		标准差	1.79×10^{-15}	1.81×10^{-15}	1.53×10^{-15}
f_6	平均值	1.07×10^{-15}	9.47×10^{-16}	9.47×10^{-16}	f_{15}	平均值	0.00	0.00	0.00
	标准差	1.66×10^{-15}	1.60×10^{-15}	1.60×10^{-15}		标准差	0.00	0.00	0.00
f_7	平均值	0.00	4.64	0.00	f_{16}	平均值	0.00	0.00	0.00
	标准差	0.00	1.15×10	0.00		标准差	0.00	0.00	0.00
f_8	平均值	0.00	0.00	0.00	f_{17}	平均值	0.00	0.00	0.00
	标准差	0.00	0.00	0.00		标准差	0.00	0.00	0.00
f_9	平均值	0.00	0.00	0.00	f_{18}	平均值	7.78×10^{3}	9.02×10^{3}	7.59×10^{3}
	标准差	0.00	0.00	0.00		标准差	5.49×10^{2}	5.53×10^{2}	4.68×10^{2}

表 8　Bonferroni-Dunn、Holm 以及 Hochberg 检验的 p-Values (TLBO-CSWL-1)

Table 8　p-Values obtained by Bonferroni-Dunn's, Holm's, and Hochberg's procedures (TLBO-CSWL-1)

算法	z	未调整 p	Bonferroni-Dunn p	Holm p	Hochberg p
TLBO-CSWL-1	0.942 8	0.345 8	0.345 8	0.345 8	0.345 8

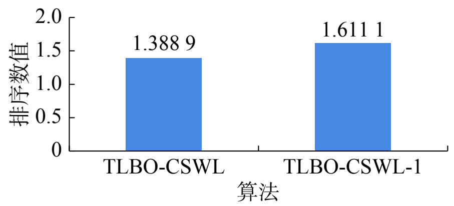


图 1　Friedman 测试排序结果 (TLBO-CSWL-1)

Fig. 1　Ranking obtained by Friedman's test(TLBO-CSWL-1)

3.2.2　与变种算法 TLBO-CSWL-2 的比较

本文对不加混沌搜索的 TLBO-CSWL-2 和 TLBO-CSWL 在同样的 18 个测试函数上进行仿真测试。结果如表 7 所示。从表 7 可以看出所提出的 TLBO-CSWL 算法在 f_2、f_5、f_6、f_7、f_{10}、f_{13}、f_{14} 和 f_{18} 上的寻优结果要比 TLBO-CSWL-2 好, 而在其余测试函数上, 两者的寻优结果相同; 以上表明 TLBO-CSWL 算法的性能要好于 TLBO-CSWL-2 算法。统计分析结果见表 9, 由表 9 可知 TLBO-CSWL 和 TLBO-CSWL-2 在统计学意义上不存在显著性差异。但是, 从图 2 的 Friedman 测试所得到的排序结果来看, 与 TLBO-CSWL-2 相比, TLBO-CSWL 具有更好的整体性能。以上统计分析表明, 混沌搜索策略对于提升 TLBO 算法的性能是有效的。

表 9　Bonferroni-Dunn、Holm 以及 Hochberg 检验的 p-Values (TLBO-CSWL-2)

Table 9　p-Values obtained by Bonferroni-Dunn's, Holm's, and Hochberg's procedures (TLBO-CSWL-2)

算法	z	未调整 p	Bonferroni-Dunn p	Holm p	Hochberg p
TLBO-CSWL-2	1.885 6	0.059 3	0.059 3	0.059 3	0.059 3

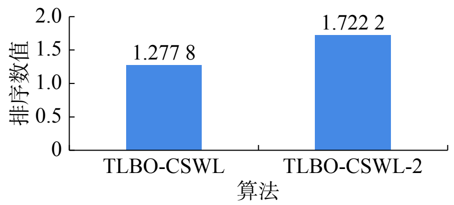


图 2　Friedman 测试排序结果 (TLBO-CSWL-2)

Fig. 2　Ranking obtained by Friedman's test(TLBO-CSWL-2)

4　在非合作博弈问题中的应用

在本实验中, 将所提算法应用于非合作博弈纳什均衡问题的求解。

4.1　博弈问题的描述

N 人有限非合作博弈纳什均衡问题, 主要是求解一种混合策略使得博弈双方均基于一定的概率来选择自己的每一个纯策略, 使得博弈双方的利益均最大化, 此时博弈模型处于稳定的状态。参考文献[7]中的问题描述, 对于 2 人有限非合作博弈问题: 设局中人 1 的混合策略为 $\boldsymbol{x}=(x_1,x_2,\cdots,x_m)$, 局中人 2 的混合策略为 $\boldsymbol{y}=(y_1,y_2,\cdots,y_n)$。$\boldsymbol{A}_{m\times n}$, $\boldsymbol{B}_{m\times n}$ 分别为局中人 1 和局中人 2 的支付矩阵, 则局中人 1 和局中人 2 的期望支付分别为 $\boldsymbol{xAy}^{\mathrm{T}}$ 和 $\boldsymbol{xBy}^{\mathrm{T}}$。$(\boldsymbol{x}^*, \boldsymbol{y}^*)$ 为双矩阵博弈问题的一个纳什均衡解的充分必要条件, 即

$$\begin{cases}\boldsymbol{x}^*\boldsymbol{Ay}^{*\mathrm{T}}\geqslant \boldsymbol{xAy}^{*\mathrm{T}},\ \forall \boldsymbol{x}\\ \boldsymbol{x}^*\boldsymbol{By}^{*\mathrm{T}}\geqslant \boldsymbol{xBy}^{*\mathrm{T}},\ \forall \boldsymbol{y}\end{cases}\tag{8}$$

算法中的每一个个体的取值表示所有局中人的混合策略, 则双矩阵博弈问题 $z=(\boldsymbol{x},\boldsymbol{y})$ 的适应度函数可以表示为

$$\begin{aligned}f(z)=&\max\left\{\max_{0\leqslant i\leqslant n}\left(\boldsymbol{A}_i\boldsymbol{y}^{\mathrm{T}}-\boldsymbol{xAy}^{\mathrm{T}}\right),0\right\}+\\&\max\left\{\max_{0\leqslant j\leqslant m}\left(\boldsymbol{xB}_j-\boldsymbol{xBy}^{\mathrm{T}}\right),0\right\}\end{aligned}\tag{9}$$

式 (9) 中, $\boldsymbol{A}_i$ 表示 $\boldsymbol{A}_{m\times n}$ 的第 i 行, $\boldsymbol{B}_j$ 表示 $\boldsymbol{B}_{m\times n}$ 的第 j 列。

根据纳什均衡的定义和性质[1]可知, 混合局势 $z^*=(\boldsymbol{x}^*,\boldsymbol{y}^*)$ 为双矩阵博弈问题的一个纳什均衡解的充分必要条件为: 存在 $z^*=(\boldsymbol{x}^*,\boldsymbol{y}^*)$, 使得 $f(z^*)=0$; 且对于任意的 $z\neq z^*$, 都有 $f(z)>0$。

4.2　案例研究

本文选取 2 个双矩阵博弈问题[30-31]。

例 1　博弈模型 1, $\Gamma_1\equiv(x_1,y_1;\boldsymbol{A}_1,\boldsymbol{B}_1)$, 支付矩阵为

$$\boldsymbol{A}_1=\begin{bmatrix}1&2&0\\0&1&2\\2&0&1\end{bmatrix},\quad \boldsymbol{B}_1=\begin{bmatrix}1&0&2\\2&1&0\\0&2&1\end{bmatrix}\tag{10}$$

例 2　博弈模型 2, $\Gamma_2\equiv(x_2,y_2;\boldsymbol{A}_2,\boldsymbol{B}_2)$, 支付矩

阵为

$$A_2 = \begin{bmatrix} 0 & 4 & 5 \\ 4 & 0 & 5 \\ 3 & 3 & 6 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 4 & 0 & 3 \\ 0 & 4 & 3 \\ 5 & 5 & 6 \end{bmatrix} \tag{11}$$

为了求解以上两个问题，选取 jDE、SaDE、CLPSO、TLBO、TLBO-CSWL 来对其进行求解。对于所有的算法，种群大小均设定为 40，最大的评价次数为 2 000，每个问题均独立运行 30 次。并使用 Wilcoxon 秩和检验方法[32]来对实验结果进行统计分析。

实验结果和统计分析结果见表 10，其中，“+”表示所提算法优于其比较算法；“–”表示所提算法差于其比较算法；“≈”表示 TLBO-CSWL 算法与其他算法性能相似。从表 10的统计结果可知，TLBO-CSWL 算法在 2 个博弈问题上的求解结果均优于 jDE、SaDE、CLPSO、TLBO 算法，说明 TLBO-CSWL 算法在解决非合作博弈问题上表现得最好。另外，对于博弈问题 1 和 2，所有算法的最好结果见表 11 和表 12。从表 11 可以看出，相对于其他算法，TLBO-CSWL算法能够得到更好的纳什均衡解。同时对于博弈模型 1，本文算法求解的适应度函数精度明显优于其他算法。另外，从表 12 可以看出，TLBO-CSWL比其他算法能找到更好的纳什均衡解，适应度函数的精度显著好于其他所有算法，这说明 TLBO-CSWL 算法在计算结果的精度比其他算法有了较大的改进。

上述分析表明，将本文提出的 TLBO-CSWL 算法应用到非合作博弈问题，取得了较满意的结果。

表 10 所有算法在博弈问题上得到的结果

Table 10 Experimental results of all algorithms on game problems

函数	结果	jDE	SaDE	CLPSO	TLBO	TLBO-CSWL
1	平均值	3.47×10^{-4} +	4.31×10^{-3} +	5.93×10^{-2} +	1.69×10^{-3} +	1.56×10^{-3}
	标准差	1.81×10^{-3}	1.52×10^{-3}	2.96×10^{-2}	1.59×10^{-3}	3.70×10^{-3}
2	平均值	6.65×10^{-4} +	1.00×10^{-3} +	1.24×10^{-2} +	3.24×10^{-6} +	3.86×10^{-10}
	标准差	4.11×10^{-4}	4.31×10^{-4}	6.72×10^{-2}	2.9×10^{-6}	3.28×10^{-10}
+		2	2	2	2	
–		0	0	0	0	
≈		0	0	0	0	

表 11 所有算法在博弈模型 1 上得到的最好结果

Table 11 The best experimental results of all algorithms on game problem 1

算法	局中人 1 混合策略	局中人 2 混合策略	适应度函数
jDE	(0.332 8, 0.333 8, 0.333 4)	(0.333 4, 0.333 2, 0.333 4)	8.52×10^{-4}
SaDE	(0.332 7, 0.333 7, 0.333 6)	(0.333 4, 0.332 6, 0.334 0)	1.81×10^{-3}
CLPSO	(0.327 0, 0.337 4, 0.335 6)	(0.335 0, 0.320 0, 0.345 1)	2.36×10^{-2}
TLBO	(0.333 4, 0.333 3, 0.333 3)	(0.333 3, 0.333 3, 0.333 4)	2.64×10^{-4}
TLBO-CSWL	(0.333 3, 0.333 3, 0.333 3)	(0.333 3, 0.333 3, 0.333 3)	1.90×10^{-6}

表 12 所有算法在博弈模型 2 上得到的最好结果

Table 12 The best experimental results of all algorithms on game problem 2

算法	局中人 1 混合策略	局中人 2 混合策略	适应度函数
jDE	(0.000 0, 0.000 1, 0.999 9)	(0.000 0, 0.000 0, 1.000 0)	1.02×10^{-6}
SaDE	(0.000 0, 0.000 1, 0.999 9)	(0.000 0, 0.000 1, 0.999 9)	3.48×10^{-4}
CLPSO	(0.012 9, 0.000 1, 0.987 0)	(0.003 9, 0.013 4, 0.982 7)	3.03×10^{-2}
TLBO	(0.000 0, 0.000 0, 1.000 0)	(0.000 0, 0.000 0, 1.000 0)	9.01×10^{-7}
TLBO-CSWL	(0.000 0, 0.000 0, 1.000 0)	(0.000 0, 0.000 0, 1.000 0)	3.53×10^{-11}

5 结束语

本文针对教与学优化算法容易早熟收敛的问题, 提出了一种基于混沌搜索和权重学习的教与学优化 (TLBO-CSWL) 算法。在该算法中, 利用当前种群的加权平均值来指导种群的进化, 并且使用正态分布随机数替代均匀随机数来提高原始 TLBO 算法的寻优性能; 另外, 将混沌搜索策略加到所提算法中, 以此来提高算法的全局搜索能力。实验仿真结果表明, 所提算法的整体性能在所有比较算法中是最好的。同时, 采用 TLBO-CSWL 算法与其变种算法 TLBO-CSWL-1、TLBO-CSWL-2 进行比较分析, 仿真结果显示本文所提出的改进策略对于提升 TLBO 算法的性能是有效的。最后, 将 TLBO-CSWL 算法应用于求解非合作博弈纳什均衡问题, 其结果表明所提算法得到的结果要好于其他算法。

参考文献:

[1] NASH J. Non-cooperative games[J]. Annals of mathematics, 1951, 54(2): 286–295.

[2] LEMKE C E, HOWSON J T JR. Equilibrium points of bimatrix games[J]. Journal of the society for industrial and applied mathematics, 1964, 12(2): 413–423.

[3] GOVINDAN S, WILSON R. A global Newton method to compute Nash equilibria[J]. Journal of economic theory, 2003, 110(1): 65–86.

[4] HERINGS P J J, PEETERS R. Homotopy methods to compute equilibria in game theory[J]. Economic theory, 2010, 42(1): 119–156.

[5] 陈士俊, 孙永广, 吴宗鑫. 一种求解 NASH 均衡解的遗传算法[J]. 系统工程, 2001, 19(5): 67–70.
CHEN Sijun, SUN Yongguang, WU Zongxin. A genetic algorithm to acquire the nash equilibrium[J]. Systems engineering, 2001, 19(5): 67–70.

[6] 邱中华, 高洁, 朱跃星. 应用免疫算法求解博弈问题[J]. 系统工程学报, 2006, 21(4): 398–404.
QIU Zhonghua, GAO Jie, ZHU Yuexing. Applying immune algorithm to solving game problem[J]. Journal of systems engineering, 2006, 21(4): 398–404.

[7] 贾文生, 向淑文, 杨剑锋, 等. 基于免疫粒子群算法的非合作博弈 Nash 均衡问题求解[J]. 计算机应用研究, 2012, 29(1): 28–31.
JIA Wensheng, YUAN Shuwen, YANG Jianfeng, et al. Solving nash equilibrium for N-persons' non-cooperative game based on immune particle swarm algorithm[J]. Application research of computers, 2012, 29(1): 28–31.

[8] 王志勇, 韩旭, 许维胜, 等. 基于改进蚁群算法的纳什均衡求解[J]. 计算机工程, 2010, 36(14): 166–168, 171.
WANG Zhiyong, HAN Xu, XU Weisheng, et al. Nash equilibrium solution based on improved ant colony algorithm[J]. Computer engineering, 2010, 36(14): 166–168, 171.

[9] RAO R V, SAVSANI V J, VAKHARIA D P. Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems[J]. Information sciences, 2012, 183(1): 1–15.

[10] RAO R V, SAVSANI V J, VAKHARIA D P. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems[J]. Computer-aided design, 2011, 43(3): 303–315.

[11] 拓守恒, 邓方安, 雍龙泉. 改进教与学优化算法的 LQR 控制器优化设计[J]. 智能系统学报, 2014, 9(5): 602–607.
TUO Shouheng, DENG Fang'an, YONG Longquan. Optimal design of a linear quadratic regulator (LQR) controller based on the modified teaching-learning-based optimization algorithm[J]. CAAI transactions on intelligent systems, 2014, 9(5): 602–607.

[12] RAO R V, PATEL V. Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm[J]. Applied mathematical modelling, 2013, 37(3): 1147–1162.

[13] WANG Lei, ZOU Feng, HEI Xinhong, et al. An improved teaching–learning-based optimization with neighborhood search for applications of ANN[J]. Neurocomputing, 2014, 143: 231–247.

[14] RAO R V, PATEL V. An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems[J]. International journal of industrial engineering computations, 2012, 3(4): 535–560.

[15] YU Kunjie, WANG Xin, WANG Zhenlei. An improved teaching-learning-based optimization algorithm for numerical and engineering optimization problems[J]. Journal of intelligent manufacturing, 2016, 27(4): 831–843.

[16] ZOU Feng, WANG Lei, HEI Xinhong, et al. Teaching–learning-based optimization with dynamic group strategy for global optimization[J]. Information sciences, 2014, 273: 112–131.

[17] CHEN Debao, LU Renquan, ZOU Feng, et al. Teaching-learning-based optimization with variable-population scheme and its application for ANN and global optimization[J]. Neurocomputing, 2016, 173: 1096–1111.

[18] WU Zongsheng, FU Weiping, XUE Ru. Nonlinear inertia weighted teaching-learning-based optimization for solving global optimization problem[J]. Computational Intelligence and Neuroscience, 2015, 2015: 1–15.

[19] SHAHBEIG S, HELFROUSH M S, RAHIDEH A. A fuzzy multi-objective hybrid TLBO–PSO approach to select the associated genes with breast cancer[J]. Signal processing, 2017, 131: 58–65.

[20] 刘军梅. 新型混沌粒子群混合优化算法[J]. 软件导刊, 2017, 16(2): 59–62.
LIU Junmei. New chaotic particle swarm hybrid optimization algorithm[J]. Software guide, 2017, 16(2): 59–62.

[21] 唐娜, 张公让, 李磊. 自适应混沌搜索的双粒子群优化算法[J]. 计算机工程与设计, 2016, 37(9): 2421–2428.
TANG Na, ZHANG Gongrang, LI Lei. Double particle swarm optimization algorithm of self-adaption and chaos search[J]. Computer engineering and design, 2016, 37(9): 2421–2428.

[22] BREST J, GREINER S, BOSKOVIC B, et al. Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems[J]. IEEE transactions on evolutionary computation, 2006, 10(6): 646–657.

[23] QIN A K, HUANG V L, SUGANTHAN P N. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE transactions on evolutionary computation, 2009, 13(2): 398–417.

[24] MENDES R, KENNEDY J, NEVES J. The fully informed particle swarm: simpler, maybe better[J]. IEEE transactions on evolutionary computation, 2004, 8(3): 204–210.

[25] LIANG J J, QIN A K, SUGANTHAN P N, et al. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions[J]. IEEE transactions on evolutionary computation, 2006, 10(3): 281–295.

[26] FRIEDMAN M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance[J]. Journal of the American statistical association, 1937, 34(200): 675–701.

[27] DUNN O J. Multiple comparisons among means[J]. Journal of the American statistical association, 1961, 56(293): 52–64.

[28] GARCÍA S, 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.

[29] WOLPERT D H, MACREADY W G. No free lunch theorems for optimization[J]. IEEE transactions on evolutionary computation, 1997, 1(1): 67–82.

[30] WEIBULL J W. Evolutionary game theory[M]. Cambridge, MA, USA: MIT Press, 1995.

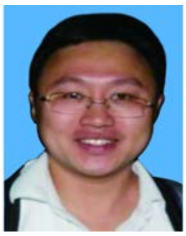
[31] GIBBONS R. A primer in game theory[M]. New York, USA: Harvester Wheatsheaf, 1992.

[32] WILCOXON F. Individual comparisons by ranking methods[J]. Biometrics bulletin, 1945, 1(6): 80–83.

作者简介：

柳缔西子，女，1995 年生，硕士研究生，主要研究方向为教与学优化算法、物流与供应链管理。

范勤勤，男，1986 年生，讲师，主要研究方向为多目标优化、机器学习、进化计算。发表学术论文 20 余篇。



胡志华，男，1977 年生，教授，博士生导师，主要研究方向为物流与港航运作优化、大数据系统与管理、计算智能与计算实验。发表学术论文百余篇，被 SCI、EI 检索 30 余篇。