计算机科学
COMPUTER SCIENCE

基于边界自适应技术的精英交互学习粒子群算法

徐杰, 周新志

引用本文

徐杰, 周新志. 基于边界自适应技术的精英交互学习粒子群算法[J]. 计算机科学, 2023, 50(11): 210-219.

XU Jie, ZHOU Xinzhi. Multi-elite Interactive Learning Based Particle Swarm Optimization Algorithm with Adaptive Bound-handling Technique [J]. Computer Science, 2023, 50(11): 210-219.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.221000129

基于边界自适应技术的精英交互学习粒子群算法

徐 杰 周新志
四川大学电子信息学院 成都 610000
(xjhkyjs@163.com)

摘 要 粒子群优化(PSO)算法依靠粒子之间的合作行为,使其在解决诸多优化问题上显示出极大的智能。然而,由于寻优机制,粒子很容易突破可行域的边界限制,若能使该行为在寻优过程中具有明确的指导意义将有助于提高算法的优化性能;更关键的是,原始粒子群优化算法中粒子的学习对象主要集中在全局最佳粒子上,这种更新机制无疑加速了种群多样性的损失,并使种群倾向于陷入局部最优。为了进一步提高求解复杂问题时的种群多样性和收敛精度,提出了一种基于边界自适应技术的精英交互学习粒子群算法(A-EIPSO)。该算法首先在原有的PSO算法中引入了新的边界处理技术,根据越界粒子的历史位置信息和越界距离自适应地赋予粒子在解空间内的分布特征;接着在多种群技术的基础上设计了一种精英学习策略来促进子群间社会信息的交换,并由精英粒子代替全局最佳粒子指导各子群内粒子的优化行为。实验结果表明,在大多数情况下,自适应处理技术保证粒子在搜索空间内实现均匀探索的同时显著提升了PSO算法的性能。此外,还将A-EIPSO在CEC2017基准测试套件上与5种先进的粒子群变体算法及2种主流的进化算法进行了比较。结果表明,A-EIPSO在不同类型函数上均表现出了优越的性能,改进了大多数优化问题的收敛精度,优于其他代表性的PSO变体算法和进化算法。

关键词:粒子群优化算法;自适应策略;边界处理技术;多种群;精英交互学习
中图法分类号 TP301.6

Multi-elite Interactive Learning Based Particle Swarm Optimization Algorithm with Adaptive Bound-handling Technique

XU Jie and ZHOU Xinzhi
College of Electronic Information, Sichuan University, Chengdu 610000, China

Abstract Particle swarm optimization(PSO) algorithm relies on the cooperation between particles, which makes it show great intelligence in solving many optimization problems. However, due to the optimization mechanism, particles are easy to break through the boundary restrictions of the feasible region. If this behavior can have a clear guiding significance in the optimization process, it will help to improve the optimization performance of the algorithm. More importantly, the learning objects of particles in the original particle swarm optimization algorithm are mainly focused on the global optimal particles. This updating mechanism undoubtedly accelerates the loss of population diversity, and makes the population tend to fall into the local optimal. In order to further improve the population diversity and convergence accuracy when solving complex problems, an elite interactive learning particle swarm optimization algorithm(A-EIPSO) based on adaptive strategy is proposed. Firstly, the algorithm introduces a new bound-handling technique into the original PSO algorithm, and adaptively endows the distribution characteristics of particles in the solution space by using the historical location information and the distance of out of bounds particles, so as to modify the position of particles to meet the requirements of effectively handling out of violated particles. Then, based on multi-swarm technology, an elites learning strategy is designed to promote the exchange of social information among subswarms, and the elite particles instead of the global optimal particles guide the optimization behavior of particles in each subswarm. Experimental results show that, in most cases, the adaptive strategy can ensure that particles can achieve uniform exploration in the search space and significantly improve the performance of PSO algorithm. In addition, A-EIPSO is compared with five advanced particle swarm optimization variant algorithms and two mainstream evolutionary algorithms on the CEC2017 benchmark suite. The results show that A-EIPSO has superior performance on different types of functions, improves the convergence accuracy of most optimization problems, and is superior to other representative PSO variant algorithms and evolutionary algorithms.

到稿日期:2022-10-16 返修日期:2023-03-30
基金项目:国家自然科学基金(U1933123);四川省科技成果转移转化示范项目(2022ZHCG0042)
This work was supported by the National Natural Science Foundation of China(U1933123) and Sichuan Science and Technology Program (2022ZHCG0042).
通信作者:周新志(xz.zhou@scu.edu.cn)

Keywords　Particle swarm optimization algorithm，Adaptive strategy，Bound-handling techniques，Multi-swarm，Elite Interactive learning

1　引言

粒子群优化（Particle Swarm Optimization，PSO）算法最初由 Eberhart 等于 1995 年引入[1-2]，是一种基于种群的随机优化技术。由于其实现简单且探索全局解的效率高，PSO 已成功被应用于解决许多问题，如分类[3]、特征选择[4]、任务分配[5]和随机优化[6]。

在该算法中，每个粒子 $i(i=1,2,\cdots,M)$ 表示一个优化问题的潜在解，并用两个向量对其进行定义，速度 $V_i=[v_i^1,v_i^2,\cdots,v_i^D]$ 和位置 $X_i=[x_i^1,x_i^2,\cdots,x_i^D]$。最初，具有随机速度值的粒子在 D 维搜索空间中的随机位置生成。在优化过程中，每个粒子将根据以下学习策略更新其速度和位置[1]：

$$v_i'^d=v_i^d+c_1 rand_1^d(pBest_i^d-x_i^d)+c_2 rand_2^d(gBest_i^d-x_i^d) \tag{1}$$

$$x_{i+1}'^d=x_i^d+v_i^d \tag{2}$$

其中，$d=1,2,\cdots,D$，c_1 和 c_2 是加速度系数，$rand_1$ 和 $rand_2$ 是在[0，1]区间内均匀分布的两个随机数，$pBest_i$ 是粒子 i 的历史最佳位置，而 $gBest_i$ 是当前迭代次数下所有粒子发现的全局最佳位置。

几乎所有的元启发式优化算法都需要某种形式的初始化。理想情况下，算法找到的最优解应该独立于初始选择，然而进化算法作为一种基于群体的元启发式优化算法，经验观测和数值模拟都表明，在解决多峰优化问题时，进化算法的最终解质量会受到问题的初始起点的布局影响，其中，相比遗传算法（Genetic Algorithm，GA），PSO 算法的性能对初始化结果更为敏感[7]。

将优化问题定义为目标函数并不是解决过程中的唯一步骤，许多实际优化问题都是有约束的，包括复杂的非线性约束和变量允许取值范围约束等，其中变量解空间上的约束通常被称为“边界”。如果问题包含边界约束，处理技术的优劣将直接反映在算法的性能表现上，并且随着决策变量数的增加，粒子离开可行区域的概率显著增大[8]。因此，一个有效且有助于粒子发现潜在最优解的边界处理技术变得尤为重要。

大量的研究表明，由于算法本身的优化机制，PSO 寻找全局最优值的能力主要取决于种群的多样性[9]，保持多样性一直是多模态优化问题的一个关键方面。在过去几十年中提出了不同的策略，尽可能使粒子保持种群多样性，这些改动大致集中在采用多种群技术和粒子学习策略上的调整。首先，多种群技术允许多个子群在某些特定时期独立演化，信息交互发生在多个小群体中，整个种群则具有较高的多样性。其次，实验证明，与基于静态且单一的索引策略相比，为处于不同状态的粒子指定不同的学习样本对算法性能有积极作用[10]。

受上述研究的启发，本文提出了一种新的基于边界自适应技术的多精英综合学习粒子群算法（Adaptive-Elite Interactive Particle Swarm Optimization，A-EIPSO）。本文主要的研究思路可概括如下：

（1）当粒子在进化过程中突破边界限制，考虑超出边界行为带来的有效信息，自适应地赋予粒子返回合理区域的分布特征，引导粒子对搜索空间中未探测区域进行更好的探索。

（2）种群采取了一种有目的性的初始化方式，试图让粒子在迭代的早期尽可能多地发生突破边界行为，确保粒子经过足够次数的修正从而降低错过最优解的可能。

（3）采用聚类动态划分种群的多种群技术并在学习策略中引入额外交互算子作为粒子更新方程中的指导因素，弥补学习策略中对象单一的缺点，让每个粒子的学习样本更加多元化。

2　相关工作

在实际应用中，研究者们针对上述问题采取相应的优化策略，并将其应用于 PSO 形成多种变体算法。下文将对近年来的研究情况进行简单的回顾。

2.1　边界处理

边界处理技术（Bound-handling Techniques）是影响算法性能的重要有效因素之一，但在相关文献中，研究工作更多地集中在处理函数约束上，而不是处理边界约束[11]。边界处理最早应用于 PSO 的著作之一由 Lampinen[12] 发表，他提出保持粒子的速度不变，将界外的粒子重置在违反侧的边界位置上，该方法也被称为 Projection。Juraez-Castillo 等[13] 提出了一种将修正后的粒子定位在可行区域内但随机偏向两个变量的位置的方法，即计算由 3 个向量形成的区域的质心 Centroid 来重新定位粒子，以避免过早收敛，该方法是特地设计用于优化差分进化，以解决一系列约束条件下的优化问题。Zhang 等[14] 没有改变任何粒子特征，而是将搜索空间扩展了无限多个副本，提出了一种周期解空间的方法，称为 Periodic。Helwig 等[15] 采用镜像搜索空间 Bounded Mirror，该方法以镜像的形式对搜索空间进行扩展，并以环形方式重新连接从而避免空间不连续。

2.2　多种群技术

多种群技术最初是结合小生境技术引入的[16]，由于其在保持种群多样性方面的良好表现而被广泛关注。多种群技术也可以被理解为粒子邻域拓扑结构的改变，这种改变影响着个体间的信息交互方式，驱动着整个种群的进化。

邻域拓扑可分为静态拓扑结构和动态拓扑结构。在静态拓扑结构研究中，Peram 等[17] 提出了一种新的基于适应度距离比的粒子群优化算法（Fitness-distance-ratio based Particle Sswarm Optimization，FDR-PSO），该算法将粒子移向距离本身固定距离内适应值更高的粒子，而不是将每个粒子吸引到任何粒子迄今发现的最佳位置；一个动态的拓扑结构更符合社会群体的进化特征，更有利于个体之间的信息流动和交互，因此 Liang 等[18] 提出了（Dynamic Multi-Swarm Particle Swarm Optimization，DMS-PSO）。该方法将整个种群划分为许多子群，并使用局部状态的子群更新粒子的速度和位置。这些子群是动态重组的，它们的信息通过重组计划及时交换，因此 DMS-PSO 在复杂问题上表现得更好。

随着对多种群技术的深入研究，研究者们开始注意到

基于子群间新颖的学习策略。Lynn 等[19]为更好地平衡粒子探索和开发，提出了一种异构综合学习粒子群优化(Heterogeneous Comprehensive Learning Particle Swarm Optimization，HCLPSO)。该算法将群体划分为两个子种群，每个子种群被指定只专注于探索或开发中的一项。在探索子群中，样本是利用种群本身内部的个人最佳粒子来生成的，而在开发子群中，则使用整个群体的个人最佳粒子来生成样本。因为探索子群并不从开发子群中的任何粒子学习，即使开发子种群过早收敛，也能保持探索子群的多样性。Liang 等[20]为增加粒子学习的不确定性，更好地维持种群多样性，提出了一种基于交叉学习策略取代全局学习策略的混合粒子群优化(Particle Swarm Optimization with Criss-cross Learning Strategy，PSO-CL)，此外还引入一种随机实例学习策略(SEL)，帮助集体信息在不同的子群之间传播，提高算法的局部利用能力。

除了应用于 PSO，多种群技术还广泛应用于其他进化算法，如人工蜂群(ABC)算法[21]。在该算法中，将蜂群划分为多个子群，使每个子群中存在觅食者。在每一次迭代中，通过并行计算来定位不同的最优值，进一步提高种群多样性，以防止算法陷入局部最优。

3 A-EIPSO 算法的设计

本文结合了边界自适应处理技术(Adaptive Bound Handling Techniques，ABHT)和多精英交互学习策略(Elite Interactive Learning Strategy，EI)，提出了一种新的算法(A-EIPSO)。本节将详细介绍该算法的关键策略并辅以算法框图予以说明。

3.1 边界自适应处理

第 2 节描述了现有的边界处理技术，但大多都停留在使离开搜索空间的粒子通过某种固定方式重新定位在可行区域，很少有研究关注在解空间约束下边界处理技术与寻优机制间的联系，专门研究碰撞行为能否为粒子寻优提供一定指导价值的研究则更少。因此，除了修正粒子的位置外，本文尝试通过边界自适应处理技术使粒子的碰撞行为不再盲目并为寻优提供积极作用，以此提高算法的优化效率。除此以外，Helwig[15]证明了过去文献记录的大多数边界处理技术会在搜索中引入显著的搜索偏差，从而影响搜索行为。特殊情况下，搜索偏差可以提升算法的优化性能，例如，如果已知最佳值将位于中心或接近边界，则可以有意使用这种偏差，从而找到最优解。然而，在缺乏这种先验知识的情况下，或许一种没有偏差的技术更具备实用性。

基于上述分析，本文提出了一种新的方法，即边界自适应处理技术。对于优化过程中超越边界的粒子，根据其发生碰撞前的状态信息为粒子提供更佳的重置选择。不同于 Juarez-castillo[13]和 Gandomi[22-23]等运用的方法，本文不再考虑全局空间，而将越界粒子重定位的允许区域限制在边界与粒子碰撞前的位置 X_i^d 之间，由此提出了一个能够自适应越界粒子与边界之间距离变化的概率分布函数。粒子按式(3)分布在允许区域。

$$X_i'^d=\begin{cases}X_i^d+\xi(up\_bound-X_i^d), & \text{if } X_i'^d>u\_bound\\ X_i^d-\xi(X_i^d-low\_bound), & \text{if } X_i'^d<l\_bound\end{cases} \tag{3}$$

其中，位置系数 ξ 满足下述分布：

$$P_d=\frac{\lambda * d_{\text{violate}}}{\alpha * \tan^{-1}\dfrac{d_c-d_{\text{violate}}}{\lambda * d_{\text{violate}}}},0\leqslant d\leqslant d_c \tag{4}$$

$$\alpha=(d-d_{\text{violate}})^2+\lambda^2 d_{\text{violate}}^2 \tag{5}$$

基于边界与粒子迭代前位置路径上的逆抛物型概率分布函数计算可行解的方法满足式(4)，其中 d_{violate} 为空间边界与越界粒子解之间的绝对距离，λ 为预先定义好的参数，d_c 为截断距离，即允许的最大越界距离，即 $d_{\text{violate}}\leqslant d_c$。

粒子修正后的新位置在可行区域内的分布特征如图 1 所示。X_i^d 表示越界粒子的位置，u_bound 和 l_bound 分别表示搜索空间的上、下边界。从图 1 可以看到，此边界处理方法考虑了粒子碰撞前的位置信息以及越界距离。粒子越界距离越大，粒子在允许区域内的分布概率密度越不均匀，在边界附近有一个显著的峰值；而离边界越近，分布概率密度则越趋于均匀，使得新位置以更大的概率落在靠近上一次可行解的位置。这样合理保留了粒子经过信息交互后得到的优化方向，并且利用了越界粒子和搜索边界关系中的有用信息，从而降低了错过最优解的可能。

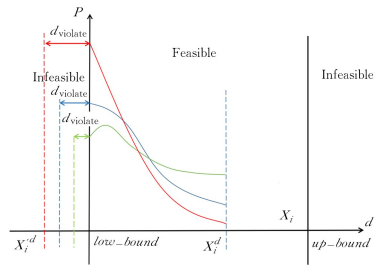

图 1 基于边界自适应处理技术的粒子定位概率密度图

Fig. 1 Probability density of particle location based on adaptive bound handling techniques

与此同时，为了配合提出的边界处理技术，进一步发挥粒子碰撞行为的价值，本文提出了一种新的初始化配置。

$$f(x,a,b)=\frac{\Gamma(a+b)x^{a-1}(1-x)^{b-1}}{\Gamma(a)\Gamma(b)} \tag{6}$$

将式(6)中 a 的值调整为 0.4，b 的值调整为 0.4，得到一个在[0,1]内成 U 型的概率密度函数作为粒子初始化的位置函数。接着，将粒子初始概率分布映射到解空间的右半区间并关于解空间中心镜像，使粒子集中分布在整个搜索范围内靠近中心和两端边界的位置，这样既保证了粒子的全局分布，又让粒子在迭代的初期尽可能多地发生碰撞，多次利用边界处理技术排除非优值解以及部分局部最优解。粒子初始化概率密度分布函数以及散点分布(粒子数量为 100)分别如图 2、图 3 所示。

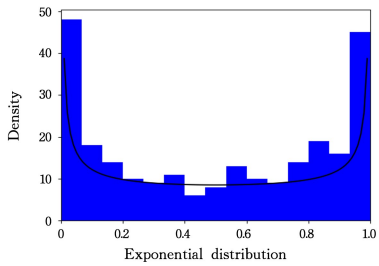

图 2 粒子初始化分布密度图

Fig. 2 Distribution density of particle's initialization

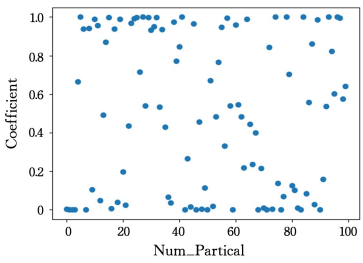

图 3　粒子初始化分布散点图

Fig. 3　Scatter distribution of particle's initialization

3.2　多种群精英交互学习

如式(1)所示，在标准 PSO 中每个粒子的飞行方向仅基于其自身的历史最优位置和全局最优位置，其中全局最优位置的强注入导致 PSO 的收敛速度较快。因此，本文试图让每个粒子的学习样本更加多元化。

3.2.1　种群划分

基于多种群技术的学习策略可以保持种群的多样性，而其中子群的划分机制及划分结果对算法性能起着重要作用[24]。本文通过一种高性能的聚类方法[25]动态地将粒子划分成若干个子群。该方法能够自动发现聚类中心，提高子群划分过程的适应性。此外，基于局部密度的概念和一步聚类分配方法，可以对任意形状的数据集进行有效聚类。

其次，考虑过多的信息可能会导致更新的方向过于模糊，这样不利于收敛[26]。因此，本文将种群数量固定为 3，这也符合本文的初始化分布策略，即使种群的数量不发生改变，但粒子在优化过程中与子群的从属关系却会发生变化。

3.2.2　精英交互学习策略

尽管多种群策略可以在寻优中显示出良好的性能，但在 DMS-PSO、HCLPSO 中子群仅依靠各自内部信息进行搜索，子群间相互独立，缺乏信息共享，无法在指导粒子学习策略上提供令人满意的特性。因此，本文提出了一种精英学习策略，通过聚类将种群划分为多个子群后，子群内又根据粒子状态对粒子进行区分，为不同类别的粒子指定不同的学习策略来指导粒子的搜索行为，并在学习策略中引入交叉算子促进集体信息在不同的子群之间传播，增加子群间信息交换的信息源数量并增大信息共享的范围，使每个子群都可以在一定程度上与其余子群保持联系，进一步提高了子群内、子群间样本的丰富性。

具体来说，各子群 sub_s，$s=[1,2,3,\cdots,S]$中的精英粒子就是所属子群内的局部最佳粒子，这类粒子通常负责探索未知的搜索空间，在此基础上进一步引入多个子群间精英粒子之间的信息交换，让单个群体学习到更多的优质社会信息，赋予这些探险者良好的勘探能力，以找到有前途的区域。精英粒子的学习策略如下：

$$V_i'^d=\omega V_i^d+c_1r_1(pB_i^d-X_i^d)+c_2r_2\left(\left(\frac{1}{N}\sum_{s=1}^{S}sub\_B_s^d\right)-X_i^d\right) \tag{7}$$

$$X_i'^d=X_i^d+V_i^d \tag{8}$$

其中，ω 是惯性权重，常数 c_1 和 c_2 是加速度系数，r_1 和 r_2 是[0，1]中两个均匀分布的随机数，S 是子群的数量，$sub\_B_s^d$ 是子群 s 中的精英粒子。可以观察到，相比式(1)，式(7)中的社会学习范例由 S 个实例的平均值取代全局最优，其中集成了来自不同子群的更多信息，与原始粒子群优化算法相比，粒子之间保持了更强的相互关系，这种修改对于提高种群多样性尤为重要。

因为考虑到精英粒子是最有可能找到最优解的粒子，它们的信息可以为找到最优解提供极具价值的指导，因此利用这些信息来指导子群中绝大部分的普通粒子的更新过程，以增加种群的局部多样性。普通粒子的学习策略的定义如下：

$$V_i'^d=\omega V_i^d+c_1r_1(pB_i^d-X_i^d)+c_2r_2(sub\_B_s^d-X_i^d) \tag{9}$$

$$X_i'^d=X_i^d+V_i^d \tag{10}$$

与标准 PSO 的速度更新式式(1)相比，式(9)使用精英粒子代替全局最佳粒子指导隶属于同一子群中的普通粒子，这样做既考虑了粒子的“认知”和“社会”两个因素，也允许种群学习更多样的搜索信息，因为此时的精英粒子通过交互学习策略集成了其余子群的良好社会信息。

3.3　算法流程

A-EIPSO 的完整流程如图 4 所示，首先根据粒子的分布特征及设置的种群数量自适应地确定子群的规模；接着，评估种群适应度，确定子群 s 中的精英粒子，通过在学习策略中引入多个子样本之间的信息交换，执行与来自其余子群中附带有效探索信息的的精英粒子间的交互操作，保持种群间信息共享，维持种群多样性；同时它还用于指导该子群中普通粒子的学习过程；与此同时，在粒子学习的过程中，如果发生越过边界的情况便采取自适应处理(ABHT)来修正越界粒子的位置。最后，当满足终止条件时，搜索过程停止。

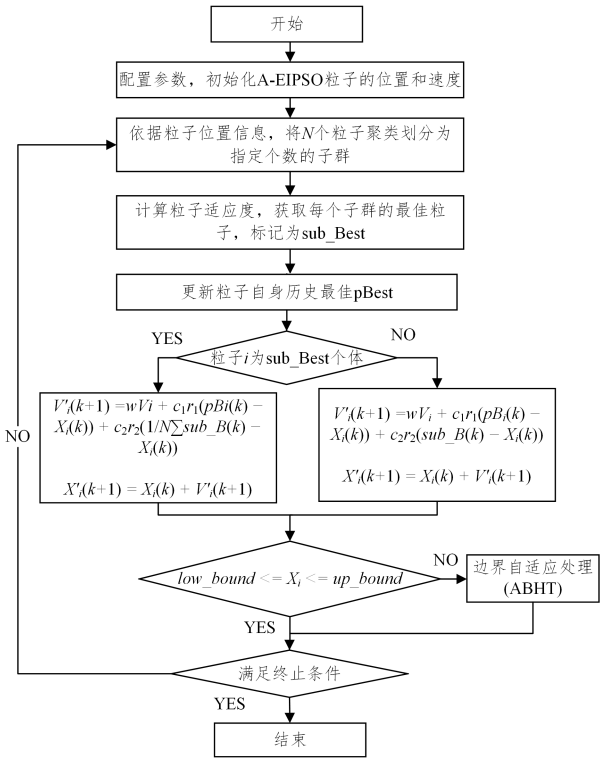

图 4　A-EIPSO 算法的流程图

Fig. 4　Flowchart of A-EIPSO

4　实验验证与比较

本节进行了多项实验以尽可能全面地评估 A-EIPSO 算法的性能。首先，本文从在可行解范围内粒子搜索特性的角度，分析了运用不同边界处理技术的 PSO 算法中粒子的搜索

倾向，其中包括第 1 节介绍的多种处理技术以及本文提出的自适应处理技术；其次，比较了基于不同边界处理技术的标准 PSO 算法在求解精度方面的表现；然后，基于 CEC2017 测试套件[27]，通过与多种经典的 PSO 变体算法即其他进化算法进行比较，进一步分析了 A-EIPSO 的综合性能，设计了统计学上的非参数检验，对实验结果进行了统计显著性检验；最后，本文还进一步对算法的计算代价进行了比较，提供了更为全面的性能综合评估。

4.1 边界处理

4.1.1 实验准备

为评估文献[11-15]中提及的各种边界处理技术在优化中的性能，本文赋予标准 PSO 算法不同的边界处理技术以处理 6 个著名的标准测试函数。表 1 列出了对函数的详细描述，包括函数编号、函数名称、维度、每个维度的搜索空间和基准函数的最小值。

表 1 标准测试函数配置

Table 1 Configuration of benchmark functions

Functions	Name	Dimension	Search space	Minimum
$f1$	Salomon	100	[−100,100]	0
$f2$	Rastrigin	100	[−5.12,5.12]	0
$f3$	Ackley	100	[−32,32]	0
$f4$	Rosenbrock	100	[−30,30]	0
$f5$	Griewank	100	[−600,600]	0
$f6$	Schwefel 1.2	100	[−100,100]	0

实验中粒子的速度和位置更新如式(1)和式(2)所示，算法中的参数则是根据 Bratton 等[28]的标准 PSO 中提出的值选择的，设置为 $c_1=c_2=1.49$，$\chi=0.72$。需要特别说明，在接下来的实验中，使用自适应边界处理技术的案例中均额外对种群做了基于式(6)的初始化分布。

4.1.2 搜索偏差特性测试

本节将分析边界处理方法对探索行为的影响。其思想是在迭代过程中对所有可行点进行采样，经过一定的采样次数后观察样本在不同区域的分布，并将其与均匀分布进行比较，从而检测到因采用不同处理策略而带来的偏差。

本文基于标准 PSO 在 6 个基准测试函数上进行了实验。将维数 D 设置为 30，群体大小设置 40，最大迭代次数设置为 300 000。为了对不同处理技术进行分析，本文将每个测试函数的搜索空间及粒子的位置信息统一标准化到相同的区间[−100,100]内，并划分为 20 个大小相等的区间。对于每个区间以及相同采样间隔，本文记录了该区间内粒子的数量并将其转换为分布密度，各算法独立运行 30 次。相同采样间隔下粒子位置样本分布的统计结果如图 5 所示，可以看到采用自适应边界处理技术的粒子样本分布在可行解域内无明显偏差，即实现搜索空间内的均匀探索，保证全局多样性。因此，自适应处理 Adaptive 也是各种边界处理方法中可减小迭代过程中忽略最优解概率的方法之一。

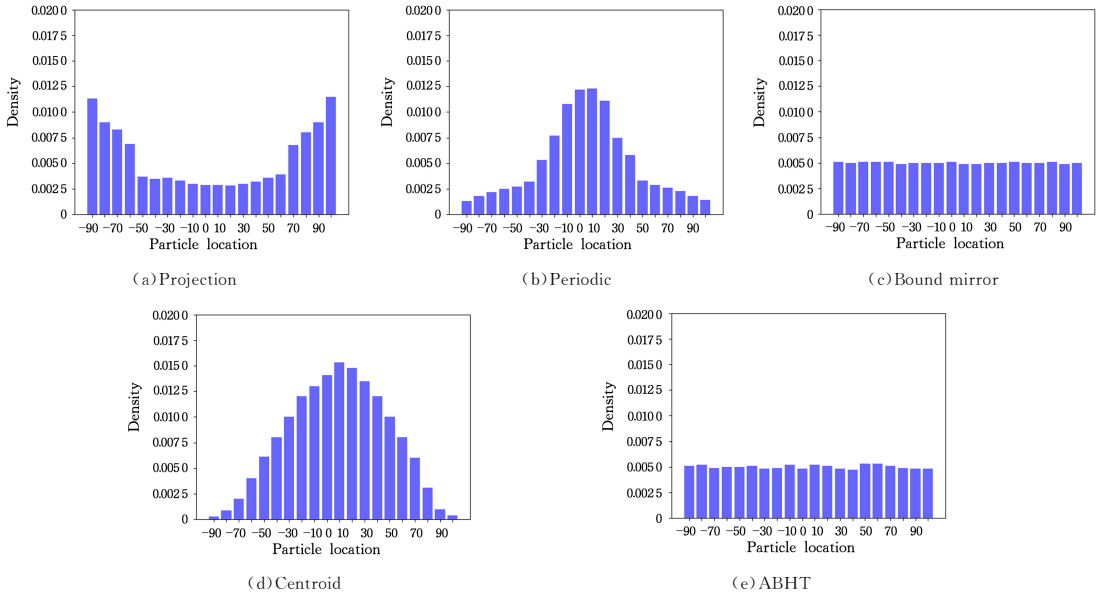

图 5 不同边界处技术下的评估样本分布特征图

Fig. 5 Distribution characteristics of evaluated samples with different bound handling technologies

4.1.3 数值实验

本文中，PSO 算法得到了新的边界处理方式。下文将使用历史文献中的边界处理技术在 6 个标准函数上进行求解精度测试，以评估这些技术的效率。所有问题均使用 100-D 参数空间进行测试，每个实验都独立重复了 30 次，以避免随机影响。本文将首先展示实验结果，然后提供解释。

表 2 列出了不同处理技术下关于每个基准函数的适应度值的平均值和标准差，以证明各种 PSO 算法的性能。此外，所提边界自适应处理技术的性能测试如图 6 所示。纵坐标表示算法平均最佳适应度值的常用对数，横坐标表示评估次数。

本文尝试从函数最优解的位置特点角度出发对结果进行分析。首先，$f2$(Rastrigin)是一个全局最优解，位于解空间中心的狭窄深谷处，其周围分布着众多局部最优解的问题，属于高度多模态问题。与倾向空间中心区域搜索的边界处理技术 Centroid 相比，自适应处理能达到同样甚至更好的优化效果，同时在其余具备不同特性的函数上的表现也优于 Centroid 技术。

表 2　测试函数上基于不同边界处理技术的平均值和标准误差(100-D)

Table 2　Average values and standard errors of different bound handling strategies on all tested benchmarks(100-D)

Function	Criteria	init to Pbest	Projection	Reinit	Periodic	Mirror	Centroid	Adaptive
$f1$	Mean	1.30×10^{1}	2.86×10^{1}	9.74	8.89	2.61	6.17	1.35
	Std	6.61×10^{-1}	1.05	3.14×10^{-2}	3.31×10^{-1}	1.46	2.54×10^{-1}	3.38×10^{-1}
$f2$	Mean	2.70×10^{2}	3.26×10^{2}	2.16×10^{2}	1.65×10^{2}	1.50×10^{2}	1.27×10^{2}	1.17×10^{2}
	Std	3.36	3.96	5.22×10^{7}	2.12	2.57	6.18	2.25
$f3$	Mean	8.87	1.10	6.37	5.87	2.31	5.91	1.07
	Std	2.64×10^{-1}	5.11×10^{-1}	6.50×10^{-2}	7.41×10^{-1}	3.26×10^{-1}	3.31×10^{-1}	3.33×10^{-1}
$f4$	Mean	2.23×10^{2}	8.22×10^{2}	1.64×10^{2}	1.23×10^{2}	2.31×10^{1}	1.02×10^{2}	1.29×10^{1}
	Std	8.23	3.25×10^{1}	9.32	9.17	3.26	3.80	1.03
$f5$	Mean	7.23×10^{-2}	9.22×10^{-5}	3.53×10^{-2}	1.47×10^{-2}	8.89×10^{-2}	1.93×10^{-2}	6.43×10^{-5}
	Std	3.52×10^{-3}	5.08×10^{-4}	2.84×10^{-3}	3.28×10^{-3}	1.25×10^{-3}	6.89×10^{-4}	1.90×10^{-4}
$f6$	Mean	5.22×10^{1}	6.59×10^{1}	2.49×10^{1}	1.48×10^{1}	6.76×10^{2}	9.98	5.54×10^{-2}
	Std	3.63	1.18×10^{1}	4.73	2.27	2.59×10^{1}	1.12	2.84×10^{-5}

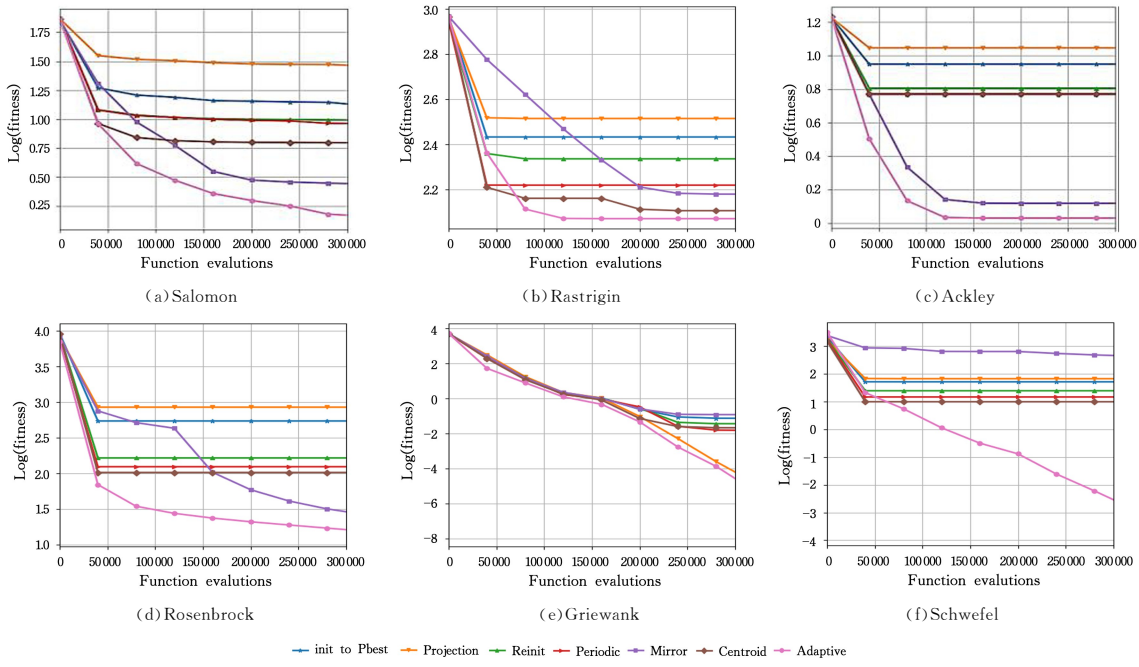

图 6　6 个测试函数的收敛曲线

Fig. 6　Convergence graphs of six test functions

其次，$f5$(Griewank)上下边界处各有一个深谷，是一个在边界上具有局部最优的问题，其他位置则较为平坦。如表 2 所列，由于上述特性阻止了许多群集接近中心位置的全局最优值，获得较好结果的边界处理技术为搜索偏向边界处 Projection 和本文提出的自适应处理 Adaptive。

最后，函数 $f6$(Schwefel 1.2)也代表了一个复杂的问题，该函数的自变量具有上位性，因此其梯度方向不会沿着轴线方向变化，同时全局最优解远离局部最优解，拥有较高的寻优难度。如表 2 所列，自适应处理技术在函数 $f6$ 上再次具有明显的优势，这表明自适应处理技术对于复杂的多模态问题具有良好的性能。

本节验证了采取自适应边界处理技术(ABHT)后，粒子能在搜索空间内实现均匀搜索的特性，并与运用其他边界处理技术的标准 PSO 算法在基准测试函数上对求解精度进行了比较。根据表 2 所列的结果显示，自适应处理技术所获得的平均适应值对于大多数函数都是突出的。

4.2　基于 CEC2017 的 PSO 变体算法比较

4.2.1　实验准备

本实验选择了 CEC2017 单目标数值优化测试套件来进一步验证 A-EIPSO 的有效性。套件中包含了 3 个单峰函数($f1-f3$)、7 个简单多峰函数($f4-f10$)、10 个混合函数($f11-f20$)和 10 个合成函数($f21-f30$)。为进行更为全面的评估与分析，本节将本文提出的 A-EIPSO 算法与 5 种先进 PSO 变体算法(FDR-PSO，DMS-PSO，HCLPSO，HCLDMS-PSO 和 PSO-CL)以及 2 种主流进化算法(ABC，SHADE)在优化性能和计算代价方面进行比较。其中 2 种主流进化算法分别是人工蜂群(ABC)和 Tanabe 等[29]提出的 CEC2014 中性能最好的 DE 改进算法 SHADE，其中 SHADE 算法成功利用历史经验指导子代选择一系列控制参数，实现控制参数的自适应调整。表 3 列出了上述所有算法和 A-EIPSO 的详细参数设置，其中作为比较算法的参数设置与相应论文中的参数配置相同。

表 3 PSO 变体算法及其他进化算法参数配置

Table 3 Parameter settings of PSO variant algorithms and other evolutionary algorithms

Algorithms	Year	Parameters settings	Ref.
ABC	2007	$limit=100, Size\ of\ employed\text{-}bee=N/2$	[21]
SHADE	2013	$Pbest=0.1, Arc\ rate=2$	[29]
FDR-PSO	2003	w:0.9～0.4, a=0.000035, b=0.5, c=0, d=1.5	[17]
DMS-PSO	2005	$w=0.729, c_1=c_2=1.49, V_{\max}=0.5*Range$	[18]
HCLPSO	2015	w:0.99～0.29, c_1:2.5～0.5, c_2:0.5～2.5, K:3～1.5, $V_{\max}=0.5*Range$	[19]
PSO-CL	2021	w:0.9～0.4, $c_1=c_2=2.0, \lambda_r=4, \lambda_c=09, \lambda_m=0.5$	[20]
HCLDMS-PSO	2020	w:0.99～0.29, w_2, c_1:2.5～0.5, c_2:0.5～2.5, $P_m=0.1, V_{\max}=0.5*Range$	[30]
A-EIPSO	—	w:0.9～0.4, $c_1=c_2=1.49$	

4.2.2 数值实验

在 CEC2017 测试套件的指导下，本文将搜索范围设置为[−100,100]，所有 PSO 变体算法和本文提出的 A-EIPSO 配置为相同的种群大小 40，而其他 2 种进化算法的种群规模设置为 100。函数维度设置为 $D=30$，功能评估的最大数量为 $10000*D$，每种算法运行 30 次，并使用平均最优值(Mean)和均方差(Std)来描述算法的性能。各种算法在测试套件中的表现细节如表 4 所列，每个测试函数的最佳结果都用加粗进行标记。同时显著性水平 $\alpha=0.05$ 的 Wilcoxon 符号秩[31]检验结果也如表 4 所列，以验证 A-EIPSO 是否能够提供比对比算法更可靠的性能。符号“+”“=”和“−”分别表示 A-EIPSO 表现“明显优于”“相似于”和“明显差于”对比算法。

表 4 CEC2017 基准函数解的精度比较(30-D)

Table 4 Comparison of solution accuracy of CEC2017 benchmark functions(30-D)

Function	Criteria	ABC	SHADE	FDR-PSO	DMS-PSO	HCLPSO	PSO-CL	HCLDMS-PSO	A-EIPSO
$f1$	Mean	1.29×10^{2}(−)	**0.00(−)**	3.75×10^{3}(+)	2.40×10^{3}(=)	7.33×10^{1}(−)	1.81×10^{3}(=)	6.78×10^{2}(−)	2.38×10^{3}
	Std	7.58×10^{1}	0.00	4.68×10^{3}	1.64×10^{3}	1.48×10^{2}	1.50×10^{2}	2.51×10^{2}	1.8×10^{3}
	Rank	3	1	8	7	2	5	4	6
$f2$	Mean	6.18×10^{4}(−)	**1.78×10^{2}(−)**	8.48×10^{9}(+)	7.29×10^{8}(+)	3.14×10^{6}(=)	3.00×10^{8}(+)	6.55×10^{7}(+)	3.01×10^{6}
	Std	1.36×10^{4}	2.57×10^{2}	3.96×10^{10}	5.22×10^{7}	3.12×10^{7}	8.18×10^{10}	3.13×10^{7}	5.54×10^{7}
	Rank	2	1	8	7	4	6	5	3
$f3$	Mean	1.33×10^{5}(+)	9.21×10^{3}(+)	1.52×10^{-6}(−)	2.23×10^{3}(+)	1.46×10^{-3}(−)	**3.42×10^{-8}(−)**	2.83×10^{1}(−)	8.86×10^{2}
	Std	1.75×10^{4}	1.28×10^{1}	2.86×10^{-6}	2.72×10^{3}	3.49×10^{-3}	1.80×10^{-7}	4.28	3.42×10^{1}
	Rank	8	7	2	6	3	1	4	5
$f4$	Mean	3.31×10^{1}(−)	**4.14×10^{1}(−)**	4.18×10^{2}(+)	6.19×10^{1}(=)	6.84×10^{1}(=)	4.21×10^{1}(−)	5.88×10^{1}(+)	6.51×10^{1}
	Std	8.28	3.07	3.25×10^{2}	5.32×10^{1}	2.17×10^{1}	2.80×10^{1}	1.62×10^{1}	3.29×10^{1}
	Rank	1	2	8	3	7	4	5	6
$f5$	Mean	8.61×10^{1}(+)	**2.48×10^{1}(−)**	5.22×10^{1}(+)	4.64×10^{1}(+)	4.13×10^{1}(+)	4.53×10^{1}(+)	2.75×10^{1}(−)	3.84×10^{1}
	Std	2.44×10^{1}	1.01×10^{1}	1.08×10^{1}	2.84×10^{1}	8.28	6.89	3.65	8.31
	Rank	8	1	7	6	4	5	2	3
$f6$	Mean	**0.00(−)**	1.45×10^{-1}(−)	6.59×10^{-2}(−)	8.25×10^{-8}(−)	3.82×10^{-13}(−)	4.78×10^{-4}(−)	**0.00(−)**	6.37
	Std	0.00	1.13×10^{-4}	1.18×10^{-1}	2.85×10^{-7}	1.33×10^{-13}	1.51×10^{-6}	3.71	1.68
	Rank	1	6	5	3	2	4	1	7
$f7$	Mean	1.14×10^{2}(+)	6.97×10^{1}(=)	1.02×10^{2}(+)	9.25×10^{1}(+)	8.58×10^{1}(+)	7.75×10^{1}(=)	**5.76×10^{1}(=)**	6.82×10^{1}
	Std	2.24×10^{1}	2.84×10^{1}	1.89×10^{1}	4.89×10^{1}	9.07	1.13×10^{1}	1.12×10^{1}	1.45×10^{1}
	Rank	8	3	7	6	5	4	1	2
$f8$	Mean	8.85×10^{1}(+)	**2.75×10^{1}(−)**	5.42×10^{1}(+)	4.83×10^{1}(=)	5.02×10^{1}(+)	4.56×10^{1}(=)	2.83×10^{1}(−)	4.63×10^{1}
	Std	2.12×10^{1}	8.31	1.04×10^{1}	6.13	8.73	8.07	1.05×10^{1}	2.28×10^{1}
	Rank	8	1	7	5	6	3	2	4
$f9$	Mean	8.21×10^{2}(+)	2.71×10^{1}(+)	1.74×10^{2}(+)	1.20×10^{-1}(−)	1.06×10^{1}(+)	2.82(=)	**7.86×10^{-2}(−)**	3.14
	Std	8.66×10^{2}	4.21×10^{2}	1.65×10^{1}	6.25×10^{-1}	8.47	3.92×10^{1}	3.42×10^{-2}	6.92×10^{-1}
	Rank	8	6	7	2	5	3	1	4
$f10$	Mean	2.32×10^{3}(+)	1.98×10^{3}(=)	3.07×10^{3}(+)	2.92×10^{3}(+)	2.31×10^{3}(+)	2.74×10^{3}(+)	2.36×10^{3}(+)	**1.76×10^{3}**
	Std	2.71×10^{2}	5.52×10^{2}	7.25×10^{2}	2.81×10^{2}	3.41×10^{2}	2.97×10^{2}	5.41×10^{2}	3.81×10^{2}
	Rank	7	2	8	6	3	5	4	1
$f11$	Mean	4.27×10^{2}(+)	8.62×10^{1}(+)	8.74×10^{1}(+)	3.78×10^{1}(=)	6.90×10^{1}(+)	3.83×10^{1}(=)	**3.38×10^{1}(=)**	3.76×10^{1}
	Std	2.66×10^{2}	2.64×10^{1}	3.96×10^{1}	2.82×10^{1}	2.05×10^{1}	3.57×10^{1}	1.28×10^{1}	1.56×10^{1}
	Rank	8	6	7	3	5	4	1	2
$f12$	Mean	4.12×10^{5}(+)	**4.58×10^{3}(−)**	2.34×10^{4}(=)	1.65×10^{5}(+)	2.90×10^{4}(=)	9.52×10^{4}(+)	6.23×10^{4}(+)	2.83×10^{4}
	Std	8.74×10^{4}	2.38×10^{3}	1.12×10^{4}	1.12×10^{5}	1.36×10^{4}	1.22×10^{4}	6.31×10^{3}	1.77×10^{4}
	Rank	8	1	2	7	4	6	5	3
$f13$	Mean	6.69×10^{3}(+)	**2.37×10^{2}(−)**	1.41×10^{4}(+)	1.06×10^{4}(+)	6.23×10^{2}(=)	2.34×10^{3}(+)	6.68×10^{3}(+)	8.25×10^{2}
	Std	2.37×10^{3}	1.88×10^{2}	1.02×10^{4}	9.26×10^{3}	2.83×10^{2}	7.18×10^{2}	4.52×10^{2}	1.46×10^{2}
	Rank	6	1	8	7	2	4	5	3
$f14$	Mean	6.25×10^{4}(+)	**1.24×10^{2}(−)**	4.48×10^{3}(+)	5.62×10^{3}(+)	5.78×10^{3}(+)	1.13×10^{4}(+)	2.55×10^{3}(+)	8.03×10^{2}
	Std	8.45×10^{3}	4.28×10^{2}	2.87×10^{3}	3.76×10^{3}	3.65×10^{3}	1.52×10^{3}	1.12×10^{3}	2.54×10^{2}
	Rank	8	1	5	5	6	7	3	2

（续表）

Function	Criteria	ABC	SHADE	FDR-PSO	DMS-PSO	HCLPSO	PSO-CL	HCLDMS-PSO	A-EIPSO
	Mean	$1.08\times10^{3}(-)$	$\mathbf{2.13\times10^{2}(-)}$	$5.26\times10^{3}(=)$	$5.33\times10^{3}(=)$	$2.98\times10^{2}(-)$	$1.44\times10^{3}(-)$	$8.47\times10^{2}(-)$	5.24×10^{3}
$f15$	Std	9.22×10^{1}	4.53×10^{1}	7.08×10^{3}	4.23×10^{3}	1.14×10^{2}	2.70×10^{2}	6.34×10^{2}	2.97×10^{3}
	Rank	4	1	7	8	2	5	3	6
	Mean	$6.82\times10^{2}(+)$	$4.47\times10^{2}(+)$	$7.00\times10^{2}(+)$	$4.46\times10^{2}(+)$	$4.81\times10^{2}(+)$	$4.51\times10^{2}(+)$	$3.38\times10^{2}(+)$	$\mathbf{2.46\times10^{2}}$
$f16$	Std	3.31×10^{2}	2.63×10^{2}	2.78×10^{2}	3.29×10^{2}	1.57×10^{2}	1.73×10^{2}	1.55×10^{2}	1.21×10^{2}
	Rank	7	4	8	3	6	5	2	1
	Mean	$2.21\times10^{2}(+)$	$1.15\times10^{2}(+)$	$2.19\times10^{2}(+)$	$\mathbf{6.68\times10^{1}(-)}$	$1.21\times10^{2}(+)$	$1.12\times10^{2}(+)$	$7.65\times10^{1}(=)$	9.68×10^{1}
$f17$	Std	6.39×10^{1}	4.52×10^{1}	1.06×10^{2}	3.95×10^{1}	7.65×10^{1}	8.70×10^{1}	1.38×10^{1}	2.01×10^{1}
	Rank	8	5	7	1	6	4	2	3
	Mean	$1.55\times10^{5}(+)$	$\mathbf{2.81\times10^{2}(-)}$	$8.18\times10^{4}(+)$	$1.19\times10^{5}(+)$	$6.99\times10^{4}(+)$	$9.39\times10^{4}(+)$	$8.24\times10^{4}(+)$	3.18×10^{4}
$f18$	Std	3.96×10^{4}	2.33×10^{1}	4.64×10^{4}	7.15×10^{4}	5.07×10^{4}	2.47×10^{4}	4.25×10^{4}	8.24×10^{3}
	Rank	8	1	4	7	3	6	5	2
	Mean	$1.19\times10^{3}(+)$	$\mathbf{1.28\times10^{2}(+)}$	$5.55\times10^{3}(+)$	$3.58\times10^{3}(+)$	$1.36\times10^{2}(-)$	$3.21\times10^{3}(=)$	$2.51\times10^{3}(=)$	2.33×10^{3}
$f19$	Std	3.85×10^{2}	4.29×10^{1}	6.51×10^{3}	2.89×10^{3}	3.39×10^{2}	3.09×10^{2}	3.82×10^{3}	2.06×10^{3}
	Rank	2	1	8	7	3	3	5	4
	Mean	$2.68\times10^{2}(+)$	$1.54\times10^{2}(+)$	$2.07\times10^{2}(+)$	$1.38\times10^{2}(+)$	$1.45\times10^{2}(+)$	$1.18\times10^{2}(+)$	$1.58\times10^{2}(+)$	$\mathbf{7.16\times10^{1}}$
$f20$	Std	4.29×10^{1}	9.36	6.53×10^{1}	5.67×10^{1}	7.56×10^{1}	9.92×10^{1}	1.09×10^{2}	1.32×10^{1}
	Rank	8	5	7	3	4	2	6	1
	Mean	$2.53\times10^{2}(+)$	$2.43\times10^{2}(=)$	$2.58\times10^{2}(+)$	$2.53\times10^{2}(+)$	$2.48\times10^{2}(=)$	$2.44\times10^{2}(=)$	$2.27\times10^{2}(=)$	$\mathbf{2.25\times10^{2}}$
$f21$	Std	3.28×10^{1}	2.88×10^{1}	1.87×10^{1}	1.36×10^{2}	2.22×10^{1}	3.00×10^{1}	2.13×10^{1}	3.54×10^{1}
	Rank	7	3	8	6	5	4	2	1
	Mean	$4.91\times10^{2}(+)$	$1.88\times10^{2}(=)$	$1.29\times10^{3}(+)$	$1.28\times10^{2}(-)$	$1.06\times10^{2}(-)$	$1.07\times10^{2}(-)$	$\mathbf{1.02\times10^{2}(-)}$	1.86×10^{2}
$f22$	Std	5.87×10^{1}	1.44×10^{1}	1.57×10^{2}	7.28×10^{1}	8.95×10^{1}	3.15	6.10×10^{1}	4.22×10^{1}
	Rank	7	6	8	4	2	3	1	5
	Mean	$4.15\times10^{2}(=)$	$4.10\times10^{2}(=)$	$4.41\times10^{2}(+)$	$3.92\times10^{2}(-)$	$4.32\times10^{2}(+)$	$4.02\times10^{2}(=)$	$\mathbf{3.86\times10^{2}(-)}$	4.08×10^{2}
$f23$	Std	5.87×10^{1}	4.44×10^{1}	1.89×10^{1}	2.33×10^{1}	1.11×10^{1}	1.07×10^{1}	2.11×10^{1}	9.35
	Rank	7	4	6	2	8	3	1	4
	Mean	$4.66\times10^{2}(=)$	$4.45\times10^{2}(-)$	$4.86\times10^{2}(+)$	$4.59\times10^{2}(=)$	$\mathbf{4.43\times10^{2}(-)}$	$4.78\times10^{2}(+)$	$4.68\times10^{2}(=)$	4.65×10^{2}
$f24$	Std	$8.44+10^{1}$	3.72×10^{1}	1.48×10^{2}	2.32×10^{2}	7.56×10^{1}	4.57×10^{1}	5.73×10^{1}	1.33×10^{1}
	Rank	5	2	8	3	1	7	6	4
	Mean	$3.82\times10^{2}(=)$	$3.84\times10^{2}(=)$	$3.91\times10^{2}(+)$	$3.85\times10^{2}(=)$	$3.87\times10^{2}(=)$	$3.84\times10^{2}(=)$	$3.86\times10^{2}(+)$	$\mathbf{3.80\times10^{2}}$
$f25$	Std	2.37×10^{2}	1.12×10^{2}	8.45×10^{1}	1.67×10^{1}	9.88×10^{-1}	1.55	1.29×10^{1}	2.25×10^{1}
	Rank	2	3	7	4	6	3	5	1
	Mean	$3.64\times10^{2}(-)$	$1.33\times10^{3}(+)$	$1.05\times10^{3}(+)$	$1.45\times10^{3}(+)$	$\mathbf{3.52\times10^{2}(-)}$	$1.28\times10^{3}(+)$	$1.25\times10^{3}(+)$	5.69×10^{2}
$f26$	Std	3.11×10^{2}	2.54×10^{2}	6.37×10^{2}	2.68×10^{3}	2.76×10^{2}	5.21×10^{2}	1.46×10^{3}	3.85×10^{2}
	Rank	2	7	4	8	1	6	5	3
	Mean	$5.16\times10^{2}(=)$	$5.12\times10^{2}(=)$	$5.61\times10^{2}(+)$	$5.12\times10^{2}(=)$	$5.15\times10^{2}(=)$	$5.11\times10^{2}(=)$	$5.06\times10^{2}(=)$	$\mathbf{5.08\times10^{2}}$
$f27$	Std	1.21×10^{2}	9.84×10^{1}	1.24×10^{1}	6.94	7.55	4.20	6.83×10^{1}	6.34
	Rank	7	5	8	4	6	3	2	1
	Mean	$4.02\times10^{2}(+)$	$3.51\times10^{2}(+)$	$3.50\times10^{2}(+)$	$3.34\times10^{2}(=)$	$3.49\times10^{2}(+)$	$3.37\times10^{2}(+)$	$3.77\times10^{2}(+)$	$\mathbf{3.17\times10^{2}}$
$f28$	Std	1.34×10^{2}	8.10×10^{1}	6.83×10^{1}	2.34×10^{2}	5.31×10^{1}	4.28×10^{1}	1.98×10^{1}	1.23×10^{1}
	Rank	8	6	5	2	4	3	7	1
	Mean	$6.32\times10^{2}(+)$	$5.33\times10^{2}(+)$	$6.14\times10^{2}(=)$	$5.69\times10^{2}(=)$	$4.98\times10^{2}(-)$	$5.55\times10^{2}(-)$	$\mathbf{4.88\times10^{2}(-)}$	5.95×10^{2}
$f29$	Std	4.12×10^{1}	2.68×10^{1}	8.36×10^{1}	3.41×10^{2}	6.17×10^{1}	8.74×10^{1}	1.68×10^{1}	4.43×10^{1}
	Rank	8	3	7	5	2	4	1	6
	Mean	$5.61\times10^{3}(+)$	$\mathbf{2.31\times10^{3}(-)}$	$5.33\times10^{3}(+)$	$4.81\times10^{3}(+)$	$3.64\times10^{3}(=)(-)$	$3.76\times10^{3}(=)$	$4.09\times10^{3}(=)$	4.03×10^{3}
$f30$	Std	1.47×10^{4}	9.81×10^{2}	2.68×10^{3}	2.12×10^{3}	6.85×10^{2}	7.92×10^{2}	3.48×10^{2}	2.03×10^{2}
	Rank	8	1	7	6	2	3	5	4
Average rank		6.07	**3.20**	6.60	4.87	3.97	4.17	3.37	3.27
Final rank		7	1	8	6	4	5	3	2
w/t/l		20/4/6	10/7/13	25/3/2	15/10/5	13/8/9	13/11/6	14/8/8	—
Algorithms		ABC	SHADE	FDR-PSO	DMS-PSO	HCLPSO	PSO-CL	HCLDMS-PSO	A-EIPSO

在3个单峰函数（$f1-f3$）上，SHADE和POS-CL分别在$f1$，$f2$和$f3$上获得最佳性能。在这类函数问题中A-EIPSO虽没有突出的表现，但由Wilcoxon符号秩检验表明，与FDR-PSO，DMS-PSO相比，A-EIPSO在2个函数上均获得了“明显优于”的符号评价。

采用多峰函数（$f4-f10$）来评估避免过早收敛的能力是一个很好的选择，因为这些函数的解空间中充斥着局部最优的因素，这也意味着粒子群优化算法需要具备更高的种群多样性，在这些问题上获得较好求解精度的可能性才会大大增加。具体而言，如表4所列，A-EIPSO在函数$f10$上获得了最佳优化结果，与此同时，函数$f5$和$f7$代表两个具有大量局部最优解的复杂多峰问题，在这两个函数上再次表现出明显的优势，这表明A-EIPSO对于复杂的多模态问题具有良好的性能。

第三类组合函数（$f11-f20$）的特点在于函数的不同组成部分具有不同的性质。实验结果表明，A-EIPSO在2个组合函数上的平均最优值取得了最佳结果，这略好于DMS-PSO和HCLDMS-PSO，两者分别在1个函数上取得了最佳结果。这些多峰问题的实验结果证实了精英交互学习策略可以有效地保持种群多样性，赋予了A-EIPSO更好的全局搜索能力，

并成功地摆脱了局部最优。

在合成函数方面，SHADE 在 $f30$ 上实现了最佳解决方案并由 Wilcoxon 符号秩检验得到明显优于 A-EIPSO 的符号评价，在 $f24$ 上有类似的实验结果。但 A-EIPSO 在 $f21$，$f25$，$f27$ 和 $f28$ 上的表现最佳，同时也是解决合成这类问题中获得最佳性能次数最多的算法。该结果验证了所提出的学习策略的有效性和 A-EIPSO 解决复杂优化问题的能力。

结合表 4 中给出的结果可以看到，在 30 个测试函数中，SHADE 和 A-EIPSO 分别在 11 个和 7 个函数上表现出了最好的性能，以数量来说 SHADE 位居首位，但可以明显观察到，随着待解决问题的复杂性递增，A-EIPSO 的性能优势却越突出。然而，在单峰函数 $f3$、两个简单的多峰函数 $f6$ 以及组合函数 $f15$ 上 A-EIPSO 算法的表现不佳，这或许可以通过“一类问题中任何性能的提升都会被另一类问题的性能抵消”的优化理论来解释[6]。

考虑到 Wilcoxon 符号检验只是一种成对比较方法，为了进一步说明所有对比算法的综合性能并对它们进行概述，本文使用 Friedman 检验[32]比较所有数据集和所有算法，以提供更可靠的证据。将所有算法都通过 pvalue 小于 0.05 的 Friedman 检验进行比较，表 5 中的 Friedman 检验结果表明，A-EIPSO 获得第二的 Friedman 秩结果，表示 A-EIPSO 的性能显著优于大多数对比算法。

表 5 各对比算法基于 CEC2017 的 Friedman 检验结果

Table 5 Friedman test results of PSO variants on CEC2017 test suite

	ABC	SHADE	FDR-PSO	DMS-PSO	HCLPSO	PSO-CL	HCLDMS-PSO	A-EIPSO
Friedman rank	5.24	**2.31**	6.02	4.57	3.52	3.66	2.83	2.68
General rank	7	1	8	6	4	5	3	2

4.2.3 运行时间对比

此外，本文根据 CEC2017 测试套件的指导方针，拟定可接受解决方案的精确度(ε)，通过记录各算法获得可接受精度所需的执行时间，对算法复杂度进行了实验对比。实验确保在相同的执行环境中比较所有算法，表 6 列出了在 30 次独立运行中所有函数的平均结果，其中最优结果加粗表示。为了便于计算，当算法在最大评估次数内没有找到达到预设精度 ε 的结果时，则将执行整个优化过程所耗费的时间作为本次算法的执行时间。

根据表 6 所列的结果，整个测试套件下 A-EIPSO 得到可接受结果的平均耗时最短，特别是在混合函数和合成函数上获得了最短的平均运行时间。结果表明，尽管 A-EIPSO 需要更频繁地处理越界粒子以及执行子群划分运算，导致在理论上具有相对较高的时间复杂度，但凭借自适应边界处理技术和精英学习策略可以有效地避免局部最优，在实践中只需要较短的执行时间即可获得优质的最终结果。

表 6 算法总体平均执行时间统计结果

Table 6 Statistical results of overall average execution time of algorithms

(单位：s)

	ABC	SHADE	FDR-PSO	DMS-PSO	HCLPSO	PSO-CL	HCLDMS-PSO	A-EIPSO
Total Average	36.83	30.90	40.74	38.48	32.90	35.15	26.41	**21.89**

结束语 本文将 PSO 算法中的边界处理技术与优化学习策略相结合，提出了一种基于边界自适应技术的精英交互学习粒子群算法(A-EIPSO)。为了证明算法的性能，本文进行了广泛的实验，从这些比较结果中可以得出如下结论。

目前现存的边界技术由于其不同的处理方式带来了粒子搜索倾向上的偏差，本文通过特殊的初始化并采用自适应处理技术，使算法避免“振荡”和“两步前进，一步后退”等现象，合理提高了优化性能，加快了收敛速度，并具备广泛的实用性。同时使粒子以最为均匀无偏差的方式对搜索空间进行遍历，不管全局最优在中心或者是靠近边界时都保证了算法的优化性能，提供了良好的平均解质量。

除此以外，基于多种群技术提出的精英交互学习策略对更新速度和位置有主要影响，精英粒子给予更多的样本，提高在随机优化问题中真正个人最佳和真正全局最佳出现并被正确选择作为导向的概率，促使所有粒子都在正确的方向上移动，而不是对每个粒子取相等数量且单一的适应值样本。通过数值实验和算法运行时间对比实验结果可见，本文在保持高度多样性和鼓励快速收敛之间存在合理的权衡。最后的非参数统计检验结果也表明 A-EIPSO 比其他方法具有明显的优越性。

参 考 文 献

[1] EBERHART R C, KENNEDY J. A new optimizer using particle swarm theory [C]// Proceedings of International Symposium on Micro Machine and Human Science(ISMMHS'95). Nagoya, Japan, 1995: 39-43.

[2] EBERHART R C, KENNEDY J. Particle swarm optimization [C]// Proceedings of IEEE International Conference on Neural Network(CNN'95). Perth, Australia, 1995: 1942-1948.

[3] HUANG K Y. A hybrid particle swarm optimization approach for clustering and classification of datasets[J]. IEEE Transactions on Power Systems, 2011, 24(3): 420-426.

[4] YAN T T, LI Z. Intelligent skin cancer detection using enhanced particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2018, 158(20): 118-135.

[5] HO S Y, LIN H S, LIAUH W H. OPSO: Orthogonal particle swarm optimization and its application to task assignment problems[J]. IEEE Transactions on Systems Man Cybernetics-Systems, B, 2008, 38(2): 288-289.

[6] ZHANG S, XU J, LEE L H. Optimal computing budget allocation for particle swarm optimization in stochastic optimization [J]. IEEE Transactions on Evolutionary Computation, 2017,

21(2):206-219.

[7] YANG X S,DEB S,ZHAO Y X,et al. Swarm intelligence:past, present and future[J]. Soft Computing, 2018, 22(14): 5923-5933.

[8] HELWIG S,WANKA R. Theoretical analysis of initial particle swarm behavior [C]// Proceedings of the International Conference on Parallel Problem Solving from Nature. DBLP, PPSN, 2008:889-898.

[9] KADIRKAMANATHAN V,SELVARAJAH K,FLEMING P J. Stability analysis of the particle dynamics in particle swarm optimizer[J]. IEEE Transactions on Evolutionary Computation, 2006,10(3):245-255.

[10] WEI B,XIA X,YU F,et al. Multiple adaptive strategies based particle swarm optimization algorithm[J]. Swarm and Evolutionary Computation,2020,57(23):100731.

[11] XU S,RAHMAT-SAMII Y. Boundary conditions in particle swarm optimization revisited[J]. IEEE Transactions on Antennas and Propagation,2007,55(3):760-765.

[12] LAMPINEN J. A constraint handling approach for the differential evolution algorithm [C]// Proceedings of the Congress on Evolutionary Computation. 2002:1468-1473.

[13] JUAREZ-CASTILLO E,ACOSTA-MESA H G,MEZURA-MONTES E. Empirical study of bound constraint-handling methods in Particle Swarm Optimization for constrained search spaces [C]// Evolutionary Computation. IEEE,2017:604-611.

[14] ZHANG W J,XIE X F,BI D C. Handling boundary constraints for numerical optimization by particle swarm flflying in periodic search space [C]// Proceedings of the IEEE Congress on Evolutionary Computation. 2004:2307-2311.

[15] HELWIG S,BRANKE J,MOSTAGHIM S. Experimental analysis of bound handling techniques in Particle Swarm Optimization[J]. IEEE Transactions on Evolutionary Computation, 2013,17(2):259-271.

[16] BOSE D,BISWAS S,KUNDU S,et al. A strategy pool adaptive artificial bee colony algorithm for dynamic environment through multi-population approach [C]// Proceedings of the International Conference on Swarm,Evolutionary,and Memetic Computing. 2012:611-619.

[17] PERAM T,VEERAMACHANENI K,MOHAN C K. Fitness-distance-ratio based particle swarm optimization [C]// Proceedings of the 2003 IEEE Swarm Intelligence Symposium. 2003: 174-181.

[18] LIANG J J,SUGANTHAN P N. Dynamic multi-swarm particle swarm optimizer[C]// Proceedings of the IEEE Swarm Intelligence Symposium. 2005:124-129.

[19] LYNN N,SUGANTHAN. Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation[J]. Swarm & Evolutionary Computation, 2015, 24(5):11-24.

[20] LIANG B,ZHAO Y,LI Y. A hybrid particle swarm optimization with crisscross learning strategy[J]. Engineering Applications of Artificial Intelligence,2021,105(3):104418.

[21] KARABOGA D,BASTURK B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm[J]. Journal of Global Optimization, 2007, 39(3):459-471.

[22] GANDOMI A H,KASHANI A R. Evolutionary bound constraint handling for particle swarm optimization[C]// International Symposium on Computational & Business Intelligence. IEEE,2016:148-152.

[23] GANDOMI A H,KASHANI A R,ZEIGHAMI F. Retaining wall optimization using interior search algorithm with different bound constraint handling[J]. International Journal for Numerical & Analytical Methods in Geomechanics,2017,41(11):1304-1331.

[24] LIANG J J,SUGANTHAN P N. Dynamic multi-swarm particle swarm optimizer[C]// Proceedings of the IEEE Swarm Intelligence Symposium. IEEE,2005:124-129.

[25] RODRIGUEZ A,LAIO A. Clustering by fast search and find of density peaks[J]. Science,2014,344(6191):1492-1496.

[26] GONG Y J,LI J J,ZHOU Y C,et al. Genetic learning particle swarm optimization [J]. IEEE Transactions on Cybernetics, 2016,46(10):2277-2290.

[27] AWAD N H,ALI M Z,LIANG J J,et al. Problem Defifinitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization[R]. Jordan University of Science and Technology, and Zhengzhou University, Technical report, Nanyang Technological University,2017.

[28] BRATTON D,KENNEDY J. Defifining a standard for particle swarm optimization [C]// Proceedings IEEE Swarm Intelligence Symposium. 2007:120-127.

[29] TANABE R,FUKUNAGA A. Success-history based parameter adaptation for differential evolution[C]// 2013 IEEE Congress on Evolutionary Computation. 2013:71-78.

[30] WANG S,LIU G,GAO M,et al. Heterogeneous comprehensive learning and dynamic multi-swarm particle swarm optimizer with two mutation operators[J]. Information Sciences, 2020, 540(12):175-201.

[31] DERRAC J,GARCÍA S,MOLINA D,et al. A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms [J]. Swarm and Evolutionary Computation,2011,1(1):3-18.

[32] DERRAC J,GARCIA S,MOLINA D. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms [J]. Swarm & Evolutionary Computation,2011,1(1):3-18.

XU Jie, born in 1996, postgraduate. His main research interests include evolutionary algorithms and intelligent information processing.

ZHOU Xinzhi, born in 1966, Ph.D, professor, Ph.D supervisor. His main research interests include the intelligent computing, intelligent control and intelligent information systems.

（责任编辑：喻藜）