智能系统学报
CAAI TRANSACTIONS ON INTELLIGENT SYSTEMS

基于概率评估差分进化的多峰值优化

王子佳, 詹志辉

引用本文:
王子佳,詹志辉. 基于概率评估差分进化的多峰值优化[J]. 智能系统学报, 2022, 17(2): 427–439.
WANG Zijia,ZHAN Zhihui. Multimodal function optimization based on DE algorithm of probabilistic evaluation mechanism[J]. *CAAI Transactions on Intelligent Systems*, 2022, 17(2): 427–439.

在线阅读 View online: https://dx.doi.org/10.11992/tis.202108007

DOI: 10.11992/tis.202108007
网络出版地址: https://kns.cnki.net/kcms/detail/23.1538.TP.20211012.1909.004.html

基于概率评估差分进化的多峰值优化

王子佳[1], 詹志辉[2]
（1. 广州大学 计算机科学与网络工程学院, 广东 广州 510006; 2. 华南理工大学 计算机科学与工程学院, 广东 广州 510006）

摘　要: 多峰值优化问题要求算法同时找到一个问题的多个全局最优解。近年来, 演化算法已被广泛用于求解多峰值优化问题。然而, 如何在极其有限的适应值评估次数内找到问题的多个全局最优解依然为演化算法带来了巨大的挑战。通过分析个体的历史更新经验, 为每个个体赋予双层适应值评估概率, 对个体进行选择性评估, 从而减少算法运行过程中无效或低效的适应值评估, 提出了一种基于概率评估差分进化的多峰值优化算法。实验结果显示, 概率评估机制可以为算法节省更多的适应值评估次数, 增加迭代过程, 效果远好于其他主流的多峰值优化算法。

关键词: 多峰值优化; 全局最优解; 演化算法; 双层适应值评估概率; 选择性评估; 差分进化算法; 历史更新经验; 高效适应值评估
中图分类号: TP18　　**文献标志码:** A　　**文章编号:** 1673−4785(2022)02−0427−13

中文引用格式: 王子佳, 詹志辉. 基于概率评估差分进化的多峰值优化 [J]. 智能系统学报, 2022, 17(2): 427–439.
英文引用格式: WANG Zijia, ZHAN Zhihui. Multimodal function optimization based on DE algorithm of probabilistic evaluation mechanism[J]. CAAI transactions on intelligent systems, 2022, 17(2): 427–439.

Multimodal function optimization based on DE algorithm of probabilistic evaluation mechanism

WANG Zijia[1], ZHAN Zhihui[2]
(1. School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China; 2. School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China)

Abstract: Multimodal optimization problems (MMOPs) require algorithms to simultaneously determine multiple global optima. Recently, evolutionary algorithms (EAs) have been widely used to solve MMOPs. However, there is still a great challenge for EAs to determine multiple global optima within very limited fitness evaluation (FE) times. To solve the inefficient FE, this paper proposes a multimodal function optimization algorithm based on the differential evolution algorithm of the probabilistic evaluation mechanism for solving MMOPs. In this algorithm, each individual will be assigned with the two-level FE probability according to its historical update experience to determine whether it needs to be evaluated. The experimental results show that the probabilistic evaluation mechanism can reduce FE times for the proposed algorithm and increase its iterative process, and its effect is much better than that of other mainstream mechanisms.

Keywords: multimodal function optimization; global optima; evolutionary algorithm; two-level fitness evaluation probability; selective evaluation; differential evolution algorithm; historical update experience; high-efficiency fitness evaluation

收稿日期: 2021−08−09.　**网络出版日期:** 2021−10−13.
基金项目: 国家自然科学基金项目 (61772207, 61873097, 62106055).
通信作者: 詹志辉. E-mail: zhanapollo@163.com.

许多实际优化问题中具有多个全局最优解, 如蛋白质结构检测[1]、电磁系统设计[2]、结构优化[3]和多行人检测[4-6]等。例如, 多行人检测通常要求

算法在一张给定的图片中尽可能多地检测到多个行人。此类问题被称为多峰值优化问题。求解多峰值优化问题具有很多实际益处。如果能同时找到一个实际问题的多个最优解，就能用多种方式来保持系统的最优性能。当其中某个最优解因为实际物理条件无法实现时，可以很快地切换到另一个最优解。

演化算法[7-14]因其基于种群的进化机制，拥有多个候选解，在求解多峰值优化问题中具有潜在的优势。然而，传统的演化算法仅仅是寻求问题的一个最优解。为了求解多峰值优化问题，最主流的思想便是小生境策略[15-28]，就是将整个种群划分为多个子种群，如 Crowding[15]、Speciation[16]、聚类[17-18]、Hill-valley[23]等。但是，现有的小生境策略对它们各自的小生境参数特别敏感，以及为了提升种群划分的准确性，算法需要采样并评估足够多的中间点，消耗了大量的适应值评估次数(fitness evaluations, FEs)。

在我们先前发表于 IEEE Transactions on Cybernetics 的研究中[29]，提出了一种自适应分布估计的差分进化算法 (adaptive estimation distribution distributed differential evolution, AED-DDE)，算法提出了一种基于自适应分布估计 (adaptive estimation distribution, AED) 的无参数小生境策略，并提出了概率局部搜索机制 (probabilistic local search, PLS) 来提升算法求解精度。AED-DDE 算法在多峰值优化问题上已经取得了很好的实验效果。然而，如何在极其有限的适应值评估次数内找到问题的多个全局最优解，依然为 AED-DDE 以及其他演化算法带来了巨大的挑战。如果可以分析个体的历史更新经验，对个体进行选择性评估，减少算法运行过程中无效或低效的适应值评估，算法性能将会得到进一步的提高。

本文基于之前 AED-DDE 的工作，进一步提出了概率评估机制 (probabilistic evaluation, PE)，并将这个机制应用在 AED-DDE 算法中，称为 PEDE。在 PEDE 算法中，每个个体会通过历史经验的学习，被赋予第一层的适应值评估概率，并根据该个体的适应值赋予第二层的适应值评估概率。每个个体在进化过程中会根据该个体的双层适应值评估概率进行选择性评估。被评估的个体更新个体的适应值，同时学习这个更新过程来调整自身的适应值评估概率；而未被评估的个体则直接根据该个体的历史经验进行个体更新。通过概率适应值评估机制，可以节省 FEs 来增加算法的迭代次数，从而提升算法求解精度。同时，通过概率评估机制，可以充分学习到个体的历史经验，更加方便种群的搜索。

在多峰值优化测试集 IEEE CEC 2013 上的实验结果显示，相比于其他的多峰值优化算法，PEDE 算法取得了更好的实验结果。同时，将概率评估机制应用到了其他基于 DE 的多峰值优化算法中，算法效果都有了一定的提升。

1　相关工作

1.1　差分进化算法

差分进化算法 (differential evolution, DE) 最初是由 Storn 和 Price 提出[30]。它通过群体内个体之间的差异来优化群体的搜索方向。DE 算法在进化过程中有变异、交叉、选择等操作，具体来说：

1) 变异

在每一代中，每个个体 $\boldsymbol{x}_i$ 称为目标向量，它通过变异操作产生它自身的变异向量 $\boldsymbol{v}_i$。3 种常用的变异策略如式 (1)~(3) 所示：

DE/rand/1：

$$\boldsymbol{v}_i = \boldsymbol{x}_{r_1} + F(\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3}) \tag{1}$$

DE/best/1：

$$\boldsymbol{v}_i = \boldsymbol{x}_{\text{best},g} + F(\boldsymbol{x}_{r_1} - \boldsymbol{x}_{r_2}) \tag{2}$$

DE/current-to-best/1：

$$\boldsymbol{v}_i = \boldsymbol{x}_i + F(\boldsymbol{x}_{\text{best}} - \boldsymbol{x}_i) + F(\boldsymbol{x}_{r_1} - \boldsymbol{x}_{r_2}) \tag{3}$$

其中，r_1、r_2 和 r_3 是 3 个不同的 $\{1, 2, \cdots, N\}$ 中的整数 (N 代表种群规模)，并且都与 i 不同。放大系数 F 是一个正实数的控制参数，用来放大差分向量。$\boldsymbol{x}_{\text{best}}$ 是指当前种群中适应值最好的个体。

2) 交叉

在变异操作后，DE 通常会在个体 $\boldsymbol{x}_i$ 和它自身的变异向量 $\boldsymbol{v}_i$ 上执行二元交叉操作，来产生试验向量 $\boldsymbol{u}_i$，如式 (4) 所示：

$$\boldsymbol{u}_{i,j,g} = \begin{cases} \boldsymbol{v}_{i,j,g}, & \text{rand} \leqslant \text{CR} \text{ 或 } j = j_{\text{rand}} \\ \boldsymbol{x}_{i,j,g}, & \text{其他} \end{cases} \tag{4}$$

其中，j_{rand} 是 $\{1, 2, \cdots, D\}$ 中的一个随机整数 (D 代表问题维度)，用来保证试验向量 $\boldsymbol{u}_i$ 至少有一维不同于 $\boldsymbol{x}_i$。参数 CR 是交叉概率，用来决定试验向量 $\boldsymbol{u}_i$ 从变异向量 $\boldsymbol{v}_i$ 的继承比例。

3) 选择

为了判断试验向量 $\boldsymbol{u}_i$ 是否可以进入下一代，试验向量 $\boldsymbol{u}_i$ 会与个体 $\boldsymbol{x}_i$ 进行比较。其中拥有较好适应值的个体进入下一代。例如，对于一个最大化问题来说，拥有较大适应值的个体会进入下一代，如式 (5) 所示：

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_{i,g}, & f(\boldsymbol{u}_{i,g}) > f(\boldsymbol{x}_{i,g}) \\ \boldsymbol{x}_{i,g}, & \text{其他} \end{cases} \tag{5}$$

其中 $f(\boldsymbol{x})$ 为适应值评估函数。

而在多数基于 DE 的多峰值优化算法中, 普遍使用的选择操作是将试验向量 $\boldsymbol{u}_i$ 与距离 $\boldsymbol{u}_i$ 最近的个体 $\boldsymbol{x}_j$ 进行比较, 拥有较好适应值的个体进入下一代, 如式 (6) 所示:

$$\boldsymbol{x}_{j,g+1}=\begin{cases}\boldsymbol{u}_{i,g}, & f(\boldsymbol{u}_{i,g})>f(\boldsymbol{x}_{j,g})\\ \boldsymbol{x}_{j,g}, & \text{其他}\end{cases} \tag{6}$$

1.2　多峰值优化算法

在求解多峰值优化问题中, 最主流的思想便是小生境策略[15-28], 就是将整个种群划分为多个子种群, 每个子种群分别去寻找该问题的一个或多个全局最优解, 如 Crowding[15]、Speciation[16]、聚类[17-18]、Hill-valley[23] 等。但是, 现有的小生境策略对它们各自的小生境参数特别敏感, 以及为了提升种群划分的准确性, 算法需要采样并评估足够多的中间点, 消耗了大量的 FEs。

基于小生境策略, Zhang 等[19] 提出了一种基于局部敏感哈希的小生境技术用来减小算法运行过程中的计算复杂度。Zhao 等[20] 提出了基于局部二值模式的小生境技术, 并设计了一种参数自适应的 DE 算法。Chen 等[21] 设计了一种基于个体的分布式框架, 在每个个体周围产生虚拟个体来形成小生境。Wang 等[22] 设计了一种基于最小生成树拓扑结构的差分进化算法, 充分捕捉个体进化过程中的全局信息和局部信息。此外, Wang 等[24] 还设计了一种基于 AP 聚类的无参小生境技术, 在避免了小生境参数敏感度的同时, 也免去了 FEs 的消耗。

除了研究小生境策略, 许多研究人员尝试提出新的变异策略来提升多峰值优化算法的性能。Biswas 等[25] 提出了基于局部信息分享机制的 DE 算法 (local information DE, LoICDE 和 LoISDE)。同时, Biswas 等[26] 进一步提出了基于邻近度和父代中心的 DE 算法 (parent-centric normalized mutation with proximity-based crowding DE, PNPCDE), 充分利用邻居的进化信息。Wang 等[27] 则针对多峰值优化问题中的选择操作, 提出了一种基于 AP 聚类的概率选择操作技术。

除了 DE, 还有很多演化算法用来求解多峰值优化问题。例如, 在文献 [31] 中, 基于聚类的小生境技术被应用在了分布估计算法中, 称之为局部多峰值分布估计算法 (locally multimodal estimation of distribution algorithm, LMCEDA 和 LMSEDA)。

此外, 也有一些研究人员通过使用多目标转化的方法来求解多峰值优化问题。Wang 等[32] 设计了一种独特的转化方式, 在每个维度上都设计了两个互相冲突的目标, 算法称之为 (multiobjective optimization for multimodal optimization problem, MOMMOP)。

1.3　AED-DDE

1.3.1　基于 AED 的自适应无参小生境技术

AED-DDE 算法提出了一种基于 AED 的自适应无参小生境技术, 每个个体将找到一个适合该个体本身的小生境大小, 并形成一个小生境来寻找一个全局最优解。具体来说, 每个个体 $\boldsymbol{x}_i$ 会首先选择距离该个体最近的 3 个个体来形成一个初始的小生境 (个体 $\boldsymbol{x}_i$ 的初始小生境大小 M_i 设置为 3)。接着, 算法采用高斯分布来估计该小生境的分布。小生境 N_i 的分布估计 D_i 如式 (7) 所示:

$$\begin{aligned}\boldsymbol{\mu}_{i,j}&=\frac{1}{M_i}\sum_{k=1}^{M_i}\boldsymbol{x}_{k,j}\\ \boldsymbol{\sigma}_{i,j}&=\sqrt{\frac{1}{M_i}\sum_{k=1}^{M_i}(\boldsymbol{x}_{k,j}-\boldsymbol{\mu}_{i,j})^2}\end{aligned} \tag{7}$$

式中: $\boldsymbol{\mu}_i$ 和 $\boldsymbol{\sigma}_i$ 表示小生境 N_i 的各维度均值和方差; $\boldsymbol{x}_k$ 表示小生境 N_i 中的第 k^{th} 个个体; M_i 表示小生境 N_i 的大小 (初始化中 M_i=3)。

在估计完小生境 N_i 的分布 D_i 后, AED 使用范围 $\boldsymbol{\mu}_i\pm3\boldsymbol{\sigma}_i$ 来判断一个给定的个体是否可以被分布 D_i 拟合。算法首先选择距离个体 $\boldsymbol{x}_i$ 第 4^{th} 近的个体 $\boldsymbol{x}_i^{4\text{th}}$ 来观察个体 $\boldsymbol{x}_i^{4\text{th}}$ 是否能落在范围 $\boldsymbol{\mu}_i\pm3\boldsymbol{\sigma}_i$ 中。如果个体 $\boldsymbol{x}_i^{4\text{th}}$ 不能落在范围 $\boldsymbol{\mu}_i\pm3\boldsymbol{\sigma}_i$ 中, 这就意味着个体 $\boldsymbol{x}_i^{4\text{th}}$ 距离个体 $\boldsymbol{x}_i$ 较远。这样, 个体 $\boldsymbol{x}_i^{4\text{th}}$ 应当被小生境 N_i 排除, 小生境 N_i 应当保持大小 M_i=3, 从而避免误导进化。

而如果个体 $\boldsymbol{x}_i^{4\text{th}}$ 落在范围 $\boldsymbol{\mu}_i\pm3\boldsymbol{\sigma}_i$ 中, 小生境 N_i 的大小 M_i 加 1, 新的个体 $\boldsymbol{x}_i^{4\text{th}}$ 加入到小生境 N_i 中, 算法继续使用式 (7) 来重新估计小生境 N_i 的分布 D_i。之后, 算法继续来看距离个体 $\boldsymbol{x}_i$ 第 5^{th} 近的个体是否可以被分布 D_i 所拟合。重复上述的过程, 直到小生境的大小不再改变为止。完整的基于 AED 的小生境策略的伪代码如算法 1 所示。

算法 1　基于 AED 的小生境技术

For 种群中的每个个体 $\boldsymbol{x}_i$;
　选择距离个体 $\boldsymbol{x}_i$ 最近的 3 个个体来形成一个初始小生境 N_i (个体 $\boldsymbol{x}_i$ 的初始小生境大小 M_i 设置为;
　使用式 (7) 来估计小生境 N_i 的分布 D_i;
　While (true);
　　If 距离个体 $\boldsymbol{x}_i$ 最近的第 $(M_i+1)^{\text{th}}$ 个个体落在范围 $\boldsymbol{\mu}_i\pm3\boldsymbol{\sigma}_i$ 中;

将距离个体 $\boldsymbol{x}_i$ 最近的第 $(M_i+1)^{\text{th}}$ 个个体加入小生境 N_i 中；

$M_i=M_i+1$；

重新使用式 (7) 来估计小生境 N_i 的分布 D_i；

Else

Break;

End If

End While

End For

经过基于 AED 的小生境技术后，每个个体将找到适合该个体本身的小生境大小，从而自适应地形成多个小生境。不同的小生境协同共进化，执行基本 DE 算法的变异和交叉操作来得到试验向量，而后使用式 (6) 执行选择操作。

1.3.2 基于 PLS 的局部搜索技术

为了进一步提高解的准确度，更加精确地找到问题的所有全局最优解，算法提出了概率局部搜索技术 (probabilistic local search, PLS)。

在 PLS 中，算法使用了基于高斯分布的局部搜索技术，如式 (8) 所示:

$$\boldsymbol{x}_{\text{new}} = N(\boldsymbol{x},\sigma) \tag{8}$$

式中：$\boldsymbol{x}_{\text{new}}$ 是高斯分布在个体 $\boldsymbol{x}$ 周围新采样的个体；高斯分布 $N(\boldsymbol{x}, \sigma)$ 表示以个体 $\boldsymbol{x}$ 作为均值，σ 作为标准差。标准差 σ 的设置如式 (9) 所示：

$$\sigma = 10^{-1-\frac{(10/D+3)\times \text{FEs}}{\text{MaxFEs}}} \tag{9}$$

在 PLS 中，算法为适应值好的个体分配较高的局部搜索概率。首先，按照个体的适应值对个体进行由坏到好的排序。接着，算法设置第 i^{th} 个个体执行局部搜索的概率为

$$P_i = r_i/N \tag{10}$$

式中：r_i 表示第 i^{th} 个个体在适应值排序中的位置；N 表示种群规模。

在执行局部搜索操作时，每个个体在各自的周围采样 2 个点。完整的 PLS 算法伪代码如算法 2 所示。

算法 2 PLS 算法

使用式 (9) 确定高斯分布的标准差 σ；

使用式 (10) 为个体设置局部搜索概率；

For 每个个体 $\boldsymbol{x}_i$:

If rand<P_i;

使用式 (8) 在 $\boldsymbol{x}_i$ 周围采样两个个体；

评估这两个个体，定义较好的个体为$\boldsymbol{x}_i'$；

FEs=FEs+2;

If 个体$\boldsymbol{x}_i'$的适应值好于个体 $\boldsymbol{x}_i$；

用个体$\boldsymbol{x}_i'$替换掉个体 $\boldsymbol{x}_i$；

End If

End If

End For

2 概率评估的差分进化算法 (PEDE)

AED-DDE 算法在多峰值优化问题上已经取得了很好的实验结果。然而，在极其有限的适应值评估次数内，AED-DDE 算法的性能仍然可以进一步提高。本文提出的 PEDE 算法基于之前的研究工作 AED-DDE[29]，并在 AED-DDE 算法的基础上引入了概率评估机制。

2.1 概率评估机制 PE

首先，在算法进化过程中，AED-DDE 并没有考虑利用个体的历史进化信息。例如，如果一个个体 $\boldsymbol{x}_j$ 持续被更新，就说明产生的试验向量 $\boldsymbol{u}_i$ 可以经常性地替换个体 $\boldsymbol{x}_j$，那么个体 $\boldsymbol{x}_j$ 的适应值很有可能并不好或者处于非最优区域；相反，如果一个个体 $\boldsymbol{x}_j$ 很久没有被更新，就说明产生的试验向量 $\boldsymbol{u}_i$ 很难替换个体 $\boldsymbol{x}_j$，那么个体 $\boldsymbol{x}_j$ 的适应值很可能已经非常好或者已经找到了最优解。如果我们可以充分挖掘并利用这些个体的历史进化信息来帮助种群的搜索，算法性能会进一步地提高。

其次，在有限的适应值评估次数内，并没有必要对所有的个体都进行评估。例如，如果一个个体 $\boldsymbol{x}_j$ 持续被更新，就说明产生的试验向量 $\boldsymbol{u}_i$ 可以经常性地替换该个体，那么可以近似认为，在不评估试验向量 $\boldsymbol{u}_i$ 的情况下，试验向量 $\boldsymbol{u}_i$ 的适应值大概率是要好于个体 $\boldsymbol{x}_j$ 的；同样地，如果一个个体 $\boldsymbol{x}_j$ 很久没有被更新，就说明产生的试验向量 $\boldsymbol{u}_i$ 很难替换个体 $\boldsymbol{x}_j$，那么我们可以近似认为，在不评估试验向量 $\boldsymbol{u}_i$ 的情况下，试验向量 $\boldsymbol{u}_i$ 的适应值大概率是要差于个体 $\boldsymbol{x}_j$ 的。也就是说，在这两种情况下，可以不去评估试验向量 $\boldsymbol{u}_i$，直接利用个体的进化信息就可以近似选择出适应值更好的个体。这个历史经验信息的积累就当作为我们第一层适应值评估概率的基础。

2.1.1 第一层适应值评估概率

首先为每个个体赋予第一层适应值评估概率 PF_j 和两个选择概率 PX_j 和 PU_j，分别代表个体 j 的第一层适应值评估概率，选择自身 $\boldsymbol{x}_j$ 进入下一代的概率，以及选择试验向量 $\boldsymbol{u}_i$ 进入下一代的概率，三者均初始化为 0.5。

在算法运行初期 (在 0.3×MaxFEs 内)，需要充分挖掘个体的历史进化信息，因此，在这个阶段里，所有的个体都要进行评估。每当产生一个试

验向量 $\boldsymbol{u}_i$ 后, 如果试验向量 $\boldsymbol{u}_i$ 的适应值要好于它最近的父代个体 $\boldsymbol{x}_j$, 则个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 会适当地增加, 而选择概率 PX_j 会适当地减小; 相反, 如果试验向量 $\boldsymbol{u}_i$ 的适应值要差于它最近的父代个体 $\boldsymbol{x}_j$, 则个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 会适当地减小, 而选择概率 PX_j 会适当地增加。我们定义个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 和 PX_j 的范围为 [0.2, 0.8], 且 $\mathrm{PU}_j+\mathrm{PX}_j=1$, 二者更新式如 (11) 所示:

$$\mathrm{PU}_j=\begin{cases}\mathrm{PU}_j+\max(0,\mathrm{N}(0.05,0.01)), & f(u_{i,g})>f(x_{j,g})\\ \mathrm{PU}_j-\max(0,\mathrm{N}(0.05,0.01)), & \text{其他}\end{cases}$$
$$\mathrm{PX}_j=1-\mathrm{PU}_j \tag{11}$$

而个体的第一层适应值评估概率 PF_j 则是根据选择概率 PU_j 和 PX_j, 选择二者中的最小值, 如式 (12) 所示:

$$\mathrm{PF}_j=\min(\mathrm{PX}_j,\mathrm{PU}_j) \tag{12}$$

也就是说, 假如个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 很大而选择概率 PX_j 很小, 那么该个体的第一层适应值评估概率就会设置为 PX_j, 即很小的值。因为 $\boldsymbol{x}_j$ 的选择概率 PU_j 很大, 也就是说, 根据前期经验, 产生的试验向量 $\boldsymbol{u}_i$ 是有很大概率可以替换掉个体 $\boldsymbol{x}_j$ 的, 那么即使不对试验向量 $\boldsymbol{u}_i$ 进行评估, 也可以近似认为试验向量 $\boldsymbol{u}_i$ 是有很大概率可以替换掉个体 $\boldsymbol{x}_j$, 故而将个体 $\boldsymbol{x}_j$ 的第一层适应值评估概率 PF_j 设置为较小的值 PX_j。同理, 假如个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 很小而选择概率 PX_j 很大, 那么该个体的第一层评估概率就会设置为 PU_j。因为 $\boldsymbol{x}_j$ 的选择概率 PU_j 很小, 也就是说根据前期经验, 产生的试验向量 $\boldsymbol{u}_i$ 是几乎不可能替换掉个体 $\boldsymbol{x}_j$ 的, 那么即使不对试验向量 $\boldsymbol{u}_i$ 进行评估, 我们可以近似认为试验向量 $\boldsymbol{u}_i$ 是几乎不可能替换掉个体 $\boldsymbol{x}_j$, 故而将个体 $\boldsymbol{x}_j$ 的第一层适应值评估概率 PF_j 设置为较小的值 PX_j。很显然, 个体的第一层适应值评估概率 PF_j 的取值范围就是 [0.2, 0.5]。

PF_j 的取值越大, 则说明个体 $\boldsymbol{x}_j$ 前期的适应值评估经验越弱, 选择概率越模糊。当 $\mathrm{PF}_j=0.5$ 时, 也就是个体 $\boldsymbol{x}_j$ 前期的适应值评估经验几乎为 0, 此时如果仍然依赖个体 $\boldsymbol{x}_j$ 前期的适应值评估经验, 很有可能误导种群的进化。也就是说, 我们希望当 $\mathrm{PF}_j=0.5$ 时, 个体 $\boldsymbol{x}_j$ 必须被评估。所以进一步在 PF_j 上增加了拉伸操作, 如式 (13) 所示:

$$\mathrm{PF}_j=2\mathrm{PF}_j \tag{13}$$

即将个体的第一层适应值评估概率 PF_j 的取值范围由 [0.2, 0.5] 线性拉伸到 [0.4, 1], 增加前期适应值评估经验较弱个体的适应值评估概率。

2.1.2 第二层适应值评估概率

除了个体的历史经验信息以外, 还需要考虑的就是个体当前的适应值。显然, 当评估次数极其有限时, 自然是希望将这有限的适应值评估次数用在更有可能寻找到问题最优解的个体上。适应值越好的个体, 接近最优解的概率也相应越高。因此, 在这里基于每个个体的适应值, 为每个个体 $\boldsymbol{x}_j$ 设计了第 2 层的适应值评估概率 PS_j。具体的设计思路与 AED-DDE 算法中的 PLS 策略类似, 按照个体的适应值对个体进行由坏到好的排序。接着, 设置个体 $\boldsymbol{x}_j$ 的第二层的适应值评估概率 PS_j 如式 (14) 所示:

$$\mathrm{PS}_j=r_j/N \tag{14}$$

式中: r_j 表示第 j^{th} 个个体在适应值排序中的位置。也就是说, 适应值越好的个体就拥有更好的第二层的适应值评估概率 PS_j。

2.1.3 完整的概率评估机制流程

在我们的算法设计中, 使用两个适应值评估概率的最大值作为个体最终的适应值评估概率 P_j, 如式 (15) 所示。

$$P_j=\max(\mathrm{PF}_j,\mathrm{PS}_j) \tag{15}$$

具体来说, 如果个体 $\boldsymbol{x}_j$ 的第 1 层适应值评估概率 PF_j 很小但是第 2 层的适应值评估概率 PS_j 很高, 这说明个体 $\boldsymbol{x}_j$ 具有很好的适应值, 寻找到最优解的概率也会很高, 此时, 即使个体 $\boldsymbol{x}_j$ 的第一层适应值评估概率 PF_j 很小, 依然为个体 $\boldsymbol{x}_j$ 设置较大的适应值评估概率, 加快算法找到问题最优解的速率。同样, 如果个体 $\boldsymbol{x}_j$ 的第一层适应值评估概率 PF_j 很高但是第 2 层的适应值评估概率 PS_j 很小, 这说明个体 $\boldsymbol{x}_j$ 虽然适应值较差, 但个体 $\boldsymbol{x}_j$ 前期的适应值评估经验较弱, 选择概率较为模糊, 此时, 为了提升算法的准确性, 同样为个体 $\boldsymbol{x}_j$ 设置较大的适应值评估概率, 进一步丰富并优化个体 $\boldsymbol{x}_j$ 的历史评估经验, 方便后续算法性能的提升。这两个适应值评估概率的结合, 可以进一步增强算法的优化性能。

在算法运行初期结束后, 就开始采用概率评估机制。在这个阶段中, 每个个体 $\boldsymbol{x}_j$ 会根据自身的适应值评估概率 P_j 来选择性评估。如果该个体 $\boldsymbol{x}_j$ 满足自身的适应值评估概率 P_j, 则对产生的试验向量 $\boldsymbol{u}_i$ 进行适应值评估, 通过比较 $\boldsymbol{u}_i$ 与 $\boldsymbol{x}_j$ 的适应值来判断是否对 $\boldsymbol{x}_j$ 进行更新, 并采用上述方法相应地更新个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 和 PX_j 以及第 1 层的适应值评估概率 PF_j。

若是该个体 $\boldsymbol{x}_j$ 并不满足自身的适应值评估概率 P_j, 则不对产生的试验向量 $\boldsymbol{u}_i$ 进行适应值评估, 而是根据前期积累的历史经验, 直接根据选择概率 PU_j 和 PX_j 来判断是否对 $\boldsymbol{x}_j$ 进行更新, 即

如果个体 $\boldsymbol{x}_j$ 满足自身的选择概率 PX_j, 则直接忽略试验向量 $\boldsymbol{u}_i$, 不对个体 $\boldsymbol{x}_j$ 进行更新。如果个体 $\boldsymbol{x}_j$ 满足自身的选择概率 PU_j, 则直接用试验向量 $\boldsymbol{u}_i$ 替换掉个体 $\boldsymbol{x}_j$。然而, 由于此时并没有对试验向量 $\boldsymbol{u}_i$ 进行适应值评估, 试验向量 $\boldsymbol{u}_i$ 没有相应的适应值, 在这种情况下, 即使个体 $\boldsymbol{x}_j$ 更新了, 依然使用它之前的适应值作为更新后个体的新适应值。同时, 由于并没有对个体的适应值进行评估, 此时个体 $\boldsymbol{x}_j$ 的选择概率 PU_j 和 PX_j 以及第一层的适应值评估概率 PF_j 不进行更新。

综上, 概率评估机制所节省下来的 FEs 可以提供给那些更需要评估的个体, 增加种群的迭代次数, 提升算法的求解精度; 同时, 概率评估机制充分捕捉了算法运行过程中每个个体的历史经验信息, 实现了信息的充分利用, 有利于算法的进一步搜索; 最后, 双层概率的结合使用互为补充, 从多方面决定个体的适应值评估概率, 丰富了算法的设计, 进一步提升了算法的优化性能。完整的概率评估机制 PE 的算法伪代码如算法 3 所示。

算法 3　PE 算法

If FEs≤ 0.3×MaxFEs:
　评估每个试验向量 $\boldsymbol{u}_i$;
　FEs=FEs+N;
　For 每个试验向量 $\boldsymbol{u}_i$;
　　找到与 $\boldsymbol{u}_i$ 与最近的父代个体 $\boldsymbol{x}_j$;
　　If $\boldsymbol{u}_i$ 的适应值好于 $\boldsymbol{x}_j$;
　　$\boldsymbol{x}_j = \boldsymbol{u}_i$; $\boldsymbol{x}_j$ 的适应值更新为 $\boldsymbol{u}_i$ 的适应值;
　　PU_j=min(PU_j + max(0, N(0.05,0.01)), 0.8);
　　Else:
　　PU_j=max(PU_j − max(0, N(0.05,0.01)), 0.2);
　　End If
　　$PX_j = 1-PU_j$;
　　PF_j = min(PX_j, PU_j);
　　$PF_j = 2\times PF_j$;
Else:
　使用式 (10) 为个体设置第二层适应值评估概率 PS_j;
　P_j = max(PF_j, PS_j);
　For 每个试验向量 $\boldsymbol{u}_i$:
　　找到与 $\boldsymbol{u}_i$ 与最近的父代个体 $\boldsymbol{x}_j$;
　　If rand ⩽ P_j:
　　评估试验向量 $\boldsymbol{u}_i$;
　　FEs=FEs+1;
　　If $\boldsymbol{u}_i$ 的适应值好于 $\boldsymbol{x}_j$:
　　$\boldsymbol{x}_j = \boldsymbol{u}_i$;
　　PU_j=min(PU_j+ max(0, N(0.05,0.01)), 0.8);
　　Else:
　　PU_j=max(PU_j - max(0, N(0.05,0.01)), 0.2);
　　End If
　　$PX_j = 1-PU_j$;
　　PF_j = min(PX_j, PU_j);
　　$PF_j = 2\times PF_j$;
　Else:
　　If rand ⩽ PU_j:
　　$\boldsymbol{x}_j = \boldsymbol{u}_i$; $\boldsymbol{x}_j$ 的适应值保持不变;
　　End If
　End If
End For
End If

2.2　完整的 PEDE 算法

结合 AED-DDE 算法与以上的概率评估机制 PE, 完整的 PEDE 算法伪代码如算法 4 所示。

算法 4　PEDE 算法

初始化种群; FEs=0; PF=0.5; PX=0.5; PU=0.5;
评估整个种群;
FEs=FEs+N;
While FEs ⩽ MaxFEs:
　使用算法 1 执行基于 AED 的自适应小生境技术;
　每个小生境进行进化并产生试验向量 $\boldsymbol{u}_i$;
　使用算法 3 执行基于 PE 的概率适应值评估机制;
　使用算法 2 执行基于 PLS 的概率局部搜索技术;
End While

1) 结合了 AED-DDE 算法中基于 AED 的小生境策略以及基于 PLS 的局部搜索策略, 可以避免种群划分的困难以及小生境参数的敏感度, 同时使得算法可以更加精确地找到所有的全局最优解。

2) 概率评估机制的提出为算法节省 FEs, 便于算法将及其有限的 FEs 提供给那些更需要评估的个体, 同时增加种群的迭代次数, 提升算法的求解精度。

3　数值实验分析

3.1　实验设置

采用目前最主流的多峰值优化测试集 IEEE CEC 2013[33] 来测试 PEDE 的算法性能。

多峰值优化问题有两个重要的评估指标, 分别是最优解寻找率 (peak ratio, PR) 和成功运行率 (success rate, SR)。对于一个给定的最大适应值评

估次数 MaxFEs 和一个准确度 ε, PR 指的是算法多次运行中, 全局最优解的平均找寻率; SR 指的是算法的成功运行比率, 其中一次成功运行表示算法可以在当次运行中找到该问题的所有全局最优解。

PEDE 的种群规模 N 的设置如表 1 所示, 而 MaxFEs 的设置则是引用 IEEE CEC 2013 多峰值优化测试集的要求。此外, PEDE 中使用的变异策略为 DE/rand/1, 放大系数 F 与交叉概率 CR 分别设置为 0.9 和 0.1。所有的算法独立运行 51 次, 对比实验结果的均值。此外, 采用准确度 ε=10^{-4} 来测试算法性能, 该准确度的使用最为普遍[15-28]。

表 1　参数设置

Table 1　Parameters setting

测试函数	N	MaxFEs
F_1~F_5	80	5.00×10^4
F_6	100	2.00×10^5
F_7	300	2.00×10^5
F_8~F_9	300	4.00×10^5
F_{10}	100	2.00×10^5
F_{11}~F_{13}	200	2.00×10^5
F_{14}~F_{20}	200	4.00×10^5

比较 PEDE 与以下 15 个多峰值优化算法, 它们分别是小生境策略算法 CDE[15]、SDE[16]、NCDE[18]、NSDE[18]、Self-CCDE[17]、Self-CSDE[17]、AED-DDE[29]; 新变异策略算法 LoICDE[25]、LoISDE[25]、R3PSO[34]、LIPS[35]、PNPCDE[26]、LMCEDA[31]、LMSEDA[31]; 以及多目标优化算法 MOMMOP[32]。这些算法都是近年来的多峰值优化中的代表性主流成果, 时间跨度从 2004 年到 2021 年。所有的算法采用相同的 MaxFEs 设置, 以此来保证比较的公平性。

3.2　实验结果

3.2.1　与主流多峰值优化算法实验结果对比

表 2 列举了 PEDE 与其他多峰值优化算法在准确度 ε=10^{-4} 下 PR 与 SR 的实验结果。为了突出显示, 最好的 PR 结果加粗显示。此外, Wilcoxon 秩和检验在 α=0.05 下用来做不同算法之间 PR 结果的显著性分析[36]。符号“+”、“−”和“≈”分别表示算法 PEDE 要显著好于 (+)、显著差于 (−) 以及无明显差异 (≈) 对比算法。从表 2 中可以看到, 在函数 F_1~F_6 和 F_{10}~F_{12} 上, 只有 PEDE 和 AED-DDE 可以在每一次运行中都找到所有的全局最优解, PR 值与 SR 值均为 1.000, 而其他的任何一个算法都不能达到相同的效果。在函数 F_7~F_9 上, Self-CCDE 和 MOMMOP 取得了令人非常满意的结果, 甚至结果要好于 PEDE, 特别是 MOMMOP, PR 值与 SR 值均为 1.000。然而, PEDE 在函数 F_7~F_9 上也取得了很好的结果, 要显著好于其他的多峰值优化算法。同时, Self-CCDE 和 MOMMOP 在函数 F_{11} 和 F_{12} 上的结果要显著差于 PEDE。在函数 F_{13} 上, PEDE 取得了与 LIPS 近似的实验结果, 但要显著好于其他所有的多峰值优化算法。同时, LIPS 在其他函数上的结果都不能优于 PEDE。在函数 F_{14}~F_{20} 上, PEDE 在函数 F_{14}、F_{16} 和 F_{20} 上都取得了最好的实验结果。需要注意的是, F_{20} 是 IEEE CEC 2013 中最复杂、维度最高的测试函数, 许多算法在 F_{20} 上甚至不能找到任何一个全局最优解。即使 PEDE 在函数 F_{15}、F_{17} 和 F_{19} 上的实验结果要略微逊色于 LMCEDA 和 LMSEDA, PEDE 的实验结果依然要显著好于其他的多峰值优化算法。同时, 与算法 LMCEDA 和 LMSEDA 相比, PEDE 在其他函数上的效果要明显好于 LMCEDA 和 LMSEDA, 尤其是在 4 个拥有着数量巨大的全局最优解的函数 F_6~F_9 上。

表 2　IEEE CEC 2013 测试集在准确度 ε=10^{-4} 下的 PR 与 SR 的实验结果

Table 2　Experimental results of IEEE CEC 2013 in PR and SR at accuracy level ε=10^{-4}

测试	PEDE		CDE		SDE		LIPS		AED-DDE		R3PSO		NCDE		NSDE	
函数	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000**(≈)	1.000	0.696(+)	0.431	0.892(+)	0.824	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000	0.773(+)	0.588	**1.000(≈)**	1.000	**1.000**(≈)	1.000	0.992(≈)	0.961	**1.000(≈)**	1.000	0.808(+)	0.726
F_3	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000**(≈)	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	0.995(≈)	0.980	0.284(+)	0.000	0.995(≈)	0.980	**1.000**(≈)	1.000	0.971(+)	0.902	**1.000(≈)**	1.000	0.255(+)	0.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000	0.961(+)	0.922	**1.000(≈)**	1.000	**1.000**(≈)	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.686(+)	0.373
F_6	**1.000**	1.000	**1.000(≈)**	1.000	0.059(+)	0.000	0.209(+)	0.000	**1.000**(≈)	1.000	0.662(+)	0.000	0.324(+)	0.000	0.056(+)	0.000
F_7	0.887	0.157	0.870(+)	0.000	0.055(+)	0.000	0.399(+)	0.000	0.838(+)	0.039	0.429(+)	0.000	0.873(+)	0.039	0.043(+)	0.000
F_8	0.805	0.000	0.000(+)	0.000	0.017(+)	0.000	0.081(+)	0.000	0.747(+)	0.000	0.431(+)	0.000	0.001(+)	0.000	0.013(+)	0.000

续表 2

测试函数	PEDE		CDE		SDE		LIPS		AED-DDE		R3PSO		NCDE		NSDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_9	0.406	0.000	0.472(−)	0.000	0.011(+)	0.000	0.104(+)	0.000	0.384(+)	0.000	0.129(+)	0.000	0.457(−)	0.000	0.005(+)	0.000
F_{10}	**1.000**	1.000	**1.000(≈)**	1.000	0.144(+)	0.000	0.755(+)	0.083	**1.000(≈)**	1.000	0.869(+)	0.137	0.997(≈)	0.961	0.088(+)	0.000
F_{11}	**1.000**	1.000	0.373(+)	0.000	0.301(+)	0.000	0.944(+)	0.686	**1.000(≈)**	1.000	0.650(+)	0.000	0.732(+)	0.020	0.226(+)	0.000
F_{12}	**1.000**	1.000	0.000(+)	0.000	0.211(+)	0.000	0.586(+)	0.000	**1.000(≈)**	1.000	0.537(+)	0.000	0.238(+)	0.000	0.132(+)	0.000
F_{13}	0.771	0.020	0.154(+)	0.000	0.307(+)	0.000	**0.775(≈)**	0.118	0.686(+)	0.000	0.654(+)	0.000	0.667(+)	0.000	0.219(+)	0.000
F_{14}	**0.667**	0.000	0.059(+)	0.000	0.212(+)	0.000	0.660(≈)	0.000	**0.667(≈)**	0.000	0.644(+)	0.000	**0.667(≈)**	0.000	0.173(+)	0.000
F_{15}	0.635	0.000	0.010(+)	0.000	0.110(+)	0.000	0.348(+)	0.000	0.637(≈)	0.000	0.206(+)	0.000	0.321(+)	0.000	0.123(+)	0.000
F_{16}	**0.667**	0.000	0.000(+)	0.000	0.101(+)	0.000	0.281(+)	0.000	**0.667(≈)**	0.000	0.451(+)	0.000	**0.667(≈)**	0.000	0.170(+)	0.000
F_{17}	0.412	0.000	0.000(+)	0.000	0.074(+)	0.000	0.164(+)	0.000	0.375(+)	0.000	0.120(+)	0.000	0.248(+)	0.000	0.096(+)	0.000
F_{18}	0.654	0.000	0.167(+)	0.000	0.013(+)	0.000	0.128(+)	0.000	0.654(≈)	0.000	0.092(+)	0.000	0.534(+)	0.000	0.167(+)	0.000
F_{19}	0.368	0.000	0.000(+)	0.000	0.098(+)	0.000	0.000(+)	0.000	0.375(−)	0.000	0.034(+)	0.000	0.306(+)	0.000	0.098(+)	0.000
F_{20}	**0.250**	0.000	0.000(+)	0.000	0.000(+)	0.000	0.000(+)	0.000	**0.250(≈)**	0.000	0.066(+)	0.000	**0.250(≈)**	0.000	0.123(+)	0.000
+			12		19		14		5		16		10		18	
−			1		0		0		1		0		1		0	
≈			7		1		6		14		4		9		2	

测试函数	PNPCDE		Self-CCDE		Self-CSDE		LoICDE		LoISDE		LMCEDA		LMSEDA		MOMMOP	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_1	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.990(≈)	0.980	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_2	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.263(+)	0.078	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_3	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.706(+)	0.333	0.971(+)	0.882	0.250(+)	0.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_5	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.961(+)	0.922	**1.000(≈)**	1.000	0.706(+)	0.412	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_6	0.559(+)	0.000	0.928(+)	0.431	0.707(+)	0.059	**1.000(≈)**	1.000	0.056(+)	0.000	0.991(+)	0.843	0.970(+)	0.529	**1.000(≈)**	1.000
F_7	0.887(≈)	0.000	0.875(+)	0.000	0.695(+)	0.000	0.691(+)	0.000	0.029(+)	0.000	0.724(+)	0.000	0.668(+)	0.000	**1.000(−)**	1.000
F_8	0.000(+)	0.000	0.997(−)	0.863	0.679(+)	0.000	0.000(+)	0.000	0.012(+)	0.000	0.350(+)	0.000	0.608(+)	0.000	**1.000(−)**	1.000
F_9	0.470(−)	0.000	0.457(−)	0.000	0.272(+)	0.000	0.115(+)	0.000	0.005(+)	0.000	0.280(+)	0.000	0.247(+)	0.000	**1.000(−)**	1.000
F_{10}	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.992(+)	0.922	**1.000(≈)**	1.000	0.083(+)	0.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_{11}	0.634(+)	0.000	0.771(+)	0.118	0.415(+)	0.000	0.660(+)	0.000	0.167(+)	0.000	0.667(+)	0.000	0.853(+)	0.157	0.712(+)	0.000
F_{12}	0.000(+)	0.000	0.375(+)	0.000	0.333(+)	0.000	0.529(+)	0.000	0.125(+)	0.000	0.750(+)	0.000	0.983(+)	0.863	0.936(+)	0.529
F_{13}	0.438(+)	0.000	0.663(+)	0.000	0.324(+)	0.000	0.523(+)	0.000	0.167(+)	0.000	0.667(+)	0.000	0.667(+)	0.000	0.667(+)	0.000
F_{14}	0.330(+)	0.000	0.657(+)	0.000	0.343(+)	0.000	0.660(≈)	0.000	0.167(+)	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000
F_{15}	0.029(+)	0.000	0.358(+)	0.000	0.208(+)	0.000	0.314(+)	0.000	0.125(+)	0.000	0.694(−)	0.000	**0.743(−)**	0.000	0.618(+)	0.000
F_{16}	0.000(+)	0.000	0.657(+)	0.000	0.078(+)	0.000	0.546(+)	0.000	0.167(+)	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	0.647(+)	0.000
F_{17}	0.000(+)	0.000	0.250(+)	0.000	0.037(+)	0.000	0.248(+)	0.000	0.074(+)	0.000	0.417(≈)	0.000	**0.632(−)**	0.000	0.502(−)	0.000
F_{18}	0.154(+)	0.000	0.327(+)	0.000	0.007(+)	0.000	0.216(+)	0.000	0.160(+)	0.000	0.654(≈)	0.000	**0.660(≈)**	0.000	0.493(+)	0.000
F_{19}	0.000(+)	0.000	0.103(+)	0.000	0.000(+)	0.000	0.049(+)	0.000	0.032(+)	0.000	0.471(−)	0.000	**0.458(-)**	0.000	0.221(+)	0.000
F_{20}	0.000(+)	0.000	0.052(+)	0.000	0.000(+)	0.000	0.123(+)	0.000	0.088(+)	0.000	0.066(+)	0.000	**0.250(≈)**	0.000	0.125(+)	0.000
+	13		12		17		13		18		8		7		8	
−	1		2		0		0		0		2		3		4	
≈	6		6		3		7		2		10		10		8	

总之，PEDE 分别要在 12、19、14、5、16、10、18、13、12、17、13、18、8、7 和 8 个函数上好于 CDE、SDE、LIPS、AED-DDE、R3PSO、NCDE、NSDE、PNPCDE、Self-CCDE、Self-CSDE、

LoICDE、LoISDE、LMCEDA、LMSEDA 和 MOMMOP。相反, PEDE 只在 1、1、1、2、2、3 和 4 个函数上要略微差于 CDE、NCDE、PNPCDE、SelfCCDE、LMCEDA、LMSEDA 和 MOMMOP。同时, PEDE 的效果要好于 AED-DDE, 说明了概率适应值评估机制的有效性。

为了有一个更加直观的视角来看 PEDE 算法的求解效果, 在 8 个可视化函数上将 PEDE 算法最终的种群分布展示出来。

如图 1 所示。

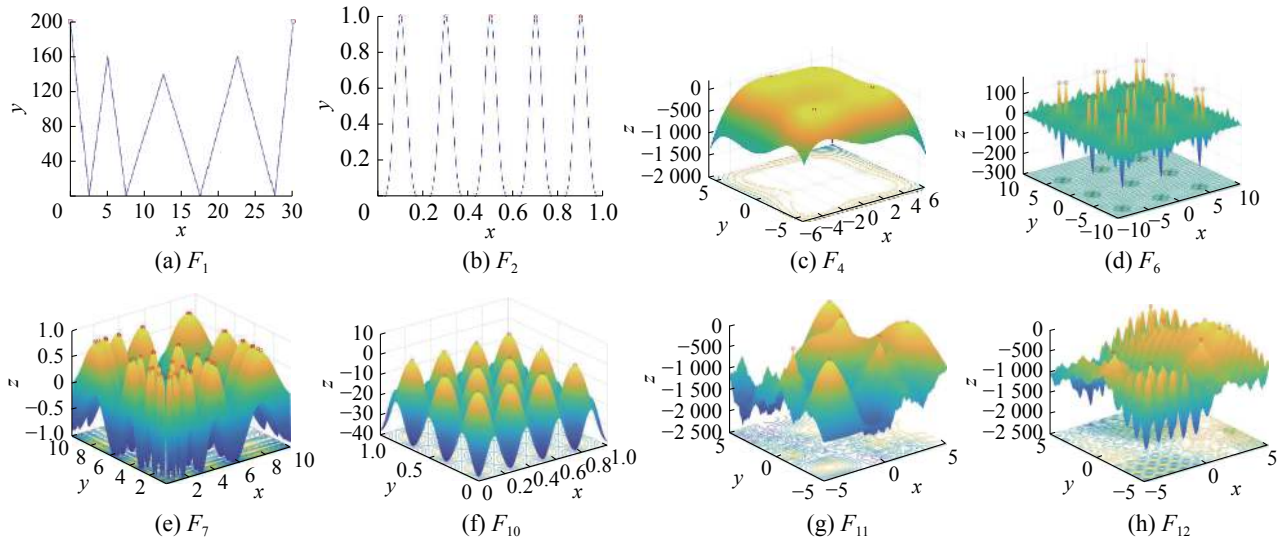

图 1　8 个测试函数上 PEDE 的最终种群分布

Fig. 1　Final population distribution of PEDE on 8 selected functions

正如我们从图 1 中看到的, PEDE 可以找到问题所有的全局最优解。即使当问题有数量巨大的全局最优解时, 如图 1(d) 和图 1(e), 在函数 F_6 和 F_7 上, PEDE 依然可以找到所有的全局最优解。当求解一些包含有数量巨大的局部最优解的复杂问题时, 如图 1(g) 和图 1(h), 在函数 F_{11} 和 F_{12} 上, PEDE 可以维持种群多样性和探索能力, 有效地避开局部最优解, 将所有的全局最优解找到。

3.2.2　与多峰值测试集冠军算法实验对比

为了进一步说明 PEDE 算法的优越性, 在本节中, 比较 PEDE 与多峰值优化测试集 IEEE CEC 2013 的冠军算法 (niching migratory multi-swarm optimizer, NMMSO[37])。为了公平比较, 直接引用 NMMSO 算法在多峰值优化测试集 IEEE CEC 2015 上的实验数据。

PEDE 与 NMMSO 在准确度 $\varepsilon=10^{-4}$ 下的 PR 与 SR 的详细对比结果如表 3 所示, 其中最好的 PR 实验结果加粗显示。

从表 3 中可以看到, PEDE 在 7 个函数上的实验结果要优于 NMMSO, 而在 7 个函数上的实验结果要略差于 NMMSO。然而, 可以看到, PEDE 可以在 9 个函数上每一次运行中都找到问题所有的全局最优解 (函数 F_1~F_6 和 F_{10}~F_{12}), PR 值与 SR 值均为 1.000, 而 NMMSO 仅可以在 7 个函数上找到问题所有的全局最优解 (函数 F_1~F_5、F_7 和 F_{10})。同时, 在 PEDE 差于 NMMSO 的测试函数上, PEDE 也实现了与 NMMSO 近似的实验结果。例如, 在函数 F_{14}、F_{17} 和 F_{19} 上, NMMSO 的 PR 实验结果分别是 0.720、0.468 和 0.450, 而 PEDE 也实现了近似的 PR 实验结果, 分别是 0.667、0.412 和 0.368。此外, 在高维复合函数 F_{16}~F_{20} 上, PEDE 在 3 个函数上要好于 NMMSO, 而 NMMSO 仅在 2 个函数上要好于 PEDE。特别地, 在函数 F_{20} 上, 也是多峰值优化测试集 IEEE CEC 2013 中最复杂的函数, PEDE 的实验结果要好于 NMMSO。

表 3　PEDE 与 NMMSO 在准确度 $\varepsilon=10^{-4}$ 下的 PR 与 SR 的实验结果对比

Table 3　Experimental results between PEDE and NMMSO in PR and SR at accuracy level $\varepsilon=10^{-4}$

测试函数	PEDE		NMMSO	
	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000(≈)**	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000
F_3	**1.000**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	**1.000(≈)**	1.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000
F_6	**1.000**	1.000	0.992(+)	0.880
F_7	0.887	0.157	**1.000(−)**	1.000
F_8	0.805	0.000	**0.899(−)**	0.020

续表 3

测试函数	PEDE		NMMSO	
	PR	SR	PR	SR
F_9	0.406	0.000	**0.978(−)**	0.120
F_{10}	**1.000**	1.000	**1.000(≈)**	1.000
F_{11}	**1.000**	1.000	0.990(+)	0.940
F_{12}	**1.000**	1.000	0.993(+)	0.940
F_{13}	0.771	0.020	**0.983(−)**	0.900
F_{14}	0.667	0.000	**0.720(−)**	0.000
F_{15}	**0.635**	0.000	0.632(+)	0.000
F_{16}	**0.667**	0.000	0.660(+)	0.000
F_{17}	0.412	0.000	**0.468(−)**	0.000
F_{18}	**0.654**	0.000	0.650(+)	0.000
F_{19}	0.368	0.000	**0.450(−)**	0.000
F_{20}	**0.250**	0.000	0.172(+)	0.000
+			7	
−			7	
≈			6	

总之，PEDE 可以实现近似甚至好于多峰值优化测试集 IEEE CEC 2013 的冠军算法的实验结果，特别是在一些复杂度较高的函数上。

3.2.3　概率评估机制性能测试

从表 2 中可以看到，PEDE 的效果要好于 AED-DDE，说明了概率适应值评估机制的有效性。在本节中，进一步测试概率评估机制的性能。将概率评估机制应用在其他的多峰值优化算法中来测试其性能。由于我们的概率评估机制适用于 DE 算法，因此在这里，将概率评估机制应用在 NCDE[18]、LoICDE[25]、Self-CCDE[17]、PNPCDE[26] 这 4 个基于 DE 的多峰值优化算法中。这 4 个算法结合概率评估机制的算法变种命名为算法-PE，例如，NCDE 算法结合概率评估机制的算法变种命名为 NCDE-PE。这 4 个算法与它们相应的算法变种在准确度 ε=10^{-4} 下的 PR 与 SR 的详细对比结果如表 4 所示。

表 4　多峰值优化算法及其在概率评估机制下的变种算法在准确度 ε=10^{-4} 下的 PR 与 SR 的实验结果

Table 4　Experimental results of multimodal optimization algorithms and their variants with PE in PR and SR at accuracy level ε=10^{-4}

函数	NCDE-PE		NCDE		LoICDE-PE		LoICDE		Self-CCDE-PE		Self-CCDE		PNPCDE-PE		PNPCDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	PR	SR	**1.000(≈)**	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000
F_3	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	0.971(+)	0.882	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000
F_6	**0.382**	0.000	0.324(+)	0.000	**1.000**	1.000	**1.000(≈)**	1.000	**0.936**	0.529	0.928(+)	0.431	**0.593**	0.000	0.559(+)	0.000
F_7	**0.883**	0.039	0.873(+)	0.039	**0.702**	0.000	0.691(+)	0.000	**0.877**	0.000	0.875(≈)	0.000	0.883	0.000	**0.887(−)**	0.000
F_8	**0.003**	0.000	0.001(+)	0.000	**0.002**	0.000	0.000(+)	0.000	0.995	0.784	**0.997(−)**	0.863	**0.003**	0.000	0.000(+)	0.000
F_9	0.454	0.000	**0.457(≈)**	0.000	**0.121**	0.000	0.115(+)	0.000	0.456	0.000	**0.457(≈)**	0.000	0.469	0.000	**0.470(−)**	0.000
F_{10}	**1.000**	1.000	0.997(≈)	0.961	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000	**1.000**	1.000	**1.000(≈)**	1.000
F_{11}	0.718	0.000	**0.732(−)**	0.020	**0.667**	0.000	0.660(+)	0.000	**0.781**	0.157	0.771(+)	0.118	**0.667**	1.000	0.634(+)	0.000
F_{12}	**0.252**	0.000	0.238(+)	0.000	0.525	0.000	**0.529(≈)**	0.000	**0.375**	0.000	**0.375(≈)**	0.000	**0.012**	0.000	0.000(+)	0.000
F_{13}	**0.667**	0.000	**0.667(≈)**	0.000	0.500	0.000	**0.523(−)**	0.000	**0.667**	0.000	0.663(+)	0.000	**0.457**	0.000	0.438(+)	0.000
F_{14}	**0.667**	0.000	**0.667(≈)**	0.000	**0.667**	0.000	0.660(≈)	0.000	**0.667**	0.000	0.657(+)	0.000	**0.333**	0.000	0.330(≈)	0.000
F_{15}	0.306	0.000	**0.321(−)**	0.000	**0.331**	0.000	0.314(+)	0.000	0.346	0.000	**0.358(-)**	0.000	0.025	0.000	**0.029(≈)**	0.000
F_{16}	**0.667**	0.000	**0.667(≈)**	0.000	**0.565**	0.000	0.546(+)	0.000	**0.667**	0.000	0.657(+)	0.000	**0.000**	0.000	**0.000(≈)**	0.000
F_{17}	**0.257**	0.000	0.248(+)	0.000	**0.250**	0.000	0.248(≈)	0.000	**0.250**	0.000	**0.250(≈)**	0.000	**0.020**	0.000	0.000(+)	0.000
F_{18}	0.529	0.000	**0.534(≈)**	0.000	**0.235**	0.000	0.216(+)	0.000	**0.359**	0.000	0.327(+)	0.000	**0.167**	0.000	0.154(+)	0.000
F_{19}	**0.318**	0.000	0.306(+)	0.000	**0.061**	0.000	0.049(+)	0.000	0.098	0.000	**0.103(≈)**	0.000	**0.000**	0.000	**0.000(≈)**	0.000
F_{20}	**0.250**	0.000	**0.250(≈)**	0.000	**0.125**	0.000	0.123(≈)	0.000	**0.064**	0.000	0.052(+)	0.000	**0.000**	0.000	**0.000(≈)**	0.000
+	6				9				7				7			
−	2				1				2				2			
≈	12				10				11				11			

从表 4 中可以看到, 概率评估机制都可以在一定程度上提升这 4 个算法的性能, 分别在 6、9、7、7 个函数上提升了 NCDE、LoICDE、Self-CCDE、PNPCDE 的算法性能, 尤其是在函数 F_6、F_{11}~F_{14}、F_{16}~F_{20} 上, 这 4 个结合概率评估机制的算法变种要显著好于原算法, 进一步说明了概率评估机制的有效性与可拓展性。

4　结束语

本文基于之前的自适应分布估计的差分进化算法这一研究工作, 进一步提出了一种概率评估机制。概率评估机制技术的提出, 充分挖掘了个体在搜索和学习过程中的历史经验信息, 对个体进行选择性评估, 节省算法的适应值评估次数, 增加算法迭代次数, 从而提升算法的求解精度。而对个体历史经验的充分学习也进一步促进了种群在搜索空间中的充分探索。

将自适应分布估计的差分进化算法与概率评估机制有机结合, 充分利用了二者的优势, 形成了本文中提出的新算法 PEDE。PEDE 不仅在多峰值优化测试集上取得了非常好的效果, 同时, 还将概率评估机制应用到其他的差分进化算法中, 效果都有了一定的提升, 也说明了概率评估机制的有效性与可拓展性。

在未来工作中, 我们希望进一步增强 PEDE 算法在复杂多峰值优化问题上的求解性能, 并将 PEDE 算法应用在多峰值实际优化问题当中。

参考文献:

[1] WONG K C, LEUNG K S, WONG M H. Protein structure prediction on a lattice model via multimodal optimization techniques[C]//Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation. Portland, USA, 2010: 155-162.

[2] WOO D K, CHOI J H, ALI M, et al. A novel multimodal optimization algorithm applied to electromagnetic optimization[J]. IEEE transactions on magnetics, 2011, 47(6): 1667–1673.

[3] 孙文新, 穆华平. 自适应群体结构的粒子群优化算法 [J]. 智能系统学报, 2013, 8(4): 372–376.
SUN Wenxin, MU Huaping. Particle swarm optimization based on self-adaptive population structure[J]. CAAI transactions on intelligent systems, 2013, 8(4): 372–376.

[4] 陈丽, 马楠, 逄桂林, 等. 多视角数据融合的特征平衡 YOLOv3 行人检测研究 [J]. 智能系统学报, 2021, 16(1): 57–65.
CHEN Li, MA Nan, PANG Guilin, et al. Research on multi-view data fusion and balanced YOLOv3 for pedestrian detection[J]. CAAI transactions on intelligent systems, 2021, 16(1): 57–65.

[5] 杨会成, 朱文博, 童英. 基于车内外视觉信息的行人碰撞预警方法 [J]. 智能系统学报, 2019, 14(4): 752–760.
YANG Huicheng, ZHU Wenbo, TONG Ying. Pedestrian collision warning system based on looking-in and looking-out visual information analysis[J]. CAAI transactions on intelligent systems, 2019, 14(4): 752–760.

[6] 伍鹏瑛, 张建明, 彭建, 等. 多层卷积特征的真实场景下行人检测研究 [J]. 智能系统学报, 2019, 14(2): 306–315.
WU Pengying, ZHANG Jianming, PENG Jian, et al. Research on pedestrian detection based on multi-layer convolution feature in real scene[J]. CAAI transactions on intelligent systems, 2019, 14(2): 306–315.

[7] WANG Zijia, ZHAN Zhihui, YU Weijie, et al. Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling[J]. IEEE transactions on cybernetics, 2020, 50(6): 2715–2729.

[8] 卢福强, 刘婷, 杜子超, 等. 模糊粒子群优化算法的第四方物流运输时间优化 [J]. 智能系统学报, 2021, 16(3): 474–483.
LU Fuqiang, LIU Ting, DU Zichao, et al. Convergence fuzzy particle swarm optimization based transportation time optimization of 4PL[J]. CAAI transactions on intelligent systems, 2021, 16(3): 474–483.

[9] 陈强, 王宇嘉, 梁海娜, 等. 目标空间映射策略的高维多目标粒子群优化算法 [J]. 智能系统学报, 2021, 16(2): 362–370.
CHEN Qiang, WANG Yujia, LIANG Haina, et al. Multi-objective particle swarm optimization algorithm based on an objective space papping strategy[J]. CAAI transactions on intelligent systems, 2021, 16(2): 362–370.

[10] 吴一全, 周建伟. 布谷鸟搜索算法研究及其应用进展 [J]. 智能系统学报, 2020, 15(3): 435–444.
WU Yiquan, ZHOU Jianwei. Overview of the cuckoo search algorithm and its applications[J]. CAAI transactions on intelligent systems, 2020, 15(3): 435–444.

[11] 钱小宇, 葛洪伟, 蔡明. 基于目标空间分解和连续变异的多目标粒子群算法 [J]. 智能系统学报, 2019, 14(3): 464–470.
QIAN Xiaoyu, GE Hongwei, CAI Ming. Decomposition and continuous mutation-based multi-objective particle swarm optimization[J]. CAAI transactions on intelligent systems, 2019, 14(3): 464–470.

[12] 裴小兵, 孙志卫. 改进区块遗传算法解决分布式车间调度问题 [J]. 智能系统学报, 2021, 16(2): 303–312.
PEI Xiaobing, SUN Zhiwei. Solving distributed-shop scheduling problems based on modified genetic algorithm[J]. CAAI transactions on intelligent systems, 2021, 16(2): 303–312.

[13] ZHAN Zhihui, WANG Zijia, JIN Hu, et al. Adaptive distributed differential evolution[J]. IEEE transactions on cybernetics, 2020, 50(11): 4633–4647.

[14] WANG Zijia, ZHAN Zhihui, KWONG S, et al. Adaptive granularity learning distributed particle swarm optimization for large-scale optimization[J]. IEEE transactions on cybernetics, 2021, 51(3): 1175–1188.

[15] THOMSEN R. Multimodal optimization using crowding-based differential evolution[C]//Proceedings of the 2004 Congress on Evolutionary Computation. Portland, USA, 2004: 1382-1389.

[16] LI Xiaodong. Efficient differential evolution using speciation for multimodal function optimization[C]//Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation. Washington, USA, 2005: 873-880.

[17] GAO Weifeng, YEN G G, LIU Sanyang. A cluster-based differential evolution with self-adaptive strategy for multimodal optimization[J]. IEEE transactions on cybernetics, 2014, 44(8): 1314–1327.

[18] QU B Y, SUGANTHAN P N, LIANG J J. Differential evolution with neighborhood mutation for multimodal optimization[J]. IEEE transactions on evolutionary computation, 2012, 16(5): 601–614.

[19] ZHANG Yuhui, GONG Yuejiao, ZHANG Huaxiang, et al. Toward fast niching evolutionary algorithms: A locality sensitive hashing-based approach[J]. IEEE transactions on evolutionary computation, 2017, 21(3): 347–362.

[20] ZHAO Hong, ZHAN Zhihui, LIN Ying, et al. Local binary pattern-based adaptive differential evolution for multimodal optimization problems[J]. IEEE transactions on cybernetics, 2020, 50(7): 3343–3357.

[21] CHEN Zonggan, ZHAN Zhihui, WANG Hua, et al. Distributed individuals for multiple peaks: a novel differential evolution for multimodal optimization problems[J]. IEEE transactions on evolutionary computation, 2020, 24(4): 708–719.

[22] WANG Zijia, ZHAN Zhihui, ZHANG Jun. Distributed minimum spanning tree differential evolution for multimodal optimization problems[J]. Soft computing, 2019, 23(24): 13339–13349.

[23] URSEM R K. Multinational evolutionary algorithms [C]//Proceedings of the 1999 Congress on Evolutionary Computation. Washington, USA, 1999: 1633-1640.

[24] WANG Zijia, ZHAN Zhihui, LIN Ying, et al. Automatic niching differential evolution with contour prediction approach for multimodal optimization problems[J]. IEEE transactions on evolutionary computation, 2020, 24(1): 114–128.

[25] BISWAS S, KUNDU S, DAS S. Inducing niching behavior in differential evolution through local information sharing[J]. IEEE transactions on evolutionary computation, 2015, 19(2): 246–263.

[26] BISWAS S, KUNDU S, DAS S. An improved parent-centric mutation with normalized neighborhoods for inducing niching behavior in differential evolution[J]. IEEE transactions on cybernetics, 2014, 44(10): 1726–1737.

[27] WANG Zijia, ZHAN Zhihui, LIN Ying, et al. Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems[J]. IEEE transactions on evolutionary computation, 2018, 22(6): 894–908.

[28] STOEAN C, PREUSS M, STOEAN R, et al. Multimodal optimization by means of a topological species conservation algorithm[J]. IEEE transactions on evolutionary computation, 2010, 14(6): 842–864.

[29] WANG Zijia, ZHOU Yuren, ZHANG Jun. Adaptive estimation distribution distributed differential evolution for multimodal optimization problems[J]. IEEE transactions on cybernetics, 2020,DOI: 10.1109/TCYB.2020.3038694.

[30] STORN R, PRICE K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of global optimization, 1997, 11(4): 341–359.

[31] YANG Qiang, CHEN Weineng, LI Yun, et al. Multimodal estimation of distribution algorithms[J]. IEEE transactions on cybernetics, 2017, 47(3): 636–650.

[32] WANG Yong, LI Hanxiong, YEN G G, et al. MOMMOP: multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems[J]. IEEE transactions on cybernetics, 2015, 45(4): 830–843.

[33] LI Xiaodong, ENGELBRECHT A, EPITROPAKIS M G. Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization[R]. Australia: RMIT University, Evolutionary Computation and Machine Learning Group, 2013.

[34] LI Xiaodong. Niching without niching parameters: particle swarm optimization using a ring topology[J]. IEEE transactions on evolutionary computation, 2010, 14(1): 150–169.

[35] QU B Y, SUGANTHAN P N, DAS S. A distance-based locally informed particle swarm model for multimodal optimization[J]. IEEE transactions on evolutionary computation, 2013, 17(3): 387–402.

[36] DERRAC J, GARCÍA S, MOLINA D, et al. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms[J]. Swarm and evolutionary computation, 2011, 1(1): 3–18.

[37] FIELDSEND J E. Running up those hills: multi-modal search with the niching migratory multi-swarm optimiser[C]//2014 IEEE Congress on Evolutionary Computation. Beijing, China, 2014: 2593−2600.

作者简介:

王子佳, 副教授, 主要研究方向为演化算法及应用。

詹志辉, 教授, 博士生导师, 主要研究方向为人工智能、进化计算、群体智能、云计算和大数据。先后荣获吴文俊人工智能优秀青年奖、IEEE 计算智能学会全球杰出博士学位论文奖、中国计算机学会优秀博士论文奖。发表学术论文 100 余篇, 其中 IEEETransactions 系列的计算机领域顶尖国际期刊论文 40 余篇。

第二届国际人工智能会议
CAAI International Conference on Artificial Intelligence 2022

由中国人工智能学会 (CAAI) 主办, 第二届国际人工智能会议 CICAI 2022 (CAAI International Conference on Artificial Intelligence 2022) 将于 2022 年 8 月在中国北京召开。本届会议组委会由国内外人工智能领域专家学者组成。国际语音识别和 AI 领域知名专家、小米集团首席语音科学家 Daniel Povey 重磅加入, 担任会议共同主席。

去年, 首届国际人工智能会议 CICAI 2021 在杭州成功举办, 受到各界广泛关注。CAAI 理事长戴琼海院士、CAAI 名誉理事长李德毅院士、CAAI 名誉副理事长沈向洋教授、北京通用人工智能研究院院长朱松纯教授等专家学者参会并做特邀报告。会议吸引了来自中国、美国、加拿大、英国、澳大利亚等国家的 307 篇有效投稿, 经过严格评审最终录用 105 篇, 录用率为 34.2%。15 篇高分录用论文被推选为口头报告(录用率为 4.8%), 且所有口头报告论文均被邀请扩展投稿到 SCI 期刊。本次会议共评出最佳论文、最佳学生论文、最佳海报论文奖项各一项。从主题报告到优秀论文分享, 这次会议不仅为各国人工智能最新研究成果交流提供了一个专业平台, 而且为研究人员、从业者、高校学生打造了一堂高端的学术课, 对营造积极健康的学术氛围, 发现和激励优秀青年人才, 推动人工智能技术创新突破具有积极作用。

今年, 第二届国际人工智能会议 CICAI 2022 将继续秉持“激发新思想、贯彻新理念、融入新格局、投身新时代”的初衷, 广纳优质学术资源、凝练前沿学术主题、积极推进国际顶级学术组织的交流与合作。具体后续安排请大家关注中国人工智能学会 CAAI 公众号的通知。