Information Sciences 714 (2025) 122189

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A probabilistic tournament learning swarm optimizer for large-scale optimization

Li-Ting Xu [a], Qiang Yang [a,*], Jian-Yu Li [b], Pei-Lan Xu [a], Xin Lin [a], Xu-Dong Gao [a], Zhen-Yu Lu [a], Jun Zhang [b,c,d,e]

[a] School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, China
[b] College of Artificial Intelligence, Nankai University, Tianjin, China
[c] Department of Electrical and Electronic Engineering, Hanyang University, Ansan, South Korea
[d] Zhejiang Normal University, Jinhua, China
[e] Chaoyang University of Technology, Taichung, Taiwan

ARTICLE INFO

Keywords:
Large-scale optimization
Probabilistic updating
Tournament learning
Particle swarm optimization
High-dimensional optimization

ABSTRACT

Large-scale optimization problems (LSOPs) pose significant challenges to particle swarm optimization (PSO) algorithms due to their high-dimensional search space and abundant attractive local optima. To solve LSOPs with high efficacy, this paper devises a probabilistic tournament learning swarm optimizer (PTLSO). Specifically, PTLSO first assigns each particle a nonlinear updating probability upon its fitness ranking. In this manner, inferior particles have exponentially higher probabilities to be updated, while superior ones preserve exponentially higher probabilities to survive. Subsequently, when a particle is triggered for updating, two different tournament selection schemes are employed to choose two different superior exemplars from all those peers with better fitness than the particle. With this random tournament learning scheme, each updated particle tends to learn from much better peers in diverse directions. Thereby, the swarm in PTLSO not only maintains high updating diversity during the evolution but also is capable of rapidly moving towards optimal points. To further help PTLSO strike an effective equilibrium between exploration and exploitation, a linear population reduction mechanism is borrowed to dynamically shrink the swarm. By this means, a large swarm is committed to traverse the broad solution space in the initial period and then a smaller and smaller number of particles enable the swarm to concentrate on subtly mining the located optimal zones as the evolution continues. With the above mechanisms, PTLSO anticipatedly presents quite good performance in tackling LSOPs. Extensive experiments carried out on the widely recognized CEC2010 and CEC2013 LSOP problems have substantiated the efficacy of PTLSO by highlighting its conspicuous superiority over 11 latest large-scale PSOs in addressing LSOPs, particularly those with complex properties. Additionally, experiments on the CEC2010 LSOPs with the dimensionality varying from 500 to 2000 have further corroborated the good scalability of PTLSO in effectively addressing LSOPs with higher dimensionalities.

* Corresponding author.
E-mail address: mmmyq@126.com (Q. Yang).

https://doi.org/10.1016/j.ins.2025.122189
Received 24 December 2024; Received in revised form 7 April 2025; Accepted 8 April 2025
Available online 11 April 2025

1. Introduction

Large-scale optimization problems (LSOPs) have gained widespread prominence in various domains in the era of the Internet of Things [1], such as investment portfolio optimization [2], power control optimization [3], and feature selection [4]. Such a kind of optimization problems give rise to considerable difficulty for optimization algorithms due to a multitude of variables involving intricate correlations, which lead to exponentially enlarged solution space and extremely complicated landscapes [5–7]. As a result, searching for the optimal solution to an LSOP in such broad and complex space is as difficult as “searching a needle in a haystack” [8,9]. Besides, attractive local or saddle regions in such vast space have greedily appealed optimization algorithms to fall into stagnation or premature convergence [9,10]. These challenges significantly diminish the effectiveness of existing optimization algorithms, especially when the given computing resources are limited. Therefore, there exists a pressing need for large-scale optimization algorithms that can efficiently and effectively solve LSOPs.

To effectively solve LSOPs, researchers have developed multitudinous large-scale evolutionary optimizers [8,10–12] by means of the unique advantages of evolutionary optimizers like their gradient-free nature, robust global search capabilities, and independence from specific problem structures. Notably, among these evolutionary optimizers, particle swarm optimization (PSO) stands out as one of the most extensively studied and sophisticated approaches [7,13,14]. Broadly speaking, in contrast to traditional PSO algorithms tailored for low dimensional optimization problems (LDOPs) that rely on personal best positions or neighbor best positions to update particles [15–17], most contemporary large-scale PSO methods utilize superior peers within the current swarm to navigate the updating processes of inferior particles [11,18,19]. Thanks to that particles are updated across iterations, the updating diversity of the swarm in existing large-scale PSOs is enhanced largely, leading to their promising performance in tackling LSOPs. From a broad perspective, current large-scale PSO approaches could be roughly partitioned into two main groups founded on the type of the superiority exhibited by the leading exemplars to the particle to be updated, namely 1) absolute superiority-based large-scale PSOs [11,19–22] and 2) relative superiority-based large-scale PSOs [7,18,23–25].

Absolute superiority-based large-scale PSOs usually first sort all particles according to their fitness values [11,19–22]. Following this, every particle is generally updated by randomly selecting guiding exemplars from all those peers with better fitness values. In this way, inferior particles are usually updated by following the guidance of absolutely better peers. Therefore, such a type of large-scale PSOs generally own fast search convergence. The only difference among this category of large-scale PSOs lies in how to determine particles to update and how to select absolutely better leading exemplars to navigate the update of the determined particles.

Instead of sorting all particles, relative superiority-based large-scale PSOs [7,18,23–25] generally first construct neighbor regions randomly for each particle. Subsequently, they let the particle compete with the peers in the neighboring regions. On the basis of the competition results, diverse updating strategies are designed to update particles by following the rule that inferior particles should follow the guidance of superior ones. Due to the relative superiority of the leading exemplars to the particle undergoing update, the updating diversity of the swarm is usually high. Therefore, such a type of large-scale PSOs usually keep a notable level of search diversity throughout the evolutionary process. The difference among this category of large-scale PSOs primarily stems from how they construct neighborhood regions and select guiding exemplars within these regions to steer the update of particles.

The above large-scale PSOs have been proved to effectively address simple LSOPs with very few correlated variables and few local regions. However, when dealing with complex LSOPs featuring numerous correlated variables and a multitude of attractive local regions, they are still at high risks of encountering premature convergence, falling into local areas, and missing finding high-accuracy solutions within limited computing resources. To solve this issue, the key lies in developing crucial and effective learning strategies for PSO to struggle for a delicate equilibrium between search diversity and convergence, so that on the one hand, particles could travel the vast space in multifarious directions to seek optimal zones quickly; on the other hand, particles could subtly mine the discovered optimal territories to attain high-precision solutions. Bearing this in mind, this paper puts forward a probabilistic tournament learning swarm optimizer (PTLSO) by probabilistically updating particles and utilizing the tournament selection to select guiding exemplars. Specifically, the crucial components of PTLSO, which are also the core innovations of this study, are summarized as below:

1) A probabilistic updating mechanism is put forward to update particles with probabilities. Specifically, an updating probability is assigned to each particle based on its fitness ranking. Consequently, particles with worse fitness possess non-linearly higher updating probabilities, while those with better fitness preserve non-linearly higher survival probabilities. In this way, relatively worse particles are likely updated to explore the vast solution space, while relatively better ones tend to survive for protecting valuable evolutionary information from being destroyed. Thereafter, high search diversity is likely kept during the evolution.
2) A tournament learning strategy is devised to update particles effectively. When a particle is selected for updating, the tournament selection strategy with two tournament sizes is employed to pick two different exemplars from peers with better fitness. In this manner, every updated particle likely follows the navigation of two much better exemplars. This contributes largely to promoting the convergence of particles towards optimal points. Additionally, thanks to the randomness of the tournament selection and the randomness of sampling the two tournament sizes from a range, high updating diversity of particles is anticipatedly sustained. Therefore, this learning strategy makes great contributions to dynamically striking a beneficial bargain between search diversity and convergence.
3) A dynamic population reduction scheme is further borrowed to gradually shrink the swarm throughout the evolution. With such a mechanism, the swarm tends to explore the broad space with a large number of particles initially in diverse directions, and then gradually shifts its focus towards digging the identified promising regions with a reduced number of particles to dig out high-accuracy solutions. Therefore, it potentially affords the optimizer a good capability of striking a favorable bargain between exploration and exploitation throughout the evolution in view of the swarm level.

By synergizing these strategies, PTLSO is expected to effectively struggle for a dynamic trade-off between search diversity and convergence and thereby gain pretty good optimization performance. The above techniques indicate that the devised PTLSO falls into the first type of large-scale PSOs. To substantiate its efficiency and efficacy, abundant experiments are carried out on the public CEC2010 [26] and CEC2013 [27] LSOPs by making comparisons with totally 11 latest large-scale PSOs. Additionally, we also carry out sufficient experiments on the CEC2010 LSOPs with different dimensionality settings to substantiate the excellent scalability of PTLSO.

The subsequent sections of this study unfold as follows. We review the two typical kinds of large-scale PSOs in Section 2. Next, we elaborate the designed PTLSO in Section 3, followed by experimental verification of the efficacy as well as the scalability of PTLSO in Section 4. Eventually, Section 5 presents the conclusion of this study.

2. Related works on large-scale PSOs

Current research on large-scale PSOs is primarily committed to designing useful learning schemes for particles, such that they could travel the broad space in various directions, swiftly move towards optimal points and thereafter subtly excavate high-precision solutions [8,9,13,25,28]. While most existing large-scale PSOs optimize all variables collectively [8,10,21,23,29], akin to traditional low-dimensional PSOs, the major difference is that most existing large-scale PSOs predominantly leverage superior peers within the swarm to navigate the update of inferior particles [20,23,28,30,31] in contrast to traditional PSOs relying on personal best positions or neighbor best positions to update particles [15,32]. Different from the historically best positions of particles, which likely keep the same for many iterations, particles within the swarm are generally updated across generations. Thereby, the search directions of the swarm in existing large-scale PSOs are generally enriched largely, which majorly contributes to their good optimization performance. Based on the superiority of the leading exemplars leveraged to update particles, current large-scale PSOs could be roughly categorized into two groups, namely absolute superiority-based large-scale PSOs [11,19–22] and relative superiority-based large-scale PSOs [7,18,23–25].

2.1. Absolute superiority-based large-scale PSOs

This kind of large-scale PSOs typically sort all particles by their fitness. Subsequently, each inferior particle follows the guidance of absolutely superior ones. In this direction, a lot of learning schemes have been developed [11,19,21,31,33] and thus it is impossible to review all of them. Here, we only review some representative and recent approaches.

One most typical large-scale PSO in this direction is the social learning PSO (SL-PSO) [19]. After sorting particles by their fitness values, SL-PSO facilitates each particle to chase a randomly selected superior peer and the average position of the entire swarm to move forward. Based on this optimizer, Jian *et al.* [20] further developed a region-based encoding strategy and an adaptive region-based search (ARS) scheme, resulting in a variant of SL-PSO, called SLPSO-ARS. Concretely, SLPSO-ARS encodes each particle with a region defined by a position and a radius. During the evolution, the radius is dynamically adjusted by leveraging the evolutionary information of each particle. With this encoding strategy and the associated region search scheme, SLPSO-ARS could let better particles perform region search to discover some nearby optimal solutions. Further, to strengthen the optimization capability of SL-PSO, Liu et al. [21] designed a sinusoidal social learning swarm optimizer (SinSLSO). Particularly, this optimizer uses a sinusoidal function to determine the learning probability of every particle by its fitness sorting. Next, for each particle triggered to update, it randomly selects two different better peers for each dimension of the particle. In this way, the updated particle learns from multiple better peers simultaneously. In [34], an exploration learning scheme and an exploitation learning mechanism were designed to pick two leading exemplars based on the sorting of particles as well as the fitness differences among particles. Accordingly, an adaptive PSO with decoupling exploration and exploitation (APSO-DEE) was developed. In [22], a heterogeneous learning PSO with the aid of reinforcement learning (AHLSO) was put forward by proposing a heterogeneous updating structure to satisfy individualized evolutionary needs. Particularly, in each generation, this optimizer utilizes the reinforcement learning to pick one learning structure. Subsequently, it evaluates particle distribution in the variable space to choose global guiding exemplars and analyses the states of particles in the fitness space to select local leading exemplars for particles to update. In [13], a convergence learning scheme was designed to pick a convergence exemplar for each particle based on its fitness ranking; then, a diversity learning scheme was designed to pick a diversity exemplar for each particle based on a diversity measure using the locally sensitive hashing technique. In this manner, a bi-directional learning PSO (BLPSO) was designed.

The aforementioned large-scale PSOs consider each individual separately. This brings benefits for high diversity maintenance but at the risk of slowing down the convergence. To alleviate this issue, researchers take advantage of grouping techniques to split particles into groups relying on their fitness sorting. To name a few typical representatives, Yang *et al.* [11] stratified particles into many levels by their fitness values. Subsequently, lower-level particles are directed by two stochastically picked higher-level peers. With these two techniques, a level-based learning swarm optimizer (LLSO) was developed. To release LLSO from the sensitivity to the level number, Wang *et al.* [29] adopted the reinforcement learning mechanism to dynamically determine a level number during the evolution. As a result, a reinforcement LLSO (RLLPSO) was developed. By considering the levels of particles in LLSO as class labels in classification, Wei *et al.* [35] combined the classification surrogate models with LLSO to solve expensive LSOPs. Based on the level separation in LLSO, Sheng *et al.* [36] further developed a multi-level population sampling scheme and a dynamic p-learning strategy for PSO (mlsdpl-PSO) to handle LSOPs. Specifically, after all particles are separated into different levels, this optimizer randomly picks a set of particles from different levels to update. Then, every particle in this set is guided by one of the top $100p$% best peers in this set with p dynamically regulated during the evolution. In [16], a dimension group-based comprehensive elite learning swarm optimizer (DGCELSO) was designed by randomly segmenting the dimensions of each absolutely inferior particle into multiple groups and then randomly selecting one absolutely superior peer to update each dimension group of this inferior particle. In this way, DGCELSO allows each absolutely inferior particle to comprehensively learn from multiple different absolutely superior peers and thus is capable of solving LSOPs effectively. In [28], particles are firstly sorted by their fitness values and then further separated into a superior set constituted by the top best half ones and an inferior set composed of the worst half ones. Then, each inferior particle is paired with one superior particle, which has the same ranking in the superior set with that of the inferior particle in the inferior set. Subsequently, every inferior particle follows the navigation of the associated superior one and the weighted average position of the entire swarm to move. Resultantly, a ranking biased learning swarm optimizer (RBLSO) was brought forward. In [14], a master–slave distributed model was adopted to maintain multiple small swarms to cooperatively travel the complex solution space. For each small swarm, particles are first sorted and then partitioned into a superior set and an inferior set. Then, each inferior particle in the inferior set is updated by learning from those peers in the superior set and the historically best solutions found by different small swarms in the slaves. In this way, particles in different small swarms search the vast solution space in parallel. In [31], after particles are sorted by their fitness values, they are separated into an elite sub-swarm and a non-elite sub-swarm. Thereafter, for each non-elite particle, an elite region is built by picking particles at random from the elite sub-swarm, and then it is navigated by the best one and all elites in the region to traverse the space. Resultantly, a random elite ensemble learning swarm optimizer (REELSO) was designed. In [33], after sorting, particles are separated into different sub-populations of equal sizes in the way that all particles in the former sub-populations are better than those in the latter ones. Then, for each particle in one sub-population, two better sub-populations are first randomly selected and then two guiding exemplars are respectively generated by randomly combining the dimensions of particles in one better sub-population. Subsequently, this particle follows the direction of the two generated exemplars to move. By this means, a superiority combination learning distributed PSO (SCLDPSO) was brought up. In [37], a heterogeneous cognitive learning PSO (HCLPSO) was designed. In HCLPSO, particles in the swarm are partitioned into a superior set and an inferior set after they are sorted. Then, every inferior particle is updated by a stochastic superior one from the superior set, which is picked by the roulette wheel selection. As for each superior particle, one random exemplar is selected from the superior set as well for this particle, but it is updated only when the selected exemplar is superior. In this manner, particles in the two sets are treated differently. In [12], an elite-directed PSO (EDPSO) was devised by first partitioning particles into three groups of different sizes based on their sorting and then treating particles in the three groups differently by letting them learn from different kinds of peers.

The above absolute superiority-based large-scale PSOs have been empirically proved to attain acceptable performance in addressing LSOPs. To summarize, in this direction, the key technique lies in how to pick superior peers to navigate the movement of inferior particles. It is deemed that a lot of effort can be devoted in this line to designing novel selection techniques to further promote the effectiveness of PSO in addressing LSOPs.

2.2. Relative superiority-based large-scale PSOs

Unlike absolute superiority-based large-scale PSOs, relative superiority-based large-scale PSOs [7,18,23–25] do not sort particles, but utilize competition strategies to determine relative superiority among particles.

The most typical one of this type is the competitive swarm optimizer (CSO) [18]. Specifically, CSO randomly pairs particles and then compares the two particles in each pair. Next, for each pair of particles, the worse particle moves forward by following the better one and the average position of the entire swarm, while the better particle keeps unchanged. Further, a three-phase CSO (TPCSO) was devised in [38] to further improve the effectiveness of CSO. Particularly, in TPCSO, all particles are first randomly separated into two equally sized sub-swarms. Then, particles undergo three learning phases. In the first phase, particles in the two sub-swarms are updated respectively using the learning strategy in CSO. In the second learning phase, every particle in one sub-swarm is updated by learning from its associated better peer in this sub-swarm and another better peer in another sub-swarm. Subsequently, in the third phase, particles in every sub-swarm move forward by following their associated better peers and the globally best individual of the entire swarm. In [30], the pairwise competition strategy in CSO is utilized to partition the swarm into a relatively good group and a relatively poor group. Then, for each poor particle, its dimensions are randomly separated into different segments. For each segment of dimensions, a random better peer is randomly selected from the relatively good set to update the particle. Therefore, every poor particle actually learns from multiple relatively better peers. As a result, a segment-based predominant learning swarm optimizer (SPLSO) was developed. In [39], Zhang *et al.* put forward a spread-based elite opposite swarm optimizer (SEOSO) to solve LSOPs. In particular, SEOSO first separates particles into multiple smaller sub-swarms. Next, for every sub-swarm, the pairwise competition strategy in CSO is utilized to determine the relatively better particle and the relatively worse particle in each pair. Next, the worse particle in each pair is updated in the same way in CSO. However, for the better particle in each pair, it is updated by using the opposite learning scheme. In [40], an elite archives-driven PSO (EAPSO) was put forward. Specifically, EAPSO first separates the swarm into an elite sub-swarm consisting of the top best half particles and a common sub-swarm containing the rest ones. Then, three elite archives are maintained during the evolution. Subsequently, six different learning schemes are designed to update particles in the common sub-swarm relying on the competition results between the particle and two random individuals selected from the archives.

Except for the above pairwise competition schemes, researchers also attempted to extend the competitions within more than two individuals. For example, in [41], a two-phase learning swarm optimizer (TPLSO) was put forward by introducing two learning phases for particles. In the first learning phase, particles are stochastically grouped into triads. Next, the worst particle in every triad follows the two better ones to move, while the medium better particle follows the best one and the mean position of the swarm to move. In the second learning phase, several best particles in the swam are taken out and thereafter, each of them is updated by learning from two random better ones among them. Fu [42] designed a homogeneous learning PSO (HLPSO) via introducing the concept of homogeneity into PSO. Specifically, particles in the swarm are structured into triads randomly. Next, the best particle in each triad is placed into a positive particle set, the medium better one is put into a homogeneous particle set, and the worst one is enclosed into a negative particle set. After that, particles in the negative set are evolved by different learning schemes based on the diversity comparison results among the three sets, while particles in the other two sets are not updated. In [24], a multiswarm optimizer with the reinforcement learning scheme (MSORL) was developed to address LSOPs. Specifically, this optimizer maintains multiple small swarms to travel the vast solution space. Throughout the evolution, particles in each small swarm are randomly grouped into triads. Among the three particles in each triad, only the medium better one or the worst one is updated, while the best one is not updated. For every triad, a reinforcement learning strategy is utilized to determine whether the medium better particle or the worst one is chosen to update. According to the selection, different updating schemes are used to update the selected particles. In [23], a stochastic dominant learning swarm optimizer (SDLSO) was put forward to tackle LSOPs. Concretely, for each particle, two different peers are randomly picked. Next, this particle is triggered to update only if the two picked peers are both better than it; otherwise, this particle is not updated. Further, a random contrastive interaction strategy was introduced into PSO, leading to RCI-PSO in [25], to solve LSOPs. Particularly, for each particle, a neighbor region is built by stochastically picking a number of peers from the current swarm. Next, this particle follows the best and the worst dominators in its neighbor region to move forward. If there are no more than two dominators in the region, this particle keeps unchanged. In [43], a PSO with a dynamic balance between convergence and diversity, shortened as PSO-DBCD, was brought up to handle LSOPs. In particular, particles are first stochastically grouped into smaller sub-swarms with the same size. Next, the best particles in all sub-swarms are kept into a convergence exemplar set, while the rest particles are combined together to be further divided into an update particle set and a diversity exemplar set based on an entropy-based diversity indicator. Subsequently, particles in the update particle set are updated by randomly selecting an exemplar from the convergence set and another exemplar from the diversity set.

The aforementioned large-scale PSOs have presented considerably good capability of addressing simple LSOPs. However, experimental results in the associated papers have indicated that their optimization abilities are still gravely challenged in face of complicated LSOPs, such as overlapping LSOPs, partially separable multimodal LSOPs, and even fully non-separable LSOPs. Therefore, to solve intricate LSOPs with high efficacy, effective large-scale optimizers are still in desideration. To cater for such demand, this paper brings up a probabilistic tournament learning swarm optimizer (PTLSO) to hopefully address complicated LSOPs with high effectiveness and efficiency.

3. Probabilistic tournament learning swarm optimizer

Most existing large-scale PSOs have been demonstrated experimentally that their effectiveness deteriorates in face of complicated LSOPs [6,22,24,29,37]. This is mainly imputed to the challenge of effectively balancing search diversity and convergence in the complex landscape of complicated LSOPs. In this study, we put forward a probabilistic tournament learning swarm optimizer (PTLSO) to make a step forward in struggling a harmonious equilibrium between exploration and exploitation to cruise the complex solution space, so as to improve the effectiveness of PSO in addressing LSOPs. The specific components of PTLSO are elaborated as below.

3.1. Probabilistic tournament learning mechanism

The main working principle of the pivotal component, namely the probabilistic tournament learning mechanism (PTL), is exhibited in Fig. 1. Concretely, PTL first sorts particles from the worst to the best. Then, it assigns every particle an updating probability based on its fitness ranking. After that, PTL has two major steps, namely the probabilistic updating strategy to trigger the update of particles, and then the tournament learning strategy to update the triggered particles. Given that the swarm contains *NP* particles, the concrete elucidations of the two major steps are presented as below.

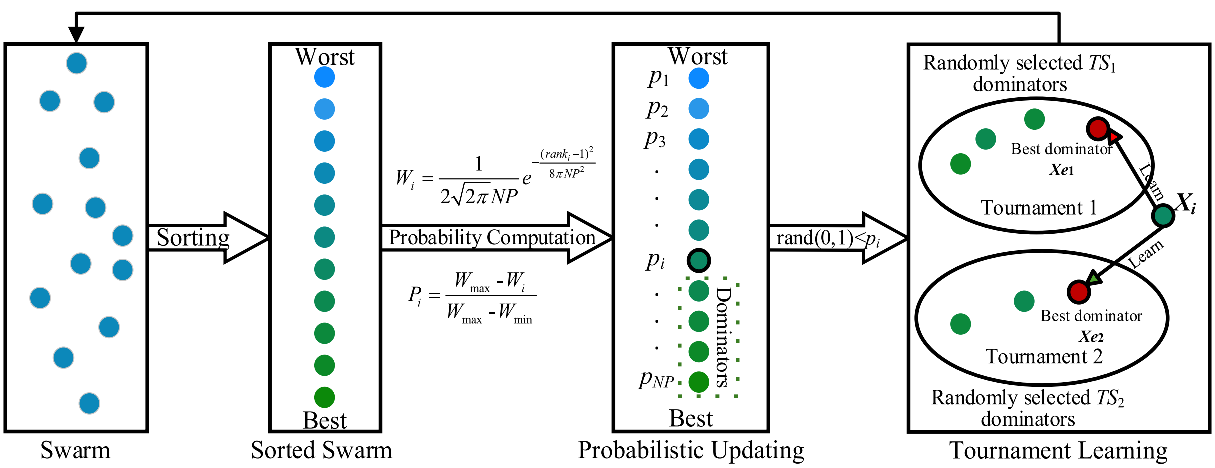

Fig. 1. Framework of PTL.

3.1.1. Probabilistic updating

PTL does not update all particles, but provides an updating probability for each particle based on its fitness ranking in the following way:

$$W_i = \frac{1}{2\sqrt{2\pi}NP}e^{-\frac{(rank_i-1)^2}{8\pi NP^2}} \tag{1}$$

$$P_i = \frac{W_i - W_{min}}{W_{max} - W_{min}} \tag{2}$$

where NP stands for the swarm size, W_i indicates the weight of the ith particle, P_i denotes its updating probability, and $rank_i$ represents its ranking index, while W_{min} and W_{max} denote the minimum and the maximum weights among all particles, respectively.

With the updating probability, for each particle, PTL first randomly samples a real value from [0, 1]. Next, if the sampled value is lower than the updating probability, this particle is triggered to update; otherwise, it survives for the next iteration of evolution. Therefore, particles within the swarm are updated probabilistically.

To observe the differences among particles with respect to their updating probabilities, we set the swarm size as 1200 and then plot the updating probabilities of all particles in Fig. 2. From Eqs. (1), (2) and Fig. 2, the following discoveries are attained:

1) The worst particle is always updated since its updating probability is 1.0, while the best particle is not updated because its updating probability is 0.0. Besides, worse particles own exponentially higher updating probabilities. On the contrary, better particles preserve exponentially lower updating probabilities. Therefore, worse particles are more likely updated to explore the solution space, while better particles are more likely retained to protect valuable evolutionary information from being destroyed.
2) With this probabilistic updating strategy, the number of particles that are triggered to update is uncertain. Therefore, in different generations, the number of updated particles is likely different. This actually makes a potential contribution to balancing the exploration and the exploitation abilities of the swarm.

3.1.2. Tournament learning

For all particles triggered to update, we design a tournament learning strategy to update them. Concretely, to pick two promising exemplars to update each of these particles, we utilize two tournament selection schemes to randomly pick exemplars from the dominators (which have higher fitness ranking values) of this particle.

Particularly, for each particle $\boldsymbol{X_i}$ triggered to update, given that the tournament sizes of the two tournament selection schemes are TS_1 and TS_2, respectively, we first randomly sample TS_1 different dominators uniformly from all peers better than $\boldsymbol{X_i}$; then, we find the best one among the TS_1 selected dominators and denote it as $\boldsymbol{X_{e1}}$. In a similar way, we randomly pick TS_2 different dominators uniformly but excluding $\boldsymbol{X_{e1}}$ and get the best one among them, which is denoted as $\boldsymbol{X_{e2}}$. Once the two guiding exemplars are attained, we compare their fitness values. If $\boldsymbol{X_{e2}}$ is better than $\boldsymbol{X_{e1}}$, we then exchange $\boldsymbol{X_{e1}}$ and $\boldsymbol{X_{e2}}$. This is to ensure that $\boldsymbol{X_{e1}}$ is better than $\boldsymbol{X_{e2}}$. Subsequently, we update particle $\boldsymbol{X_i}$ in the following manner:

$$V_i^d \leftarrow R_1^d V_i^d + R_2^d(X_{e_1}^d - X_i^d) + \beta R_3^d(X_{e_2}^d - X_i^d) \tag{3}$$

$$X_i^d \leftarrow X_i^d + V_i^d \tag{4}$$

where V_i^d stands for the dth dimension of the velocity of the ith ranked particle, and X_i^d symbolizes the dth dimension of its position; R_1, R_2 and R_3 are three random values uniformly generated from [0, 1]; $\boldsymbol{X_{e1}}$ and $\boldsymbol{X_{e2}}$ are designated as the two guiding exemplars

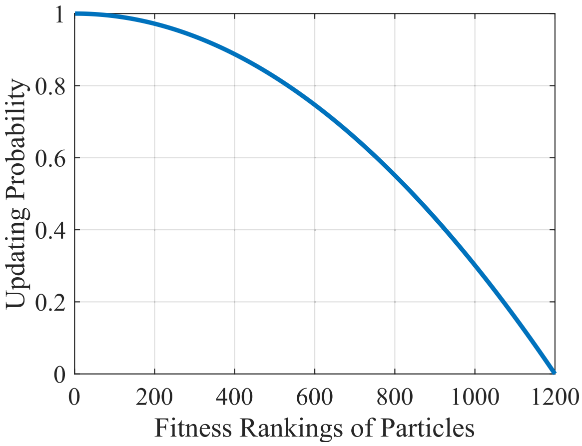

Fig. 2. Updating probabilities of different particles in PTL.

stochastically picked by the two tournament selection schemes, respectively. β is a real parameter utilized to regulate the influence of $\boldsymbol{X_{e2}}$ on the updated particle, and is typically set within [0,1].

Besides the above update formula, some extra details with respect to the tournament learning strategy deserve attention here:

1) We update particles from the worst one to the best one. In this way, the already updated particles have no effect on the later particles triggered to update.
2) Given that the larger one between TS_1 and TS_2 is TS_1, then the top best $(TS_1 + 1)$ particles are not updated though they have certain probabilities to update according to Eq. (2). This is because for these top best particles, the two tournament selection schemes cannot pick two different guiding exemplars to direct their update.
3) In PTL, two distinct tournament selection schemes are used to pick two different leading exemplars for every particle triggered to update. The purpose of this action is to enhance the learning pressure of each updated particle, such that it can learn from much better peers to move towards optimal points fast. Given the considerably vast problem space of LSOPs, this learning scheme empowers the updated particles to efficiently explore the landscape and identify promising zones. In addition, the two selected better peers likely offer different learning directions for every updated particle. This is conducive to avoiding the updated particle moving along only one road to explore the vast solution space and thus is profitable for enriching the search diversity of particles. Together, this tournament learning scheme is potentially conducive to harmonizing the exploration and exploitation capacities of the swarm to find satisfactory solutions to LSOPs. The effectiveness of this learning scheme will be demonstrated experimentally in Section 4.4.
4) The collaboration between the probabilistic updating strategy and the tournament learning scheme enables particles to explore and exploit the expansive solution space efficiently with an appropriate equilibrium between diversity maintenance and convergence acceleration. This further enables PTLSO to achieve promising performance in addressing LSOPs.

Upon examining Eq. (3), the learning rate β governs the impact of the second guiding exemplar $\boldsymbol{X_{e2}}$ on $\boldsymbol{X_i}$. By enlarging β, the influence of $\boldsymbol{X_{e2}}$ is enhanced. On the one hand, it is very helpful for avoiding the updated $\boldsymbol{X_i}$ moving towards the first guiding exemplar $\boldsymbol{X_{e1}}$ greedily. On the other hand, the learning directions of $\boldsymbol{X_i}$ are enriched, which is also profitable for the updated $\boldsymbol{X_i}$ to cruise the vast solution space diversely.

However, setting a proper value for β is very difficult, because a fixed value of β is not suitable for particles to explore and exploit the vast space appropriately. In particular, a small β may bring damage to search diversity, because the influence of $\boldsymbol{X_{e2}}$ is weakened; by contrast, it is beneficial for accelerating the convergence of the swarm because each updated particle moves greedily towards the first exemplar. Conversely, a large β enhances the impact of $\boldsymbol{X_{e2}}$, and thus brings benefit to diversity promotion; in contrast, it results in slow convergence of particles to locate optimal areas and find out high-precision solutions. Therefore, we deem that a dynamic setting for β appears more advantageous for striking a harmonious equilibrium between exploration and exploitation. To achieve such a purpose, this paper designs a dynamic scheme for β as below:

$$\beta = Gaussian(0.4, 0.01) \tag{5}$$

where Gaussian (0.4, 0.01) is a Gaussian distribution generator by setting the mean as 0.4 and the standard deviation as 0.01. It deserves mentioning that we need to keep β within [0, 1]. Therefore, once the sampled value is out of this range, it is regenerated by Eq. (5) until it is within the range. Another thing we should pay attention to is that we stochastically generate a real value of β for each particle triggered to update. In this way, different updated particles preserve different values of β.

With this dynamic strategy for β, for one thing, a suitable equilibrium between exploration and exploitation is likely attained during the evolution, which is likely beneficial to the satisfactory performance of PTLSO; for another, the efforts for fine-tuning this parameter can be saved, which results in better usability of PTLSO.

As for the two tournament sizes (TS_1 and TS_2), we also find it is considerably hard to set proper values for them. To address this issue, this paper randomly samples two values for them from [2,6]. Such a range is defined for the tournament size generation because on the one hand, the minimal tournament size is 2; on the other hand, according to the “Six Degrees of Separation” theory [44] that one can connect to any stranger in the world through a maximum of six individuals, we set the maximum tournament size as 6.

It should be mentioned that we randomly sample the two tournament sizes from [2,6] for each particle triggered to update. Therefore, different updated particles likely have different tournament sizes. This is of great benefit for promoting the learning diversity of particles. On the other hand, when large tournament sizes are generated, the updated particle learns from much better dominators. This is beneficial for convergence acceleration; on the contrary, when small tournament sizes are sampled, the updated particle learns from slightly better dominators. This brings profits to diversity promotion. Together, this dynamic sampling strategy for the two tournament sizes makes a potential contribution to helping PTLSO struggle an appropriate trade-off between search diversity and convergence.

3.2. Dynamic population reduction

In the literature [13,21], the population reduction strategies have been extensively deployed to further promote the optimization capabilities of EAs. Inspired by this, this paper also utilizes the population reduction strategy to further help PTLSO achieve good performance. Specifically, this paper utilizes the following linear population reduction strategy:

$$NP = NP_{\max} + (NP_{\min} - NP_{\max}) \times \frac{fes}{FES_{\max}} \tag{6}$$

where NP represents the current swarm size, NP_{max} denotes the maximum swarm size, NP_{min} is the minimum swarm size, fes indicates the already used number of fitness evaluations, and FES_{max} represents the given maximum number of fitness evaluations. Here, we set $NP_{min} = 0.1 * NP_{max}$. It should be mentioned that when NP becomes smaller, superfluous particles are removed randomly except for the best one until only NP particles are retained.

With this dynamic population reduction strategy, in the early stages, a large swarm of particles are maintained to explore the vast problem space in various directions. This facilitates particles to fully traverse the vast space of LSOPs to locate promising areas fast. Conversely, in the later stages, a small swarm of particles are kept to exploit the discovered optimal zones, so as to dig out high-precision solutions. Therefore, on the one hand, with the evolution continuing, the swarm gradually shrinks. Hence, the swarm transitions from an exploratory phase across the expansive solution space to an exploitative phase focusing on the identified optimal regions. On the other hand, with the continuously reduced swarm, fewer and fewer fitness evaluations are consumed in each generation. This leads to that the swarm undergoes more and more generations of evolution. This is of great help for the swarm to subtly mine the discovered optimal zones to dig out high-precision solutions. Comprehensively, this dynamic population reduction strategy helps PTLSO strike a favorable equilibrium between exploration and exploitation at the swarm level.

Algorithm1: PTLSO

```
Input: Maximum swarm size NP_max and Maximum number of fitness evaluations FES_max
1:  Initialize NP_max particles at random in the solution space and evaluate their fitness; NP=NP_max; fes = NP_max
2:  While (fes <= FES_max)
3:    Sort particles from the worst to the best concerning their fitness;
4:    Calculate the update probability (P_i) of every particle by Eqs. (1) and (2);
5:    For (i = 1: NP-7) //Update particles from the worst to the best
6:      If (rand(0,1) ≤ P_i)
7:        Randomly sample two tournament sizes (TS_1 and TS_2) from [2,6] for particle X_i;
8:        Randomly select TS_1 different dominators from all those superior peers to X_i and get the best one X_e1;
9:        Randomly select TS_2 different dominators from all those superior peers to X_i and get the best one X_e2;
10:         Swap X_e1 and X_e2 if X_e2 is better than X_e1;
11:         Randomly sample a value for β according to Eq. (5);
12:         Update particle X_i by Eq. (3) and Eq. (4);
13:         Calculate the fitness of X_i and fes++;
14:      End If
15:    End For
16:    Calculate NP according to Eq. (6);
17:    Randomly remove particles (except for the best one) out of the swarm until only NP particles are retained;
18: End While
19: Attain the solution bx with the best fitness value in the final swarm;
Output: The found optimal solution bx
```

3.3. Complete PTLSO

By integrating the PTL scheme and the dynamic population reduction mechanism, we attain the complete PTLSO, as shown in Algorithm 1. Upon examining the pseudocode, it is observed that the time complexity of PTLSO per iteration, without considering the time complexity of fitness evaluations, is O ($NP\log NP + NP \times D$). Concretely, in the main loop (Line 2 ~ Line 18), it requires O ($NP\log NP$) to rank all particles (Line 3). After that, it needs O(NP) to calculate the updating probabilities of all particles (Line 4). Subsequently, in the update of particles, it consumes O(($TS_1 + TS_2$) × NP) to select exemplars for all the NP particles, and O ($NP \times D$) to update the triggered particles. After all particles are updated, it needs O(1) to calculate the new swarm size and remove superfluous particles. Here, it should be mentioned that we do not truly remove the superfluous particles, but switch those particles to be removed from their current positions in the array to the last ones and then the first indexed NP particles are the ones used for the next iteration of evolution. In total, the overall time complexity of PTLSO is O($NP\log NP + NP + (TS_1 + TS_2) \times NP + NP \times D$). Since TS_1 and TS_2 are both significantly lower than the swarm size NP and the dimension size D, the final complexity of PTLSO is O($NP\log NP + NP \times D$).

3.4. Difference between PTLSO and existing large-scale PSOs

Since PTLSO sorts all particles by their fitness, it is an absolute superiority-based large-scale PSO variant. Comparing it with existing absolute superiority-based large-scale PSO variants, we find that it has the following significant differences:

1) The first key difference between PTLSO and existing absolute superiority-based large-scale PSOs lies in the probabilistic updating scheme that assigns a non-linear updating probability to every particle based on its fitness sorting. Most absolute superiority-based large-scale PSOs, such as REELSO [31], LLSO [11], RLLPSO [29], SCLDPSO [33], etc., update all inferior particles and only keep the top several best ones un-updated. That is, inferior particles have no survival probabilities. This may result in that the swarm is over-updated, which may do harm to the high diversity preservation. Unlike these large-scale PSO variants, PTLSO assigns an updating

probability to each particle based on its fitness sorting. Besides, the updating probability declines exponentially with the increasement of the fitness sorting. Therefore, in PTLSO, every particle preserves an implicit survival probability. This brings considerable benefits for PTLSO to keep high search diversity. Additionally, thanks to this probabilistic updating scheme, the number of updated particles is uncertain in each generation. Together, from this perspective, PTLSO is anticipated to keep higher search diversity to fully explore the vast solution space of LSOPs.

2) The second key difference between PTLSO and existing absolute superiority-based large-scale PSOs depends on the tournament learning scheme that uses two different tournament selection schemes to choose two distinct exemplars to update every particle triggered to update. Most absolute superiority-based large-scale PSOs, such as SL-PSO [19], SLPSO-ARS [20], SinSLSO [21], uses two better peers randomly selected from all those dominators to update every particle. This brings low learning pressure to particles, which is not profitable for them to traverse the vast solution space fast to dig out high-precision solutions. Instead, in PTLSO, to enhance the learning pressure, we take advantage of two different tournament selection schemes to pick two much better peers to update every particle triggered to update. By this means, every triggered particle is ensured to follow much better exemplars to move forward and thus the search convergence to optimal points is likely accelerated.

4. Experiments

To validate the efficacy of PTLSO, abundant experiments are executed on two public LSOP problem suites, namely the CEC2010 [26] suite consisting of 20 LSOPs and the CEC2013 [27] suite containing 15 LSOPs, in this section. The main characteristics of the two suites are condensed in Supplementary Tables S1 and S2, respectively. It should be mentioned that the LSOPs within the CEC2013 set pose greater challenges in terms of optimization compared to those in the CEC2010 set.

To compare with PTLSO, 11 latest large-scale PSOs are picked from the literature. Specifically, they are AHLSO [22], MSORL [24], HCLPSO [37], REELSO [31], SCLDPSO [33], TPCSO [38], mlsdpl-PSO [36], RLLPSO [29], APSO-DEE [34], TPLSO [41], and SLPSO-ARS [20]. Among them, AHLSO, HCLPSO, REELSO, SCLDPSO, mlsdpl-PSO, RLLPSO, APSO-DEE, and SLPSO-ARS are absolute superiority-based large-scale PSO algorithms as reviewed in Section 2.1, while MSORL, TPCSO, and TPLSO are relative superiority-based large-scale PSO approaches as reviewed in Section 2.2. It should be mentioned that the devised PTLSO is an absolute superiority-based large-scale optimizer. By comparing with these two types of large-scale PSOs, the efficacy of PTLSO is hopefully substantiated comprehensively.

Unless otherwise specified, all optimizers in the experiments are allocated the same total of $3000 \times D$ fitness evaluations with D standing for the dimension size. For fair comparisons, every optimizer undergoes 30 independent runs, with its performance evaluated by the median, the mean, and the standard deviation (referred to as std) calculated across these 30 independent executions. Additionally, to determine the significance of the performance difference between PTLSO and every compared method on every LSOP, the Wilcoxon rank-sum test is conducted at a significance level of $\alpha = 0.05$, which is used to identify whether there are significant differences in performance between PTLSO and the compared methods on each specific LSOP. Moreover, to assess the significance of the differences among all methods concerning the overall performance on an entire LSOP suite, the Friedman test is employed at a significance level of $\alpha = 0.05$. This test provides average rankings for all optimizers, which is to determine whether there are significant differences in their performance across the entire LSOP suite.

4.1. Parameter fine-tuning for PTLSO

In PTLSO, the only parameter requiring fine-tuning is the maximum swarm size (NP_{max}). To assess its impact on the performance of PTLSO, experiments are executed on the 1000-D CEC2010 problems by ranging NP_{max} from 900 to 1500. Table S3 exhibits the experimental results of PTLSO with distinct NP_{max} settings in solving the CEC2010 LSOPs in terms of the mean values across 30 separate executions. In this table, the best result on every LSOP is bolded. In addition, the “*Rank*” row indicates the ranking of every configuration derived from the Friedman test.

Observing Table S3, we find that PTLSO with $NP_{max} = 1200$ attains the lowest ranking among all the 7 configurations. This finding underscores that $NP_{max} = 1200$ helps PTLSO attain the best performance in addressing the 1000-D CEC2010 LSOPs. Therefore, we set $NP_{max} = 1200$ for PTLSO to tackle 1000-D LSOPs.

4.2. Comparisons with latest large-scale PSOs

This section conducts comprehensive comparisons between PTLSO and the 11 selected large-scale PSOs. To ensure fairness, the parameters in all the 11 selected optimizers are configured as the suggested ones in the corresponding papers since they were also evaluated on the CEC2010 and the CEC2013 LSOPs. Tables S4 and S5 provide the specific experimental results of PTLSO and the 11 selected large-scale optimizers on the 1000-D CEC2010 and CEC2013 LSOP suites, respectively. The “p-value” is derived from the Wilcoxon rank-sum test. A p-value below 0.05 means a substantial performance distinction between PTLSO and the compared optimizers on a specific LSOP. Conversely, when the p-value is above 0.05, it suggests that PTLSO and the compared method perform equivalently. Based on this, the bolded p-values mean that PTLSO outperforms the compared optimizers significantly on the respective LSOPs. Based on the p-values, “+”, “=”, and “-” above the p-values represent that PTLSO is notably superior to, equivalent with, and evidently inferior to the compared methods on the corresponding LSOPs, respectively. Then, “*w/t/l*” calculates the counts of “+”, “=”, and “-”, respectively. That is, “*w/t/l*” indicates that PTLSO significantly outperforms the compared method on “*w*” LSOPs, achieves equivalent performance with the compared method on “*t*” LSOPs, and shows significant inferiority to the compared method on “*l*”

LSOPs, respectively. Additionally, the "*Rank*" row displays the average ranking of every method derived from the Friedman test. To facilitate analysis, Table 1 presents the summarized statistical comparison outcomes between PTLSO and the 11 compared approaches on the CEC2010 and the CEC2013 suites. Besides, to observe the comparisons between PTLSO and the 11 compared approaches in view of their convergence behaviors, we plot their convergence curves on all the CEC2010 and the CEC2013 LSOPs in Figs. S1 and S2, respectively.

Based on Table 1, Table S4, and Fig. S1 on the 20 CEC2010 LSOPs, the following conclusions are drawn:

1) Concerning "*Rank*", PTLSO ranks the first place among all the 12 optimizers. Further observation uncovers that PTLSO attains a significantly lower rank value than the 11 compared algorithms. This demonstrates that PTLSO excels not only across the entire CEC2010 suite but also exhibits notably superior performance compared to the 11 selected large-scale optimizers.
2) In view of the overall "*w/t/l*", PTLSO exhibits significant dominance to 10 compared large-scale approaches on at least 15 LSOPs. Particularly, PTLSO significantly dominates 4 compared large-scale methods (HCLPSO, APSO-DEE, TPLSO, and SLPSO-ARS) on all the 20 LSOPs; it significantly outperforms 4 compared large-scale algorithms (AHLSO, MSORL, mlsdpl-PSO, and RLLPSO) on 19 LSOPs. These observations further demonstrate the significant superiority of PTLSO over the 11 chosen large-scale optimizers.
3) Regarding the "*w/t/l*" on various LSOP types, PTLSO exhibits exceptional performance on various LSOP types, excelling in both simple and complex LSOPs compared to the 11 chosen large-scale optimizers. To be concrete, on the 3 fully separable LSOPs (F_1, F_2, and F_3), PTLSO is substantially superior to 7 compared optimizers on all the 3 problems. On the 6 partially separable unimodal LSOPs, the performance of PTLSO is substantially better than 9 compared optimizers on all the 6 problems. On the 9 partially separable multimodal LSOPs, the performance of PTLSO is significantly superior to the 11 selected optimizers on no fewer than 7 problems. On the 2 fully non-separable LSOPs (F_{19} and F_{20}), PTLSO is significantly dominant to 10 compared large-scale approaches on both problems. As a whole, PTLSO shows particular supremacy to the 11 selected large-scale optimizers in solving LSOPs.
4) Observing Fig. S1, we discover that PTLSO attains the best performance concerning both convergence speed and solution precision on 4 LSOPs (F_8, F_{15}, F_{16}, and F_{19}). It particularly outperforms 10 compared algorithms and shows equivalent performance or slight inferiority to only one compared algorithm pertaining to both convergence speed and solution precision on 11 LSOPs (F_1, F_2, F_4, $F_5 \sim F_7$, $F_{10} \sim F_{12}$, F_{18}, and F_{20}). As a whole, regarding the efficiency, PTLSO also exhibits noticeable superiority over the 11 compared large-scale optimizers.

Based on Table 1, Table S5, and Fig. S2 on the 15 CEC2013 LSOPs, the following conclusions are drawn:

1) Regarding "*Rank*", in face of such difficult LSOPs, PTLSO consistently ranks the first place and gains an evidently lower rank value than the 11 compared approaches. This demonstrates that PTLSO not just performs the best on the entire CEC2013 set, but also substantially outperforms the 11 compared approaches.
2) Concerning the overall "*w/t/l*", confronted with the 15 LSOPs, PTLSO noticeably outperforms the 11 compared optimizers on no fewer than 10 problems. Remarkably, PTLSO defeats 6 compared optimizers (AHLSO, HCLPSO, RLLPSO, APSO-DEE, TPLSO, and SLPSO-ARS) down on all the 15 LSOPs. This further substantiates the significant superiority of PTLSO over the 11 compared approaches in addressing the CEC2013 LSOPs.
3) As for the "*w/t/l*" on various classes of LSOPs, the statistical results reveal that PTLSO not only performs markedly better than the 11 compared optimizers in addressing simple LSOPs but also displays noticeable superiority in addressing complicated LSOPs. To be concrete, on the 3 fully separable problems (F_1, F_2, and F_3), PTLSO notably outperforms 9 compared approaches on all of them. On the 3 partially separable unimodal problems (F_4, F_8, and F_{11}), PTLSO significantly defeats down 10 compared large-scale algorithms on all of them. On the 5 partially separable multimodal problems ($F_5 \sim F_7$, F_9, and F_{10}), PTLSO achieves noticeable wins in competition with the 11 compared approaches on no fewer than 4 of them. On the 3 overlapping LSOPs ($F_{12} \sim F_{14}$), PTLSO noticeably outperforms 10 compared optimizers on all of them. On the only 1 fully non-separable LSOP (F_{15}), PTLSO is notably dominant to all the 11 compared approaches. In brief, PTLSO exhibits eminent superiority over the 11 compared large-scale optimizers in addressing complicated LSOPs.
4) From Fig. S2, PTLSO converges evidently faster to higher-precision solutions than all the 11 selected approaches on 6 LSOPs (F_3, F_5, F_6, F_8, F_{10}, and F_{15}). Besides, it shows evident dominance to 10 compared optimizers and exhibits equivalent performance with or slight inferiority to just 1 compared optimizer on 9 LSOPs (F_1, F_2, F_4, F_7, F_9, $F_{11} \sim F_{14}$) regarding both convergence speed and solution precision. These observations demonstrate that PTLSO solves LSOPs with much higher efficiency than the 11 compared methods.

To sum up, the comparative analysis between PTLSO and the 11 compared optimizers reveal that PTLSO is very promising for tackling LSOPs. Particularly, PTLSO is particularly adept at solving complicated LSOPs. This is demonstrated from two perspectives. For one thing, on the CEC2010 LSOPs or the CEC2013 LSOPs, PTLSO exhibits noticeable superiority over the 11 compared approaches in addressing partially separable, fully non-separable, and overlapping problems; for another, compared with the superiority of PTLSO over the 11 compared approaches on the CEC2010 suite, the superiority of PTSLO on the CEC2013 suite is much more evident. Since the CEC2013 LSOPs are much more difficult to address than the CEC2010 LSOPs, it implicitly proves that PTLSO is exceptionally skilled at tackling intricate LSOPs.

The above superiority of PTLSO is inseparable from the proposed PTL mechanism, which enables PTLSO to struggle for a good equilibrium between search diversity and convergence. Concretely, the probabilistic update strategy offers higher probabilities for better particles to survive, and higher probabilities for worse particles to update. In this way, worse particles pay more attention to

Table 1

Summarized comparison results between PTLSO and the 11 selected algorithms on the CEC2010 and the CEC2013 sets.

Benchmark Set	Problem Property	Index	PTLSO	AHLSO	MSORL	HCLPSO	REELSO	SCLDPSO	TPCSO	mlsdpl-PSO	RLLPSO	APSO-DEE	TPLSO	SLPSO-ARS
1000-*D* CEC2010	Fully Separable Unimodal (F_1)	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Separable Multimodal (F_2,F_3)	*w/t/l*	–	2/0/0	1/1/0	2/0/0	1/0/1	2/0/0	1/0/1	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0
	Partially Separable Unimodal (F_4,F_7,F_9,F_{12}, F_{14},F_{17})	*w/t/l*	–	6/0/0	6/0/0	6/0/0	0/1/5	3/3/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0
	Partially Separable Multimodal (F_5,F_6,F_8,F_{10}, F_{11},F_{13},F_{15},F_{16},F_{18})	*w/t/l*	–	8/0/1	9/0/0	9/0/0	7/0/2	8/1/0	8/0/1	8/1/0	8/1/0	9/0/0	9/0/0	9/0/0
	Fully Non-Separable Unimodal (F_{19})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Non-Separable Multimodal (F_{20})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Overall	*w/t/l*	–	19/0/1	19/1/0	20/0/0	10/1/9	15/4/1	18/0/2	19/1/0	19/1/0	20/0/0	20/0/0	20/0/0
	Overall	*Rank*	**1.80**	5.70	7.10	6.95	4.40	6.30	6.65	6.15	8.10	5.30	10.10	9.45
1000-*D* CEC2013	Fully Separable Unimodal (F_1)	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Separable Multimodal (F_2,F_3)	*w/t/l*	–	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	1/0/1	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0
	Partially Separable Unimodal (F_4,F_8,F_{11})	*w/t/l*	–	3/0/0	3/0/0	3/0/0	1/2/0	3/0/0	3/0/0	3/0/0	3/0/0	3/0/0	3/0/0	3/0/0
	Partially Separable Multimodal ($F_5 \sim F_7$,F_9, F_{10})	*w/t/l*	–	5/0/0	4/1/0	5/0/0	5/0/0	5/0/0	5/0/0	4/1/0	5/0/0	5/0/0	5/0/0	5/0/0
	Overlapping Unimodal (F_{13},F_{14})	*w/t/l*	–	2/0/0	2/0/0	2/0/0	0/2/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0
	Overlapping Multimodal (F_{12})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/1/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Non-Separable Unimodal (F_{15})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Overall	*w/t/l*	–	15/0/0	14/1/0	15/0/0	10/5/0	14/0/1	13/1/1	14/1/0	15/0/0	15/0/0	15/0/0	15/0/0
	Overall	*Rank*	**1.40**	6.33	7.87	7.47	4.73	5.93	6.40	6.20	7.40	6.00	9.40	8.87

exploring the vast solution space, while better particles retain their valuable information to guide worse ones. Besides, the tournament learning scheme takes advantage of much better peers to update worse particles. On the one side, owing to the random tournament selection and the stochastic generation of the tournament sizes, different particles likely learn from different better peers and hence the updating diversity of particles is promoted substantially. On the other side, since every updated particle follows much better peers to move forward, the convergence of the swarm is likely accelerated. Together, the close cooperation between the two mechanisms affords the guarantee for the good optimization performance of PTLSO. In addition, the linear population reduction strategy also provides help for PTLSO to keep an effective equilibrium between search convergence and diversity. Particularly, the effectiveness of these crucial components is to be investigated in the following section.

4.3. Scalability investigation

This section investigates the scalability of PTLSO in solving LSOPs. To this end, we first extend the dimensionality of the CEC2010 LSOPs from 500 to 2000. Consequently, we get three new CEC2010 suites, namely the 500-*D*, the 1500-*D*, and the 2000-*D* CEC2010 suites. Then, we undertake comparison experiments to evaluate PTLSO against the 11 chosen large-scale PSOs on the three new CEC2010 suites. To ensure fairness, on the one hand, the maximum number of fitness evaluations is set as $3000 \times D$ with D standing for the dimension size for all optimizers; on the other hand, the population sizes for all optimizers are fine-tuned across the three suites with the detailed results provided in Tables S6–S8, respectively. After analyzing the experimental results, we get the optimal population sizes for all optimizers on the three suites as shown in Table 2.

Subsequently, with the optimal parameter settings, we compare the results of PTLSO with those of the 11 selected large-scale optimizers. Tables S9–S11 afford the specific comparison results on the 500-*D*, 1500-*D*, and 2000-*D* CEC2010 sets, respectively. To facilitate analysis, Table 3 displays the summarized key statistical outcomes on the three suites. The analysis of these tables yields the following insights:

1) In terms of "*Rank*", across all the three suites, PTLSO consistently ranks the first place among all optimizers. Particularly, its rank is noticeably lower than those of the 11 selected large-scale optimizers on each suite. These findings uncover that PTLSO gains the most satisfactory performance among all methods in solving the 500-*D*, 1500-*D*, and 2000-*D* CEC2010 LSOPs. Besides, its overall performance on each suite is significantly superior to the 11 selected approaches. This proves that PTLSO has an excellent scalability in addressing LSOPs.
2) Concerning the overall "*w/t/l*", excluding REELSO, PTLSO is noticeably superior to the other 10 compared optimizers on no fewer than 16 LSOPs in the 500-*D* CEC2010 suite, more than 17 LSOPs in the 1500-*D* CEC2010 suite, and no fewer than 14 LSOPs in the 2000-*D* CEC2010 suite. This further substantiates that PTLSO owns a very good scalability in solving LSOPs.
3) Concerning the "*w/t/l*" on every LSOP type, PTLSO excels not only in handling simple LSOPs but also in addressing complex ones. Particularly, on the three CEC2010 suites, PTLSO displays notably superiority over the 11 compared approaches, particularly in addressing partially separable and fully non-separable LSOPs.

All in all, the above comparison results confirm the significant superiority of PTLSO to the 11 selected latest large-scale PSOs in addressing higher and higher dimensional LSOPs. This proves the excellent scalability of PTLSO in tackling LSOPs. This advantageous property of PTLSO is mainly attributed to the designed PTL mechanism. Particularly, such a learning scheme not only largely enhances the updating diversity of particles, but also guides the movement of particles with much better peers. As a result, PTLSO has a good ability to balance search diversity and convergence dynamically in a suitable way to find the optimum to an LSOP.

4.4. Investigation experiments on PTLSO

4.4.1. Efficacy of the proposed PTL mechanism

This section performs experiments to validate the efficacy of PTL. For this goal, several variants of PTLSO are first developed to make comparisons with the original PTLSO.

First, in PTL, each particle is assigned an updating probability. To verify the efficacy of this principle, we remove the assigned probability. That is, all particles are updated except for the top best ($TS_{\max} + 1$) ones. The top best ($TS_{\max} + 1$) particles are not updated because we cannot use the tournament learning scheme to find available exemplars to guide their update. This variant is denoted as "PTLSO-WP".

Table 2
Swarm size settings for all optimizers to handle the CEC2010 LSOPs with different dimensionality settings.

D Alg	NP											
	PTLSO (our)	AHLSO (2024)	MSORL (2024)	HCLPSO (2023)	REELSO (2023)	SCLDPSO (2023)	TPCSO (2023)	mlsdpl-PSO (2022)	RLLPSO (2022)	APSO-DEE (2021)	TPLSO (2021)	SLPSO-ARS (2021)
D = 500	900	500	900	500	600	500	300	300	500	600	400	150
D = 1000	1200	500	900	500	800	400	500	400	500	1000	600	200
D = 1500	1500	600	900	600	1000	700	400	500	800	1000	600	250
D = 2000	1200	600	900	600	1100	400	500	600	900	1400	800	300

Table 3
Summarized comparison results between PTLSO and the 11 selected optimizers on the CEC2010 suite with different dimensionality settings.

Benchmark Set	Problem Property	Index	PTLSO	AHLSO	MSORL	HCLPSO	REELSO	SCLDPSO	TPCSO	mlsdpl-PSO	RLLPSO	APSO-DEE	TPLSO	SLPSO-ARS
500-*D*CEC2010	Fully Separable Unimodal (F_1)	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/1/0	0/1/0	1/0/0	1/0/0	0/1/0	1/0/0	1/0/0	1/0/0
	Fully Separable Multimodal (F_2,F_3)	*w/t/l*	–	2/0/0	2/0/0	2/0/0	1/0/1	2/0/0	1/1/0	2/0/0	1/1/0	2/0/0	2/0/0	2/0/0
	Partially Separable Unimodal (F_4,F_7,F_9,F_{12},F_{14},F_{17})	*w/t/l*	–	6/0/0	6/0/0	6/0/0	0/2/4	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0
	Partially Separable Multimodal (F_5,F_6,F_8,F_{10},F_{11},F_{13},F_{15},F_{16},F_{18})	*w/t/l*	–	8/0/1	8/1/0	9/0/0	7/0/2	8/0/1	9/0/0	9/0/0	7/1/1	9/0/0	9/0/0	9/0/0
	Fully Non-Separable Unimodal (F_{19})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Non-Separable Multimodal (F_{20})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/1/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Overall	*w/t/l*	–	19/0/1	19/1/0	20/0/0	8/4/8	18/1/1	19/1/0	20/0/0	16/3/1	20/0/0	20/0/0	20/0/0
	Overall	*Rank*	**1.80**	5.05	7.35	7.00	4.15	6.05	6.95	7.00	7.00	6.10	9.45	10.10
1500-*D* CEC2010	Fully Separable Unimodal (F_1)	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/0/1	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Separable Multimodal (F_2,F_3)	*w/t/l*	–	2/0/0	1/1/0	2/0/0	1/0/1	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0
	Partially Separable Unimodal (F_4,F_7,F_9,F_{12},F_{14},F_{17})	*w/t/l*	–	6/0/0	4/0/2	6/0/0	0/1/5	6/0/0	6/0/0	5/0/1	6/0/0	6/0/0	6/0/0	6/0/0
	Partially Separable Multimodal (F_5,F_6,F_8,F_{10},F_{11},F_{13},F_{15},F_{16},F_{18})	*w/t/l*	–	8/0/1	9/0/0	9/0/0	6/2/1	8/1/0	8/0/1	9/0/0	7/0/2	9/0/0	9/0/0	9/0/0
	Fully Non-Separable Unimodal (F_{19})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0
	Fully Non-Separable Multimodal (F_{20})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/1/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Overall	*w/t/l*	–	19/0/1	17/1/2	20/0/0	8/4/8	18/1/1	19/0/1	19/0/1	18/0/2	19/0/1	20/0/0	20/0/0
	Overall	*Rank*	**2.00**	5.50	7.00	7.80	4.30	6.20	6.80	6.25	7.10	5.45	10.05	9.55
2000-*D* CEC2010	Fully Separable Unimodal (F_1)	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Separable Multimodal (F_2,F_3)	*w/t/l*	–	2/0/0	1/0/1	1/0/1	1/0/1	2/0/0	1/0/1	2/0/0	1/0/1	1/0/1	2/0/0	2/0/0
	Partially Separable Unimodal (F_4,F_7,F_9,F_{12},F_{14},F_{17})	*w/t/l*	–	6/0/0	4/1/1	6/0/0	0/1/5	1/0/5	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0
	Partially Separable Multimodal (F_5,F_6,F_8,F_{10},F_{11},F_{13},F_{15},F_{16},F_{18})	*w/t/l*	–	7/0/2	8/0/1	7/0/2	6/1/2	9/0/0	4/2/3	9/0/0	5/1/3	5/1/3	8/0/1	6/1/2
	Fully Non-Separable Unimodal (F_{19})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0
	Fully Non-Separable Multimodal (F_{20})	*w/t/l*	–	1/0/0	1/0/0	1/0/0	0/0/1	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	1/0/0	0/0/1
	Overall	*w/t/l*	–	18/0/2	16/1/3	17/0/3	9/2/9	15/0/5	14/2/4	19/0/1	15/1/4	15/1/4	19/0/1	16/1/3
	Overall	*Rank*	**3.15**	5.50	6.95	7.50	4.05	6.95	6.00	8.25	7.00	4.55	9.30	8.80

Second, in PTL, each updated particle is directed by two much better peers stochastically picked by the tournament selection. To evaluate the efficacy of this approach, we first remove the tournament selection and then randomly select two different better peers to direct the movement of each particle triggered to update. We denote this variant as “PTLSO-WT”. Subsequently, we develop another variant by using only one tournament selection instead of two tournament selections. That is, we randomly select only one better peer to update each particle by using the tournament selection. We denote this variant as “PTLSO-OT”.

Third, as a baseline method, we remove the probabilistic update rule and the tournament learning scheme simultaneously from PTLSO. That is, each particle (except for the top two) follows two random superior peers to update. We denote this variant as “PTLSO-WPT”.

With the above developed variants, experiments are conducted on the 1000-*D* CEC2010 LSOPs to compare them against the original PTLSO. Table S12 displays the specific experimental results. Analysis of this table reveals the following observations:

1) Concerning “*Rank*”, PTLSO demonstrates the lowest average rank among all the 5 variants. Further observation displays that its rank is significantly lower than those of the 4 developed variants. This emphasizes the significant superiority of the PTL scheme over the 4 developed schemes in enhancing the overall performance of PTLSO, which just substantiates the commendable effectiveness of the devised PTL.
2) When considering the number of LSOPs where each algorithm excels, PTLSO has an overwhelming advantage. Specifically, it gains the best results on 17 LSOPs, while the 4 developed variants obtain the best performance on at most 4 LSOPs. This finding further demonstrates the great effectiveness of PTL.
3) Comparing PTLSO with PTLSO-WP, we find that PTLSO performs significantly better. This proves the great usefulness of the probabilistic updating rule in the PTL strategy. This strategy is helpful because it provides powerful strengths to enrich the swarm search diversity by offering a probability for each particle to survive.
4) Comparing PTLSO with PTLSO-WT, we discover a significant performance improvement of PTLSO over PTLSO-WT. This substantiates that the tournament learning scheme is of great help for PTLSO. This is because it provides powerful learning pressure for the updated particles by letting them learn from much better peers. With this strategy, the swarm likely moves towards optimal points fast.
5) Comparing PTLSO with PTLSO-OT, we observe that PTLSO significantly outperforms this variant. This just proves the usefulness of using two tournament selection schemes for PTLSO to achieve good performance. This is because these two tournament selection schemes could pick diverse yet much better peers for these particles triggered to update for efficient exploration of the solution space as well as in diverse directions.

In summary, the above experimental analysis has confirmed the significant effectiveness of the proposed PTL mechanism, which plays an irreplaceable role in aiding PTLSO gain excellent optimization performance.

4.4.2. Efficacy of the population reduction strategy

This section is to prove the efficacy of the population reduction strategy in helping PTLSO achieve good performance. To this purpose, we first fix the population size with different values varying from 500 to 1200 instead of using the reduction strategy. Subsequently, experiments are carried out on the 1000-*D* CEC2010 problems to compare PTLSO incorporating the reduction scheme against PTLSO with the fixed population sizes. The detailed results are shown in Table S13. Analysis of this table reveals the following findings:

1) Comparing the rank values of different PTLSO variants, we find that PTLSO with the population reduction strategy has the smallest rank, which is also far smaller than those of PTLSO with different fixed population sizes. This demonstrates that the population reduction strategy helps PTLSO achieve the best overall performance. Besides, this dynamic population size strategy exhibits noteworthy superiority to the fixed population sizes.
2) Regarding the number of LSOPs where each algorithm excels, PTLSO with the population reduction strategy attains the best results on 14 LSOPs, surpassing the PTLSO variants with fixed population sizes, which excel on at most 3 LSOPs. This underlines the considerable superiority of the population reduction scheme over fixed population sizes.

As a whole, the experimental results prove the pivotal role of the population reduction scheme in improving the performance of PTLSO. This strategy enables PTLSO to strike an effective equilibrium between search diversity and convergence at the population level. Initially, a large swarm is kept to explore the broad space diversely, while in later stages, a small swarm is sustained to subtly dig the located optimal zones for digging out high-precision solutions. As the evolution progresses, the swarm shrinks gradually, leading to fewer fitness evaluations per generation and thus facilitating deeper and further evolution and the discovery of superior solutions. Consequently, accompanied by the population reduction strategy, PTLSO demonstrates significant performance enhancements in addressing LSOPs.

5. Conclusion

This research paper designs a probabilistic tournament learning swarm optimizer (PTLSO) to effectively address LSOPs. Concretely, PTLSO gives every particle an updating probability according to its fitness rank. With this strategy, worse particles preserve exponentially higher updating probabilities, while better ones own exponentially higher survival probabilities. Therefore, worse particles are likely updated to explore the solution space, while better particles are likely retained to protect their valuable evolutionary information. Subsequently, a tournament learning scheme was designed by using the tournament selection strategy with two tournament sizes to select two different better peers to update each of those particles that are triggered to update. With this strategy, each updated particle learns from two much better peers and thus the movement of the swarm towards optimal points is likely accelerated. Further, a population reduction scheme was further designed by linearly decreasing the population size as the evolution continues. This scheme guides the swarm gradually from expansive exploration of the vast space to focused exploitation of the discovered optimal zones. Through the united integration of these techniques, PTLSO is expected to struggle a harmonious equilibrium between search diversity and convergence throughout evolution and hence is capable of solving LSOPs effectively and efficiently.

Comparative experiments have been conducted to compare PTLSO against totally 11 latest large-scale PSOs on the CEC2010 and the CEC2013 suites. Analysis of the experimental results has confirmed the noticeable superiority of PTLSO to the 11 large-scale optimizers in addressing LSOPs. Particularly, PTLSO shows excellent advantages in solving complex LSOPs, for instance partially separable, fully non-separable, and overlapping problems. In addition, the scalability of PTLSO in tackling higher dimensional LSOPs has further been proved by extensive experiments on the CEC2010 LSOPs via spanning dimensions from 500 to 2000.

Future endeavors will focus on applying PTLSO to solve practical LSOPs, like high-dimensional feature selection [45], neural architecture search [46], and integrating the devised PTL strategy into other evolutionary algorithms, such as differential evolution (DE) [47,48], ant colony optimization (ACO) [49], and estimation of distribution algorithms (EDA) [50], to enhance their optimization effectiveness.

CRediT authorship contribution statement

Li-Ting Xu: Writing – original draft, Software, Methodology, Investigation. **Qiang Yang:** Writing – original draft, Supervision, Methodology, Formal analysis, Conceptualization. **Jian-Yu Li:** Writing – review & editing, Methodology, Formal analysis. **Pei-Lan Xu:** Writing – review & editing, Validation, Formal analysis. **Xin Lin:** Writing – review & editing, Validation, Formal analysis. **Xu-Dong Gao:** Writing – review & editing, Validation, Formal analysis. **Zhen-Yu Lu:** Writing – review & editing, Formal analysis. **Jun Zhang:** Writing – review & editing, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2022ZD0120002, in part by the National Research Foundation of Korea under GrantNo. RS-2025-00555463, in part by the Tianjin Top Scientist Studio Project under Grant 24JRRCRC00030, in part by the Tianjin Belt and Road Joint Laboratory under Grant 24PTLYHZ00250, and in part by Jiangsu Students' Platform for Innovation and Entrepreneurship Training Program (202410300184Y).

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ins.2025.122189.

Data availability

Data will be made available on request.

References

[1] P.Z. Sotenga, K. Djouani, A.M. Kurien, A virtual network model for gateway media access control virtualisation in large scale internet of things, Internet Things 21 (2023) 100668.
[2] W. Liu, Y. Zhang, K. Liu, B. Quinn, X. Yang, Q. Peng, Evolutionary multi-objective optimisation for large-scale portfolio selection with both random and uncertain returns, IEEE Trans. Evol. Comput. (2024) 1.
[3] Y. Chen, J. Zhu, Y. Liu, L. Zhang, J. Zhou, Distributed hierarchical deep reinforcement learning for large-scale grid emergency control, IEEE Trans. Power Syst. 39 (2) (2024) 4446–4458.
[4] L. Li, M. Xuan, Q. Lin, M. Jiang, Z. Ming, K.C. Tan, An evolutionary multitasking algorithm with multiple filtering for high-dimensional feature selection, IEEE Trans. Evol. Comput. 27 (4) (2023) 802–816.
[5] Y. Li, L. Li, Q. Lin, K.-C. Wong, Z. Ming, C.A. Coello Coello, A self-organizing weighted optimization based framework for large-scale multi-objective optimization, Swarm Evol. Comput. 72 (2022) 101084.
[6] L. Li, et al., Neural net-enhanced competitive swarm optimizer for large-scale multiobjective optimization, IEEE Trans. Cybern. 54 (6) (2024) 3502–3515.
[7] W. Guo, L. Li, M. Chen, W. Ni, L. Wang, D. Li, Balancing convergence and diversity preservation in dual search space for large scale particle swarm optimization, Appl. Soft Comput. 169 (2025) 112617.

[8] J.-R. Jian, Z.-H. Zhan, J. Zhang, Large-scale evolutionary optimization: a survey and experimental comparative study, Int. J. Mach. Learn. Cybern. 11 (3) (2020) 729–745.
[9] M.N. Omidvar, X. Li, X. Yao, A review of population-based metaheuristics for large-scale black-box global optimization—Part I, IEEE Trans. Evol. Comput. 26 (5) (2021) 802–822.
[10] J. Liu, R. Sarker, S. Elsayed, D. Essam, N. Siswanto, Large-scale evolutionary optimization: a review and comparative study, Swarm Evol. Comput. 85 (2024) 101466.
[11] Q. Yang, et al., A level-based learning swarm optimizer for large-scale optimization, IEEE Trans. Evol. Comput. 22 (4) (2018) 578–594.
[12] Q. Yang, et al., Elite directed particle swarm optimization with historical information for high-dimensional problems, Mathematics 10 (9) (2022) 1384.
[13] S. Liu, Z.-J. Wang, Y.-G. Wang, S. Kwong, J. Zhang, Bi-directional learning particle swarm optimization for large-scale optimization, Appl. Soft Comput. 149 (2023) 110990.
[14] Q. Yang, et al., A distributed swarm optimizer with adaptive communication for large-scale optimization, IEEE Trans. Cybern. 50 (7) (2020) 3393–3408.
[15] Q. Yang, et al., Stochastic triad topology based particle swarm optimization for global numerical optimization, Mathematics 10 (7) (2022) 1032.
[16] Q. Yang, et al., A dimension group-based comprehensive elite learning swarm optimizer for large-scale optimization, Mathematics 10 (7) (2022) 1072.
[17] D. Tian, Q. Xu, X. Yao, G. Zhang, Y. Li, C. Xu, Diversity-guided particle swarm optimization with multi-level learning strategy, Swarm Evol. Comput. 86 (2024) 101533.
[18] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204.
[19] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inf. Sci. 291 (2015) 43–60.
[20] J.-R. Jian, Z.-G. Chen, Z.-H. Zhan, J. Zhang, Region encoding helps evolutionary computation evolve faster: a new solution encoding scheme in particle swarm for large-scale optimization, IEEE Trans. Evol. Comput. 25 (4) (2021) 779–793.
[21] N. Liu, J.-S. Pan, S.-C. Chu, P. Hu, A sinusoidal social learning swarm optimizer for large-scale optimization, Knowl.-Based Syst. 259 (2023) 110090.
[22] Y. Sun, H. Cao, An agent-assisted heterogeneous learning swarm optimizer for large-scale optimization, Swarm Evol. Comput. 89 (2024) 101627.
[23] Q. Yang, et al., An adaptive stochastic dominant learning swarm optimizer for high-dimensional optimization, IEEE Trans. Cybern. 52 (3) (2022) 1960–1976.
[24] X. Wang, F. Wang, Q. He, Y. Guo, A multi-swarm optimizer with a reinforcement learning mechanism for large-scale optimization, Swarm Evol. Comput. 86 (2024) 101486.
[25] Q. Yang, et al., Random contrastive interaction for particle swarm optimization in high-dimensional environment, IEEE Trans. Evol. Comput. 28 (4) (2024) 933–949.
[26] K. Tang, X. Li, P.N. Suganthan, Z. Yang, T. Weise, Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization, Nat. Inspired Comput. Appl. Lab. 24 (2010) 1–18.
[27] X. Li, K. Tang, M.N. Omidvar, Z. Yang, K. Qin, H. China, Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization, Gene 7 (33) (2013) 8.
[28] H. Deng, L. Peng, H. Zhang, B. Yang, Z. Chen, Ranking-based biased learning swarm optimizer for large-scale optimization, Inf. Sci. 493 (2019) 120–137.
[29] F. Wang, X. Wang, S. Sun, A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization, Inf. Sci. 602 (2022) 298–312.
[30] Q. Yang, et al., Segment-based predominant learning swarm optimizer for large-scale optimization, IEEE Trans. Cybern. 47 (9) (2016) 2896–2910.
[31] Q. Yang, et al., A random elite ensemble learning swarm optimizer for high-dimensional optimization, Complex Intell. Syst. 9 (5) (2023) 5467–5500.
[32] Q. Yang, et al., Predominant cognitive learning particle swarm optimization for global numerical optimization, Mathematics 10 (10) (2022) 1620.
[33] Z.-J. Wang, et al., Superiority combination learning distributed particle swarm optimization for large-scale optimization, Appl. Soft Comput. 136 (2023) 110101.
[34] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, Q. Wu, An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization, Swarm Evol. Comput. 60 (2021) 100789.
[35] F.F. Wei, et al., A classifier-assisted level-based learning swarm optimizer for expensive optimization, IEEE Trans. Evol. Comput. 25 (2) (2021) 219–233.
[36] M. Sheng, Z. Wang, W. Liu, X. Wang, S. Chen, X. Liu, A particle swarm optimizer with multi-level population sampling and dynamic p-learning mechanisms for large-scale optimization, Knowl.-Based Syst. 242 (2022) 108382.
[37] E. Zhang, et al., Heterogeneous cognitive learning particle swarm optimization for large-scale optimization problems, Inf. Sci. 633 (2023) 321–342.
[38] C. Huang, X. Zhou, X. Ran, Y. Liu, W. Deng, W. Deng, Co-evolutionary competitive swarm optimizer with three-phase for large-scale complex optimization problem, Inf. Sci. 619 (2023) 2–18.
[39] L. Zhang, Y. Tan, Spread-based elite opposite swarm optimizer for large scale optimization, Cognit. Rob. 2 (2022) 112–118.
[40] Y. Zhang, Elite archives-driven particle swarm optimization for large scale numerical optimization and its engineering applications, Swarm Evol. Comput. 76 (2023) 101212.
[41] R. Lan, Y. Zhu, H. Lu, Z. Liu, X. Luo, A two-phase learning-based swarm optimizer for large-scale optimization, IEEE Trans. Cybern. 51 (12) (2020) 6284–6293.
[42] W.-Y. Fu, Accelerated high-dimensional global optimization: a particle swarm optimizer incorporating homogeneous learning and autophagy mechanisms, Inf. Sci. 648 (2023) 119573.
[43] D. Li, L. Wang, W. Guo, M. Zhang, B. Hu, Q. Wu, A particle swarm optimizer with dynamic balance of convergence and diversity for large-scale optimization, Appl. Soft Comput. 132 (2023) 109852.
[44] S. Milgram, Six degrees of separation, Psychol. Today 2 (1967) 60–64.
[45] D. Duan, et al., An adaptive matrix-based evolutionary computation framework for EEG feature selection, Memet. Comput. 17 (1) (2025) 1.
[46] Z. Lu, et al., Evolving block-based convolutional neural network for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–21.
[47] Q. Yang, et al., Random neighbor elite guided differential evolution for global numerical optimization, Inf. Sci. 607 (2022) 1408–1438.
[48] J. Zhang, et al., Proximity ranking-based multimodal differential evolution, Swarm Evol. Comput. 78 (2023) 101277.
[49] Q. Yang, et al., Adaptive multimodal continuous ant colony optimization, IEEE Trans. Evol. Comput. 21 (2) (2017) 191–205.
[50] Q. Yang, et al., Multimodal estimation of distribution algorithms, IEEE Trans. Cybern. 47 (3) (2017) 636–650.