Applied Soft Computing 48 (2016) 584–596

Contents lists available at ScienceDirect

Applied Soft Computing

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

Particle swarm optimization using dynamic tournament topology

Lin Wang[a,b], Bo Yang[a,*], Jeff Orchard[b]

[a] Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China
[b] David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 19 July 2014
Received in revised form 1 July 2016
Accepted 25 July 2016
Available online 1 August 2016

Keywords:
Particle swarm optimization
Dynamic tournament topology
Evolutionary computation
Artificial neural network

ABSTRACT

Particle swarm optimization (PSO) is a nature-inspired global optimization method that uses interaction between particles to find the optimal solution in a complex search space. The swarm's evolving solution is represented by the *best* solution found by any particle. However, using this *best* solution often limits the search area. In this paper, we propose a dynamic tournament topology strategy to improve PSO. In our method, each particle is guided by several *better* solutions, chosen from the entire population. The selection of the *better* particles is stochastic, but still favors particles with better solutions. Experimental results on benchmark functions indicate that the proposed method is promising. Furthermore, the application of our dynamic tournament topology strategy in optimization of artificial neural networks indicates that this method has favorable performance.

1. Introduction

Optimization is a fundamental challenge in many real-world problems. Several optimization algorithms, such as Powell's method [1], have been examined to solve optimization problems. However, traditional algorithms have difficulty determining a satisfactory solution when dealing with highly complex real-world problems (multimodal, high dimensional, and noisy). Nature-inspired methods have been proposed to solve problems that are hard for traditional methods [2–6]. As one example (inspired by the foraging of birds), particle swarm optimization (PSO) [2] yields impressive results. The algorithm has been proven to be a practical optimization tool with several success stories in various applications, such as communication, finance, energy, medicine, materials science, and remote sensing [7–13].

This study proposes a dynamic tournament topology strategy to improve PSO (DTT-PSO). Instead of the *gbest*, *lbest*, or other *bests*, a tournament strategy is introduced to choose several above-average (*better*) guides from the population. Each potential individual has a chance to inform the others. The vast majority of particles has the opportunity to be picked as a guide to inform others. The method also incorporates merits from random topology and fully informed strategy. Each particle simultaneously receives information from all *better* guides. Furthermore, particle guides continuously change with evolution. The method increases population diversity and decreases the probability of dropping into local optima.

The rest of this paper is organized as follows. Section 2 reviews the basics, improvements, applications, and challenges of PSO. Section 3 defines the benchmark functions. Section 4 describes the details of the dynamic tournament topology strategy. Section 5 outlines and discusses the experimental results and shows its application to optimizing neural networks. Finally, Section 6 provides the conclusion.

2. Motivation

The solution of a problem in PSO is represented as a *particle* in the solution space. Each particle moves throughout the solution space following *two bests*: one that records the particle's personal best solution (denoted *pbest*), and another that records the global best solution (denoted *gbest*). However, particle or swarm historical memory in traditional algorithms is represented by the "*best*" position found by the particle itself or a group of particles. Suppose a particle is in the basin of attraction of a local optimum that is not the global optimum. If this particle happens to exhibit the best fitness of all the points, then it will enlist the others to follow it. However, this will mislead the swarm into a local optimum, and possibly away from the global optimum.

This problem gives rise to the following question: *can we replace the best position with better positions to further improve the performance of PSO by thoroughly searching the potential optimal regions?*

This study proposes a dynamic tournament topology strategy. No concept of *gbest* or *lbest* exists in DTT-PSO. Instead, we propose

* Corresponding author.
E-mail address: yangbo@ujn.edu.cn (B. Yang).

http://dx.doi.org/10.1016/j.asoc.2016.07.041
1568-4946/© 2016 Elsevier B.V. All rights reserved.

the use of a tournament strategy to distribute the guidance of the swarm more evenly. Every particle (except for the worst ones) can potentially become a *better* guide. The method also adopts advantages from random topology [14] and the fully informed strategy [15]. Each particle simultaneously receives information from all *better* guides and changes its guides frequently during evolution.

On one hand, this strategy avoids the myopic tendency to follow a single global best particle [2]. Rather, it selects several better guides randomly to enable a more diversified search. The randomness of the tournament topology ensures that the best particle *tends* to inform the others, but not always. On the other hand, our method adopts a dynamic topology in which the neighborhood of a particle frequently changes according to the tournament strategy, similar to local PSO [16]. This dynamic random topology ensures a wide exchange rather than a directional flow of information between particles. The DTT strategy helps pull the PSO out of local optima, and toward the global optimum. The next subsection describes the DTT-PSO in detail.

3. Review of particle swarm optimization

3.1. Basics of PSO

PSO is a nature-inspired global optimization method designed by Kennedy et al. [2] in 1995. The method can determine the optimum area in a complex solution space by the interaction between particles. Compared with the genetic algorithm (a classical nature-inspired optimization algorithm), no selection, recombination, and mutation occurs in the PSO frame. This method views an individual in a population as a particle in the solution space, with zero mass and volume. These particles fly over the solution space at a given speed, updated according to the experience of the particles and the entire population. In traditional PSO, the term “experience” refers to the best position found by the particle (*pbest*) and by the entire population (*gbest*). All particles follow two *bests* to update their velocities and positions. Therefore, as a differential system [17,18], PSO involves two types of behaviors: cognitive and social. The formula to update a particle’s velocity and position is

$$\begin{cases} v_{id}^d = v_{id}^d + \varphi_1 r(pbest_{id}^d - x_{id}^d) + \varphi_2 r(gbest^d - x_{id}^d), \\ x_{id}^d = x_{id}^d + v_{id}^d, \end{cases} \tag{1}$$

where v is the velocity vector, x is the particle’s position in the solution space, d represents the dth-dimension, $pbest_{id}$ represents the best position found by the particle id, and $gbest$ represents the best position found by the entire population. The variable r is a random positive number drawn from the uniform distribution [0, 1]. φ_1 and φ_2 represent the acceleration constants. The PSO algorithm is shown in Algorithm 1.

Algorithm 1. Algorithm of particle swarm optimization

```
Input: Population size PopSize, and acceleration constants $\varphi_1, \varphi_2$
Output: The best solution.
1  Initialization;
2  while termination condition has not been met do
3      Evaluate the fitness for each particle according to its position vector x;
4      Update the best position pbest for each particle;
5      Update the best position gbest for the whole population;
6      for id=1 to PopSize do
7          for d=1 to Dim do
8              $v_{id}^d = v_{id}^d + \varphi_1 r(pbest_{id}^d - x_{id}^d) + \varphi_2 r(gbest_{id}^d - x_{id}^d)$;
9              $v_{id}^d = \min(VMAX^d, \max(-VMAX^d, v_{id}^d))$;
10             $x_{id}^d = x_{id}^d + v_{id}^d$;
11         end
12     end
13 end
14 Return the best position found by all of particles.
```

3.2. Applications and improvements

PSO is a practical approach with several success stories in many applications. Zhang et al. [19] proposed an improved adaptive particle swarm optimization to solve the reservoir operation problem. Bin et al. [20] proposed a novel binary particle swarm optimization to solve the haplotype inference by pure parsimony problem (HIPP). Davoodi et al. [21] proposed a new optimization approach, based on a hybrid algorithm combining improved quantum-behaved PSO and simplex algorithms for solving the power system load flow problem. Cervantes et al. [22] proposed an adaptive Michigan PSO and examined its application in nearest neighborhood classification. Wang et al. [23] proposed the use of nearest neighbor classification in conjunction with a neural-network classifier and uses PSO as the optimizer. Li et al. [24] suggested a novel fuzzy neural network, which adopts PSO to facilitate parameter estimation and improve accuracy, for system state forecasting. Chen et al. [25] proposed an online modeling algorithm for nonlinear and nonstationary systems with RBF neural network, in which the quantum PSO algorithm is introduced to optimize the tunable center vector and diagonal covariance matrix. Zhan et al. [26] used PSO for each population and developed a coevolutionary multiswarm PSO to solve multi-objective optimization problems. Lu et al. [27] investigated decision making and finite-time motion control for a group of robots, which introduced PSO to determine the probable position of the odor source. Wang et al. [28] distilled the middle-age hydration kinetics for cement hydration using phased hybrid evolution method and adopted PSO to search for the best value of coefficients for a given form of kinetics.

Despite the success of PSO in certain applications, traditional PSO still has room for improvements in updating velocity and topology structure. Several studies have focused on further improving the performance of PSO by integrating mechanisms from other algorithms, adding new strategies or techniques, and designing topologies. Mendes et al. [15] proposed a novel fully informed particle swarm (FIPS) that adopts a fully informed mechanism. The method allows every neighbor of a particle to contribute information for updating the velocity. Updating the velocity can be considered a cumulative effect of the information contributed by neighbors. This method increases the source of information for each particle, so it avoids dropping into the local optimum during evolution, and it improves the ability of converging and global searching for PSO. The traditional topologies in PSO are static; therefore, Liang and Siganthan [14] proposed a PSO method that adopts the dynamic topological strategy. This method divides the original population into several small swarms that regroup after every fixed number of iterations during the evolution process. On one hand, the division strategy ensures the diversity of the entire population. On the other hand, the regrouping strategy enables the exchange of information between small swarms. Liang et al. [29] also developed a comprehensive learning particle swarm optimizer (CLPSO), which has the distinction of independently optimizing the vector for each dimension. A particle receives information from the others with a certain probability and updates its velocity accordingly.

Leu et al. [30] proposed a grey evolutionary analysis to analyze the population distribution of particle swarm optimization during the evolutionary process. Lim and Isa [31] developed a teaching and peer-learning PSO algorithm by improving the cutting edge teaching-learning-based optimization algorithm and adapting the enhanced framework into the PSO. Calazan et al. [32] presented a novel massively parallel coprocessor for PSO’s implementation using reconfigurable hardware. Zhan et al. [33] presented an adaptive PSO that consists of a real-time evolutionary state estimation and an elitist learning strategy. Valdez et al. [34] proposed a new hybrid approach by combining particle swarm optimization and genetic algorithm using fuzzy logic to integrate the results of both methods and for parameters tuning.

4. Methodology

This section presents the proposed DTT-PSO method. First, the motivation of this method is introduced. Then, the details and mechanisms of the proposed DTT strategy are described. The user-defined parameters are then discussed in the following subsection. Finally, special cases of DTT-PSO are presented and discussed.

4.1. Dynamic tournament topology strategy

This method adopts the tournament strategy to randomly reconstruct topology, and it incorporates the fully informed strategy to update the velocity. In each iteration, topology reorganization will be conducted with probability P for each particle. When topology reorganization is performed for a particle, M relatively *better* guides are chosen as neighbors. The identity of those neighbors is determined using a tournament strategy.

For a tournament, $\lfloor K * PopSize \rfloor$ contestants are randomly chosen from the population, where K is the ratio of contestants and *PopSize* is the size of the population. The contestant with the best *pbest* in the tournament is chosen as one of the neighbors and will have the right to inform the current particle in the subsequent iterations (until the next topology change). This process of random sampling and selection is repeated M times to determine the M neighbors. Notably, the tournament is conducted with replacement, so the same particle may be chosen several times.

The velocity update of a particle depends on all M neighbors, determined by the accumulated information from all its neighbors. The velocity formula of the dth dimension of a particle in DTT-PSO is

$$v_{id}^{d} = \omega v_{id}^{d} + \varphi_0 r(pbest_{id}^{d} - x_{id}^{d}) + \sum_{i=1}^{M} \varphi_i r(pbest_{dn(id,i)}^{d} - x_{id}^{d}) \tag{2}$$

where $pbest_{id}$ represents the best position found by the particle id itself, $dn(id, i)$ represents the ith neighbor of particle id, and $pbest_{dn(id,i)}$ represents the best position found by the ith neighbor of particle id. A random inertia weight $\omega \in [0, \omega_{max}]$ is adopted in DTT-PSO to restrict the velocity change and improve the searching flexibility. Conventionally, ω_{max} is be set to 1 to balance the precision of the search. $(\varphi_0, \varphi_1, \ldots, \varphi_M)$ represent the acceleration constants, all set to the same value,

$$\varphi_0 = \varphi_1 = \cdots = \varphi_M = \frac{\varphi}{M+1}, \tag{3}$$

where φ constitutes the sum of the acceleration constants. The formula for velocity then becomes

$$v_{id}^{d} = \omega v_{id}^{d} + \frac{\varphi}{M+1} r(pbest_{id}^{d} - x_{id}^{d}) + \sum_{i=1}^{M} \frac{\varphi}{M+1} r(pbest_{dn(id,i)}^{d} - x_{id}^{d}). \tag{4}$$

The detailed algorithm of DTT-PSO is presented in the Algorithm 1, where r represents a random positive number drawn from the uniform distribution [0, 1].

Algorithm 2. Algorithm of DTT-PSO

```
Input: Population size PopSize, and acceleration constants $\varphi_0, \varphi_1, \cdots, \varphi_M$, and
       number of neighbors $M$, ratio of contestants $K$, probability of
       reorganization $P$.
Output: The best solution.
1  Initialization;
2  while termination condition has not been met do
3      Evaluate the fitness for each particle according to its position vector $x$;
4      Update the best position pbest for each particle;
5      for id=1 to PopSize do
6          if $r < P$ then
7              for i=1 to M do
8                  for j=1 to $\lfloor K * PopSize \rfloor$ do
9                      Choose an individual as contestant from the population at
                       random and add it into the tournament.
10                 end
11                 The contestant, which has the best fitness in tournament, is chosen
                   as $dn(id, i)$.
12             end
13         end
14     end
15     for id=1 to PopSize do
16         for d=1 to Dim do
17             $v_{id}^{d} = \omega v_{id}^{d} + \frac{\varphi}{M+1} r(pbest_{id}^{d} - x_{id}^{d}) + \sum_{i=1}^{M} \frac{\varphi}{M+1} r(pbest_{dn(id,i)}^{d} - x_{id}^{d})$;
18             $v_{id}^{d} = min(VMAX^{d}, max(-VMAX^{d}, v_{id}^{d}))$;
19             $x_{id}^{d} = x_{id}^{d} + v_{id}^{d}$;
20         end
21     end
22 end
23 Return the best position found by all of particles.
```

4.2. Mechanism

The tournament strategy extends and distributes the notion of the population's *best* solution. The danger of being fixated on a single population *best* position – as done in traditional PSO implementations – is that this *best* solution might lead the entire swarm into a suboptimal local trap. The DTT strategy creates an opportunity for a particle that is located in the global optimum area (but does not have the best fitness) to inform the other particles. Furthermore, this strategy also encourages input from a variety of non-best particles. A relatively good particle will stand out more using this strategy than in traditional PSO. In this way, the DTT strategy balances exploration and exploitation because it spreads the influence of leadership across more individuals, rather than just one. It preserves the exploration ability by maintaining diversity, but can exploit a potential optimum region if multiple *better* particles discover it. This strategy also adopts the fully informed strategy in which a particle simultaneously receives information from *better* guides. The strategy helps the swarm dilute the influence from a single *best* particle and distributes that influence across a society of *better* particles.

Furthermore, the dynamic random topology ensures a wide exchange of information between particles, rather than a directional flow. The reorganization of the topology is performed at different times for different particles, helping to diversify the sampling of the search space. If all the topologies were updated at the same time, then the neighborhoods would all be based on the same set of fitness values available at the time.

4.3. User-defined parameters

Aside from the classical parameters of PSO, our method involves three additional user-defined parameters: the probability of reorganization P, the number of neighbors M, and the ratio of contestants K. Figs. 1 and 2 illustrate the effects of parameters on the results for unimodal benchmark functions F_1, F_2 and multimodal

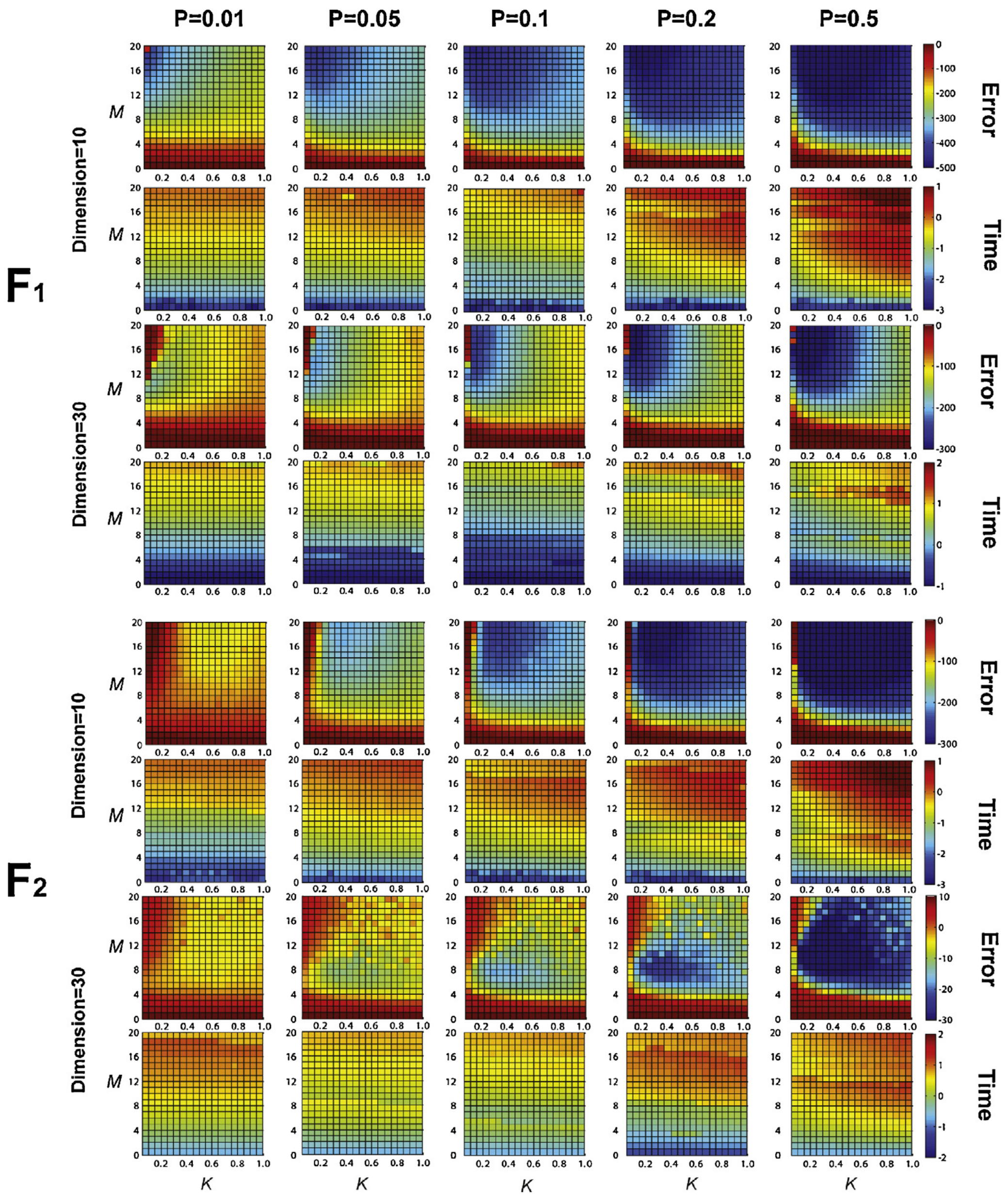

Fig. 1. The impact of parameter values on the results for unimodal benchmark functions F_1, F_2. Error and time represent the average error to the best value and average computation time (seconds). The natural logarithms of original values are used to enhance contrast.

benchmark functions F_7, F_8 on 10 and 30 dimensions. P, M, and K are discussed in detail in this subsection in relation to Figs. 1 and 2. The maximum number of fitness evaluations is set to 100,000, the population size is set to 50, and φ is set to 4.1. The experiment is run 30 times for each parameter setting to reduce statistical sampling effects. It should be noted that the negative values appear in figures because the original values are displayed in natural logarithm-scale to enhance contrast.

The frequency of topological reorganization is represented by P ($P \in [0, 1]$). At each iteration, the probability of selecting a new neighborhood is P. DTT-PSO adopts asynchronous reorganization. On the one hand, a larger P is suggested for unimodal problems to accelerate convergence. When P is set too large (e.g., $P \approx 1$), the topology is frequently reorganized. In this case, a particle is pulled by different neighbors at different iterations, and this situation leads to vibration, hindering local search. Furthermore, the frequent reorganization also increases the computation time. However, it should be noted that the influence of the increase of P on computation time is smaller in a 30-dimensional problem than in a 10-dimensional problem.

On the other hand, experimental results indicate that setting P to a smaller value for multimodal problems maintains the stability of topology during local search. When P is set too small (e.g., $P \approx 0$), the topology is rarely reorganized. In this case, DTT-PSO loses its dynamic characteristic, and a particle cannot receive useful information from other particles – only from its fixed neighbors. We have found that selecting P from the interval (0, 0.2] is adequate to balance the accuracy and computation time.

The number of neighbors is represented by M ($M \in \mathbb{Z}^+$, $M \geqslant 0$). This parameter controls the number of neighbors that simultaneously influences a particle. Each particle is fully informed by all of its neighbors [15]. According to Figs. 1 and 2, a larger M helps each particle receive information from multiple sources and aids in finding potential regions. When M is set too large, the particles tend to lose cognitive ability. The behavior of the different particles becomes correlated and results in similar particle behavior and

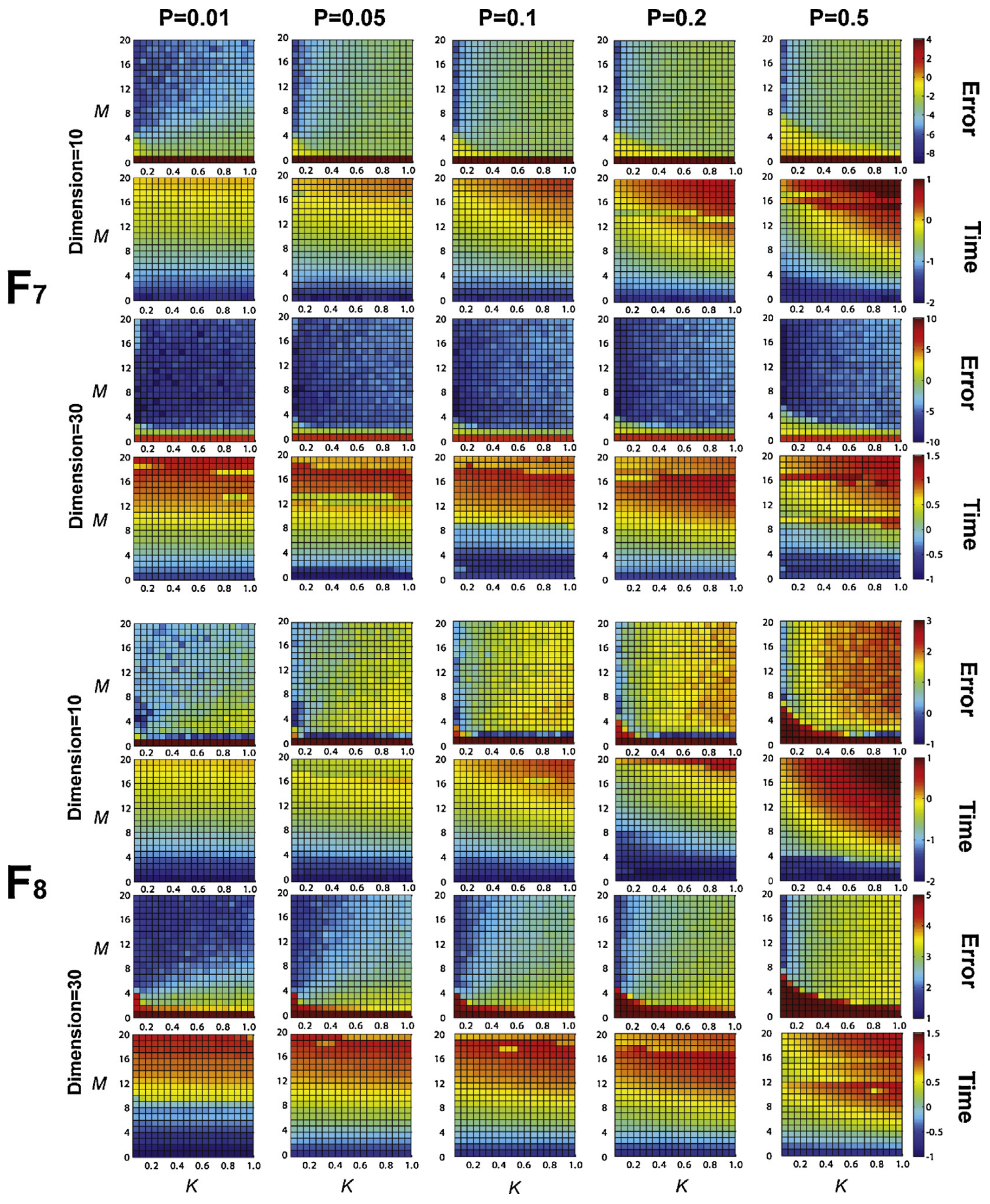

Fig. 2. The impact of parameter values on the results for multimodal benchmark functions F_7, F_8. The results are transformed to natural logarithm to enhance the contrast. Error and time represent the average error to the best value and average computation time (seconds). The natural logarithms of original values are used to enhance contrast.

premature convergence. Furthermore, as far as the time resources are concerned, a large M increases the time-complexity in updating velocities and positions. On the other hand, smaller M (e.g., $M = 0$) makes DTT-PSO gradually degenerate to a cognitive-only model and each particle only receives information from itself. In this situation, the particle cannot get useful information from the others. It can be observed from Figs. 1 and 2 that the error is reduced when M is set to a suitable value, but the time consumption increases with the increase of M. Experiments suggest that selecting M from the interval [5, 15] reduces the error with an acceptable computation time.

The ratio of contestants is represented by K ($K \in \mathbb{R}^+$, $\lfloor K * PopSize \rfloor \geqslant 1$). Thus, $\lfloor K * PopSize \rfloor$ represents the number of contestants. This parameter regulates the speed of convergence during the evolution process. If K is set too large, then neighborhoods will tend to be populated by a small set of particles with the very highest fitness values. This happens because it is very probable that at least one of those high-fitness particles will be selected for a given tournament. In this case, the probability of premature convergence increases. If K is set too small, the tournament will lose its characteristic of competition. For example, if $\lfloor K * PopSize \rfloor = 1$, each individual who is chosen into a tournament automatically wins because it is the only contestant in the tournament. In this case, the flow of information that indicates the possible best position is very slow and further slows down the convergence. From Figs. 1 and 2, we see that the increase of P enlarges the scope of selectable K on unimodal problems but narrows down the scope on multimodal problems. The combination of larger P and larger K encourages the global best to win tournaments. It accelerates the convergence for unimodal problems and leads to premature convergence for multimodal problems. Moreover, it can also be observed that the computation time increases when K is large, and this trend is more pronounced when P and M are also large. Experiments suggest that setting K to a larger value for unimodal problems speeds up convergence, and setting K to a smaller value helps in multimodal problems because it leads to a more diversified search.

4.4. Special cases of parameters

When the parameters are set according to the following combinations, DTT-PSO degenerates to special, interesting cases. The degeneration manifests that DTT-PSO is a generalization of the traditional PSO method.

- $M = 0$
  In this case, DTT-PSO is equivalent to a cognitive-only model. A particle does not receive any information from other particles.
- $P = 1$, $M = 1$, $K \to \infty$
  In this case, DTT-PSO is equivalent to the classical model: global PSO. With high probability, all particles will be included in each tournament, so the globally best particle always wins. Since each neighborhood consists of only one particle, the best particle is the only one of influence. The topology is reorganized at each iteration and ensures the update of the global best particles.
- $P = 1$, $M \to \infty$, $K \to \infty$
  In this case, DTT-PSO is equivalent to a social-only model. With high probability, the globally best particle will always be chosen and win the tournament. All particles receive the information of the global best location at each iteration and update their velocity accordingly.
- $P = 0$
  In this case, DTT-PSO is equivalent to a static fully informed model, which includes the particle itself in the neighborhood. The topology is fixed at the initial stage and remains unchanged during the evolution.

5. Experiments

5.1. Experiments on benchmark functions

5.1.1. Benchmark functions

The benchmark functions used to test DTT-PSO and the other methods are described in this subsection. Four unimodal and 12

Table 1
Formulas and properties of benchmark functions.

	Function	Search range	Best value
Unimodal functions	$F_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	0
	$F_2(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$	0
	$F_3(x) = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	$[-1.28, 1.28]^D$	0
	$F_4(x) = \sum_{i=1}^{D}\left(\lfloor x_i + 0.5 \rfloor\right)^2$	$[-100, 100]^D$	0
Multimodal functions	$F_5(x) = \sum_{i=1}^{D-1}\left(100\left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right)$	$[-2, 2]^D$	0
	$F_6(x) = -20e^{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)} + 20 + e$	$[-32, 32]^D$	0
	$F_7(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$	0
	$F_8(x) = \sum_{i=1}^{D}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[-5, 5]^D$	0
	$\begin{cases} F_9(x) = \sum_{i=1}^{D}\left(y_i^2 - 10\cos(2\pi y_i) + 10\right) \\ y_i = \begin{cases} x_i & \lvert x_i\rvert < \frac{1}{2} \\ \frac{round(2x_i)}{2} & \lvert x_i\rvert \geq \frac{1}{2} \end{cases} \quad (i = 1, 2, \cdots, D) \end{cases}$	$[-5, 5]^D$	0
	$\begin{cases} F_{10}(x) = \sum_{i=1}^{D-1}\left(100\left(z_i^2 - z_{i+1}\right)^2 + (z_i - 1)^2\right) + 390 \\ z = x - o + 1 \end{cases}$	$[-100, 100]^D$	390
	$\begin{cases} F_{11}(x) = \sum_{i=1}^{D}\frac{z_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}}) + 1 - 180 \\ z = (x - o) * M \end{cases}$	$[0, 600]^D$	−180
	$\begin{cases} F_{12}(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) - 330 \\ z = x - o \end{cases}$	$[-5, 5]^D$	−330
	$\begin{cases} F_{13}(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) - 330 \\ z = (x - o) * M \end{cases}$	$[-5, 5]^D$	−330
	$\begin{cases} F_{14}(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}\left[a^k\cos(2\pi b^k(z_i + 0.5))\right]\right) \\ \qquad -\sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}\left[a^k\cos(2\pi b^k * 0.5)\right]\right) + 90 \\ a = 0.5, b = 3, k\max = 20 \\ z = (x - o) * M \end{cases}$	$[-0.5, 0.5]^D$	90
	$\begin{cases} F_{15}(x) = F_7(F_5(z_1, z_2)) + F_7(F_5(z_2, z_3)) + \cdots \\ \qquad + F_7(F_5(z_{D-1}, z_D)) + F_7(F_5(z_D, z_1)) - 130 \\ z = x - o + 1 \end{cases}$	$[-3, 1]^D$	−130
	$\begin{cases} F_{16}(x) = F(z_1, z_2) + F(z_2, z_3) + \cdots \\ \qquad + F(z_{D-1}, z_D) + F(z_D, z_1) - 300 \\ z = (x - o) * M \end{cases}$	$[-100, 100]^D$	−300

multimodal benchmark functions were chosen [35,36], including seven CEC05 functions ($F_{10} - F_{16}$). Table 1 illustrates the formulas and properties of these functions.

5.1.2. Experimental settings

The benchmark functions defined in the previous subsection are used to test the performance of DTT-PSO. Here we report the results from all the benchmark datasets we tested on. Seven PSO and genetic algorithms were also used to compare the benchmark functions.

(1) Genetic algorithm (GA) [37], which is a widely used evolutionary algorithm.
(2) Traditional global PSO algorithms with inertia weight (GPSO) [2].
(3) Traditional local PSO algorithms with inertia weight and ring neighborhood (LPSO) [16].
(4) Fully informed PSO [15] whose velocity is calculated according to the information from all of the neighbors. The goodness weighted FIPS with URing (FIPS (URing)) and USquare (FIPS (USquare)) topologies are adopted.
(5) Dynamic multi-swarm PSO (DMS-PSO) [14] is a dynamic topological method that reorganizes neighborhoods during evolution.
(6) Comprehensive learning PSO (CLPSO) employs a comprehensive learning strategy that optimizes every dimension of the vectors and exchanges the information of the best between different particles [29].
(7) PSO with dynamic tournament topology strategy (DTT-PSO).

The results of GPSO, LPSO, FIPS (URing), FIPS (USquare), DMS-PSO, and CLPSO are extracted from [38]. For a fair comparison among these algorithms, the maximum fitness evaluation for DTT-PSO is set to 200,000, and the population size is set to 64, consistent with [38]. The value of the acceleration constant φ is set to 4.1. P is set to 0.05, M is set to 6, and K is set to 0.1. The experiments were conducted in terms of the 16 benchmark functions on 10 and 30 dimensions and were independently run 30 times for each method on each function to reduce statistical variation. All experiments were conducted in MATLAB with the use of the same machine.

5.1.3. Results and discussions

The comparison of optimization accuracy, including mean and standard deviation, between the proposed DTT-PSO and the other six methods is illustrated in Tables 2 and 3. For 10-dimensional problems, despite F_1, F_2, and F_6, whose results are under 10^{-8} and are generally considered noise, DTT-PSO is better than the other methods on F_3, F_5, F_{10}, F_{11} in terms of mean error. DTT-PSO is equal to the other benchmark functions on F_4. For 30-dimensional problems, despite F_1, and F_6, whose results are under 10^{-8}, DTT-PSO is better than the other methods on F_2, F_3, F_5, F_{11}, F_{14}, F_{16} and equal with GPSO, LPSO, FIPS(URing), FIPS(USquare), and CLPSO on

Table 2
The comparison of optimization accuracy, including mean error and standard deviation, between the proposed DTT-PSO and the other six methods on 10-dimensional problems.

	F_1	F_2	F_3	F_4
GA	4.06E−11(±3.08E−11)	1.30E−09(±7.25E−10)	9.22E−02(±4.49E−02)	**0.00E+00(±0.00E+00)**
GPSO	6.93E−89(±2.61E−88)	5.44E−31(±1.84E−30)	5.14E−04(±1.79E−04)	**0.00E+00(±0.00E+00)**
LPSO	1.90E−40(±3.53E−40)	5.63E−09(±6.70E−09)	1.22E−03(±5.16E−04)	**0.00E+00(±0.00E+00)**
FIPS(URing)	5.22E−28(±6.40E−28)	2.13E−10(±1.45E−10)	6.70E−04(±2.93E−04)	**0.00E+00(±0.00E+00)**
FIPS(Usquare)	9.68E−76(±1.58E−75)	5.60E−28(±9.95E−28)	4.61E−04(±2.23E−04)	**0.00E+00(±0.00E+00)**
DMSPSO	3.00E−19(±3.99E−19)	1.40E−02(±1.56E−02)	1.44E−03(±5.50E−04)	**0.00E+00(±0.00E+00)**
CLPSO	3.46E−102(±1.41E−101)	4.42E−14(±1.57E−13)	7.57E−04(±6.72E−04)	**0.00E+00(±0.00E+00)**
DTT-PSO	**3.46E−175(±0.00E+00)**	**1.54E−62(±4.99E−62)**	**1.25E−04(±9.93E−05)**	**0.00E+00(±0.00E+00)**
	F_5	F_6	F_7	F_8
GA	7.65E−02(±3.38E−02)	7.63E−06(±4.49E−06)	**5.05E−12(±5.73E−12)**	5.31E−01(±6.78E−01)
GPSO	2.19E+00(±9.46E−01)	2.66E−15(±0.00E+00)	6.04E−02(±2.23E−02)	1.26E+00(±1.28E+00)
LPSO	3.73E+00(±1.82E−01)	2.66E−15(±0.00E+00)	3.24E−02(±1.81E−02)	2.04E+00(±8.71E−01)
FIPS(URing)	2.59E+00(±1.67E−01)	3.37E−11(±9.12E−11)	1.80E−02(±1.81E−02)	2.67E−01(±4.47E−01)
FIPS(Usquare)	3.09E−01(±4.49E−02)	1.01E−15(±1.80E−15)	2.94E−03(±4.64E−03)	**4.57E−02(±1.83E−01)**
DMSPSO	4.23E+00(±3.30E−01)	1.65E−10(±1.12E−10)	1.44E−01(±5.04E−02)	3.69E+00(±1.51E+00)
CLPSO	3.44E+00(±1.49E+00)	2.66E−15(±0.00E+00)	2.72E−02(±1.98E−02)	1.76E+00(±1.24E+00)
DTT-PSO	**4.35E−03(±6.37E−04)**	**0.00E+00(±0.00E+00)**	5.25E−03(±7.96E−03)	1.06E+00(±9.03E−01)
	F_9	F_{10}	F_{11}	F_{12}
GA	4.77E+00(±1.79E+00)	1.67E+01(±1.69E+01)	1.67E+01(±1.77E+01)	1.93E+02(±5.19E+00)
GPSO	**6.67E−02(±2.54E−01)**	8.24E+00(±1.80E+01)	3.49E+02(±5.57E+02)	6.88E−01(±3.54E−01)
LPSO	1.93E+00(±1.05E+00)	1.46E+00(±1.69E+00)	2.51E+00(±1.95E+00)	**2.97E−02(±2.54E−02)**
FIPS(URing)	3.58E+00(±8.82E−01)	3.57E+00(±1.45E+00)	4.13E+01(±4.72E+01)	3.72E−01(±1.83E−01)
FIPS(Usquare)	2.30E+00(±9.92E−01)	7.41E−01(±1.32E+00)	9.35E−01(±1.57E+00)	1.60E−01(±3.65E−01)
DMSPSO	4.90E+00(±8.41E−01)	1.19E+01(±2.76E+01)	2.65E−01(±6.38E−02)	4.20E+00(±1.35E+00)
CLPSO	2.07E+00(±1.05E+00)	1.15E+01(±1.63E+01)	1.27E+03(±4.64E−13)	1.99E+00(±1.59E+00)
DTT-PSO	3.50E+00(±1.38E+00)	**3.81E−01(±2.06E+00)**	**1.15E−01(±4.91E−02)**	1.31E+01(±4.09E+00)
	F_{13}	F_{14}	F_{15}	F_{16}
GA	3.45E+00(±7.85E+00)	1.16E+02(±1.68E+01)	1.19E+03(±2.12E+03)	2.77E+00(±1.94E+00)
GPSO	1.19E+00(±1.15E+00)	1.06E+02(±8.42E+00)	4.40E+02(±6.82E+02)	6.70E−01(±1.92E−01)
LPSO	2.10E+00(±1.02E+00)	8.96E+01(±5.59E+00)	4.49E+00(±4.46E+00)	**6.06E−01(±2.53E−01)**
FIPS(URing)	**2.38E−01(±4.36E−01)**	1.50E+01(±5.36E+00)	4.98E+03(±2.19E+03)	1.47E+00(±3.68E−01)
FIPS(Usquare)	1.45E+01(±1.07E+01)	4.05E+00(±1.13E+00)	1.81E+00(±4.65E−01)	2.56E+00(±2.48E−01)
DMSPSO	8.19E+00(±2.46E+00)	**2.22E+00(±9.97E−01)**	8.54E−01(±1.74E−01)	2.15E+00(±3.40E−01)
CLPSO	1.40E+01(±6.45E+00)	4.11E+00(±1.12E+00)	**3.99E−01(±1.11E−01)**	2.63E+00(±5.89E−01)
DTT-PSO	1.32E+01(±4.30E+00)	3.34E+00(±1.12E+00)	8.71E−01(±3.02E−01)	3.22E+00(±4.47E−01)

The bold font indicates the method which achieves the best performance on each function.

Table 3
The comparison of optimization accuracy, including mean error and standard deviation, between the proposed DTT-PSO and the other six methods on 30-dimensional problems.

	F_1	F_2	F_3	F_4
GA	6.40E−09(±2.46E−09)	2.00E−02(±1.29E−02)	1.11E+00(±4.51E−01)	2.00E−01(±4.07E−01)
GPSO	1.04E−21(±3.64E−21)	1.56E+01(±9.47E+00)	9.83E−03(±3.38E−03)	**0.00E+00(±0.00E+00)**
LPSO	5.13E−08(±4.62E−08)	4.25E+02(±1.08E+02)	2.54E−02(±5.16E−03)	**0.00E+00(±0.00E+00)**
FIPS(URing)	3.44E−05(±1.20E−05)	3.96E+03(±8.23E+02)	1.13E−02(±2.88E−03)	**0.00E+00(±0.00E+00)**
FIPS(Usquare)	3.53E−42(±2.98E−42)	6.15E−02(±3.91E−02)	4.11E−03(±1.58E−03)	**0.00E+00(±0.00E+00)**
DMSPSO	2.34E−01(±1.99E−01)	1.40E+03(±2.35E+02)	5.50E−02(±1.45E−02)	3.93E+00(±2.45E+00)
CLPSO	7.21E−21(±5.23E−21)	1.79E+02(±7.47E+01)	3.55E−03(±9.66E−04)	**0.00E+00(±0.00E+00)**
DTT-PSO	**1.21E−94(±1.64E−94)**	**1.95E−06(±2.25E−06)**	**8.05E−04(±3.36E−04)**	**0.00E+00(±0.00E+00)**

	F_5	F_6	F_7	F_8
GA	2.23E+01(±2.34E+01)	5.95E−05(±1.20E−05)	3.50E−10(±1.66E−10)	4.94E+00(±2.02E+00)
GPSO	2.43E+01(±1.77E+00)	5.85E−12(±6.97E−12)	1.48E−02(±1.31E−02)	2.81E+01(±6.36E+00)
LPSO	2.52E+01(±4.99E−01)	1.05E−04(±6.75E−05)	9.25E−04(±2.56E−03)	3.62E+01(±7.15E+00)
FIPS(URing)	2.56E+01(±4.26E−01)	1.21E−03(±2.68E−04)	2.97E−03(±5.20E−03)	1.02E+02(±1.35E+01)
FIPS(Usquare)	2.28E+01(±2.76E−01)	4.20E−15(±1.79E−15)	4.59E−05(±2.09E−04)	2.51E+01(±5.06E+00)
DMSPSO	2.72E+01(±3.11E−01)	6.88E−01(±6.95E−01)	4.71E−01(±2.48E−01)	6.54E+01(±7.85E+00)
CLPSO	2.56E+01(±4.64E−01)	1.87E−11(±6.56E−12)	**0.00E+00(±0.00E+00)**	**2.26E+00(±1.57E+00)**
DTT-PSO	**1.62E+01(±3.13E−01)**	**2.66E−15(±0.00E+00)**	1.56E−03(±4.17E−03)	5.77E+00(±1.87E+00)

	F_9	F_{10}	F_{11}	F_{12}
GA	2.24E+01(±3.35E+00)	3.17E+02(±1.64E+02)	2.56E+02(±6.34E+01)	3.54E+03(±2.83E+01)
GPSO	1.66E+01(±5.94E+00)	1.81E+02(±3.29E+02)	4.41E+02(±9.62E+02)	**1.94E−02(±9.49E−03)**
LPSO	3.55E+01(±7.73E+00)	6.25E+01(±2.72E+01)	1.86E+02(±2.39E+02)	4.79E−02(±1.26E−02)
FIPS(URing)	9.57E+01(±1.01E+01)	3.50E+02(±6.39E+02)	1.42E+04(±2.51E+04)	2.09E+00(±1.60E+00)
FIPS(Usquare)	3.34E+01(±5.46E+00)	**5.18E+01(±4.78E+01)**	1.96E+01(±3.86E+01)	8.37E+01(±7.15E+01)
DMSPSO	4.84E+01(±6.58E+00)	5.36E+03(±7.71E+03)	1.20E+00(±1.69E−01)	6.77E+01(±7.42E+00)
CLPSO	**5.20E+00(±2.19E+00)**	8.13E+01(±5.48E+01)	4.70E+03(±2.75E−11)	4.94E+00(±2.21E+00)
DTT-PSO	9.47E+00(±2.24E+00)	6.48E+01(±7.28E+01)	**1.10E−02(±4.85E−03)**	1.11E+02(±1.65E+01)

	F_{13}	F_{14}	F_{15}	F_{16}
GA	5.35E+02(±6.97E+01)	3.46E+01(±3.61E+00)	2.38E+01(±9.50E+00)	1.46E+01(±1.49E−02)
GPSO	**7.82E+01(±3.59E+01)**	1.95E+01(±3.57E+00)	2.81E+00(±5.89E−01)	1.24E+01(±4.94E−01)
LPSO	1.31E+02(±2.38E+01)	2.09E+01(±1.37E+00)	5.10E+00(±1.14E+00)	1.25E+01(±2.03E−01)
FIPS(URing)	2.01E+02(±1.97E+01)	3.99E+01(±1.42E+00)	1.65E+01(±1.71E+00)	1.40E+01(±1.29E−01)
FIPS(Usquare)	1.69E+02(±6.61E+01)	3.48E+01(±2.31E+00)	2.67E+01(±8.75E+00)	1.26E+01(±2.42E−01)
DMSPSO	1.13E+02(±9.34E+00)	2.71E+01(±1.20E+00)	9.35E+00(±1.15E+00)	1.23E+01(±2.86E−01)
CLPSO	9.44E+01(±1.78E+01)	3.21E+01(±2.08E+00)	**1.73E+00(±3.10E−01)**	1.30E+01(±2.46E−01)
DTT-PSO	8.47E+01(±1.30E+01)	**1.31E+01(±1.86E+00)**	3.26E+00(±8.65E−01)	**1.21E+01(±3.19E−01)**

The bold font indicates the method which achieves the best performance on each function.

F_4 in terms of mean error. Noticeably, although DTT-PSO is not the best strategy on F_{14} or F_{16} for 10-dimensional problems, it is the best one on the same functions for high-dimensional problems (30-dimensional), a result indicating that the dynamic tournament topology can help PSO solve complex problems. Compared with the promising results on mean error, the standard deviation of DTT-PSO is less impressive, yielding sufficiently low standard deviation on a few functions (F_1, F_2, F_3, F_5, F_6 for 10-dimensional problems and F_1, F_2, F_3, F_6 for 30-dimensional problems). Therefore, future work will involve investigating strategies to make the DTT-PSO more stable.

The performance of DTT-PSO is worse than most methods on F_{12} for both 10-dimensional and 30-dimensional cases. The landscape of F_{12} has multiple local optima around the global optimum, with values very close to the value of the global optimum [36]. The *pbest* of particles in these nearby local optima can easily yield a better value than a *pbest* near the global optimum, and hence mislead the swarm into the non-optimal basin. In comparison with DTT-PSO, the LPSO and GPSO, which rank first on 10-dimensional F_{12} and 30-dimensional F_{12}, respectively, do not adopt fully informed or dynamic topological strategies and thus enable themselves to concentrate on converging to *gbest* or *lbest*. This hypothesis can be further verified from the phenomenon that DTT-PSO is worse than many other methods on F_9, F_{13}, F_{16} for 10-dimensional problems, and on F_7 for 30-dimensional problems. However, it should be noted that with the increase of dimension, the DTT-PSO gradually exhibits its advantage for finding a better solution in comparison with the other methods. Although the complexities of the problems are increased with the expansion of dimension, the preserved diversity for the swarm in DTT-PSO still enables it to discover potential regions in the search space.

To thoroughly and fairly compare DTT-PSO and other methods, statistical t-tests [39] were performed, and the results are shown in Table 4. The number of benchmark functions showing that DTT-PSO is significantly better than, almost the same as, and significantly worse than the other algorithms is illustrated in this table. The level of significance is 0.05. The “Merit” score was calculated by subtracting the “worse” score from the “better” score. The result shows the degree and difference by which DTT-PSO is better than the other methods. The table shows that DTT-PSO is better than the compared algorithms. Furthermore, the value of merit increases with the expansion of dimension (10-dimensional to 30-dimensional functions), a result implying that the dynamic tournament topology strategy enables PSO to solve complex problems.

To demonstrate the convergence speed of DTT-PSO, the evolving process of mean accuracy for each algorithm on all benchmark functions is shown in Figs. 3 and 4. Compared with the convergence curve produced by the other methods, the speed of DTT-PSO is faster on functions F_1, F_2, F_3, F_4, F_5, F_6, and F_{11} for both 10- and 30-dimensional problems. DTT-PSO is faster than the other algorithms on 30-dimensional F_{11}, F_{14}, and F_{16}, but not on the same 10-dimensional problems, a result indicating that the proposed strategy is applicable to solving complex problems. However, the only fly in the ointment is on F_{10}; DTT-PSO is the fastest but only for the 10-dimensional problem.

Table 4
The statistical analysis *t*-tests. The number of benchmark functions (out of the 16 tested) that the DTT-PSO is significantly better than (Better), about the same as (Same), and significantly worse than (Worse) the other algorithms. The Merit score is calculated by subtracting Worse from Better, so can be a positive or negative value.

		GA	GPSO	LPSO	FIPS (URing)	FIPS (USquare)	DMSPSO	CLPSO
10D	Better	11	8	11	10	8	10	8
	Same	2	4	1	2	4	2	4
	Worse	3	4	4	4	4	4	4
	Merit	8	4	7	6	4	6	4
30D	Better	13	10	12	13	12	15	9
	Same	2	4	3	2	3	0	2
	Worse	1	2	1	1	1	1	5
	Merit	12	8	11	12	11	14	4

Since the introduction of the dynamic tournament topology increases the complexity of the algorithm, the computation time was also evaluated. Figs. 5 and 6 illustrate the comparison of average computation time before reaching the fixed error levels, which are determined according to the median of compared methods and defined in Table 5. It can be observed that DTT-PSO ranks at the middle level on 10-dimensional problems. However, it becomes the fastest method on 30-dimensional problems. The results indicate that, with the increase of complexity of problems, the relative influence of the dynamic tournament topology on computation time decreases gradually while its influence on accuracy increases.

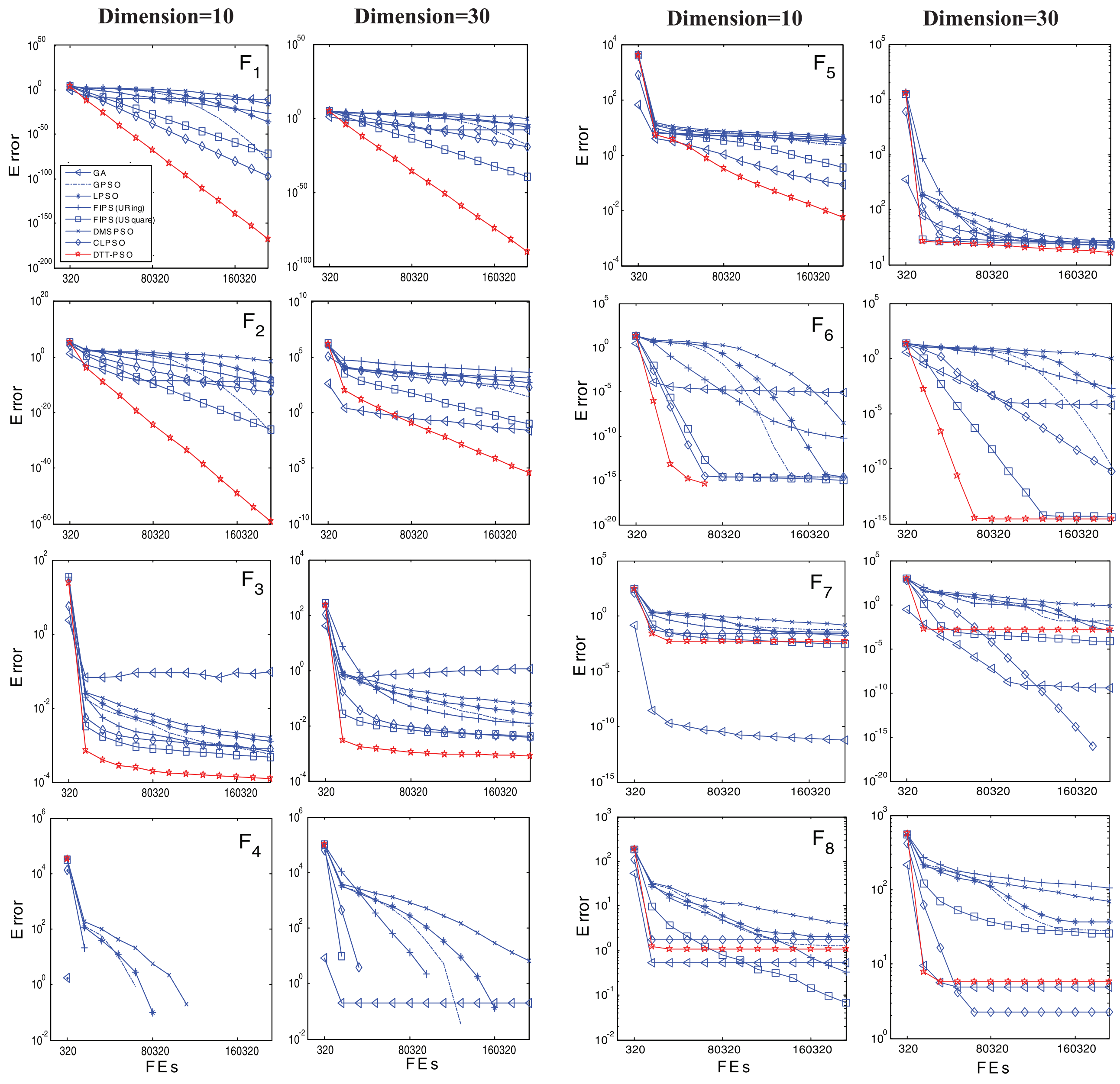

Fig. 3. The comparison of evolutionary curve of error on F_1 to F_8 problems.

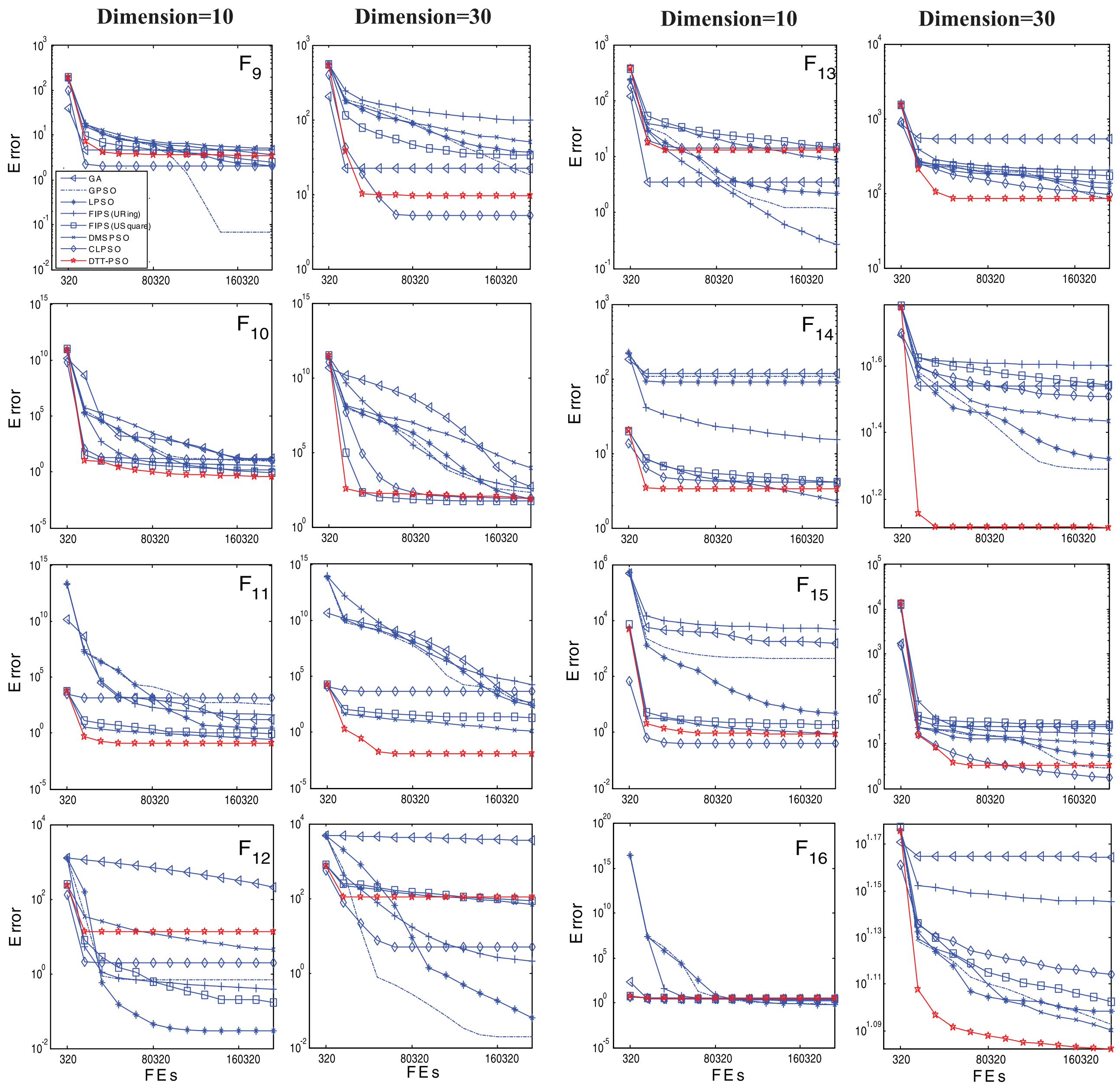

Fig. 4. The comparison of evolutionary curve of error on F_9 to F_{16} problems.

The tournament strategy extends the definition of the best position and enables a particle that is located in the global optimum area to inform the other particles, even if it does not have the best fitness. This strategy also encourages the individual whose personal best is larger to inform other individuals. Furthermore, the dynamic characteristic ensures a wide exchange rather than a directional flow of information between individuals. The dynamic tournament topology enables the swarm to find a better solution and helps avoid dropping into a local optimum.

5.2. Application in optimization of neural network

In this subsection, we applied DTT-PSO to the design of a neural network. The task here is to design a neural network for a robot

Table 5
Fixed error levels of problems. The levels are determined according to the median of compared methods.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
10D	1.00E−40	1.00E−09	7.00E−04	0.00E+00	2.50E+00	2.70E−15	2.00E−02	1.20E+00
30D	1.00E−09	1.00E+02	1.00E−02	0.00E+00	2.50E+01	1.00E−05	1.00E−03	2.70E+01
	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
10D	3.00E+00	5.00E+00	1.00E+01	1.00E+00	5.00E+00	1.00E+01	3.00E+00	2.50E+00
30D	3.00E+01	1.00E+02	2.00E+02	1.00E+01	1.20E+02	3.00E+01	7.00E+00	1.25E+01

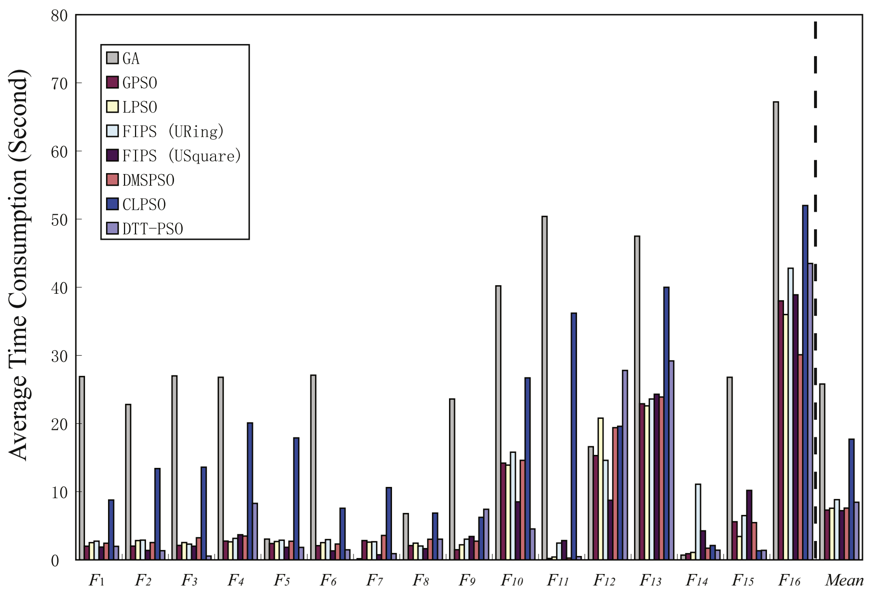

Fig. 5. Average computation time (seconds) before reaching the fixed error levels on 10-dimensional problems. The total time of one trial is used as the computation time if the required accuracy is not reached.

that forages in a two-dimensional discrete environment [40]. The food items are distributed randomly in the environment at regular time intervals. A feedforward neural network is adopted for this task. Its inputs include the action from the previous time step, and the sensory inputs of the first, second, and third nearest food at the current time step. The sensory inputs consist of the egocentric angle and distance from the robot to that food. The output layer encodes the current planned action using the binary encoding

- 1 1: step forward
- 1 0: turn left
- 0 1: turn right
- 0 0: do nothing

The environment and network are shown in Fig. 7. A three-layer feedforward neural network with 5 nodes in its hidden layer was used in this experiment. The objective is to design a neural network that directs the robot to eat as many food items as possible during its life.

We cannot use conventional backpropagation here because there is no direct teaching signal. However, nature-inspired algorithms can be used to change the network to optimize fitness. Since the total number of food items is constant for a given lifetime, the total number of *unconsumed* food items is our cost function. We compared DTT-PSO to the same set of alternative optimization algorithms mentioned in Section 5.1.2. For all compared methods, the maximum number of fitness evaluations is set to 10,000, and the population size is set to 20. For DMS-PSO, each swarm's population size is set to 5, and the number of swarms is set to 4. For DTT-PSO and FIPS, the value of acceleration constant φ is set to 4.1. For GPSO, LPSO, and DMS-PSO, each acceleration constant is set to 2. For CLPSO, each acceleration constant is set to 1.49445. For

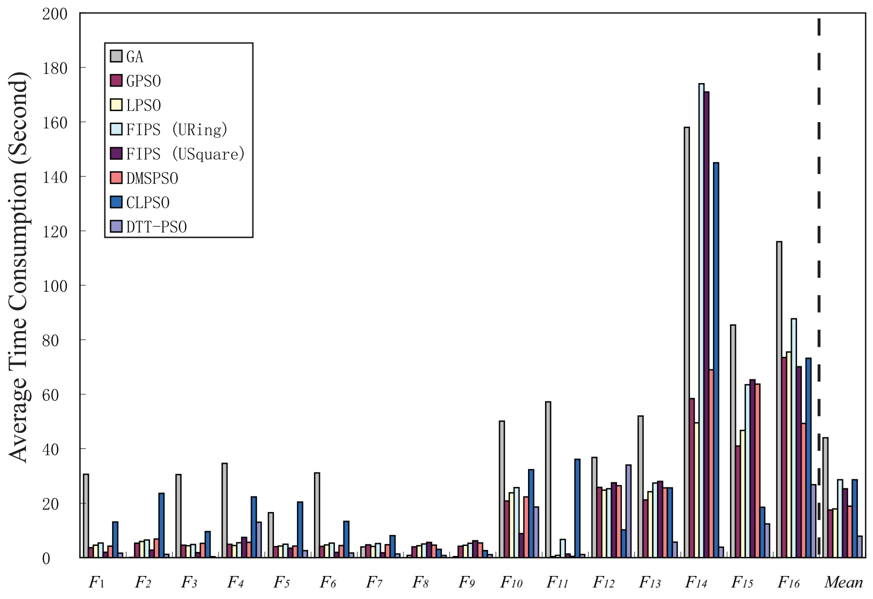

Fig. 6. Average computation time (seconds) before reaching the fixed error levels on 30-dimensional problems. The total time of one trial is used as the computation time if the required accuracy is not reached.

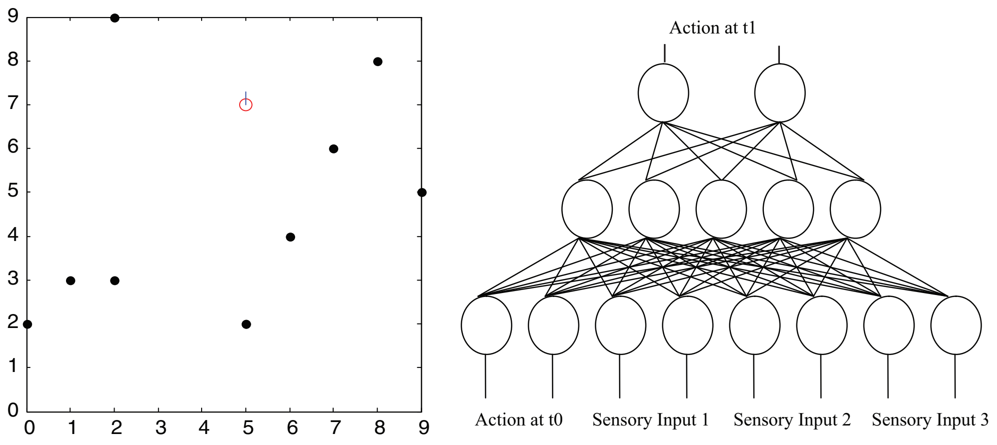

Fig. 7. The environment of foraging task and the neural network structure of the robot.

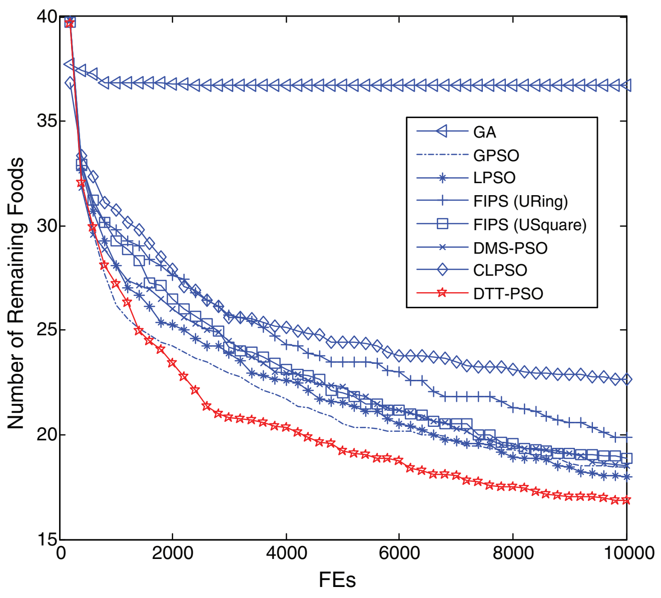

Fig. 8. The comparison of evolutionary curve of number of remaining foods between all methods on robot foraging task.

DTT-PSO, *P* is set to 0.2, *M* is set to 5, and *K* is set to 0.4. For GA, the crossover fraction is set to 0.8. The experiment is run 30 times for all compared methods. Fig. 8 illustrates the evolution curves of the seven methods, plotting the number fitness evaluations versus the historical best number of remaining foods, averaged over the 30 runs. Comparing the curves shown in this figure, the DTT-PSO demonstrates better performance among all compared methods on this task. This result illustrates that the DTT-PSO is also effective for finding neural network weights.

6. Conclusions

A dynamic tournament topology strategy to improve PSO has been proposed in this study. The strategy generalizes the definition of the best position and ensures a wide exchange of information. Instead of focussing on the historical best solution, the proposed strategy chooses a number of guides selected randomly, but with preference going to particles with higher fitness. Moreover, each particle can be guided by a different set of better particles, encouraging the sharing of information across a wide network. Compared with traditional PSO, this strategy preserves the diversity of the population, and enables the swarm to find a better solution, avoiding falling into a local optimum. Its effectiveness is especially pronounced when solving complex problems.

Sixteen well-known benchmark functions are used to evaluate the performance of the proposed approach. We also compared its performance against a host of different algorithms. Experimental results illustrate that the proposed dynamic tournament topology strategy demonstrates favorable performance, especially in terms of accuracy and convergence speed. Furthermore, as far as the time resources are concerned, although the introduction of this strategy increases time complexity of the optimization, the relative influence of dynamic tournament topology on computation time decreases gradually while its accuracy increases.

Furthermore, the DTT-PSO is applied to the optimization of a neural network for a simple robot foraging task. DTT-PSO demonstrates a better performance among all compared methods on this task and shows that DTT-PSO is able to solve real problems.

For future work, more real-world applications from other fields will help to further investigate the effectiveness of DTT-PSO. In order to improve the performance of robot foraging task, the self organization neural network will be used instead of feedforward neural network. Furthermore, we plan to investigate an adaptive method for selecting and changing user defined parameters to improve the method's stability.

Acknowledgement

This work was supported by National Natural Science Foundation of China under Grant Nos. 61573166, 61572230, 61373054, 61472164, 81301298, 61302128, 61472163, Shandong Provincial Natural Science Foundation, China, under Grant ZR2015JL025, Science and technology project of Shandong Province under Grant No. 2015GGX101025, Jinan Youth Science & Technology Star Project under Grant No. 2013012, Natural Sciences and Engineering Research Council of Canada (NSERC).

Appendix A.

The source code of DTT-PSO, written in Matlab, is available at https://sourceforge.net/projects/dttpso/files.

References

[1] M.J.D. Powell, An efficient method for finding the minimum of a function of several variables without calculating derivatives, Comput. J. 7 (2) (1964) 155–162.
[2] J. Kennedy, R.C. Eberhart, A new optimizer using particle swarm theory, in: Proc. Sixth Int. Symposium on Micromachine and Human Science, 1995, pp. 39–43.

[3] M. Randall, Differential evolution for a constrained combinatorial optimisation problem, Int. J. Metaheurist. 1 (4) (2011) 279–297.
[4] R.-C. David, R.-E. Precup, E.M. Petriu, M.-B. Radac, S. Preitl, Gravitational search algorithm-based design of fuzzy control systems with a reduced parametric sensitivity, Inform. Sci. 247 (2013) 154–173.
[5] J. Zhou, L. Chen, C.L. Philip Chen, Y. Zhang, H.-X. Li, Fuzzy clustering with the entropy of attribute weights, Neurocomputing 198 (2016) 125–134.
[6] M.R.D. Savio, A. Sankar, N.R. Vijayarajan, A novel enumeration strategy of maximal bicliques from 3-dimensional symmetric adjacency matrix, Int. J. Artif. Intell. 12 (2) (2014) 42–56.
[7] J. Ciurana, G. Arias, T. Ozel, Neural network modeling and particle swarm optimization (PSO) of process parameters in pulsed laser micromachining of hardened AISI H13 steel, Mater. Manuf. Process. 24 (3) (2009) 358–368.
[8] M. Eslami, H. Shareef, A. Mohamed, Power system stabilizer design using hybrid multi-objective particle swarm optimization with chaos, J. Central South Univ. Technol. 18 (5) (2011) 1579–1588.
[9] J.-C. Hung, Adaptive Fuzzy-GARCH model applied to forecasting the volatility of stock markets using particle swarm optimization, Inform. Sci. 181 (20) (2011) 4673–4683.
[10] T.S. Lim, V.C. Koo, H.T. Ewe, et al., A SAR autofocus algorithm based on particle swarm optimization, Prog. Electromagn. Res. B 1 (2008) 159–176.
[11] H.A. Nguyen, H. Guo, K.-S. Low, Real-time estimation of sensor node's position using particle swarm optimization with log-barrier constraint, IEEE Trans. Instrum. Meas. 60 (11) (2011) 3619–3628.
[12] Y. del Valle, G.K. Venayagamoorthy, S. Mohagheghi, et al., Particle swarm optimization: basic concepts, variants and applications in power systems, IEEE Trans. Evol. Comput. 12 (2) (2008) 171–195.
[13] S.H. Zainud-Deen, W.M. Hassen, E.M. Ali, et al., Breast cancer detection using a hybrid finite difference frequency domain and particle swarm optimization techniques, Prog. Electromagn. Res. B 3 (2008) 35–46.
[14] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, in: Proc. Swarm Intell. Symp., 2005, pp. 124–129.
[15] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evol. Comput. 8 (3) (2004) 204–210.
[16] J. Kennedy, R. Mendes, Population structure and particle swarm performance, in: Proc. IEEE Congress on Evolutionary Computation, 2002, pp. 1671–1676.
[17] T. Li, Yu.V. Rogovchenko, Oscillation criteria for even-order neutral differential equations, Appl. Math. Lett. 61 (2016) 35–41.
[18] T. Li, Yu.V. Rogovchenko, Oscillation of second-order neutral differential equations", Math. Nachrich. 288 (2015) 1150–1162.
[19] Z.B. Zhang, Y.Z. Jiang, S.H. Zhang, S.M. Geng, H. Wang, G.Q. Sang, An adaptive particle swarm optimization algorithm for reservoir operation optimization, Appl. Soft Comput. 18 (2014) 167–177.
[20] W. Bin, Z. Jing, Haplotype inference using a novel binary particle swarm optimization algorithm, Appl. Soft Comput. 21 (2014) 415–422.
[21] E. Davoodi, M.T. Hagh, S.G. Zadeh, A hybrid improved quantum-behaved particle swarm optimization-simplex method (IQPSOS) to solve power system load flow problems, Appl. Soft Comput. 21 (2013) 171–179.
[22] A. Cervantes, I.M. Galvan, P. Isasi, AMPSO: a new particle swarm method for nearest neighborhood classification, IEEE Trans. Cybern. 39 (5) (2009) 1082–1091.
[23] L. Wang, B. Yang, Y. Chen, X. Zhang, J. Orchard, Improving neural-network classifiers using nearest neighbor partitioning, IEEE Trans. Neural Netw. Learning Syst. (2016), http://dx.doi.org/10.1109/TNNLS.2016.2580570 (in press).
[24] D. Li, W. Wang, F. Ismail, Fuzzy neural network technique for system state forecasting, IEEE Trans. Cybern. 43 (5) (2013) 1484–1494.
[25] H. Chen, Y. Gong, X. Hong, Online modeling with tunable RBF network, IEEE Trans. Cybern. 43 (3) (2013) 935–947.
[26] Z.-h. Zhan, J. Li, J. Cao, J. Zhang, et al., Multiple populations for multiple objectives: a coevolutionary technique for solving multiobjective optimization problems, IEEE Trans. Cybern. 43 (2) (2013) 445–463.
[27] Q. Lu, S. Liu, X. Xie, J. Wang, Decision making and finite-time motion control for a group of robots, IEEE Trans. Cybern. 43 (2) (2013) 738–750.
[28] L. Wang, B. Yang, A. Abraham, Distilling middle-age cement hydration kinetics from observed data using phased hybrid evolution, Soft Comput. (2016), http://dx.doi.org/10.1007/s00500-015-1723-4 (in press).
[29] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.
[30] M.S. Leu, M.F. Yeh, S.C. Wang, Particle swarm optimization with grey evolutionary analysis, Appl. Soft Comput. 13 (10) (2013) 4047–4062.
[31] W.H. Lim, N.A.M. Isa, Teaching and peer-learning particle swarm optimization, Appl. Soft Comput. 18 (3) (2014) 39–58.
[32] R.M. Calazan, N. Nedjah, L.M. Mourelle, A hardware accelerator for particle swarm optimization, Appl. Soft Comput. 14 (2014) 347–356.
[33] Z.-H. Zhan, J. Zhang, Y. Li, H.S.-H. Chung, Adaptive particle swarm optimization, IEEE Trans. Cybern. 39 (6) (2009).
[34] F. Valdez, P. Melin, O. Castillo, An improved evolutionary method with fuzzy logic for combining particle swarm optimization and genetic algorithms, Appl. Soft Comput. 11 (2011) 2625–2632.
[35] X. Yao, Y. Liu, G.M. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (2) (1999) 82–102.
[36] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore and KanGAL, 2005, Report Number 2005005.
[37] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
[38] L. Wang, B. Yang, Y. Chen, Improving particle swarm optimization using multi-layer searching strategy, Inform. Sci. 274 (2014) 70–94.
[39] G.E.-P. Box, J.S. Hunter, W.G. Hunter, Statistics for Experiments: Design, Innovation, and Discovery, 2nd ed., Wiley, New York, 2005.
[40] S. Nolfi, D. Parisi, J.L. Elman, Learning and evolution in neural networks, Adapt. Behav. 3 (1) (1994) 5–28.