Gene Targeting Particle Swarm Optimization for Large-Scale Optimization Problem

Zhi-Fan Tang
School of Computer Science and Engineering
South China University of Technology
Guangzhou, 510006, China
helen39@163.com

Liu-Yue Luo (Co-First Author)
School of Computer Science and Engineering
South China University of Technology
Guangzhou, 510006, China
FluffyL@163.com

Xin-Xin Xu
School of Computer Science and Technology
Ocean University of China
Qingdao, 266100, China
xuxinxin0107@163.com

Jian-Yu Li (Corresponding Author)
College of Artificial Intelligence
Nankai University
Tianjin, 300350, China
jianyulics@foxmail.com

Jing Xu
College of Artificial Intelligence
Nankai University
Tianjin, 300350, China
xujing@nankai.edu.cn

Jing-Hui Zhong
School of Computer Science and Engineering
South China University of Technology
Guangzhou, 510006, China

Jun Zhang
College of Artificial Intelligence
Nankai University
Tianjin, 300350, China
junzhang@ieee.org

Zhi-Hui Zhan (Corresponding Author)
College of Artificial Intelligence
Nankai University
Tianjin, 300350, China
zhanapollo@163.com

***Abstract*—With the development of information technology, solving large-scale optimization problems (LSOPs) has become an urgent issue in the artificial intelligence (AI) for solving practical problems. LSOPs are optimization problems with high dimensions and large search space. Although many improved evolutionary computation (EC) algorithms have been proposed as promising AI approaches for dealing with LSOPs, they still face two challenges, which are easily falling into local optima and having slow convergence speed. Focusing on these two challenges, this paper proposes to integrate the gene targeting (GT) technology in the biotechnology with the particle swarm optimization (PSO), so as to propose a gene targeting particle swarm optimization (GTPSO) algorithm. The GTPSO works well in solving LSOPs due to the following three novel designs. Firstly, a Monte Carlo probabilistically targeting strategy is used to determine the poorly-performing dimension(s) of the globally best particle, so as to balance the targeting chance of all the dimensions. Secondly, a dual GT (DGT) strategy is proposed, including two GT strategies called GT1 and GT2, to modify the poorly-performing dimension(s), which can balance diversity and convergence at the same time. Thirdly, a greedy strategy is used to accept the targeted new solution only if it is better than the original globally best particle, so as to ensure the quality of the evolution. In the experiment, the GTPSO algorithm is evaluated on 12 benchmark functions and obtains excellent results, which demonstrates the effectiveness of the GT strategy and the GTPSO algorithm.**

Keywords— Gene targeting (GT), evolutionary computation (EC), particle swarm optimization (PSO), large-scale optimization problem (LSOP)

This work was supported in part by the National Natural Science Foundations of China (NSFC) under Grant 62176094 and Grant U23B2039, and in part by the Tianjin Top Scientist Studio Project under Grant 24JRRCRC00030.

Zhi-Fan Tang, Liu-Yue Luo, and Jing-Hui Zhong are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, P. R. China.

Jian-Yu Li, Jing Xu, Zhi-Hui Zhan, and Jun Zhang is with the College of Artificial Intelligence, Nankai University, Tianjin 300350, P. R. China.

Xin-Xin Xu is with the School of Computer Science and Technology, Ocean University of China, Qingdao 266100, P. R. China.

I. Introduction

In recent years, artificial intelligence (AI) becomes a promising approach to solve the optimization problems faced in the real world. Evolutionary computation (EC) algorithms, including evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms [1-3], are representative AI approaches that have been widely used in a variety of global optimization problems with excellent performance. Particle swarm optimization (PSO) [4] is one of the most widely used SI algorithms, which imitates the bird foraging. Considering for solving an optimization problem in D-dimensional search space, the position of the ith particle at tth generation is denoted as $x_i(t) = (x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,D}(t))$ and the velocity is represented by $v_i(t) = (v_{i,1}(t), v_{i,2}(t), \ldots, v_{i,D}(t))$. The rules for updating the velocity and position of the ith particle can be summarized as follows:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 (pbest_i(t) - x_i(t)) + c_2 r_2 (gbest(t) - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

r_1 and r_2 are two random numbers in [0,1]; ω is inertia weight; c_1 and c_2 are two learning factors that control the influence of the personal historical optimal position of the ith particle (i.e., $pbest_i$) and the global optimal position of the swarm until now (i.e., $gbest$), respectively.

Because of the characteristics of simple and efficiency, PSO has been widely used in various optimization problems. Based on the characteristics of different problems, researchers have proposed different variants of PSO, mainly including different parameter settings [5], topology structure settings [6], multi-swarm strategies [7], parallel and distributed techniques [8][9], and hybrid with other algorithms [10].

The optimization problems with high dimensions and large search space for the solutions is called large-scale optimization problems (LSOPs). In the era of big data, the complexity of optimization problems increases exponentially as the dimensions of optimization problems increasing rapidly. As a result, the performance of PSO and other traditional EC algorithms will rapidly deteriorate. Therefore, improving algorithms' performance in solving optimization problems with the increasing dimensions has become a research hotspot.

In general, there are two categories of methods to improve the EC algorithm efficiency for solving LSOPs: using cooperative coevolution (CC) frameworks or designing new search strategies. CC framework was first proposed in [11] and has been introduced into different EC algorithms [12][13], including CC-genetic algorithms [14], CC-differential evolution (DE) [15] and CC-PSO [16].

CC framework provides an effective approach for solving LSOPs, but it still has significant limitations and drawbacks, such as the high dependency on decomposition strategies, increased computational complexity, and effectiveness only for separable problems. And researchers also consider designing novel non-CC strategies to enhance the population diversity for full exploration and avoid falling into local optima.

In non-CC methods, researchers have designed various new operators [17] and strategies [18]. Moreover, some works adopt distributed paradigms [19] and multiple populations [20] to deal with LSOPs. However, how to balance exploration and exploitation is still a challenge that wait to be addressed.

Recently, Wang *et al*. [21] are inspired by the gene targeting (GT) [22] technology in biotechnology and introduce it into DE to design an effective GTDE for solving LSOPs. The GT is an experimental method in biotechnology that modifies specific genes to improve the characteristics of organisms and has achieved remarkable results. In detail, GT first uses the deoxyribonucleic acid (DNA) homologous recombination technology to construct a homologous targeting vector after obtaining the location of the disease-causing gene. Then, these homologous targeting vectors are inserted into embryonic stem cells to form mutated embryonic stem cells. Similarly, GTDE modifies the best solution in every generation of DE, thus breaking out the poorly-performing dimensions and balancing evolution across all dimensions.

GT improves the convergence of the algorithm, but how to jump out the local optima is also an urgent challenge for solving LSOPs. The social learning strategy, which concluded from social learning PSO(SLPSO), can help avoid falling into the local optimal solution prematurely in LSOPs by making full use of information on different particles in the swarm via social learning. Although SLPSO can better maintain the diversity of the swarm for exploration and prevent premature convergence, it has poor convergence in the later evolutionary stage, resulting in insufficient solution accuracy. Therefore, some enhanced algorithms with social learning have also been proposed. For example, Jian *et al.* [24] propose a region coding scheme combined with social learning to generate more solutions in every generation to speed up the convergence. Molina *et al.* [25] propose iteratively use a variety of local search strategies based on social learning strategy to enhance the convergence.

This paper combines GT and the social learning strategy to develop the GTPSO algorithm for solving LSOPs. The main contributions are as follows:

(1) This paper proposes a dual GT (DGT) method, which contains two novel GT strategies, i.e., GT1 and GT2. Among them, GT1 can limit the search range of the target solution to deviate not too much from the whole population, and GT2 is used to help the target solution better conduct small-scale random searches around the stagnation state.

(2) A new metric named effective evolution ratio (EER) metric is proposed to investigate the effectiveness of GT in improving the success rate of evolution.

(3) An algorithm called GTPSO is proposed, which integrates GT with social learning strategy. GTPSO can not only better maintain the diversity of the population, but also enhance the optimization accuracy of the algorithm, which provides efficient solutions for LSOPs.

The rest of this paper is organized below. The second section provides a brief overview of related work. The proposed GTPSO algorithm is described in detail in Section III. Section IV describes the experimental settings and results analysis. Section V summarizes the paper and discusses future research directions.

II. Background and Related Work

A. SLPSO

Among the numerous optimization algorithms used to solve LSOPs, SLPSO is one of the most widely used algorithms. SLPSO inherits the advantages of easy implementation of PSO and improves it to address the problem of insufficient population diversity in LSOPs. In SLPSO, except for the particle with best fitness currently, each particle undergoes evolutionary learning from the particles in the current population that perform better than itself, a strategy known as social learning mechanism.

In SLPSO, the particles within the current swarm are initially sorted from best to worst based on their fitness values at the start of each generation. When a particle has multiple superior particles, it randomly selects one as a guide for updating. Different dimensions can be guided by different particles. Considering particle *i* chooses to learn from particle *k* at the *d*th dimension. The update process is as follows:

$$v_{i,d}(t+1) = r_1 v_{i,d}(t) + r_2(x_{k,d}(t) - x_{i,d}(t)) + \varepsilon r_3(\overline{x_d}(t) - x_{i,d}(t)) \tag{3}$$

$$x_{i,d}(t+1) = \begin{cases} x_{i,d}(t) + v_{i,d}(t+1), & \mathit{if}\ \ rand(0,1) < P_i \\ x_{i,d}(t), & \mathit{otherwise} \end{cases} \tag{4}$$

where r_1, r_2, and r_3 are three random numbers in [0,1]; $\overline{x_d}$ is the mean position of all the particles in the *d*th dimension; ε is proportional to the problem dimension and defined as:

$$\varepsilon = \beta \cdot \frac{D}{M} \tag{5}$$

where a small value of $\beta = 0.01$ is used to avoid premature convergence in this work. D is the dimension of the optimization problem, and $M = 100$. The P_i is the learning probability for *i*th particle, and it is defined as:

$$P_i = (1 - \frac{i-1}{N})^{\mu \bullet \log(\left\lceil \frac{D}{M} \right\rceil)} \tag{6}$$

The *i*th particle will update its position only if a randomly generated value in [0,1] is smaller than P_i. N is the population number set as $N = M + \lfloor 0.1D \rfloor$ and $\mu = 0.5$.

B. GT

The concept of GT was first put forward in biological [22], which means to modify the information of biological genes, so as to change of biological traits, and then achieve the desired biological performance. Taking CRISPR-Cas9 gene editing as an example, GT constructs a homologous targeting plasmid containing a specific gene after obtaining the specific gene position that needs to be modified. The plasmid is then moved into the embryonic stem cell, forming a mutated embryonic stem cell that modifies disease-causing genes. Due to the targeted modification of the path, the mutated spared stem cells often exhibit better characteristics or traits than the original cells.

Inspired by GT technology, Wang *et al.* [21] propose a GTDE algorithm that applies the ideas of GT to the evolution of DE, helping the population balance the evolution between all dimensions in LSOPs. Although the GT has shown promising results on DE, how to apply the GT to PSO has not attracted enough attentions. Therefore, this paper makes the first attempt to apply GT technology to the PSO and tries to find a better evolutionary direction by modifying the solutions in the population and get a better solution for the optimization problem.

III. Proposed GTPSO Method

A. DGT Strategy

Inspired by GT in biotechnology, we design a DGT strategy to modify the particles and help the algorithm have better performance for solving LSOPs.

Before conducting the DGT, it is necessary to select a suitable particle as the embryonic stem cell, which means DGT modification should only use on this particle. We choose the particle with the best fitness currently as the embryonic stem cell, named target particle. There are three reasons for this choice. First, the target particle has the current global optimal position of the swarm, so it can be regarded as potential with its current performance. Second, the target particle plays an important role in guiding the evolution direction of the population, so using this best particle as the target solution might have a greater positive influence than other particles. Finally, the search space around the target particle may have more probability of finding the better solution as its performance is better than any other particles in current population. Based on the above, the target particle is chosen to carry out the GT strategy. The process of the GT strategy can be concluded as the following three steps.

1) Bottleneck dimension selection

In solving LSOPs, the poorly-performing dimension(s) which limit(s) the particle to find global optimal solution can be regarded as the causative gene(s), considered to be the bottleneck dimension(s). In GT strategy, we use the Monte Carlo method [26] with probability P_j to decide whether the j^{th} dimension is a bottleneck dimension.

The Monte Carlo method is a mathematical method used to predict the possible result of uncertain events, which is suitable for us to balance all the dimensions. Herein, we design a sample probability P_j, which is a random variable drawn from a univariate Gaussian distribution with 0.01 as the mean and 0.01 as the standard deviation for all the dimensions. For each dimension, we generate a uniformly distributed random value in [0,1]. The current dimension will be marked as a bottleneck dimension if the generated random value is smaller than P_j, otherwise, it is regarded as a normal dimension.

Although the Monte Carlo method is simple, it allows every dimension to have the probability of breaking out the bottleneck, which is a good approach to balance all the dimensions.

2) DGT modification

This step aims to modify the homologous targeting vector, and it is conducted by using the DGT modification method in GTPSO. In order to balance the exploitation and exploitation of the population, based on the idea of cooperative game, we propose the GT1 and GT2 that works cooperatively in the DGT.

GT1 is mainly used to maintain the stability of the target particle, making its position deviate not too much from the population. This is considered to make full use of the historical information obtained from the evolution until current process. The updating formula for GT1 is shown as:

$$v_{tar,d}(t+1) = \omega v_{i,d}(t) + c_1 r_1 (pbest_{k_1,d}(t) - pbest_{k_2,d}(t)) + c_2 r_2 (\overline{x_d}(t) - x_d(t)) \quad (7)$$

where v_{tar} is the velocity of the target particle, k_1 and k_2 are two randomly selected particles from the swarm.

GT2 helps the target particle to conduct a local search nearby, thus helping the target particle out of stasis when the population falls into local optima and enhancing the diversity of the population. The updating formula for GT2 is shown as:

$$v_{tar,d}(t+1) = Gaussian(\frac{1}{2}(v_{k_1,d}(t) + v_{k_2,d}(t)), \frac{1}{2}(v_{k_1,d}(t) - v_{k_2,d}(t))) \quad (8)$$

The new value of v_{tar} is a Gaussian distributed random number with a mean as the midpoint of two randomly selected particles k_1 and k_2 at d^{th} dimension and a standard deviation as half of the distance between these two particles at d^{th} dimension.

To preserve the evolutionary information of the population, the mutation strategy should be paid attention. P_m is a probability that is set to help determine which GT mutation is used, and it is set relatively small as 0.01. We generate a uniform distribution random value in [0,1] to help choose the specific GT strategy. If the value is smaller than the P_m, we choose GT1 to mutate. Otherwise, we use GT2.

Due to the two different goals of GT1 and GT2, there is a conflict effect between the two strategies. Besides, both GT1 and GT2 are used to update the target solution through evolution, so there is a cooperative effect between the two strategies in the population. Thus, the balance of stability and diversity in the process of population evolution is realized. By setting GT1 and GT2 respectively, the DGT strategy can help improve the algorithm's ability of solving LSOPs.

3) Target particle update

After the modification, the homologous targeting vector will be inserted into the embryonic stem cell, which means that the velocity and position of the target particle will be updated. To further ensure the effectiveness of the algorithm, we use the greedy strategy to update the target particle. The velocity and position of the target particle is only updated when the target particle updated by the DGT strategy has a better evolution fitness than before.

Based on the above three steps, we implement the whole process of DGT strategy and provide a target particle that have more change to find different better positions. To make this, we set a gene targeting times N_{GT}, which allows the DGT strategy to repeat more times in every generation to provide more potential global optimal solutions.

Algorithm 1 shows the pseudo-code of the DGT strategy. First, we adopt the Monte Carlo method with P_j to select the bottleneck dimensions. Second, the DGT modification is used to help the target particle have a greater probability of finding the global optimal solution and improve the algorithm's ability for solving LSOPs. The convergence of the algorithm is promised by the GT1 strategy and the GT2 strategy lets the population still have robuster exploration ability. Third, we compare the new fitness of the target particle updated by DGT modification with the before one and choose the better one to save. The procedure is repeat for N_{GT} times.

Algorithm 1 Dual Gene Targeting

```
Input: the target particle $x_{tar}$, gene targeting times $N_{GT} = 400$,
$P_m = 0.01$
Output: the updated target particle $x_{tar}$
Begin
1:  For $i = 1$ to $N_{GT}$:
2:      For $d = 1$ to $D$:
3:          $P_j$ = Gaussian(0.01, 0.01);
4:          Randomly select two particles $x_{k1}$ and $x_{k2}$;
5:          If $rand(0, 1) < P_j$
6:          //$d^{th}$ dimension is targeted as bottleneck dimension
7:              If $rand(0, 1) < P_m$
8:                  Generate the $v_{tar,d}$ using the GT-1 strategy in (7);
9:              Else:
10:                 Generate the $v_{tar,d}$ using the GT-2 strategy in (8);
11:             End If
12:             $x'_{tar,d} = x_{tar,d} + v_{tar,d}$;
13:         Else: //$d^{th}$ dimension is not targeted as bottleneck dimension
14:             $x'_{tar,d} = x_{tar,d}$;
15:         End If
16:     End For
17:     If $f(x'_{tar}) \le f(x_{tar})$:
18:         $x_{tar} = x'_{tar}$;
19:     End If
20:     $FEs = FEs + 1$;
21: End For
End
```

B. GTPSO

The DGT strategy mainly focus on the modification of the target particle, but one of the advantages of PSO is that it can use the intelligence of the swarm. As a result, only focusing on the updating of the target particle is not enough, instead, we also need to consider other particles' evolution, so as to keep the diversity of the population and make fully exploration within the large search space.

Based on the above, social learning strategy is a suitable strategy to combine with DGT. In the social learning strategy, for every dimension d of the current particle, we generate a uniform distributed random value in [0,1], and randomly choose a particle which has better fitness than the current one to update the d^{th} dimension of i^{th} particle according to Eq (3). This strategy can fully use the information of the swarm.

Algorithm 2 GTPSO

```
Input: the fitness evolutions $FEs$, the maximum fitness evolutions
$MaxFEs$, the population size $N$, the particle dimension $D$, $M = 100$, $\beta =$
0.01
Output: the final solution $gbest$
Begin
1:  $\varepsilon = \beta D/M$;
2:  Population initialization;
3:  $FEs = FEs + N$;
4:  While $FEs \le MaxFEs$:
5:      Sort all the particles according to fitness from best to worst;
6:      For $i = 1$ to $N$:
7:          If $i == 1$:
8:              Execute DGT strategy using Algorithm 1;
9:          Else:
10:             For $d = 1$ to $D$:
11:                 If $rand(0, 1) \le P_i$:
12:                     $k = rand\_int[1, i - 1]$;
13:                     Update $d^{th}$ dimension of particle $i$; //Eq.(3)-(6)
14:                 End If
15:             End For
16:             Calculate and update the fitness value of particle $i$;
17:             $FEs = FEs + 1$;
18:         End If
19:     End For
20: End While
End
```

The whole GTPSO algorithm can be summarized in the following steps. In every generation, we first sort all the particles from best to worst by their fitness. DGT strategy is used to update the particle with the best fitness currently. And we use the social learning strategy to update other particles. The procedure is repeated until meeting the maximum number of *FEs*. The pseudo-code of the whole algorithm GTPSO is shown in **Algorithm 2**.

IV. Experiment

A. Experimental Setting

The experiments are carried out on the benchmark functions in Table I, including five unimodal functions and seven multimodal functions. All the functions are of D = 1000 dimensions. To obtain more reasonable results, each experiment is run 30 times independently.

The *MaxFEs* is set as 3,000,000 for all competitors and functions, the population size N is set as 100 in GTPSO, while the frequency of the DGT-based modification N_{GT} is set as 400 in every generation. As GTDE [21] is an algorithm which is also inspired by GT for solving LSOPs, and SLPSO [23] uses the social learning strategy and combined it with PSO, we compare the results of GTPSO with GTDE and SLPSO. To ensure a fair comparison, we use the original parameter settings from the algorithms' papers, which are optimized for large-scale optimization benchmarks. To enhance clarity, the best results are highlighted in **boldface**.

B. Comparison Results

Comparison results on the 12 benchmark functions are shown in Table II. The '+', '-', and '=' represent GTPSO is significantly better, worse, and similar to the corresponding

TABLE I
THE 12 BENCHMARK FUNCTIONS

Name	Function	Range
Sphere	$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
Quadric	$f_2(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$
Sch's 2.21	$f_3(x)=\max_i\left(\lvert x_i\rvert, 1\le i\le D\right)$	$[-100, 100]^D$
Step	$f_4(x)=\sum_{i=1}^{D}\left(\lfloor x_i+0.5\rfloor\right)^2$	$[-100, 100]^D$
Noise	$f_5(x)=\sum_{i=1}^{D} ix_i^4 + random[0,1)$	$[-1.28, 1.28]^D$
Rosenbrock	$f_6(x)=\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	$[-10, 10]^D$
Schwefel	$f_7(x)=\sum_{i=1}^{D} -x_i\cdot\sin\left(\sqrt{\lvert x_i\rvert}\right)+418.9829\times D$	$[-500, 500]^D$
Rastrigin	$f_8(x)=\sum_{i=1}^{D}\left[x_i^2-10\cos\left(2\pi x_i\right)+10\right]$	$[-5.12, 5.12]^D$
Ackley	$f_9(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)+20+e$	$[-32, 32]^D$
Griewank	$f_{10}(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$[-600, 600]^D$
Generalized penalized 1	$f_{11}(x)=\frac{\pi}{D}\left\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_D-1)^2\right\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ where $y_i=1+\frac{1}{4}(x_i+1)$, $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, x_i>a\\0, \quad -a\le x_i\le a\\k(-x_i-a)^m, x_i<-a\end{cases}$	$[-50, 50]^D$
Generalized penalized 2	$f_{12}(x)=0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right]+(x_D-1)^2\left[1+\sin^2(2\pi x_D)\right]\right\}+\sum_{i=1}^{D}u(x_i,5,100,4)$	$[-50, 50]^D$

TABLE II
RESULTS ON THE 12 BENCHMARK FUNCTIONS

Function	GTPSO		SLPSO		GTDE	
	mean	std	mean	std	mean	std
f_1	**4.01E+00**	**3.06E+00**	4.17E+02(+)	1.33E+02	2.51E+05(+)	3.26E+04
f_2	**5.02E+05**	**8.94E+04**	1.13E+06(+)	7.55E+04	7.95E+06(+)	1.95E+06
f_3	8.59E+01	8.83E+00	9.22E+01(+)	3.40E+00	**4.80E+01(-)**	**4.27E+00**
f_4	**1.04E+02**	**6.53E+01**	1.04E+03(+)	2.42E+02	2.43E+05(+)	2.65E+04
f_5	**1.79E+00**	**5.36E-01**	1.21E+01(+)	1.45E+00	2.10E+03(+)	4.93E+02
f_6	**3.14E+03**	**1.15E+03**	3.05E+04(+)	2.22E+03	2.19E+06(+)	5.65E+05
f_7	**1.29E+05**	**3.77E+04**	2.37E+05(+)	1.03E+04	3.68E+05(+)	4.31E+03
f_8	**2.70E+02**	**8.73E+01**	1.19E+03(+)	4.92E+01	9.53E+03(+)	2.48E+02
f_9	**4.86E-01**	**4.67E-01**	2.60E+00(+)	1.47E-01	1.43E+01(+)	3.90E-01
f_{10}	**2.16E-01**	**1.48E-01**	2.89E+00(+)	5.50E-01	2.20E+03(+)	3.00E+02
f_{11}	**4.93E+00**	**1.11E+00**	1.03E+01(+)	1.29E+00	3.78E+07(+)	2.23E+07
f_{12}	1.02E+09	1.44E+08	6.49E+10(+)	1.44E+09	**5.44E+08(-)**	**1.82E+08**
Total (+/-/=)			12/0/0		10/2/0	

algorithm according to Wilcoxon's rank-sum test at 0.05 level. It can be seen that the proposed GTPSO performs better than SLPSO and GTDE on the vast majority of functions.

To verify the effectiveness of different parts of GTPSO, we compared GTPSO with GTPSO only using GT1 or only using GT2. The comparison results on the 12 benchmark functions are shown in Table III.

TABLE III
ABLATION EXPERIMENT RESULTS

Function	GTPSO		GTPSO-GT1		GTPSO-GT2	
	mean	std	mean	std	mean	std
f_1	**4.01E+00**	**3.06E+00**	5.22E+01(+)	6.43E+01	6.01E+00(=)	7.66E+00
f_2	**5.02E+05**	**8.94E+04**	1.11E+06(+)	1.14E+05	5.23E+05(+)	1.02E+05
f_3	**8.59E+01**	**8.83E+00**	9.08E+01(=)	4.18E+00	8.89E+01(+)	2.42E+00
f_4	**1.04E+02**	**6.53E+01**	3.24E+02(+)	1.96E+02	1.22E+02(=)	5.69E+01
f_5	1.79E+00	5.36E-01	5.54E+00(+)	1.95E+00	**1.72E+00(=)**	**5.18E-01**
f_6	3.14E+03	1.15E+03	1.73E+04(+)	6.02E+03	**3.04E+03(=)**	**1.44E+03**
f_7	**1.29E+05**	**3.77E+04**	1.54E+05(+)	3.45E+04	1.32E+05(+)	3.98E+04
f_8	**2.70E+02**	**8.73E+01**	9.30E+02(+)	7.94E+01	7.24E+02(=)	1.44E+02
f_9	**4.86E-01**	**4.67E-01**	1.44E+00(+)	4.75E-01	5.61E-01(=)	4.86E-01
f_{10}	**2.16E-01**	**1.48E-01**	1.25E+00(+)	6.68E-01	1.87E-01(=)	1.35E-01
f_{11}	**4.93E+00**	**1.11E+00**	7.23E+00(+)	1.29E+00	4.81E+00(=)	1.16E+00
f_{12}	**1.02E+09**	**1.44E+08**	2.06E+10(+)	1.29E+09	1.08E+09(+)	1.33E+08
Total (+/-/=)			11/0/1		4/0/8	

As shown in Table III, it is evident that the combined action of GT1 and GT2 yields superior performance on the vast majority of functions, which highlights the distinct contributions of GT1 and GT2 in enhancing GTPSO in different aspects. This outcome underscores the importance of employing both GT1 and GT2 simultaneously.

C. Further Analysis of GT

To facilitate a more comprehensive analysis of the results, we propose an indicator called EER per generation to examine the performance of GT and SLPSO across different functions. During the algorithm operation, the EER of every generation is calculated using Eq. (9) and Eq. (10).

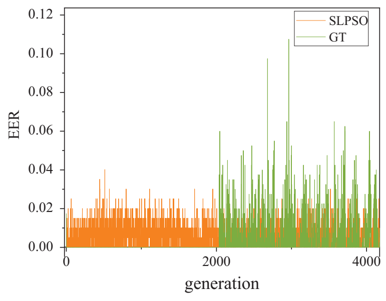
$$EER_{GT}=\frac{N_{validGT}}{N_{totalGT}} \tag{9}$$

$$EER_{SLPSO}=\frac{N_{validSLPSO}}{N_{totalSLPSO}} \tag{10}$$

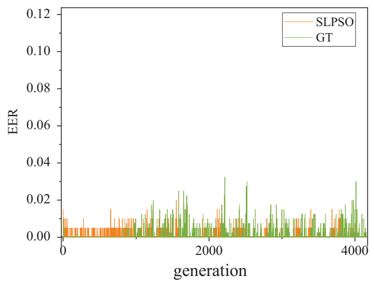
$N_{validGT}$ and $N_{validSLPSO}$ are the number of times that all the particles successfully replace their original *pbest* after the action of GT and SLPSO, while the $N_{validGT}$ and $N_{validSLPSO}$ are the numbers of times that generation undergoes GT or SLPSO evolution.

Fig. 1 shows the EER of GT and SLPSO in every generation on different functions. The EER of SLPSO is generally stable on the vast majority of functions, such as f_1, while the GT gradually becomes effective and surpasses SLPSO after SLPSO has evolved to a certain stage. In f_2 the whole evolution process is guided almost entirely by GT.

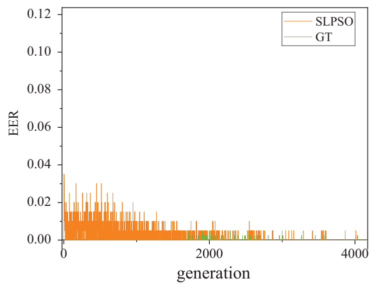
In some functions, i.e., f_4, GT has a smaller EER, probably because f_4 is a downward rounding function. Small disturbances in some dimensions have little effect on improving the target particle when solving this kind of functions. However, from the above Table III, we can also see that the GTPSO significantly outperforms its variants that do not have GT1 or GT2. This may be because that although the GT does not have significant EER on the target particle, it can help the population obtain better solutions. Based on the above, the GT is promising for improving the PSO.



a) EER of GT and SLPSO in f_1



b) EER of GT and SLPSO in f_2



c) EER of GT and SLPSO in f_4

Fig. 1 The EER of GT and SLPSO in the 1st, 2nd, and 4th benchmark problem

V. Conclusion

Inspired by GT, this paper proposes a simple and effective algorithm called GTPSO for solving LSOPs. In the algorithm design, we propose the DGT strategy. It includes two mutation schemes, including the use of excellent solutions and average information of the population to maintain the stability of the population, and the use of Gaussian distribution to generate random numbers to assist particles in jumping out of the local optimal solution and maintain the diversity of the population. GTPSO maintains the simple structure of the PSO algorithm, and shows good performance in a variety of experiments. It has effectively contributed to solving LSOPs.

In future work, we will consider finding more suitable traits to guide the search for bottleneck dimensions and the use of GT, such as starting GT after the population evolution reaches a certain stage, so as to avoid wasting the resource of fitness evaluation in the early evolutionary stage.

References

[1] Z. -H. Zhan, J. -Y. Li, S. Kwong, and J. Zhang, "Learning-aided evolution for optimization," *IEEE Trans. Evol. Comput.*, 2022, DOI: 10.1109/TEVC.2022.3232776.

[2] Z. -H. Zhan, L. Shi, K. C. Tan, and J. Zhang, "A survey on evolutionary computation for complex continuous optimization," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 59-110, Jan. 2022.

[3] W. Deng, H. Liu, J. Xu, H. Zhao, and Y. Song, "An improved quantum-inspired differential evolution algorithm for deep belief network," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 10, pp. 7319-7327, Oct. 2020.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.

[5] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," *Proc. IEEE Int. Conf. Evol. Comput*, 1998, pp. 69-73.

[6] N. Zeng, Z. Wang, W. Liu, H. Zhang, K. Hone, and X. Liu, "A dynamic neighborhood-based switching particle swarm optimization algorithm," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 9290-9301, Sept. 2022.

[7] W. Ye, W. Feng, and S. Fan, "A novel multi-swarm particle swarm optimization with dynamic learning strategy", *Appl. Soft Comput.*, vol. 61, pp. 832-843, Dec. 2017.

[8] J. -Y. Li, Z. -H. Zhan, R. -D. Liu, C. Wang, S. Kwong, and J. Zhang, "Generation-level parallelism for evolutionary computation: a pipeline-based parallel particle swarm optimization," *IEEE Trans. Cybern.*, vol. 51, no. 10, pp. 4848-4859, Oct. 2021.

[9] Z. -J. Wang et al., "Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2715-2729, June. 2020.

[10] Y. Xu and D. Pi, "A reinforcement learning-based communication topology in particle swarm optimization", *Neural Comput. Appl.*, vol. 32, pp. 10007-10032, Jul. 2020.

[11] M. A. Potter and K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," in *Proc. International Conference on Parallel Problem Solving from Nature*, 1994, pp. 249-257.

[12] J. -Y. Li, Z. -H. Zhan, K. C. Tan and J. Zhang, "Dual Differential Grouping: A More General Decomposition Method for Large-Scale Optimization," *IEEE Trans. Cybern.*, vol. 53, no. 6, pp. 3624-3638, Jun. 2023.

[13] X. Zhang et al., "Graph-Based Deep Decomposition for Overlapping Large-Scale Optimization Problems," *IEEE Trans. Syst. Man Cybern.*, vol. 53, no. 4, pp. 2374-2386, Apr. 2023.

[14] Y. Yang and X. -J. Yu, "Cooperative coevolutionary genetic algorithm for digital IIR filter design," *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1311-1318, Jun. 2007.

[15] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 378-393, Jun. 2014.

[16] F. Van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[17] J. -Q. Yang, C. -H. Chen, J. -Y. Li, D. Liu, T. Li, and Z. -H. Zhan. "Compressed-encoding particle swarm optimization with fuzzy learning for large-scale feature selection," *Symmetry*. Vol. 14, pp. 1142. Jun. 2022.

[18] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 191–204, Feb. 2015.

[19] Z. -J. Wang, Z. -H. Zhan, S. Kwong, H. Jin, and J. Zhang, "Adaptive granularity learning distributed particle swarm optimization for large-scale optimization," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1175-1188, Mar. 2021.

[20] Y.-F. Ge et al., "Distributed differential evolution based on adaptive mergence and split for large-scale optimization," *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 2166–2180, Jul. 2018.

[21] Z. -J. Wang, J. -R. Jian, Z. -H. Zhan, Y. Li, S. Kwong, and J. Zhang, "Gene targeting differential evolution: a simple and efficient method for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 4, pp. 964-979, Aug. 2022.

[22] M. R. Capecchi, "Gene targeting in mice: functional analysis of the mammalian genome for the twenty-first century," *Nat. Rev. Genet.*, vol. 6, no. 6, pp. 507-512, Jun. 2005.

[23] R. Cheng and Y. C. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Inf. Sci.*, vol. 291, pp. 43–60, Jan. 2015.

[24] J. -R. Jian, Z. -G. Chen, Z. -H. Zhan, and 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.*, vol. 25, no. 4, pp. 779-793, Aug. 2021.

[25] D. Molina and F. Herrera, "Iterative hybridization of DE with local search for the CEC'2015 special session on large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2015, pp. 1974-1978

[26] J. Jiménez, "Monte Carlo science," *J. Turbul.*, vol. 21, pp. 54