Particle Swarm Optimization with Moving Particles on Scale-Free Networks

Di Wu, Nan Jiang, Wenbo Du, *Member, IEEE*,
Ke Tang, *Senior Member, IEEE*, and Xianbin Cao, *Senior Member, IEEE*

Abstract—PSO is a nature-inspired optimization algorithm widely applied in many fields. In this paper, we present a variant named MP-PSO, in which some particles are allowed to move on a scale-free network and change the interaction pattern during the search course. In contrast to traditional PSOs with fixed interaction sources, MP-PSO shows better flexibility and diversity, where the structure of the particle swarm could change adaptively and balance exploration and exploitation to a large extent. Experiments on benchmark functions show that MP-PSO outperforms other PSO variants on solution quality and success rate, especially for multimodal functions. We further investigate effects of the moving strategy from a microscopic view, finding that the cooperation mechanism of particles located on hub and non-hub nodes plays a crucial role during the optimization process. In particular, owing to the movement of particles on non-hub nodes, the exploration can be guaranteed to some extent even in the final stage, which may be benefit for optimization. We demonstrate the applicability of MP-PSO by using it to solve an important optimization problem, arrival sequencing and scheduling, in the field of air traffic control.

Index Terms—Network structure, particle swarm optimization, scale-free network, swarm structure, moving strategy

1 Introduction

Optimization plays an important role in many real-world problems in the field of science and engineering [1], [2], [3]. The target of a typical optimization problem can be simply abstracted as seeking for the global optimal solution of one objective function. With the development of technology and the increasing difficulty of the problems, traditional optimization methods, including conjugate gradient method, linear programming and Lagrange multiplier method [4], [5], has gradually failed to meet people's demands or available results. To solve these complicated problems, many intelligent optimization methods were proposed in past decades, including particle swarm optimization (PSO) [6], [7].

Inspired by bird flocks and fish schools and first presented by Kennedy and Eberhart, PSO is a widespread optimization algorithm dealing with complex practical matters in many fields [8], [9], [10]. In PSO, a flock of particles fly in the searching space during the optimization process and try to find out the optimal solution cooperatively. Each particle has its certain position in the space, which is evaluated by fitness value. During iterations, the velocity and position of a particle are updated according to the best position discovered before by both itself and other interacted particles. Generally, a PSO process terminates with a predefined iteration or a predefined goal of fitness value.

For its simplicity and high efficiency, PSO has attracted many researchers since its appearance, and many variants of PSO have been proposed to better balance exploration and exploitation [11], [12], [13]. Some earlier researches commonly focus on parameter selection [11], [14], [15], [16]. Shi and Eberhart introduced the inertia weight coefficient to control the scope of the search and managed to balance global and local searching [11], [15]. Clerc and Kennedy proposed a variant with constriction coefficient and proved its similar algebraic effects as the variant with the inertia weight coefficient [16]. Apart from model coefficients, some innovative learning strategies are adopted in some studies [12], [17], [18], [19]. Mendes et al. noticed the excessive dependence on the best-performed neighbor of each particle, and introduced the fully informed particle swarm (FIPS) [19], where all neighbors of a particle contribute to its velocity updating. Liang et al. proposed comprehensive learning particle swarm optimizer (CLPSO) [12], in which particles learn from different neighbors on each dimension separately, to make the swarm avoid getting trapped into local optima to some extent. Considering concepts of quantum mechanics, Sun et al. and Yang et al. introduced quantum behavior into particle swarm optimization algorithm [20], [21], showing the advantages and leading many further research studies [22], [23], [24]. The topology structure is another important aspect [6], [13], [25], [26], [27], [28], [29], [30], since it determines the interaction pattern of the swarm in PSO. As is widely known, the canonical version of PSO, introduced by Kennedy and Eberhart, adopts a fully-connected topology structure (Fig. 1a) [6], where each particle

- *D. Wu, N. Jiang, W. Du, and X. Cao are with the School of Electronic and Information Engineering, Beihang University, Beijing 100191, China. E-mail: shd_leo@163.com, {jn16021104, wenbodu, xbcao}@buaa.edu.cn.*
- *K. Tang is with the Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, Guangdong 518055, China. E-mail: tangk3@sustc.edu.cn.*

Manuscript received 14 Dec. 2017; revised 26 May 2018; accepted 28 June 2018. Date of publication 16 July 2018; date of current version 5 Mar. 2020. (Corresponding authors: Wenbo Du and Xianbin Cao.)
Recommended for acceptance by A. Arenas.
Digital Object Identifier no. 10.1109/TNSE.2018.2854884

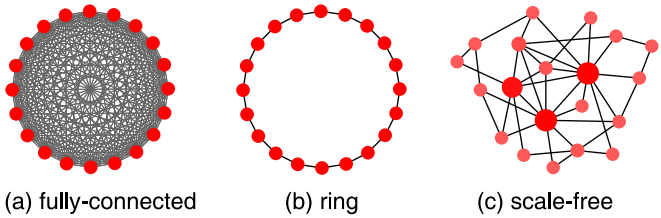


Fig. 1. Representative topology structures adopted in PSO.

could directly interact with any other particles in the whole swarm. Although canonical PSO converges fast, it tends to be easily trapped in local optima because of the lack of time for a comprehensive searching in the solution space. To address such a defect, Kennedy later proposed PSO with ring (circle) structure (Fig. 1b) [13], where each particle can only interact with two neighbor particles rather than with all particles directly. Admittedly, particles within the ring structure could explore the searching space more carefully to avoid local optima and find out more promising regions, but the convergence speed is hard to be guaranteed. Some other regular structures such as wheel, star and von Neumann were demonstrated to be able to make good results when solving some specific problems [27]. Moreover, with the development of network science, especially the proposal of small-world and scale-free model in 1998 and 1999 respectively [31], [32], people got more inspirations on the population structure of PSO. Gong and Zhang proposed small-world PSO which permits particles to occasionally interact with non-neighbor particles by small-world randomization [30].

An important property of real networks is scale-free-ness, i.e., the node degree follows a power law [32]. The degree heterogeneity of scale-free networks (Fig. 1c) allows the hub nodes (whose degrees are larger than a certain threshold) to guide the system for exploitation and the non-hub nodes (whose degrees are less than the threshold) to explore the solution space. Employing scale-free networks, Liu et al. introduced PSO with scale-free interactions (SFPSO) [29], where scale-free structure makes the algorithm achieve a good tradeoff between exploration and exploitation during the optimization process. Also adopting scale-free network, selectively-informed PSO (SIPSO) proposed by Gao et al. [28] showed that when structural hub particles and non-hub particles employ different strategies, where hub particles get information from all neighbors while non-hub particles only learn from the best-performed neighbor, the system achieves a better cooperation and a promising overall performance.

However, particles in most of the variants mentioned above are with fixed interaction patterns during the optimization process. Particles can only learn from the neighbors on the static topology structure, which limits the information interaction in the swarm. On the basis of scale-free structure whose advantages have been widely proved, a PSO variant is expected to make particles flexibly change interaction sources to better handle different situations during the optimization process. Consequently, we propose a particle swarm optimization with moving particles on scale-free networks (MP-PSO), where particles are consistently restricted in a static scale-free base network but are partly free to adaptively change the swarm network with the moving strategy during iterations. In MP-PSO, the adaptive moving strategy helps the particle swarm cope with varies of complicated and uncertain situations, and particles are more likely to get useful information during each iteration. Numerical experiments show that MP-PSO has a superior performance, especially on multimodal problems.

The rest of this paper is organized as follows. Section 2 introduces MP-PSO in detail and gives some premised information. Section 3 compares the results of six PSOs, demonstrates some discussions mainly concerning the property of the algorithm macroscopically and microscopically. Section 4 applies MP-PSO to solve the arrival sequencing and scheduling problem to demonstrate its applicability. Section 5 makes a conclusion.

2 Material and Methods

2.1 Canonical PSO

First, we give a brief introduction of the canonical PSO with constriction coefficient [6], [16]. Suppose the dimension of the searching space is D and the size of particle population is N_0, and the ith particle has its position $\boldsymbol{x}_i = (x_i^1, x_i^2, \ldots, x_i^D)$ and its velocity $\boldsymbol{v}_i = (v_i^1, v_i^2, \ldots, v_i^D)$. During iterations, the velocity and position of each particle are updated on the basis of the following methods:

$$v_i^d \leftarrow \chi * (v_i^d + c_1 * r_1 * (p_i^d - x_i^d) + c_2 * r_2 * (p_g^d - x_i^d)) \quad (1)$$

$$x_i^d \leftarrow x_i^d + v_i^d, \quad (2)$$

where χ controls the speed of convergence, c_1 and c_2 are acceleration coefficients, and r_1 and r_2 are independent random numbers in range $[0, 1]$. In the methods above, $\boldsymbol{p}_i = [p_i^1, \ldots, p_i^d, \ldots, p_i^D]$ refers to the best position particle i has reached ever before and $\boldsymbol{p}_g = [p_g^1, \ldots, p_g^d, \ldots, p_g^D]$ refers to the best position all particles discovered in a fully-connected structure.

During the whole optimization process, each particle has a dynamic fitness value corresponding with its current position. $\boldsymbol{p}_i$ and $\boldsymbol{p}_g$ of each particle are also updated during each iteration if a better fitness value is available. This process iterates continuously until a certain number of time or a certain goal of fitness value is met.

2.2 MP-PSO

Base Network. In the initial stage, a base scale-free network with N_B nodes is established adopting BA scale-free model (Fig. 2a), which is proposed by Barabási and Albert in 1999 [32]. With m_0 initial interconnected nodes, the network adds nodes one after another separately. When new nodes are added, each connects to m ($m < m_0$) different existing nodes and the probability of building connection with node i, P_i, is related to the degree of node i (k_i) and of other existing nodes as follows:

$$P_i = k_i / \sum_j k_j, \quad (3)$$

where j traverses all existing nodes. In other words, new nodes are more likely to connect with high-degree hubs, and the number of edges among all nodes approximately follow power law property in the end.

Swarm Network. In MP-PSO, N_S ($N_S < N_B$) particles are randomly assigned in the base network (Fig. 2b). Particles, as well as edges between them, make up the swarm network (Fig. 2c). Obviously, the structure of swarm network

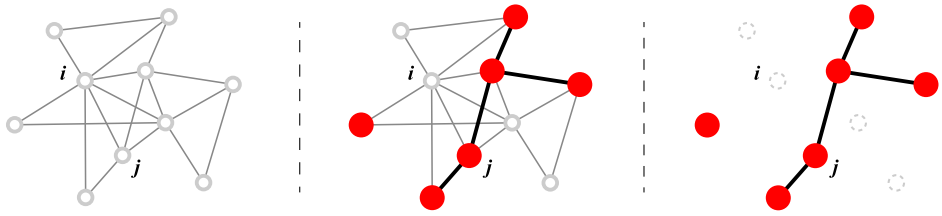


Fig. 2. Networks in MP-PSO. Light edges represent edges in the base network, and bold edges represent edges in the swarm network. White nodes represent vacant nodes, and red nodes represent nodes occupied by particles. In (c) dashed nodes are not involved in the swarm network.

determines the information spread pattern during the optimization process. The velocity update method of each particle in MP-PSO is as follows:

$$v_i^d \leftarrow \chi * (v_i^d + c_1 * r_1 * (p_i^d - x_i^d) + c_2 * r_2 * (p_{si}^d - x_i^d)), \quad (4)$$

where $\boldsymbol{p}_{si}$ refers to the best position discovered by the ith particle and its neighbors in the swarm network, and the position update method is the same as Equation (2).

Since there are two networks in our MP-PSO, namely base network and swarm network, here we define two concepts concerning degree, k_B and k_S. k_B of a node refers to its degree in the base network, while k_S is the degree in the swarm network. For example, in Fig. 2, k_B of node i and j is 7 and 4 respectively (Fig. 2a), and their k_S are 0 and 2 (Fig. 2c). Note that k_B of each node is constant during the optimization process since the base network is static, while k_S can be varied during iterations with the moving strategy, which will be discussed detailedly in the next part.

Moving Strategy. The key point of MP-PSO is the moving strategy, according to which the swarm network adaptively changes. First, we define a time counter t_i for each particle i, which denotes the number of generations that its $\boldsymbol{p}_i$ ceases improving. If t_i reaches a preset value T_{gap}, particle i is *qualified*. A qualified particle can move to a vacant neighbor node if there is at least one vacant position in its base network neighbors, carrying its position and velocity information as well as $\boldsymbol{p}_i$ and $\boldsymbol{p}_{si}$. Consequently, the structure of swarm network is changed, as is shown in Fig. 3. Note that the destination of a particle is chosen randomly if more than one vacant neighbor nodes are available, and if a particle cannot move because of the lack of a vacant neighbor node, it remains on the current topology position and t_i will keep increasing without a reset, until it moves successfully to a vacant neighbor node in further iterations or its $\boldsymbol{p}_i$ is updated. The movement of particles can lead to two effects: converging and diverging. Figs. 3a and 3b show the converging effect, where the movement can make the swarm network denser and the information can spread more fluently, corresponding to the facility of exploitation. Conversely, Figs. 3c and 3d show the diverging effect, where the movement can make the swarm network sparser. Obviously, the diverging effect is beneficial for exploration. During the optimization process, the interaction pattern will change due to the particles' movements and it may make a good tradeoff between exploration and exploitation.

Parameter Setting. According to common practices, we adopt $\chi = 0.7298$ and $c_1 = c_2 = 2.05$ to update the velocity and position of each particle during the optimization process [8], [19], [29], [34], [35]. To build up the scale-free base network in MP-PSO, we adopt parameters $m_0 = 5$ and $m = 2$. Obviously, the filling ratio (the ratio of the swarm network size N_S to the base network size N_B) may affect the optimization performance of MP-PSO. If the filling ratio is too high, the moving range of particles is so limited that the swarm network is similar with the static topology structure in SFPSO; if the filling ratio is too low, the swarm network is so sparse that particles can hardly interact with others, leading to a random search process. After extensive experiments, we select $N_B = 80$ as the base network size and $N_S = 50$ as the swarm network size respectively. All experiment results are averaged by 50 times after 5000 iterations. T_{gap} represents the movement threshold, and higher values of T_{gap} make particles hard to move. After massive experiments, we set $T_{gap} = 4$. All these values are as default in the rest of this paper.

2.3 Benchmark Functions

To test the performance of MP-PSO, 16 benchmark functions are used in this paper (Table 1), including 5 unimodal functions (f_1-f_5), 6 multimodal functions (f_6-f_{11}) and 5 rotated multimodal functions (f_{12}-f_{16}) [12], [20], [28], [33].

Specifically, f_1 is an easy problem for most algorithms to solve, while f_2 is relative difficult and is sometimes treated as a multimodal function when with a high D [12], [36]. f_3 is

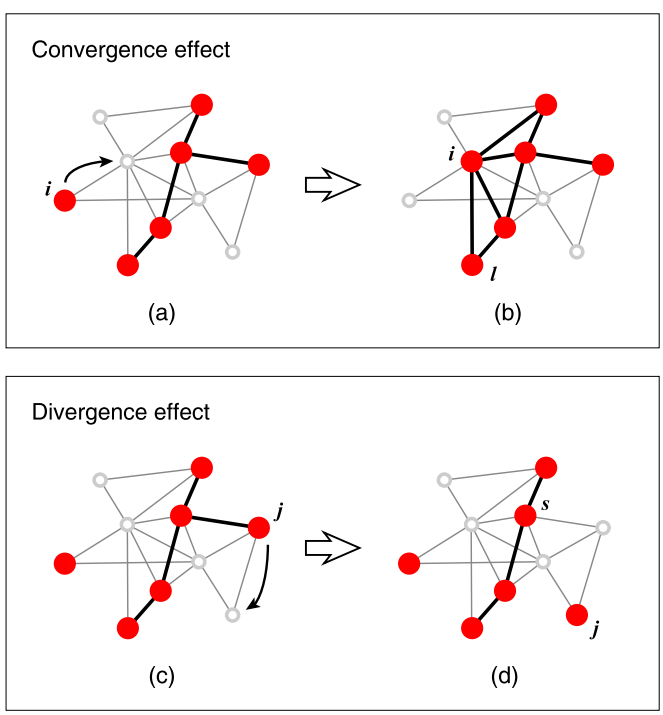


Fig. 3. Moving process and two effects of the moving strategy in MP-PSO. (a)-(b) show the converging effect where particle i moves to a vacant neighbor node. (c)-(d) show the diverging effect as j's movement.

TABLE 1
Optimization Benchmark Functions

Name	Formula	Searching Space	Dimension	Goal Value
Unimodal functions				
Sphere[33]	$f_1(\boldsymbol{x}) = \sum_{i=1}^{D} {x_i}^2$	$[-100, 100]^D$	30	0.01
Rosenbrock[33]	$f_2(\boldsymbol{x}) = \sum_{i=1}^{D-1} 100({x_{i+1}} - {x_i}^2)^2 + (x_i - 1)^2$	$[-30, 30]^D$	30	100
Schwefel's P2.22[28]	$f_3(\boldsymbol{x}) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	30	0.01
De Jong[20]	$f_4(\boldsymbol{x}) = \sum_{i=1}^{D} i x_i^4$	$[-1.28, 1.28]^D$	30	0.05
Quartic[28]	$f_5(\boldsymbol{x}) = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^D$	30	0.05
Multimodal functions				
Shaffer[20]	$f_6(\boldsymbol{x}) = 0.5 + \frac{(\sin\sqrt{{x_1}^2 + {x_2}^2})^2 - 0.5}{1 + 0.001({x_1}^2 + {x_2}^2)^2}$	$[-100, 100]^D$	2	0.00001
Rastrigin[33]	$f_7(\boldsymbol{x}) = \sum_{i=1}^{D} {x_i}^2 - 10\cos(2\pi x_i) + 10$	$[-5.12, 5.12]^D$	30	100
Griewank[33]	$f_8(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{D} {x_i}^2 - \prod_{i=1}^{D} \cos\frac{x_i}{\sqrt{i}} + 1$	$[-600, 600]^D$	30	0.05
Ackley[33]	$f_9(\boldsymbol{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} {x_i}^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D} \cos 2\pi x_i) + 20 + e$	$[-32, 32]^D$	30	0.01
Schwefel[33]	$f_{10}(\boldsymbol{x}) = 418.9829 \times D + \sum_{i=1}^{D} x_i \sin(\lvert x_i \rvert^{\frac{1}{2}})$	$[-500, 500]^D$	30	2000
Weierstrass[33]	$f_{11}(\boldsymbol{x}) = \sum_{i=1}^{D} (\sum_{k=0}^{kmax} [a^k \cos(2\pi b^k (x_i + 0.5))]) - D \times \sum_{k=0}^{kmax} [a^k \cos(\pi b^k)]$ $a = 0.5, b = 3, kmax = 20$	$[-0.5, 0.5]^D$	30	0.01
Rotated Multimodal functions				
Rotated Rastrigin[33]	$f_{12}(\boldsymbol{x}) = \sum_{i=1}^{D} {y_i}^2 - 10\cos(2\pi y_i) + 10$	$[-5.12, 5.12]^D$	30	100
Rotated Griewank[33]	$f_{13}(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{D} {y_i}^2 - \prod_{i=1}^{D} \cos\frac{y_i}{\sqrt{i}} + 1$	$[-600, 600]^D$	30	0.05
Rotated Ackley[33]	$f_{14}(\boldsymbol{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} {y_i}^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D} \cos 2\pi y_i) + 20 + e$	$[-32, 32]^D$	30	0.01
Rotated Schwefel[33]	$f_{15}(\boldsymbol{x}) = 418.9829 \times D + \sum_{i=1}^{D} z_i \quad z_i = \begin{cases} y_i \sin(\lvert y_i \rvert^{\frac{1}{2}}) & \lvert y_i \rvert < 500 \\ 0.001(\lvert y_i \rvert - 500)^2 & \lvert y_i \rvert \geq 500 \end{cases}$ $\boldsymbol{y} = \boldsymbol{M} \times (\boldsymbol{x} - 420.96) + 420.96$	$[-500, 500]^D$	30	2000
Rotated Weierstrass[33]	$f_{16}(\boldsymbol{x}) = \sum_{i=1}^{D} (\sum_{k=0}^{kmax} [a^k \cos(2\pi b^k (y_i + 0.5))]) - D \times \sum_{k=0}^{kmax} [a^k \cos(\pi b^k)]$ $a = 0.5, b = 3, kmax = 20$	$[-0.5, 0.5]^D$	30	1

similar with f_1 but with non-differentiable points, and f_5 adds a random interference on the basis of f_4. Most multimodal problems are much more difficult. f_9 has some minor local optima. f_6, f_7 and f_{10} involve a large number of deep local optima and f_8 makes linkages among variables, while only a set of points in f_{11} are differentiable [12]. For rotated multimodal functions, algorithms cannot solve the problems by dealing with each dimension separately. To establish a rotated function in this paper, an orthogonal matrix $\boldsymbol{M}$ is generated using Salomon's method [37], and the position of each particle $\boldsymbol{x}$ is left multiplied by $\boldsymbol{M}$ to get the rotated position $\boldsymbol{y}$ (except for f_{15}), which is used to calculate the fitness value. The rotated functions have similar shapes of the original functions, but the problems are more difficult for algorithms to solve [12].

3 Results and Discussions

The performance of MP-PSO is compared with three PSOs with static interaction patterns: fully-connected (FPSO), ring (RPSO) and scale-free (SFPSO), and two PSOs with different learning strategies: selectively-informed (SIPSO) and quantum behavior (QPSO). Note that two important parameters are set as reasonable values $k_c = 5$ and $g = 0.96$ in SIPSO and QPSO respectively [20], [28], and the number of particles in all PSOs is set to 50 in the following experiments. We first examine the result of solution quality R (Table 2). As all selected benchmark functions are minimization problems, smaller results reflect better performance. Here the best sets are marked in bold. FPSO performs better on 3 of 5 unimodal functions due to its high speed on information spreading. MP-PSO can obtain best results on 6 of 11 multimodal and rotated multimodal problems (including 3 multimodal functions and 3 rotated multimodal functions), remarkably outperforming other algorithms. It can be attributed to the flexible moving strategy, which may make the algorithm jump out of local optima via the cooperation of converging and diverging effects. Fig. 4 depicts the number of times that each algorithm performs top-Z among all the 16 benchmark functions [38]. It can be observed that the curve of MP-PSO is never beneath that of any other PSO algorithms, and it first reaches 16 at $Z = 5$, which shows its outstanding performance in this aspect.

Table 3 shows the results of the convergence speed Q (iteration times required to meet the goal value), where '–' indicates that the algorithm fails to reach the goal value even once. MP-PSO only fails on f_{15}, while FPSO/RPSO/SFPSO/SIPSO/QPSO cannot reach the goal value on 4/3/3/2/2 functions respectively. As for the success rate (Table 4), MP-PSO ranks first on 10 functions and second on 5 functions, while other PSOs perform poorly on some functions more or less. Obviously, MP-PSO performs overall the best concerning results of the three criterions.

TABLE 2
Optimization Performance Results on Criterion R

Function	FPSO[6]	RPSO[13]	SFPSO[29]	SIPSO[28]	QPSO[20]	MP-PSO
			Unimodal functions			
f_1	$3.35E-99$	$9.52E-45$	$3.52E-60$	$\mathbf{3.11E-109}$	$2.96E-90$	$6.09E-62$
	$\pm 1.83E-98$	$\pm 1.34E-44$	$\pm 1.83E-59$	$\pm 1.37E-108$	$\pm 1.47E-89$	$\pm 9.69E-62$
f_2	$\mathbf{1.17E+01}$	$2.10E+01$	$2.33E+01$	$1.67E+01$	$6.52E+01$	$2.88E+01$
	$\pm 1.81E+01$	$\pm 2.20E+01$	$\pm 3.90E+01$	$\pm 4.70E+00$	$\pm 1.48E+02$	$\pm 2.47E+01$
f_3	$7.88E-27$	$8.60E-27$	$4.54E-25$	$7.80E-60$	$\mathbf{1.56E-67}$	$1.01E-36$
	$\pm 4.32E-26$	$\pm 5.65E-27$	$\pm 2.32E-24$	$\pm 4.53E-59$	$\pm 8.28E-67$	$\pm 1.16E-36$
f_4	$\mathbf{2.12E-159}$	$1.98E-67$	$8.89E-93$	$1.46E-111$	$2.63E-100$	$1.31E-94$
	$\pm 1.46E-158$	$\pm 5.94E-67$	$\pm 6.28E-92$	$\pm 7.29E-111$	$\pm 1.42E-99$	$\pm 7.92E-97$
f_5	$\mathbf{2.89E-03}$	$9.92E-03$	$7.56E-03$	$3.86E-03$	$3.62E-03$	$3.39E-03$
	$\pm 1.10E-03$	$\pm 3.24E-03$	$\pm 2.40E-03$	$\pm 1.41E-03$	$\pm 1.74E-03$	$\pm 9.02E-04$
			Multimodal functions			
f_6	$1.41E-03$	$1.94E-04$	$1.94E-04$	$5.23E-10$	$1.54E-03$	$\mathbf{0} \pm 0$
	$\pm 3.40E-03$	$\pm 1.37E-03$	$\pm 1.37E-03$	$\pm 2.61E-09$	$\pm 3.63E-03$	
f_7	$7.16E+01$	$6.59E+01$	$5.44E+01$	$\mathbf{1.88E+01}$	$2.20E+01$	$4.28E+01$
	$\pm 1.48E+01$	$\pm 1.30E+01$	$\pm 1.70E+01$	$\pm 1.47E+01$	$\pm 6.71E+00$	$\pm 9.96E+00$
f_8	$2.73E-02$	$\mathbf{2.47E-04}$	$1.73E-02$	$7.12E-03$	$1.56E-02$	$6.32E-03$
	$\pm 3.64E-02$	$\pm 1.41E-03$	$\pm 2.79E-02$	$\pm 4.25E-03$	$\pm 1.90E-02$	$\pm 8.85E-03$
f_9	$1.23E+00$	$7.99E-15$	$8.86E-01$	$9.15E-15$	$1.06E-14$	$\mathbf{7.76E-15}$
	$\pm 7.81E-01$	± 0	$\pm 8.56E-01$	$\pm 5.24E-15$	$\pm 3.45E-15$	$\pm 9.01E-16$
f_{10}	$3.53E+03$	$3.44E+03$	$3.67E+03$	$4.02E+03$	$\mathbf{2.27E+03}$	$3.24E+03$
	$\pm 6.68E+02$	$\pm 5.55E+02$	$\pm 6.55E+02$	$\pm 6.45E+02$	$\pm 5.06E+02$	$\pm 7.98E+02$
f_{11}	$4.03E+00$	$6.63E+00$	$4.07E+00$	$1.63E+00$	$1.35E+00$	$\mathbf{1.13E+00}$
	$\pm 1.88E+00$	$\pm 2.87E+00$	$\pm 1.70E+00$	$\pm 1.04E+00$	$\pm 1.86E+00$	$\pm 1.03E+00$
			Rotated multimodal functions			
f_{12}	$1.80E+02$	$1.76E+02$	$1.67E+02$	$1.54E+02$	$1.99E+02$	$\mathbf{1.42E+02}$
	$\pm 2.82E+01$	$\pm 1.40E+01$	$\pm 2.36E+01$	$\pm 2.73E+01$	$\pm 1.90E+01$	$\pm 1.98E+01$
f_{13}	$9.37E-02$	$\mathbf{0} \pm 0$	$7.03E-02$	$4.20E-03$	$1.67E-01$	$1.05E-03$
	$\pm 1.68E-01$		$\pm 1.48E-01$	$\pm 1.62E-02$	$\pm 2.63E-01$	$\pm 5.52E-03$
f_{14}	$9.91E-02$	$\mathbf{7.99E-15}$	$9.89E-14$	$8.82E-15$	$1.14E-14$	$\mathbf{7.99E-15}$
	$\pm 3.16E-01$	± 0	$\pm 3.20E-15$	$\pm 2.22E-15$	$\pm 3.59E-15$	$\pm 1.47E-15$
f_{15}	$5.86E+03$	$5.23E+03$	$5.24E+03$	$5.81E+03$	$5.76E+03$	$\mathbf{4.93E+03}$
	$\pm 5.91E+02$	$\pm 6.21E+02$	$\pm 8.31E+02$	$\pm 6.67E+02$	$\pm 8.09E+02$	$\pm 8.09E+02$
f_{16}	$1.92E+01$	$1.06E+01$	$1.24E+01$	$\mathbf{5.30E-02}$	$2.93E+01$	$3.11E+00$
	$\pm 8.49E+00$	$\pm 1.12E+01$	$\pm 9.73E+00$	$1.28E-01$	$\pm 1.01E+01$	$\pm 5.05E+00$

To investigate the underlying mechanism of the excellent performance of MP-PSO, we analyze the optimization process from a network science view. Here a representative multimodal function Rastrigin (f_7) is chosen in all following experiments.

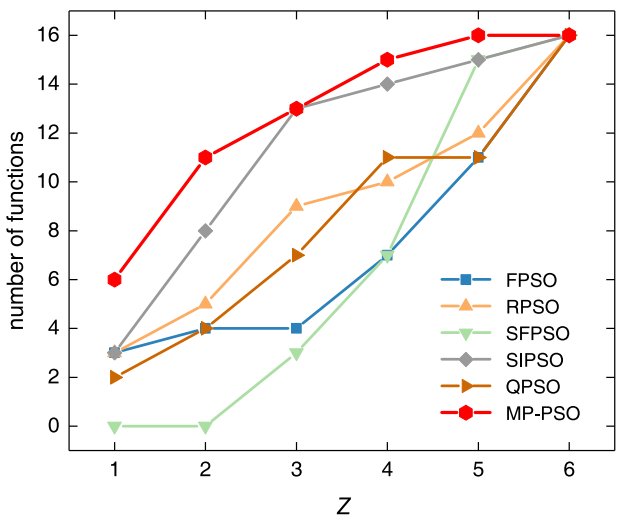


Fig. 4. The top-Z curves of the algorithms among the 16 benchmark functions, where $Z = 1, 2, \ldots, 6$.

As can be learned from Fig. 3, the base network in MP-PSO is static, while the swarm network changes with the particle movements. In general, the denser the swarm network is, the

TABLE 3
Optimization Performance Results on Criterion Q

Func.	FPSO	RPSO	SFPSO	SIPSO	QPSO	MP-PSO
			Unimodal functions			
f_1	317	698	518	**283**	363	508
f_2	515	806	940	**258**	586	733
f_3	478	763	669	325	**300**	562
f_4	103	233	176	**63**	526	169
f_5	361	1097	736	**108**	708	531
			Multimodal functions			
f_6	533	584	694	351	601	**293**
f_7	**163**	437	365	179	164	300
f_8	**304**	682	490	325	342	481
f_9	409	896	704	**327**	462	599
f_{10}	–	–	528	–	464	**455**
f_{11}	–	–	–	847	**631**	1084
			Rotated multimodal functions			
f_{12}	–	–	–	**2576**	–	3937
f_{13}	1817	773	863	745	906	602
f_{14}	755	939	726	**389**	562	592
f_{15}	–	**1644**	–	–	–	–
f_{16}	1647	2808	1752	1760	1191	**938**

TABLE 4
Success Rate after 5000 Iterations

Func.	FPSO	RPSO	SFPSO	SIPSO	QPSO	MP-PSO
Unimodal functions						
f_1	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
f_2	0.98	**1.00**	0.96	**1.00**	**1.00**	**1.00**
f_3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
f_4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
f_5	**1.00**	**1.00**	**1.00**	0.98	**1.00**	**1.00**
Multimodal functions						
f_6	0.86	0.98	0.94	**1.00**	0.74	0.94
f_7	0.98	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
f_8	0.92	**1.00**	0.90	**1.00**	0.98	**1.00**
f_9	0.30	**1.00**	0.38	**1.00**	**1.00**	0.96
f_{10}	0.00	0.00	0.02	0.00	**0.28**	0.02
f_{11}	0.00	0.00	0.00	0.06	0.12	**0.30**
Rotated multimodal functions						
f_{12}	0.00	0.00	0.00	**0.36**	0.00	0.18
f_{13}	**1.00**	**1.00**	0.84	0.96	0.68	0.98
f_{14}	0.86	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
f_{15}	0.00	**0.06**	0.00	0.00	0.00	0.00
f_{16}	0.02	0.50	0.24	0.52	0.04	**0.88**

faster the information spreads. Fig. 5a shows the variation of $\bar{k}_S$ during the optimization process. It is found that $\bar{k}_S$ continuously increases, indicating that the swarm network concentrates with iterations. From Fig. 5b, we can observe that there are 5 or 6 connected components of the swarm network in the initial state but only about 2 components after the evolution, showing the same concentrating trend as in Fig. 5a. Although the swarm network gets denser during the optimization process, however, the curve of N_m in Fig. 5c indicates that it does not concentrate blindly, which may induce premature. Instead, there are even more moving particles in the last period (please see Fig. 5c). The increasing number of moving particles presents some possibilities of exploration.

Fig. 6 shows four snapshots of the base network with particles at $t = 0$, $t = 5$, $t = 2000$ and $t = 5000$ respectively. Obviously, with 4 connected components, a random beginning (Fig. 6a), the swarm network converges rapidly to one single component once the particle moving is first permitted at $t = 5$ (Fig. 6b). During the later period of optimization process, the giant component still exists, but more particles occasionally separate from the giant component and explore by themselves, making the swarm explore the searching space and may find more promising regions (Figs. 6c and 6d).

As can be found in Fig. 6, particles located on hub nodes are always within the giant component, while particles separating from the giant component are mainly located on non-hub nodes. It seems that the structural properties of nodes have significant impacts on the behaviors of particles located on them during iterations. Hence, we further investigate the effects of k_B, the node degree in the base network, on particle moving behaviors.

Fig. 7a shows the frequency that the node is occupied by a particle. Obviously, nodes with larger k_B obtain higher f_{OCC}, indicating that hub nodes are more possibly to be occupied. In the scale-free network, the degree distribution is heterogeneous. Once a hub node with large k_B becomes vacant, numerous particles on its neighbor nodes will occupy it. In Fig. 7b, the curve of move desire f_{QUA} decreases with the increase of k_B, showing that particles on hub nodes are more inert to move. For particles located on hub nodes, they can collect information from more sources, and thus can get more high-quality information to update their $\boldsymbol{p}_i$. Conversely, the

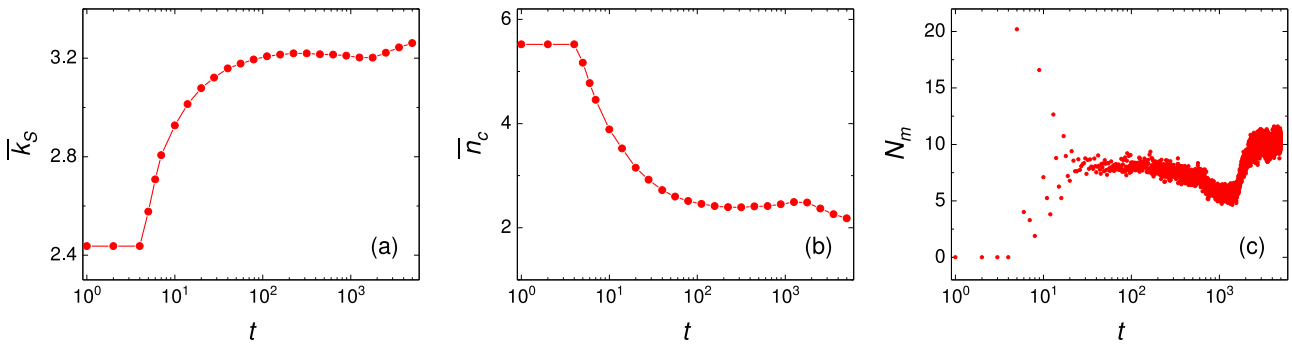


Fig. 5. (a) Variation of $\bar{k}_S$, which denotes the average k_S for all nodes in the swarm network. (b) Variation of $\bar{n}_c$, which denotes the average number of connected components in the swarm network. (c) Variation of N_m, which denotes the number of moving particles during each iteration.

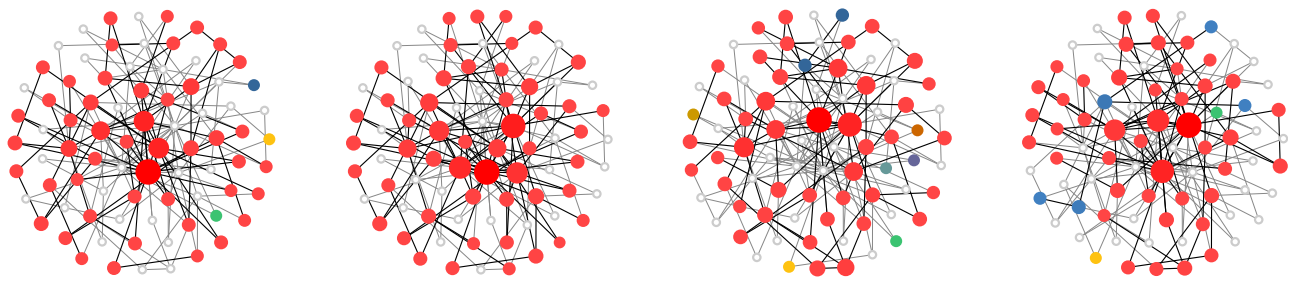


Fig. 6. Snapshots during an optimization process. The size of the colored nodes is positively correlated to k_S, and the different colors represent distinct components in the swarm network (red components denote the giant components).

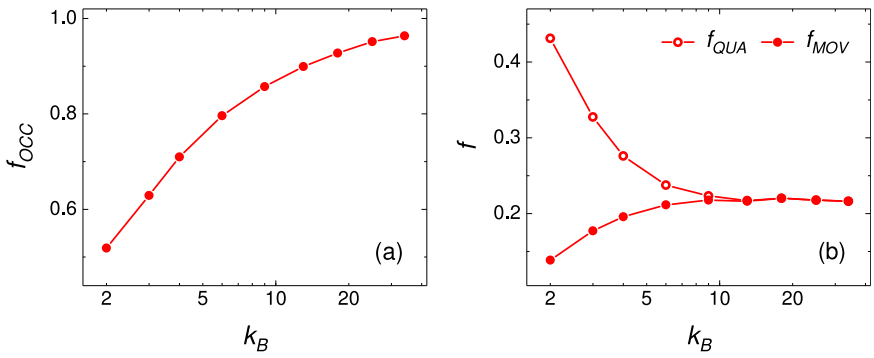


Fig. 7. (a) $f_{OCC} = t_{OCC}/5000$ denotes the frequency that the node is occupied by a particle in 5000 iterations, where t_{OCC} is the occupied times of the node during the whole process. (b) $f_{QUA} = t_{QUA}/t_{OCC}$ denotes the frequency that the particle located on the node is qualified, where t_{QUA} is the qualified times that a particle i meets $t_i \geq T_{gap}$, and $f_{MOV} = t_{MOV}/t_{OCC}$ denotes the frequency that the particle moves away from the node, where t_{MOV} is the iteration times that the particle located on the node moves away.

move frequency f_{MOV} shows a just opposite trend (Fig. 7b). We can conclude that moving behaviors of particles on hub and non-hub nodes are quite different. For a hub particle (such as particle s in Fig. 3d), the move desire and the move frequency are almost the same, indicating that it can move once it wants to move. This is guaranteed by its large neighbor sets in the base network. However, particles on non-hub nodes present strong move desires but much lower move frequency. For a non-hub particle (such as particle l in Fig. 3b), its $\boldsymbol{p}_i$ can hardly update due to the poor information sources. Although the move desire can be very strong, it cannot move if there does not exist a vacant node in its neighbor set. Furthermore, even if a non-hub particle is allowed to move (such as particle j in Fig. 3c), since the neighboring hub nodes are always occupied, it will possibly leave the giant component to explore by itself (Fig. 3d). As a result, the particles on hub nodes take on the main responsibility of the optimization, while particles on non-hub nodes guarantee an appropriate exploration ability of the swarm during the evolution. The cooperation between particles on hub and non-hub nodes achieves an overall better performance.

4 A Case Study on the Arrival Sequencing and Scheduling (ASS) Problem

To demonstrate the applicability of MP-PSO, we use it to solve the arrival sequencing and scheduling (ASS) problem, which has attracted considerable discussion in the field of Air Traffic Control (ATC) during past decades [39], [40], [41].

4.1 Problem Description

The ASS problem can be simplified as generating efficient landing sequences and landing times of a certain number of arriving flights, to minimize both the total delay of all arriving flights (T_{delay}) and the total time of the entire process (T_{length}) [41]. For the sake of safety, the minimum permissible time between two successive landing flights is restricted by the landing time interval (LTI), depending on the feature of the two aircrafts [41], [42].

The goal of ASS problem is usually to minimize T_{delay}, and T_{length} is also sometimes adopted as the index for optimization [41]. In our experiment, we adopt T_{delay} and the objective function can be simply defined as follows:

$$\min T_{delay} = \sum_{d=1}^{D} t_A(X_d) - t_P(X_d), \tag{5}$$

s.t.

$$t_A(X_d) = \max(t_P(X_d), t_A(X_{d-1}) + S(X_{d-1}, X_d)), \tag{6}$$

where X_d is the dth landing flight in the optimized sequence, $S(X_{d-1}, X_d)$ indicates the LTI between flight X_{d-1} and X_d, while $t_A(X_d)$ and $t_P(X_d)$ are the actual and predicted landing time of the corresponding flight.

To perform ASS, the simplest method is first-come-first-serve (FCFS), which shares the same order based on the predicted landing time. Although the schedule it establishes is relative fair and safe, some useful information is ignored [41], and thus optimization methods are required to solve the problem better.

4.2 Data Sets

In our following experiments, the total number of flights is set to $N_F = 50$. For simplicity, we choose four types of common commercial aircraft (A, B, C and D) and the LTI between them are shown in Table 5 [41]. For a landing sequence data, $\boldsymbol{t_P}$ is the predicted landing time, whose elements obey uniform distribution in range of $(0, 5000)$, and *type* is the type of aircraft, and the ratio of the four types of aircraft is 5:3:1:1.

4.3 Simulation Results

The performance of MP-PSO is compared with other PSO algorithms by optimizing the original landing sequence, as well as with FCFS method. To simply illustrate how FCFS method and optimization algorithms work, Table 6 gives the predicted landing sequence and optimization results in a single test. Note that T_{delay} of each method and algorithm is calculated and

TABLE 5
Minimum LTI between Four Types of Common Commercial Aircraft

$S(i,j)$(s)		type of the following aircraft j			
		A	B	C	D
type of the leading aircraft i	A	96	200	181	228
	B	72	80	70	110
	C	72	100	70	130
	D	72	80	70	90

TABLE 6
Predicted Landing Sequence and Results of FCFS Method and Algorithms in a Single Test

Predicted			FCFS		FPSO		RPSO		SFPSO		SIPSO		QPSO		MP-PSO	
No.	Type	t_P (s)	No.	t_A (s)	No.	t_A (s)	No.	t_A (s)	No.	t_A (s)	No.	t_A (s)	No.	t_A (s)	No.	t_A (s)
1	B	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
2	A	258	2	258	2	258	2	258	2	258	2	258	2	258	2	258
3	B	474	3	474	3	474	3	474	3	474	3	474	3	474	3	474
4	A	480	4	546	4	546	4	546	4	546	4	546	4	546	4	546
5	A	1039	5	1039	5	1039	5	1039	5	1039	5	1039	5	1039	5	1039
6	A	1097	6	1135	6	1135	6	1135	6	1135	6	1135	6	1135	6	1135
7	A	1169	7	1231	8	1231	7	1231	7	1231	8	1231	7	1231	8	1231
8	A	1171	8	1327	7	1327	8	1327	8	1327	7	1327	8	1327	7	1327
9	B	1174	9	1527	9	1527	9	1527	9	1527	9	1527	9	1527	9	1527
10	D	1315	10	1637	12	1607	12	1607	12	1607	12	1607	10	1637	12	1607
11	A	1467	11	1709	13	1687	13	1687	13	1687	13	1717	16	1727	13	1687
12	B	1547	12	1909	10	1797	17	1797	16	1797	16	1807	17	1817	10	1797
13	B	1616	13	1989	16	1887	10	1887	17	1887	17	1887	13	1897	17	1887
14	A	1627	14	2061	17	1977	16	1977	18	1967	10	1997	18	1977	16	1977
15	A	1630	15	2157	20	2057	18	2057	20	2048	18	2077	20	2057	20	2057
16	D	1691	16	2385	18	2137	20	2137	15	2120	20	2157	12	2137	18	2137
17	D	1734	17	2475	19	2209	14	2209	19	2216	19	2229	14	2209	15	2209
18	B	1783	18	2555	15	2305	11	2305	11	2312	14	2325	11	2305	14	2305
19	A	1923	19	2627	11	2401	15	2401	14	2408	11	2421	19	2401	21	2401
20	B	2048	20	2827	21	2497	21	2497	21	2504	21	2517	15	2497	19	2497
21	A	2283	21	2899	14	2593	22	2697	22	2704	15	2613	21	2593	11	2593
22	B	2616	22	3099	22	2793	24	2777	24	2784	23	2709	23	2689	23	2689
23	A	2643	23	3171	24	2873	26	2847	29	2864	25	2805	28	2785	25	2785
24	B	2694	24	3371	29	2953	25	2919	27	2974	28	2986	25	2881	28	2881
25	A	2705	25	3443	27	3063	28	3015	10	3064	26	3058	26	3062	26	3062
26	C	2717	26	3624	26	3133	19	3111	26	3134	24	3258	24	3162	24	3162
27	D	2733	27	3754	23	3205	23	3207	30	3206	22	3338	29	3242	22	3242
28	A	2763	28	3826	30	3301	30	3303	28	3302	29	3418	22	3322	29	3322
29	B	2785	29	4026	28	3397	29	3503	25	3398	27	3528	31	3403	31	3403
30	A	3010	30	4098	25	3493	31	3583	32	3494	30	3600	27	3513	27	3513
31	B	3403	31	4298	32	3589	27	3693	23	3590	32	3696	32	3585	32	3585
32	A	3465	32	4370	33	3702	32	3765	31	3790	33	3792	30	3681	30	3681
33	A	3702	33	4466	34	3798	33	3861	35	3860	34	3888	36	3777	36	3777
34	A	3738	34	4562	36	3894	38	3957	37	3930	31	4088	33	3873	33	3873
35	C	3743	35	4743	38	3990	37	4138	38	4002	39	4168	34	3969	38	3969
36	A	3777	36	4815	37	4171	35	4208	33	4098	41	4238	38	4065	34	4065
37	C	3785	37	4996	41	4241	41	4278	34	4194	35	4308	40	4161	40	4161
38	A	3848	38	5068	35	4311	39	4378	36	4290	37	4378	43	4257	43	4257
39	B	4052	39	5268	31	4411	42	4488	44	4386	38	4450	37	4438	35	4438
40	A	4121	40	5340	39	4491	44	4560	40	4482	40	4546	41	4508	37	4508
41	C	4140	41	5521	42	4601	40	4656	45	4578	43	4642	45	4580	41	4578
42	D	4160	42	5651	44	4673	45	4752	41	4759	36	4738	46	4761	46	4648
43	A	4162	43	5723	43	4769	43	4848	46	4829	45	4834	39	4861	44	4720
44	A	4286	44	5819	40	4865	50	4959	39	4929	44	4930	47	4941	45	4816
45	A	4465	45	5915	45	4961	48	5159	47	5009	46	5111	49	5021	47	5016
46	C	4648	46	6096	48	5161	49	5239	49	5089	47	5211	48	5101	49	5096
47	B	4788	47	6196	49	5241	47	5319	42	5199	48	5291	42	5211	48	5176
48	B	4903	48	6276	47	5321	46	5389	48	5279	49	5371	35	5281	39	5256
49	B	4942	49	6356	46	5391	36	5461	50	5351	42	5481	44	5353	42	5366
50	A	4959	50	6428	50	5463	34	5557	43	5447	50	5553	50	5449	50	5438
	T_{delay}			39807		16667		18446		16826		16399		16444		**15895**

TABLE 7
Simulation Results on ASS Problem

Method	Mean (s)	SD (s)	Best (s)	Worst (s)
FCFS	39807	–	–	–
FPSO	18783.18	1301.25	16667	22579
RPSO	22883.56	3064.30	18446	32597
SFPSO	21607.14	3667.50	16826	34900
SIPSO	18602.32	2030.79	16399	25040
QPSO	17704.52	1108.74	16444	20435
MP-PSO	**17631.82**	**957.20**	**15895**	**19379**

presented at the bottom of Table 6. Obviously, T_{delay} of MP-PSO is less than that of FCFS method and other algorithms.

Table 7 shows the statistic results for 50 individual runs, indicating T_{delay} of FCFS method and all optimization algorithms. In Table 7 the best sets are marked in bold, where one can see that MP-PSO performs better than other PSO algorithms. The overall performances demonstrate the applicability of MP-PSO when solving real-world problems.

5 Conclusion

In this paper we propose MP-PSO, a variant version of scale-free networked PSO with a moving strategy. In MP-PSO,

particles are permitted to adaptively move in a static scale-free base network, to change their interacted neighbors for more useful information. We select 16 widely-used benchmark functions to test the performance of MP-PSO and it shows better results than other five PSOs with fixed interaction sources or different interaction modes to a large extent, especially on multimodal functions. Our analysis presents that the swarm in MP-PSO generally gets denser during the optimization process, which benefits for information spreading, while particles are significantly more active in the final stage adopting the moving strategy, indicating some possibilities of exploration. We further study the moving behavior of particles, demonstrating that particles on hub and non-hub nodes play different but cooperative roles. In particular, the particles on hub nodes take the main responsibility of the optimization, while particles on non-hub nodes guarantee an appropriate exploration ability of the swarm. The cooperation between particles on hub and non-hub nodes helps to achieve an overall better performance. The applicability of MP-PSO to real-world optimization problems is demonstrated by solving the arrival sequencing and scheduling problem at the end of the paper.

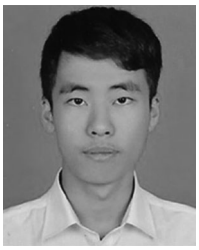
Acknowledgments

This paper is supported by National Key Research and Development Program of China (Grant No. 2016YFB1200100), National Natural Science Foundation of China (Grant Nos. 61425014, 61521091, 91538204, 61671031,61722102).

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Ann Arbor, MI, USA: Univ. Michigan Press, 1975.
[2] F. Glover and M. Laguna, *Tabu Search*. New York, NY, USA: Springer, 1999.
[3] P. J. Van Laarhoven and E. H. Aarts, *Simulated Annealing*. Netherlands: Springer, 1987.
[4] W. Forst and D. Hoffmann, *Optimization - Theory and Practice*. New York, NY, USA: Springer, 2010.
[5] R. K. Arora, *Optimization: Algorithms and Applications*. London, UK: Chapman & Hall, 2015.
[6] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Nov. 1995, pp. 1942–1948.
[7] Y. Mei, X. D. Li, and X. Yao, "Cooperative co-evolution with route distance grouping for large-scale capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 435–449, Jun. 2014.
[8] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, 2007.
[9] R. Eberhart and Y. H. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. Congr. Evol. Comput.*, May 2001, pp. 81–86.
[10] A. R. Jordehi, "Particle swarm optimisation for dynamic optimisation problems: A review," *Neural Comput. Appl.*, vol. 25, no. 7, pp. 1507–1516, 2014.
[11] Y. H. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evol. Comput. Proc.*, May 1998, pp. 69–73.
[12] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.
[13] J. Kennedy, "Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance," in *Proc. Congr. Evol. Comput.*, Jul. 1999, pp. 1931–1938.
[14] J. Liu, Y. Mei, and X. D. Li, "An analysis of the inertia weight parameter for binary particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 666–681, Oct. 2016.
[15] Y. H. Shi and R. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. Congr. Evol. Comput.*, May 2001, pp. 101–106.
[16] M. Clerc and J. Kennedy, "The particle swarmfig1-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.
[17] B. Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 387–402, Jun. 2013.
[18] C. H. Li, S. X. Yang, and T. N. Trung, "A self-learning particle swarm optimizer for global optimization problems," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 42, no. 3, pp. 627–646, Jun. 2012.
[19] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, Jun. 2004.
[20] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proc. Congr. Evol. Comput.*, Jun. 2004, pp. 325–331.
[21] S. Yang, M. Wang, and L. Jiao, "A quantum particle swarm optimization," in *Proc. Congr. Evol. Comput.*, Jun. 2004, pp. 320–324.
[22] J. Sun, W. Xu, and B. Feng, "A global search strategy of quantum-behaved particle swarm optimization," in *Proc. Conf. Cybern. Intell. Syst.*, Dec. 2004, pp. 111–116.
[23] J. Sun, W. Xu, and B. Feng, "Adaptive parameter control for quantum-behaved particle swarm optimization on individual level," in *Proc. Int. Conf. Syst. Man Cybern.*, Oct. 2005, pp. 3049–3054.
[24] L. dos Santos Coelho, "A quantum particle swarm optimizer with chaotic mutation operator," *Chaos, Solitons Fractals*, vol. 37, no. 5, pp. 1409–1418, 2008.
[25] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 35, no. 6, pp. 1272–1282, Dec. 2005.
[26] X. D. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, Feb. 2010.
[27] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. Congr. Evol. Comput.*, May 2002, pp. 1671–1676.
[28] Y. Gao, W. B. Du, and G. Yan, "Selectively-informed particle swarm optimization," *Sci. Reports*, vol. 5, 2015, Art. no. 9295.
[29] C. Liu, W. B. Du, and W. X. Wang, "Particle swarm optimization with scale-free interactions," *PLoS ONE*, vol. 9, no. 5, 2014, Art. no. e97822.
[30] Y. J. Gong and J. Zhang, "Small-world particle swarm optimization with topology adaptation," in *Proc. 15th Genetic Evol. Comput. Conf.*, Jul. 2013, pp. 25–31.
[31] D. J. Watts and S. H. Strogatz, "Collective dynamics of small-world networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[32] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Sci.*, vol. 286, no. 5439, pp. 509–512, 1999.
[33] J. J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Proc. IEEE Swarm Intell. Symp.*, Jun. 2005, pp. 68–75.
[34] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2007, pp. 120–127.
[35] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, 2003.
[36] W. B. Du, W. Ying, G. Yan, Y. B. Zhu, and X. B. Cao, "Heterogeneous strategy particle swarm optimization," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 64, no. 4, pp. 467–471, Apr. 2017.
[37] R. Salomon, "Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms," *BioSystems*, vol. 39, no. 3, pp. 263–278, 1996.
[38] K. Tang, P. Yang, and X. Yao, "Negatively correlated search," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 542–550, Mar. 2016.
[39] G. Andreatta and G. Romanin-Jacur, "Aircraft flow management under congestion," *Transportation Sci.*, vol. 21, no. 4, pp. 249–253, 1987.
[40] H. Balakrishnan and B. G. Chandran, "Algorithms for scheduling runway operations under constrained position shifting," *Operations Res.*, vol. 58, no. 6, pp. 1650–1665, 2010.
[41] X. B. Hu and W. H. Chen, "Genetic algorithm based on receding horizon control for arrival sequencing and scheduling," *Eng. Appl. Artif. Intell.*, vol. 18, no. 5, pp. 633–642, 2005.
[42] L. Bianco, P. DellOlmo, and S. Giordani, "Scheduling models and algorithms for TMA traffic management," in *Modelling and Simulation Air Traffic Management*. Berlin, Germany: Springer, 1997, pp. 139–167.

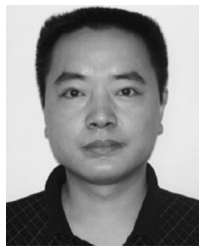
Di Wu is currently working toward the BE degree in the School of Electronic and Information Engineering, Beihang University, Beijing, China. His research interests include intelligent optimization and network science.



Nan Jiang is currently working toward the BE degree in the School of Electronic and Information Engineering, Beihang University, Beijing, China. His research interests include intelligent optimization and network science.

Wenbo Du received the BS and PhD degrees from the School of Computer Science and Technology, University of Science and Technology of China, in 2005 and 2010, respectively. He is an associate professor with the School of Electronic and Information Engineering, Beihang University, China. His research interests include network science and bio-inspired computation. He is a member of the IEEE.

Ke Tang (S'05-M'07-SM'13) received the BEng degree from Huazhong University of Science and Technology, Wuhan, China, in 2002, and the PhD degree from Nanyang Technological University, Singapore, in 2007. Since 2007, he has been with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China, where he is currently a professor. His research interests include evolutionary computation, machine learning, and their real-world applications. He is a senior member of the IEEE.



Xianbin Cao (M'08-SM'10) received the BEng and MEng degrees in computer applications and information science from Anhui University, Hefei, China, in 1990 and 1993, respectively, and the PhD degree in information science from the University of Science and Technology of China, Hefei, in 1996. He is currently a professor with the School of Electronic and Information Engineering, Beihang University, Beijing, China. His current research interests include intelligent transportation systems, airspace transportation management, and intelligent computation. He is a senior member of the IEEE.