An Analysis of Binary Particle Swarm Optimizers for Task Assigning Problem in Wireless Sensor Networks

Xu-long Zeng, Wei-Neng Chen*, *Member IEEE*, and Jun Zhang, *Senior Member IEEE*
School of Advanced Computing, Sun Yat-sen University
Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education
*Corresponding Author, email: chenwn3@mail.sysu.edu.cn

Abstract—**The tightly restricted resource in wireless sensors networks (WSN) makes it challenging to schedule the task assignment for better performance. Binary particle swarm optimizers (BPSO) along with its modified version (MBPSO) have shown promising performance to this problem, but premature convergence remains a key issue. To improve performance of BPSO for task assigning in WSN, this paper first develops various extended BPSOs by using different topologies and the comprehensive learning strategy. An integrated comparison among these candidate approaches and the MBPSO is carried out. In addition, the choice of transfer function highly affects the global optimizing ability of BPSO. Thus the significance of transfer functions with different shapes adopted in BPSO is discussed. Through sufficient simulations and analysis, it is found that the BPSO with the comprehensive learning strategy and a V-shaped transfer function is very promising, especially toward large-scale problems.**

Keywords-particle swarm optimization, wireless sensor networks, combinatorial optimization

I. Introduction

The computing capability and energy storage of sensors are usually limited in wireless sensor networks. Meanwhile, there are some computationally intense processing tasks that require more computing ability and energy that exceed the capability of a sole sensor node [1]. So we have to decompose tasks into smaller sub-tasks that can be executed in a single sensor node. Sub-tasks are then assigned to sensor nodes and processed in sensor nodes. Such in-network processing schemes are considerably efficient [2].

The isomerism of node capability leads to the discrepancy of the overall WSN performance of different task assigning scheme. Sequentially, finding the task assigning scheme with the optimal performance is a problem worth deep researching. Our aim is to obtain longer WSN lifetime, smaller overall energy consumption and better energy consumption balance. The three parts composes the fitness function that defines a good task assigning scheme.

Recent studies have been focusing on the utilization of stochastic meta-heuristic approaches like particle swarm optimization (PSO) to the problem. PSO is a meta-heuristic algorithm with excellent characters [3]. PSO can be easily implemented into real-world applications. PSO has manifested its dominance in various WSN problems [4]. As the problem is discrete, researches tried to apply the binary version of PSO (BPSO) to it and gained ideal performance [5]. Jun Yang proposed a modified binary particle swarm optimization (MBPSO) with a transfer function differs from that of the original version of BPSO [6]. Toward the task assigning issue, MBPSO is capable of outperforming BPSO and genetic algorithm (GA) in finding the global optimum.

BPSO approaches are advantageous and promising toward the task assigning problem in WSN. Nevertheless, there are still some aspects that are badly in need of further study. The drawback of premature remains to be fixed. As the transfer function in BPSO plays an essential role and it strongly affect the algorithm performance, the issue of transfer function choice and how it influences the algorithm performance needs to be researched and analyzed in depth.

Aimed at the above-mentioned issues, various kinds of modified BPSO approaches toward the task assigning problem in WSN are proposed in this work. Three binary versions of local PSO based on different topology structures are developed. Moreover, a BPSO with the comprehensive learning strategy (BCLPSO) is proposed. They are all applied to the problem to further improve the particle diversity of BPSO, and so as to avoid local optima. We also propose using V-shaped functions instead of S-shaped ones in BPSO as transfer functions.

This work was supported in part by the NSFC Nos. 61379061, 6141101191, in part by Natural Science Foundation of Guangdong No. S2013040014949, in part by Specialized Research Fund for the Doctoral Programs No. 20130171120016, an in part by the Science and Technology Innovation Outstanding Young Scholars of Guangdong, No. 2014TQ01X550.

978-1-4799-8697-2/15 $31.00 © 2015 IEEE
DOI 10.1109/SMC.2015.344

Comparisons between the two forms of transfer functions are carried out, and based on this, analysis of the effectiveness of transfer functions is given.

In our experiments, it is manifested that the proposed BCLPSO algorithm with the V-shaped function is very promising. The global searching ability of BCLPSO is greatly improved by the comprehensive learning strategy and the V-shaped function. This approach is especially advantageous to large-scale problems. As wireless sensor networks are comparably large today, the superiority of BCLPSO is apparent.

The remainder of this paper is structured as follows. In the next section, the problem definition is given. Basic concepts of BPSO algorithm, the neighborhood topology of local BPSO and the comprehensive learning strategy are introduced in Section 3. Section 4 introduces S-shaped transfer functions and V-Shaped functions in details and a comparison is made. Simulations are carried out in Section 5, where analysis and review are also given. And finally, the conclusion is drawn in Section 6.

II. PROBLEM DEFINITION

The problem definition follows [6], and here the outline is given.

A. Modeling Tasks

A directed acyclic graph (DAG) notation is used to present series of tasks that compose a WSN application. For each of the task, workload of computing and communicating are used as featuring components. In a given DAG that represent the series of tasks, the node vector $W = \{W_i : i = 1, 2, ...m\}$ represents all the tasks to be processed. Directed edges in DAG denote the executing sequence. That is, the task attached to the front of an arrow shall not be processed until the one attached to the rear of the same arrow is done.

B. Modeling WSN Networks

A weighted undirected graph notation is introduced here to represent wireless sensor networks. Each sensor node is denoted by a graphic node with three components, i.e., the computing speed v, the rate of work e and the initial energy storage E_{init}. The weight of each edge in the graph denotes the distance between the two related sensor nodes.

C. Specification of Fitness Function

The fitness function consists of three components: the overall task processing time, the overall energy consumption and the energy distribution. Definitions of these three factors in fitness function are given respectively.

1) Overall task processing time

Tasks are decomposed into sub-tasks and then assigned to a bunch of sensor nodes. We use W_{ij} to represent the sub-task of task W_i that assigned to the sensor node *j*. T_{ij} is the time consumed by W_{ij}, which consists of two parts: T_{ij}^p the computational time consumption and T_{ij}^m the communication time consumption. Formula for T_{ij}^p is given as follows:

$$T_{ij}^p = W_{ij}^p / v_j \quad (1)$$

Here v_j is the computing speed of node *j*. As for T_{ij}^c, the formula is shown in (2).

$$T_{ij}^c = dl_{ij} / bw + T_{ij}^{qu} \quad (2)$$

In (2), dl_{ij} is the data amount of sub-task W_{ij} while bw stands for the bandwidth. T_{ij}^{qu} is the queuing time caused by the limited bandwidth. Combining formula (1) and (2), the time consumption of sub-task W_{ij} is acquired by (3):

$$T_{ij} = T_{ij}^c + T_{ij}^p \quad (3)$$

Sequentially the overall time consumption for a WSN application is the accumulation of all the T_{ij}, which is illustrated in (4) .

$$T = \sum T_{ij} \quad (4)$$

2) Overall Energy Consumption

Energy consumption is composed by two parts i.e., the computational energy consumption E^p and the communication energy consumption E^c. E_{ij}^p is determined by the product of a sensor node's rate of work e_j and the correlative computing time consumption T_{ij}^p. A popular communication energy model is introduced in [7] to modify the energy consumption of inter-node data exchange. For the communication energy consumption E_{ij}^c, there are two standalone parts: the data sending part and data receiving part, denoted by (5) and (6) respectively.

$$E_{ij}^s = (e_{elec} + \varepsilon_{amp} \times dist^2) \times dl_{ij} \quad (5)$$

$$E_{ij}^r = e_{elec} \times dl_{ij} \quad (6)$$

E_{elec} and ε_{amp} are parameters that specified by the radio characteristics of sensor nodes. And here we have:

$$E_{ij}^c = E_{ij}^s + E_{ij}^r \quad (7)$$

Therefore, we conclude the energy consumption of task i on node j as :

$$E_{ij} = E_{ij}^p + E_{ij}^c \quad (8)$$

And finally the energy consumption is given as:

$$E = \sum E_{ij} \quad (9)$$

3) Energy Distribution

The energy distribution shows the balance of energy consumption among sensor nodes, thus a standard deviation measurement is introduced to evaluate the proportionality. The formal expression of energy distribution is given by formula (10) where $Eva(E_j)$ is the average energy consumption of all sensor nodes.

$$SD = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(E_j - Eva(E_j))^2} \quad (10)$$

D. Constraints in the Problem of Task Assigning in WSN

As a real-world application problem, our issue is constrained by two factors: the number of chosen nodes should not be smaller than the minimum number of needed nodes to fulfill the task, and the chosen nodes should be connected with each other to ensure the data exchange among nodes. The definition of these two constraints follows what's given in [6]. We need to confirm that the chosen nodes in the WSN are in the same connected graph, either by straight links or by multi-hop links. This can be done by a depth-first-search operation or by the *Theorem 1* in [6].

E. Formulation of Fitness Function

Evaluation of fitness function for a certain solution X is given as follows:

$$Fitness(X) = w_1 \times T + w_2 \times E + w_3 \times SD \tag{11}$$

In formula (11), *T*, *E* and *SD* are normalized by being divided by the values under the circumstance of the maximum task workload and the lowest processing speed. And w_1, w_2, w_3 are the weight coefficients, where $\sum w_i = 1, (i = 1,2,3)$, $w_i \in (0,1)$. Therefore, our aim is to find the optimal X with the minimal $Fitness(X)$.

III. BINARY PARTICLE SWARM OPTIMIZERS

A. The Local Version BPSO

In PSO algorithms, particles cooperate with each other as a collective and fly to the optimal position together [3]. The idea of local PSO is to force particles to learn from the local best position in its neighbors rather than from the global best one. This helps to slow down the communication among particles and keeps their diversity. With this modification, the updating of particle velocities is altered into formula (12), where $lbest_i^j$ refers to the dimension j of the best position in the neighborhood of particle i. The neighborhood is like Fig.1(a).

$$V_i^j \leftarrow \omega V_i^j + c_1 r_1^j (pbest_i^j - X_i^j) + c_1 r_2^j (lbest_i^j - X_i^j) \tag{12}$$

In ring topology BPSO, particles are organized as a ring, and the neighborhood for each particle includes three individuals, i.e., the one before it, the one behind it and the particle itself. Fig. 1(b) illustrated an example neighborhood of RBPSO.

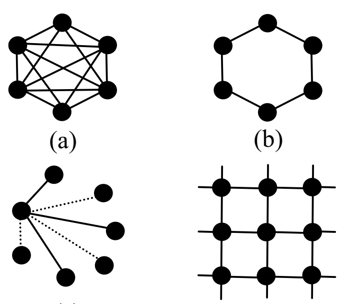


Figure 1. Different topology of local BPSO

The random neighborhood topology is self-evident. Fig. 1(c) gives a simple model of a random neighborhood of 4 particles including the particle itself. Particles connected by full lines are currently in a neighborhood, and particles connected by dotted lines may be a neighborhood in the next iteration. The number of particles in a neighborhood is constant, while we randomly decide which particles to conform a neighborhood in iterations during the whole run [8].

For von Neumann structure neighborhood topology, there are four other individuals in one particle's neighborhood excluding itself, and particles are tied together and form square grids as shown in Fig. 1(d). For each of the particle, its neighbors are the one above it, the one below it, the one on the left side and the one on the right side.

B. Mutation in Local BPSO

Mutation operation is a technique applied in genetic algorithms, which can be utilized in PSO to maintain particle diversity. It is an effective way to avoid local optimum. Here a mutation operator is introduced to the local versions of BPSO. The mutating formula is given by (13).

$$x_i^j = \begin{cases} 1 - x_i^j & \text{if } rand < muRate \\ x_i^j & otherwise \end{cases} \tag{13}$$

In (13), *rand* is a randomly generated number within [0,1] and *muRate* is the predefined mutation rate that describe the probability of executing the mutation.

```
Procedure MBCLPSO:
randomly initialize X and V of all particles;
set parameters;
set f_i, gbest, pbest, Iteration=0;
for Iteration=0..MaxIteration:
    renew w according to (13);
    for each particle:
        if flag_i ≥ Lg:
            renewFi();
            flag_i = 0;
        end if;
        for each dimension d:
            renew V_i^d according to (16);
            renew X_i^d according to (17) and (18);
        end for;
        if fitness(X_i)>fitness(pbest_i):
            pbest_i =X_i;
            flag_i = 0;
        if fitness(X_i)>fitness(gbest):
            gbest =X_i;
        else:
            flag_i = flag_i + 1;
        end if;
    end for;
end for;
output;
End procedure;
```

Figure 2. Pseudo-code of MBCLPSO

C. The Comprehensive Learning Strategy

In traditional PSO, the particle with the optimal fitness may not be optimal in some dimensions. A large amount of useful information is deserted when particles learn from the global optimum $gbest$. Liang developed a version of PSO called the comprehensive learning PSO where particles learn from all the $pbests$ instead of $gbest$ or solely their own $pbest$. The diversity of particles is ensured and premature is prevented by this modification. CLPSO outperforms a lot other variants of PSO in many multi-modal problems. The velocity updating for CLPSO follows formula (14).

$$V_i^j = \omega \cdot V_i^j + c \cdot rand_i^j \cdot (pbest_{fi(j)}^j - X_i^j) \quad (14)$$

In (14), $f_i = [f_i(1), f_i(2), ... f_i(D)]$ denotes which particle's $pbest$ the ith particle should learning from. When one dimension of a particle is being updated, if a randomly generated value is smaller than the pre-configured Pc_i, we stochastically choose two particles excluding itself, and run a tournament selection between them to choose the one with better fitness value. Afterwards, the particle renew this dimension of its velocity by learning from the identical dimension of the chosen particle velocity. Or if the random value is greater than Pc_i, the particle simply learn from its own $pbest$ when renewing this dimension. If all the dimensions of a particle are learned from itself, a random dimension is chosen to learning from another random particle. The contents of f_i are not renewed at each iteration. Instead, the learning gap Lg determines that f_i will be updated every other Lg iterations that without getting a better fitness value[9]. By using the comprehensive learning strategy, we developed the algorithm of BCLPSO, which is concluded into the flowchart in Fig. 2. In Fig. 2, $flag_i$ records the number of iterations without renewing $pbest_i$ for particle i, and Lg is the refreshing gap defining the rate of updating f_i.

IV. TRANSFER FUNCTION

As the guidance of position renewing, the shape of the transfer function strongly effect the algorithm performance. In the original version of BPSO, a sigmoid function as in (15) is employed and the position updating is (16).

$$S(V_i^j) = \frac{1}{1 + \exp(-v_i^j)} \quad (15)$$

$$X_i^{j+1} = \begin{cases} 0 & \text{if } rand \geq S(v_i^j) \\ 1 & otherwise \end{cases} \quad (16)$$

$S(V_i^j)$ denotes the probability of converting a bit of the position to 1 and $rand$ is a randomly generated value. Both of these values are within the range [0,1].

As the algorithm goes on, velocities converge to 0, making $S(V_{i,j}^{d+1})$ converge to 0.5, which means the converting becomes completely random. This feature tends to slow down the convergence of the whole swarm. Furthermore, the already obtained good positions are easily forgotten when using formula (16) as the position updating formula. This tends to weaken the global optimizing ability of BPSO.

There are some other candidate S-shaped functions given in TABLE I as S1, S2 and S3 that might be employed as the transfer function in BPSO. On the other hand, the V-shaped function family takes a totally different form of function image, which is given in TABLE I as V1, V2 and V3. When velocities of particles are input to transfer functions, and the outputs are regarded as the position converting probability, V-shaped functions well conform to the fact that velocities and the probability of position reversing converge to zero simultaneously.

In [6], a modified BPSO for task assigning in WSN is illustrated. The V-shaped transfer function they used is V1 in TABLE.I. The works in [6] also made an improvement to the position updating formula as in (17). This improvement enables particles to stay where they are when the evolution is about to end. The memory of former obtained good positions is kept, helping the particle swarm to converge faster.

$$X_i^{j+1} = \begin{cases} 0 & \text{if } rand \leq T(V_i^j) \text{ and } V_i^j \leq 0 \\ 1 & \text{if } rand \leq T(V_i^j) \text{ and } V_i^j > 0 \\ X_i^j & \text{if } rand > T(V_i^j) \end{cases} \quad (17)$$

Yet there are a lot other S-shaped and V-shaped functions with different characteristics, which may be even more suitable for the task assigning issue in WSN. We employ all functions in TABLE I to find the optimal transfer function for BPSO in the application of task assigning issue of WSN. When developing algorithms using V-shaped functions, the position updating formula of (17) is utilized.

TABLE I. The S-shaped and V-shaped functions

Name	Function formula
S1	$T(x) = 1/(1+\exp(-x))$
S2	$T(x) = 1/(1+\exp(-2x))$
S3	$T(x) = 1/(1+\exp(-x/2))$
V1	$T(x) = \begin{cases} 1 - 2/(1+\exp(-x)) & \text{if } x \leq 0 \\ 2/(1+\exp(-x)) - 1 & \text{else} \end{cases}$
V2	$T(x) = \left\lvert \frac{2}{\pi} \arctan(\frac{\pi}{x} \cdot x) \right\rvert$
V3	$T(x) = \lvert \tanh(x) \rvert$

V. EXPERIMENTAL RESULTS AND DISCUSSION

A. Particle Coding and Basic Configuration

To fit in for the application problem, a particle is denoted as a two-dimensional binary matrix. The element b_{ij} in the *i*-th row and the *j*-th column is either 1 or 0, meaning whether node j is chosen for task i. All particles are randomly initialized at first, so they may fail to meet the constraints. We continually use the formula of (18) to repair them. In (18), X_i is one whole row in a particle, $randint(d)$ generates d integers valued 1 or 0, Col is the total number of columns in a particle, and $sg(y)$ is 1 if $y > 0$ or 0 otherwise.

$$X_i = sg(X_i + randint(Col)) \quad (18)$$

During the run of BPSO, there may be some particles that don't satisfy the constraints. All these defective particles are repaired after each iteration. The encoding of particles makes each row of a particle represent a task allocating scheme for one task. Thus when one row of a particle does not satisfy the constraints, it is replaced by the identical row from the best-ever position of the swarm.

TABLE II. Problem scale setting

Mark	Sensor number	Task number
P1	30	6
P2	40	15
P3	50	20
P4	60	30

TABLE III. Algorithm parameter configuration

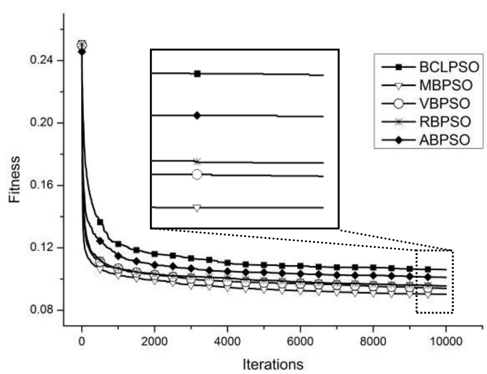
Parameter name	Value/formula
Lg	7
c	1.49445
c_1	2
c_2	2
Max inertia weight	0.9
Min inertia weight	0.4
Velocity range	[-6,6]
Max iteration	10000
Swarm size	50
muRate	0.02

For a valid and fair comparison, the simulation environment setting is identical to [6]. The value of e_{elec} is set as $50\,nJ/b$ and ε_{amp} is $10\,pJ/b/m^2$. The maximum transfer range is 100 m. The processing velocity of sensor nodes are within [30,100] *MCPS* while the rate of work is constrained to [4,10] *mW*. For each task, computational load is randomly initialized in [300,600] *KCC* and the communication load is set to be between 500 and 800 bytes. Upper limit of computational load is 40 *KCC* and communication load is 50 bytes. The scale of problems is confirmed by the number of sensor nodes and the number of tasks, as they confirm the shape of the solution space. In our simulation, four typical problems scales are considered as in TABLE II.

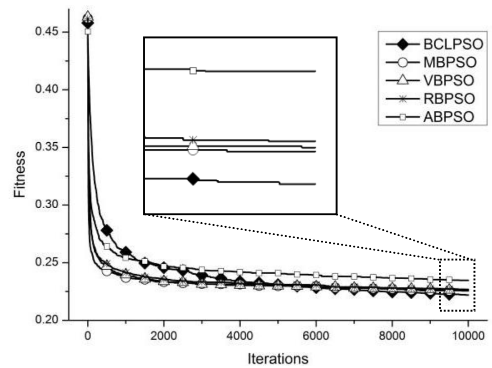
As for parameter configuration of all the BPSO, values or formulations are given in TABLE III, where *Lg* is for BCLPSO only. The *Pc* setting follows (19). The three proposed versions of local PSO and BCLPSO are developed for the task assigning issue in WSN. And the MBPSO from [6] is developed for comparison. All simulations are carried out in MATLAB with 30 runs.

$$Pc_i = 0.05 + \frac{0.45 \times (e^{\frac{10(i-1)}{ps-1}} - 1)}{e^{10} - 1} \qquad (19)$$

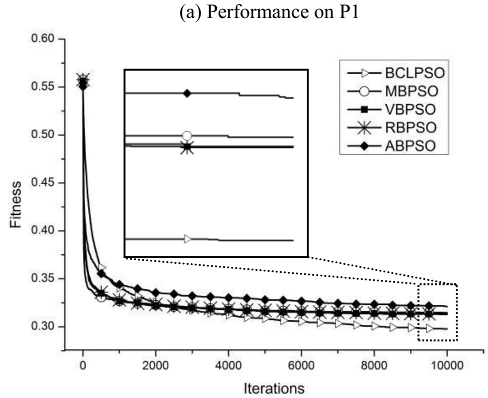
In order to compare the performance of different BPSO to the problem and analyze the characters and factors that cause the discrepancy, MBPSO, BCLPSO and the three topological BPSO are developed. In RBPSO, particles conform a closed cycle. Then the size of neighborhood is 3. The neighborhood size of ABPSO is set as 4. Hence there will randomly be 3 other particles in one particle's neighborhood. It's evident that the neighborhood size of VBPSO is 4. For a reasonable comparison, all the BPSO take the identical transfer function V1. The mutation rate is set as 0.02 in MBPSO and all local BPSO. Aimed at the problems with different scales, convergence characters of BPSOs are illustrated in Fig. 3.



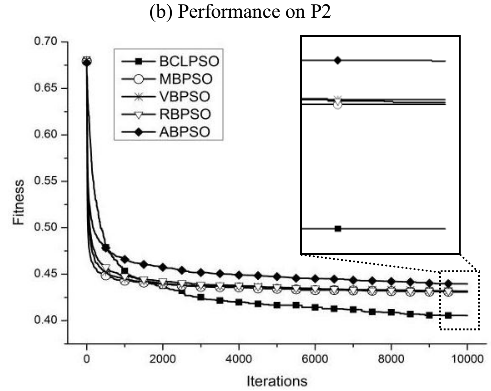
(a) Performance on P1



(b) Performance on P2



(c) Performance on P3



(d) Performance on P4

Figure 3. Performance of BPSOs in WSN task assigning.

It is seen from Fig. 3 that the three proposed local BPSO and BCLPSO are all effective for the problem, among which BCLPSO is the most promising one. In P1, where the problem scale is comparably small, MBPSO performs best, yet the superiority is not quite obvious against local BPSO and BCLPSO. When problem scale becomes larger, BCLPSO outperforms all other approaches in finding the global optimum, and the larger the problem scale, the more apparent the superiority is. Although the convergence speed is slightly slower, the trade-off is tolerable, and if the iteration is prolonged, we can see even greater superiority. As a complex multi-modal and real-world problem, the task assigning issue is better solved by BCLPSO than other approaches. The significance of BCLPSO against the best result gained by others is tested. Seeing from TABLE IV, BCLPSO is significantly better than others.

TABLE IV. Performance significance

	ABPSO	RBPSO	VBPSO	MBPSO	BCLPSO	S
P1	0.1011 0.0057	0.0955 0.0067	0.0940 0.0042	**0.0903** **0.0055**	0.1060 0.0059	1
P2	0.2346 0.0050	0.2267 0.0082	0.2260 0.0069	0.2256 0.0035	**0.2219** **0.0068**	1
P3	0.3210 0.0051	0.3129 0.0038	0.3131 0.0063	0.3146 0.0070	**0.2977** **0.0047**	1
P4	0.4395 0.012	0.4311 0.0048	0.4317 0.0066	0.4307 0.0039	**0.4055** **0.0062**	1

B. *Analysis on Transfer Functions*

To analyze the influence of transfer functions, simulations are done by using all the candidate transfer functions in TABLE I on BCLPSO. V-shaped functions show much better performance against S-shaped ones. Comparisons are given in TABLE V. The modification of transfer function and position updating formula preserves the knowledge about good positions. Also, the randomness at the end of the iteration is avoided. Under the guide of the preserved knowledge, particles are able to fly to the global optimum. For the three candidate V-shaped functions, they shows their own advantage in different problem scales. V1 and V2 are better than V3 according to TABLE V.

TABLE V. Best solutions fitness by different transfer functions.

Function	P1	P2	P3	P4
S1	0.1518	0.3133	0.4166	0.5973
S2	0.1512	0.3246	0.3984	0.5703
S3	0.1511	0.3428	0.4311	0.6079
V1	**0.1057**	0.2223	0.2881	**0.4139**
V2	0.1064	**0.2148**	**0.2776**	0.4164
V3	0.1059	0.2196	0.2840	0.4218

V3 drops much faster to 0 when input tends to 0. This feature weakens the local searching ability of the swarm. Solution fitness of V3 is close to that of V1 and V2 with a tiny gap owing to the loss of local searching ability. Still, the V-shaped function family is far more advantageous than the S-shaped one toward the problem of task allocating in WSN.

VI. CONCLUSIONS

A BPSO with the comprehensive learning strategy and three versions of local BPSO are proposed for the task assigning problem in WSN. Through simulations we find that all the proposed approaches are capable of solving this problem. Among them, the BCLPSO approach performs the best. Due to the particle diversity and extensive searching space gained by learning from all the particles' best-ever position, BCLPSO is competent as a promising solution to the problem. V-shaped functions are way better than S-shaped functions when applied as BPSO transfer function in the task assigning issue. For the three V-shaped functions, V1 and V2 are comparably more suitable. To sum up, the proposed BCLPSO with a V-shaped transfer function, which outperforms other approaches like MBPSO, is very promising to the task assigning problem in WSN.

REFERENCE

[1] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," Communications Magazine, IEEE, vol. 40, pp. 102-114, 2002.

[2] W. Guo, N. Xiong, H.-C. Chao, S. Hussain, and G. Chen, "Design and analysis of self-adapted task scheduling strategies in wireless sensor networks," Sensors, vol. 11, pp. 6533-6554, 2011.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Neural Networks, 1995. Proceedings., IEEE International Conference on, 1995, pp. 1942-1948 vol.4.

[4] R. V. Kulkarni and G. K. Venayagamoorthy, "Particle Swarm Optimization in Wireless-Sensor Networks: A Brief Survey," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 41, pp. 262-267, 2011.

[5] A. Salman, I. Ahmad, and S. Al-Madani, "Particle swarm optimization for task assignment problem," Microprocessors and Microsystems, vol. 26, pp. 363-371, 11/10/ 2002.

[6] Y. Jun, Z. Hesheng, L. Yun, P. Cheng, and S. Wei, "Task Allocation for Wireless Sensor Network Using Modified Binary Particle Swarm Optimization," Sensors Journal, IEEE, vol. 14, pp. 882-892, 2014.

[7] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on, 2000, p. 10 pp. vol. 2.

[8] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999, p. 1938 Vol. 3.

[9] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," Evolutionary Computation, IEEE Transactions on, vol. 10, pp. 281-295, 2006.