Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 160 – 170

doi:10.3906/elk-1304-213

Lifetime maximization of wireless sensor networks using particle swarm optimization

Aleem Kabeer MIR[1], **Muhammad ZUBAIR**[2,*], **Ijaz Mansoor QURESHI**[3]
[1]Faculty of Computing, Riphah International University, Islamabad, Pakistan
[2]Department of Electronic Engineering, International Islamic University, Islamabad, Pakistan
[3]Department of Electrical Engineering, Air University, Islamabad, Pakistan

Received: 22.04.2013 • **Accepted/Published Online:** 21.10.2013 • **Final Version:** 01.01.2016

Abstract: Wireless sensor networks have multiple applications in intelligent environment and structural monitoring. The major challenge in wireless sensor networks is the power constraint. This paper deals with minimizing the energy utilization of wireless sensor nodes and maximizing their overall life span. The objective of our proposed scheme is to find a method for grouping sensors into the maximum number of distinct sensor cover sets to totally monitor the required area. This problem can be solved by using the disjoint cover set problem. Present optimization techniques take much time and deliver unsatisfactory results in large-scale networks. This paper proposes a technique that delivers optimal results in minimal computation time. The results of the proposed technique are superior to existing techniques ranging from four to more than eighteen times better in various cases.

Key words: Wireless sensor networks, disjoint complete cover set, particle swarm optimization

1. Introduction

A wireless sensor network contains a different number of sensors placed at different positions to monitor the physical and environmental changes [1]. Sensors are placed in an environment in which battery charging or replacement is almost impossible. Sensor placement can be random or preplanned [2]. In the case of random deployment, more than one sensor node may be sensing the same area, thus resulting in inefficient utilization of the nonrechargeable batteries of the sensor nodes. A lot of work has been done regarding energy conservation. Although there is a significant amount of literature addressing the issue of efficient energy management [3] in wireless sensor networks, the sensor scheduling technique [4] is used for efficient energy usage in wireless sensor networks. In this technique, sensors are divided into a number of disjoint groups and each group turns ON when the previous group's energy is less than the specified threshold energy level. As a result, more sensor groups will sustain the network for a long period of time. In wireless sensor networks, this technique of finding disjoints sensor cover sets is known as the disjoint set cover (DSC) problem and it is a well-known nondeterministic polynomial complete problem [5]. Different people have applied different techniques to solve this DSC problem.

Slijepcevic and Potkonjak [5] tried to solve the DSC problem by proposing a greedy deterministic approach, known as the most constrained minimally constraining (MCMCC) algorithm. The same problem was approached by Cardei and Du [6] using the mixed integer programming (MCMIP) algorithm. MCMIP produced more disjoint sets than the MCMCC algorithm, but the latter had lower execution time. In addition

*Correspondence: muhammad.zubair@iiu.edu.pk

to the above mentioned algorithms, researchers have also utilized evolutionary algorithms to solve the coverage problem. In this regard, Lai et al. [7] were among the pioneers. They proposed the genetic algorithm for maximum disjoint sets (GAMDSC) to solve the DSC problem. However, the GAMDSC is infeasible with an increased number of sensors. The schedule transition hybrid genetic algorithm (STHGA) was proposed by Hu et al. [8] to solve the DSC problem.

In this paper, we solve the above-mentioned DSC problem by using a hybrid particle swarm optimization algorithm (PSODSC). There is a strong reason for choosing particle swarm optimization (PSO) [9] to solve the DSC problem. From our previous experience in [10–12] and from [13], we have observed that PSO is faster in convergence than the GA. It reaches the optimal solution in less time than the GA. The major difference between the GA and PSO is that that the former has more operations to perform than the latter. The GA also involves more parameters to work with while PSO has fewer parameters for implementation. Finally, the GA operates by generating a number of generations (also known as a vertical approach), while PSO operates on just one generation of candidate solutions (also known as a horizontal approach). Results of our proposed algorithm show that both the time complexity for large-scale wireless sensor networks and efficiency are achieved at the same time. The paper is organized as follows: Section 2 explains the DSC problem, Section 3 describes the components of the proposed PSODSC in detail, simulation results of the proposed algorithm are given and discussed in Section 4, and Section 5 concludes the paper.

1. Problem definition

We consider the target area having dimensions $L \times W$, as shown in Figure 1. Suppose a set S consisting of N number of wireless sensors is scattered in the desired region. The circular discs show the coverage of sensors; the area of each disc is the same, meaning that all sensors have equal coverage. The goal of our proposed algorithm is to search the maximum number of sensors of set $\tilde{D}$ with each set containing disjoint sensors. The corresponding disjoint sets S_j should satisfy the following conditions:

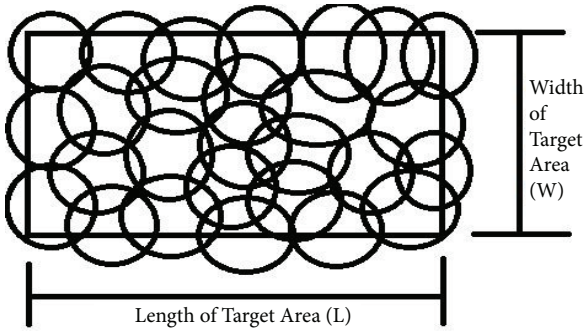

Figure 1. Target area L × W.

1) Each set $S = S_1 \cup S_2 \cup S_3 \ldots S_D \subseteq S$,

2) $S_j = \{S_{j1}, S_{j2}, \ldots, S_j, \ldots S_{j\max}\}$.

   Each sensor set must monitor the whole $L \times W$ target area. $S_{j\max}$ is the set of those sensors that are working in the j^{th} set.

3) No sensor should be used in multiple sensor cover sets, i.e. $S_j \cap S_k = \varphi$ where $j \neq k$ and $j, k = 1, 2, 3, 4, \ldots \tilde{D}$.

In Figure 1, most of the sensors cover the same target area, which forms fields. Figure 2 gives a closer look and explains how seven sensors form sixteen fields. When a total N number of sensors are activated, fields are formed and $\tilde{D}$ is found by using Eq. (1):

$$\tilde{D} = min\left(|F_j|\right) \quad j = 1, 2, \ldots, n_F, \tag{1}$$

where F_j represents those sensors that cover the field j, and n_F represents the total number of fields formed. In Figure 2, the values of $|F_1|, |F_2|, |F_3|, |F_4|, |F_5|, |F_6|, |F_7|, \ldots, |F_{16}|$ are 1, 2, 1, 2, 1, 3, 3, 2, 2, 1, 2, 2, 1, 2, and 1, respectively, so the value of $D_{\max}$ is 1.

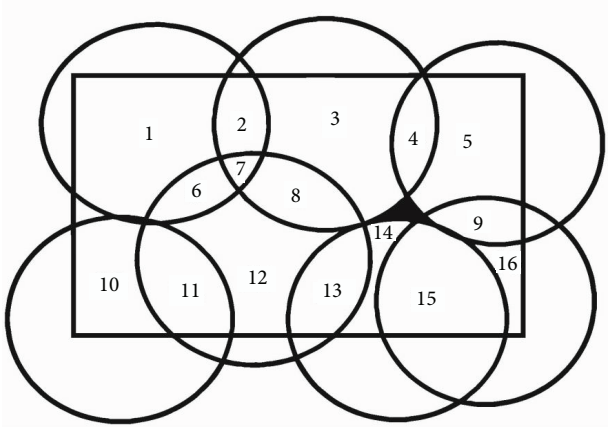

Figure 2. Fields formed by all sensors.

The total area covered by a sensor is represented by the coverage percentage of that sensor and this area is calculated by dividing the whole $L \times W$ target area into small, precalculated areas called grids. Only the grids that are fully covered by a sensor will be considered covered. Figure 3 shows three sensors which are active (ON) and fully covers 9, 9, and 8 grids. Hence, a total of 23 grids will be covered.

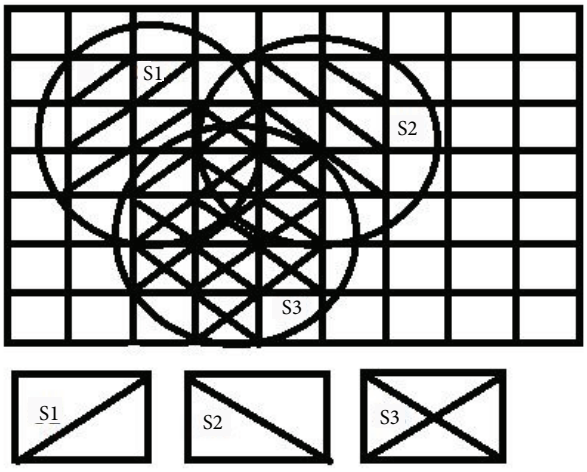

Figure 3. Number of grids covered by sensors S1, S2, and S3.

2. Proposed hybrid particle swarm optimization algorithm

This paper proposes a PSODSC algorithm for the DSC problem for extending the overall lifetime of a sensor network. In PSODSC, the key of design is the particle representation of sensors. Additional operators, called forward encoding scheme and schedule transition operations, are applied to the offspring.

In a forward encoding scheme, sensors are placed as an element in a particle. Each particle contains a number of different sensors covering different grids. When a particle contains such sensors that collectively cover the whole target area, $L \times W$, this particle will be saved and its sensors will be turned ON at their turn.

The implementation flowchart of the proposed PSODSC is shown in Figure 4.

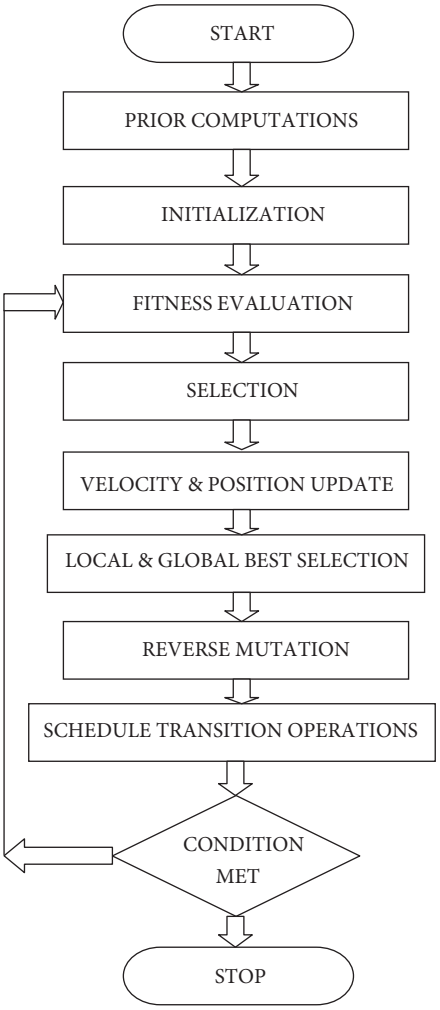

Figure 4. Flowchart of the proposed PSODSC.

2.1. Prior computations

The whole target area was divided into fields. The term "field" is used to represent that area that is covered by all sensors. For our computational facility, the whole target area was divided into grids. Fields were then computed on the basis of these grids, and then finally we attempted to find the set of those sensors that cover all the grids. The primary objective was to find a large number of such sensor sets.

2.2. Initialization and representation of particles

Each particle is encoded as a list of integers. Each element of a particle represents one sensor, and the value of the elements indicates the cover set to which the sensor belongs. Initially, a population with P set of particles is generated. Each particle P_i in the population is represented as in Eq. (2):

$$P_i = (p_{i1}, p_{i2}, p_{i3}, p_{i4}, \ldots, p_{iN}), \tag{2}$$

where p_{ij} represents scheduling numbers for sensors $j = 1, 2 \ldots N$, where N is the total number of sensors deployed and $i = 1, 2, 3, 4, \ldots, p$, where p is the total number of particles in the population. Initially, each sensor is assigned value 1 for scheduling number, i.e. P_i = (1, 1, 1,..., 1), i = 1, 2,..., p means that all sensors are working. For every particle, K_1 elements are randomly selected, where K_1 is a user-defined parameter. If the K_1 selected sensors are redundant with respect to the present set, then the values of the selected sensors will be incremented by 1, which means that these sensors will be in the second schedule. This process is applied to all particles and then the initial population with P particles is generated.

2.3. Fitness evaluation

The initial fitness evaluation function of a particle P_i in P population is found by using in Eq. (3):

$$f_i = \beta^* \psi_i + (1 - \beta)^* \eta_i, \tag{3}$$

where ψ_i, the number of DSC sets, is $\psi_1, \psi_2, \ldots, \psi_i$, formed at the end of the initialization process, and $(\eta_i \in 0, 1)$ shows the coverage percentage of the incomplete sensors cover set $(\psi_i + 1)^{th}$. If the coverage percentage of the incomplete cover set η_i is 1, then the value of ψ_i adds to 1; otherwise its coverage percentage value will be added to η_i. The greater the value of ψ_i, the higher the fitness of particle P_i. The fitness of the initial population P_i, where i = 1, 2, 3, ... p, is calculated using the above fitness evaluation formula. The fitness values of all the particles are sorted in descending order and local best particles and global best particles are declared. These local best particles and global best particles are nominated for initial population.

2.4. Selection of particles

Select the two particles with the best fitness values from the currently generated population. Let the selected particles be p_i and p_j. Now randomly select all the elements from the selected particles and form new offspring p_k by recombining the elements. The fitness of new offspring p_k is then calculated by using Eq. (3) and only the offspring with better fitness values than their parents are added in the new population. In this way, the new population with p_i particles is created. Now the fitness of the new population is calculated according to the above fitness evaluation expression and fitness values are sorted in descending order to find local and global bests.

2.5. Velocity and position update

Now the velocity and positions of the particles are updated using Eqs. (4) and (5), respectively.

Updating velocity:

$$V_{im}(n) = V_{im}(n-1) + \varnothing_1 (p_{im} - b_{im}(n-1)) + \varnothing_2 (p_{gm} - b_{im}(n-1)), \tag{4}$$

where $V_{im}(n-1)$ is the previous velocity, $p_{im} - b_{im}(n-1)$ is the distance from local best, and $p_{gm} - b_{im}(n-1)$ is the distance from global best. The constants $\varnothing_1$ and $\varnothing_2$ and their values are problem-dependent. $V_{im}(0)$ can be taken as zero for the initial step.

Updating position:

$$S\left(V_{im}\right) = 1/1 + e_{im}^{-V}. \tag{5}$$

$$If\ S\left(V_{im}\right)\ >\ rand()\, then\, b_{im}\left(n\right) = +1\, else\, b_{im}\left(n\right) = -1.$$

Here, $b_{im}\left(n\right)$ is the position value. The value of $S\left(V_{im}\right)$ is between 0 and 1. The velocity and position updating steps will be carried out for all $i = 1, 2, 3, \ldots, p$ particles and for all $j = 1, 2, 3, \ldots, N$ elements.

2.6. Reverse mutation

The reverse mutation operation randomly selects sensors from an incomplete cover set $\varepsilon\eta_i$ and places these sensors into a randomly selected complete cover set ψ_i. If the selected incomplete cover set $\varepsilon\eta_i$ is already a complete cover set ψ_i, then reverse mutation operation is not performed. Otherwise, it is performed only once in every generation. For every element e_{ij} in the incomplete cover set $\varepsilon\eta_i$, if $q_1 < \mu$, the mutated element value is generated as in Eq. (6):

$$e_{ij} = floor\left(c_i q_2\right) + 1, \tag{6}$$

where q_1 and q_2 are uniform random numbers between [0, 1] and the floor $(c_i q_2)$ represents the largest integer that is less than or equal to $c_i q_2$.

2.7. Schedule transition operations

The schedule transition operations consist of three search operations: mixed schedule transition, forward schedule transition, and critical schedule transition. These schedule transition operations schedule sensor redundancy information among all particles.

a) Mixed schedule transition: This transition scheme schedules redundant sensors among complete cover sets as shown in Figure 5. A sensor is selected randomly from a particle. If it is redundant, it is rescheduled to another randomly selected complete cover set. If the present scheduling number is the same as the newly selected scheduling number, then this redundant sensor will be scheduled to an incomplete cover set $(\varepsilon\eta)$. This process is repeated k_2 times so that all redundant sensors from all complete cover sets (ψ) will be moved to $\varepsilon\eta$.

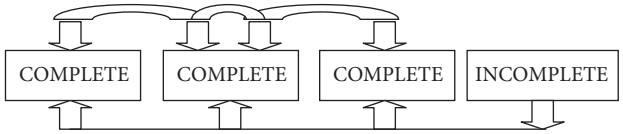

Figure 5. Mixed schedule operations.

b) Forward schedule transition: This transition scheme schedules redundant sensors among ψ and $\varepsilon\eta$, as shown in Figure 6. In this way, the coverage percentage of $\varepsilon\eta$ increases. For every particle P_i, k_1 sensors are randomly selected. If the selected sensor is redundant, the selected sensor is scheduled into the $\varepsilon\eta$. In this way, the coverage percentage of incomplete cover sets increases.

Figure 6. Forward schedule operations.

c) Critical schedule transition: This transition scheme is designed especially for covering critical fields. Redundant sensors from ψ are rescheduled to critical fields. Once a critical field is covered by at least one sensor, then this $\varepsilon\eta$ has higher chances to become ψ by using the above-mentioned two scheduling techniques.

3. Experimental results and discussion

The results of the proposed PSODSC algorithm were compared with the previous two algorithms, GAMDSC and STHGA, which have been applied to the same problem for finding a solution to the DSC problem. It can be seen that the performance of the PSODSC algorithm is much better than the above-mentioned two algorithms.

We conducted a number of simulations to compare the results of the proposed PSODSC algorithm with GAMDSC and STHGA. The simulation platform was MATLAB 7.8.0 (R2009a) in Windows XP/Pentium 4, 2.8 GHz. Each simulation case included 100 runs of the PSODSC algorithm.

Previously, GAMDSC and STHGA were proposed for the same DSC problem. Results of our proposed scheme are compared with both these schemes and can be viewed in Tables 1 and 2. The detailed comparison of different parameters, like number of fields formed, average simulation time, number of disjoint sets, and successful percentage, is as follows: Table 1 summarizes the test results of the proposed PSODSC and GAMDSC with different numbers of sensors N and sensing ranges R.

By comparing the number of fields of both the algorithms, it can be noted that the number of fields formed by the proposed PSODSC algorithm is greater than or almost equal to the GAMDSC (see columns 4 and 9 of Table 1).

When total N numbers of sensors are deployed in the whole target area, fields are formed. On the basis of these fields, we calculate the total number of disjoint cover sets ($\tilde{D}$). The value of this $\tilde{D}$ is different in each deployment. The average simulation time of the algorithm is the total time required/consumed by the proposed PSODSC algorithm to find the precalculated number of disjoint cover sets. This time will be different for different numbers of deployments. Average simulation time increases by the increase of the number of sensors deployed.

The average simulation time of an algorithm represents the amount of time required for finding the disjoint sets of sensors (ψ). The average simulation times of both algorithms are shown in columns 7 and 12 of Table 1. When we compare the average simulation times of both algorithms, it is clear that the average simulation time of the proposed PSODSC algorithm is always much less than that of the GAMDSC for all the cases from serial number 1 to 9 of Table 1. The difference increases more rapidly as the number of sensors increases. When the number of sensors increases by 500, the performance of GAMDSC is almost negligible. The shorter average simulation time means the proposed algorithm is much more effective for the DSC problem than the GAMDSC.

$\tilde{D}$ is the number of disjoint sets of sensors (ψ) formed. If we compare these $\tilde{D}$ values of the proposed PSODSC algorithm with the GAMDSC algorithm, it can be clearly seen that the $\tilde{D}$ values of the proposed PSODSC algorithm are always greater in number than GAMDSC (see columns 6 and 11 from serial number 1 to 9 of Table 1). There is much difference in some cases, like case numbers 5 to 9 in Table 1. This difference shows the better performance of the proposed PSODSC algorithm compared to GAMDSC, which means that the proposed algorithm is more suitable for solving the DSC problem than GAMDSC.

The successful percentage (OK %) field shows the overall performance of the algorithm in all respects. GAMDSC gets 100% in only one case, serial number 1 in Table 1. It gets 0% in all the remaining cases from

Table 1. Results of proposed PSODSC algorithm and GAMDSC with different numbers of sensors N and sensing ranges R.

	Test cases		Proposed PSODSC					GAMDSC				
S. no.	N (No. of sensors deployed)	R (sensing ranges)	N_f (no. of fields)	$\tilde{D}$ calculated	$\tilde{D}$ found	T (Avg. simulation time, ms)	OK % (successful %)	N_f (no. of fields)	$\tilde{D}$ calculated	$\tilde{D}$ found	T (Avg. simulation time, ms)	OK % (successful %)
1	100	20	374	11	11	7.2725	100	385	7	7	126	100
2	300	15	674	15	15	67.44	100	673	15	13	9713	0
3	300	20	397	27	27	66.95	100	400	29	26	10080	0
4	400	10	1563	8	8	43.720	100	1556	8	6	13764	0
5	400	15	675	21	21	86.813	100	676	20	16	13268	0
6	500	8	2389	8	8	81.827	100	2400	6	4	17413	0
7	500	10	1580	12	12	142.457	100	1586	8	5	18527	0
8	1000	5	6043	5	5	237.013	100	6076	3	0	38105	0
9	1000	8	2493	17	17	323.57	100	2498	6	3	37830	0

Table 2. Results of proposed PSODSC and STHGA with different number of sensors and sensing ranges.

	Test cases		Proposed PSODSC					GAMDSC				
S. no.	N (No. of sensors deployed)	R (sensing ranges)	N_f (no. of fields)	$\tilde{D}$ calculated	$\tilde{D}$ found	T (Avg. simulation time, ms)	OK % (successful %)	N_f (no. of fields)	$\tilde{D}$ calculated	$\tilde{D}$ found	T (Avg. simulation time, ms)	OK % (successful %)
1.	100	20	374	11	11	7.2725	100	385	7	7	33	100
2.	300	15	674	15	15	67.44	100	673	16	16	400	100
3.	300	20	397	27	27	66.95	100	400	32	32	468	100
4.	400	10	1563	8	8	43.720	100	1556	9	9	797	100
5.	400	15	675	21	21	86.813	100	676	23	23	767	100
6.	500	8	2389	8	8	81.827	100	2400	7	7	1588	100
7.	500	10	1580	12	12	142.45	100	1586	15	15	11386	100
8.	1000	5	6043	5	5	237.01	100	6076	5	5	4534	100
9.	1000	8	2493	17	17	323.57	100	2498	17	17	5901	100

serial number 2 to 9 in Table 1. Thus, GAMDSC is not suitable for solving the DSC problem. However, the proposed PSODSC algorithm gets 100% success in every case from serial number 1 to 9 in Table 1.

By comparing the number of fields of both algorithms, it can be noted that the number of fields calculated by the proposed PSODSC algorithm are greater than or almost the same as those of STHGA (Table 2).

Now if we compare the $\tilde{D}$ values of the proposed PSODSC algorithm with STHGA, it can be clearly seen that the $\tilde{D}$ values of the proposed PSODSC algorithm are greater than or nearly the same as those of STHGA (see columns 6 and 11 from serial number 1 to 9 in Table 2). Taking cases 1 and 6 as examples, the proposed PSODSC algorithm formed more disjoint complete cover sets than the STHGA. One thing is very clear from the results: as the sensing range of the sensors decreases, the number of disjoint complete cover sets also decreases (see columns 6 and 11, cases 4, 6, and 8 in Table 2). Thus, if we would like to increase the number of disjoint complete cover sets, we have to increase the sensing ranges of sensors (see columns 6 and 11, cases 1, 2, 3, and 5 in Table 2).

The average simulation times of both the PSODSC algorithm and STHGA are shown in columns 7 and 12 of Table 2. As the number of sensors increases, the average simulation time also increases (see cases 8 and 9 in Table 2). The average simulation time of the proposed PSODSC is less than the average simulation time of STHGA in all the cases from 1 to 9 (see Table 2) because PSO is more convergent and reaches the optimal solution faster than GA. The reduced average simulation time means that the proposed PSODSC algorithm is much more effective for solving the DSC problem than the STHGA.

Both PSODSC and STHGA get 100% success in all the cases from serial number 1 to 9 in Table 2.

In Figure 7, a relation between average simulation time and number of complete cover sets is shown. As the average simulation time increases, the number of complete cover sets decreases, because in the beginning most of the sensors have already been used in the disjoint cover sets. Thus, in the end, there are very few sensors that are not still used in any disjoint set.

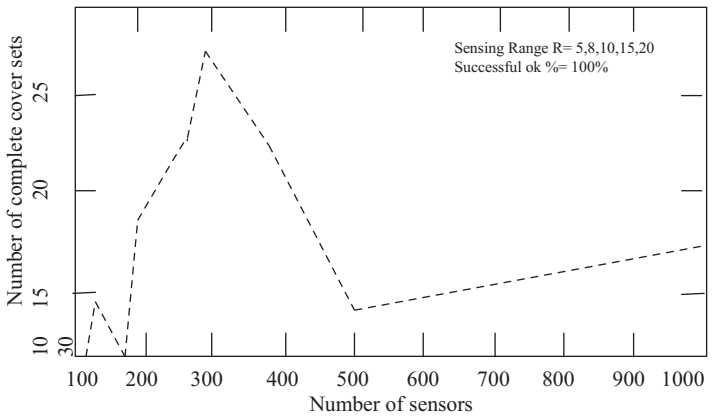

Figure 7. Relation between the total sensors deployed and number of complete cover sets achieved.

As the number of sensors increases, the number of disjoint complete cover sets also increases; more sensors form more fields, and hence there are more disjoint cover sets. However, this process continues for some initial time. Eventually, increasing the number of sensors is useless. Figure 8 clearly shows this relation.

Relations between the numbers of fields formed and disjoint complete cover sets are shown in Figure 9. In the beginning, more fields are formed, but with the passage of time, the number of fields decreases.

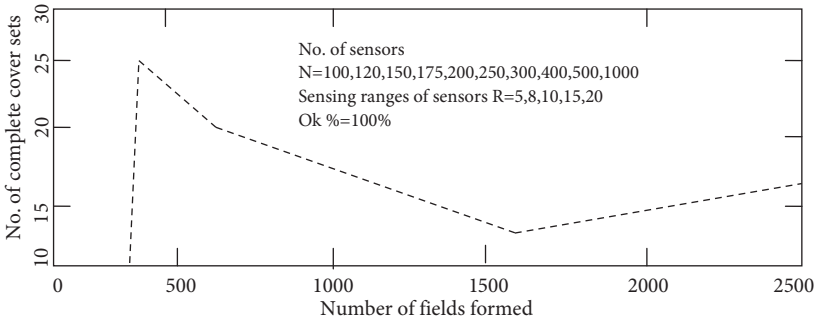

Figure 8. Relation between the total fields formed and number of complete cover sets achieved.

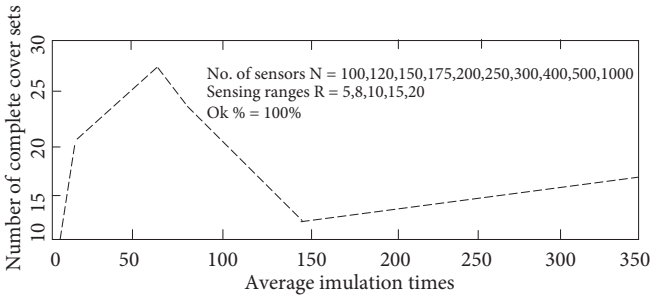

Figure 9. Relation between the average simulation time and number of complete cover sets achieved.

4. Conclusions

We proposed a hybrid particle swarm optimization algorithm for DSC, called PSODSC, to solve the DSC problem. Simulation results show that the PSODSC algorithm performs better than the GAMDSC and STHGA and gets optimal results in much less time. The improvement in time is in a range of 4 times to more than 18 times faster. The PSODSC algorithm also works well in large-scale wireless sensor networks. Although PSODSC can get near-optimal solutions, there is still room for improvement. For large-scale dense wireless sensor networks, there should be more efficient utilization of PSO to avoid trapping and time complexity issues. In the future, we intend to investigate the hybridization of PSO with various other evolutionary techniques, especially with differential evolution and ant colony optimization.

References

[1] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. Comput Netw 2002; 38: 393–422.

[2] Younis M, Akkaya K. Strategies and techniques for node placement in wireless sensor networks: a survey. Ad Hoc Netw 2008; 6: 621–655.

[3] Anastasi G, Conti M, Di Francesco M, Passarella A. Energy conservation in wireless sensor networks: a survey. Ad Hoc Netw 2009; 7: 537–568.

[4] Wang L, Xiao Y. A survey of energy-efficient scheduling mechanisms in sensor networks. Mobile Netw Appl 2006; 11: 723–740.

[5] Slijepcevic S, Potkonjak M. Power efficient organization of wireless sensor networks. In: IEEE 2001 Wireless Communications Conference; 11–14 June 2001; Helsinki, Finland. New York, NY, USA: IEEE. pp. 472–476.

[6] Cardei M, Du DZ. Improving wireless sensor network lifetime through power aware organization. Wirel Netw 2005; 11: 333–340.

[7] Lai CC, Ting CK, Ko RS. An effective genetic algorithm to improve wireless sensor network lifetime for large-scale surveillance applications. In: IEEE 2007 Congress on Evolutionary Computation; 25–28 September 2007; Singapore. New York, NY, USA: IEEE. pp. 3531–3538.

[8] Hu XM, Zhang J, Yu Y, Chung HSH, Li YL, Shi YH, Luo XN. Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks. IEEE T Evolut Comput 2010; 14: 766–781.

[9] Kennedy J, Eberhart R. Swarm Intelligence. San Diego, CA, USA: Academic Press, 2001.

[10] Zubair M, Choudhry MAS, Malik AN, Qureshi IM. Particle swarm with soft decision for multiuser detection of synchronous multicarrier CDMA. IEICE T Commun 2008; E91-B: 1640–1643.

[11] Choudhry MAS, Zubair M, Malik AN, Qureshi IM. Near optimum detector for DS-CDMA system using particle swarm optimization. IEICE T Commun 2007; E90-B: 3278–3282.

[12] Zubair M, Choudhry MAS, Malik AN, Qureshi IM. Particle swarm optimization assisted multiuser detection along with radial basis function. IEICE T Commun 2007; E90-B: 1861–1863.

[13] Russell C, Eberhart YS. Comparison between genetic algorithms and particle swarm optimization. Lect Notes Comp Sc 1998; 1447: 611–616.