Information Sciences 180 (2010) 4818–4833

Contents lists available at ScienceDirect

Information Sciences

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

A memetic algorithm for extending wireless sensor network lifetime

Chuan-Kang Ting *, Chien-Chih Liao

Department of Computer Science and Information Engineering, National Chung Cheng University, Chia-Yi 62102, Taiwan

ARTICLE INFO

Article history:
Received 8 September 2009
Received in revised form 5 June 2010
Accepted 14 August 2010

Keywords:
Memetic algorithm
Evolutionary algorithm
Wireless sensor network
Lifetime extension

ABSTRACT

Extending the lifetime during which a wireless sensor network (WSN) can cover all targets is a key issue in WSN applications such as surveillance. One effective method is to partition the collection of sensors into several covers, each of which must include all targets, and then to activate these covers one by one. Therefore, more covers enable longer lifetime. The problem of finding the maximum number of covers has been modeled as the SET K-COVER problem, which has been proven to be NP-complete. This study proposes a memetic algorithm to solve this problem. The memetic algorithm utilizes the Darwinian evolutionary scheme and Lamarckian local enhancement to search for optima given the considerations of global exploration and local exploitation. Additionally, the proposed algorithm does not require an upper bound or any assumption about the maximum number of covers. The simulation results on numerous problem instances confirm that the algorithm significantly outperforms several heuristic and evolutionary algorithms in terms of solution quality, which demonstrate the effectiveness of the proposed algorithm in extending WSN lifetime.

1. Introduction

Recent advances in hardware miniaturization, communication technologies, and low-cost mass production have facilitated the emergence of wireless sensor networks (WSNs) that consist of small, inexpensive, battery-powered, and wirelessly connected sensors. The WSNs have brought up various new applications, including surveillance, home security, and environmental monitoring [3,4,25,42]. WSN sensors are deployed randomly or systematically to collect information about their surroundings within their sensing range. They can transmit the collected information via wireless communication; some can even process the data before transmission. Despite their widely varying characteristics, all sensors essentially collect, transmit, and relay information. A promising WSN application is long-term surveillance in hostile or distant environments. Using WSNs for military surveillance, for example, involves deploying numerous sensors throughout the region of interest by aircraft to detect enemy activity or equipment. However, a key consideration in the design of WSNs is the power supply since replacing batteries in sensors is often impractical.

Although a considerable number of studies have addressed energy efficiency issues in generic wireless ad hoc networks, distributed sensing applications impose new constraints on sensor network coverage [10]. For instance, surveillance applications may require at least one sensor in each location in a geographic region of interest [8], while object tracking applications may require at least three sensors [7]. Data sampling applications may require coverage of a given percentage of monitored regions. In addition to sensing coverage, network connectivity is another important property of WSNs. Connectivity enables sensor nodes to relay collected information back to data sinks. Zhang and Hou [45] proved that if the communication range is at least twice the sensing range, then full coverage of a convex area implies connectivity of the WSN. Hence,

* Corresponding author.
E-mail addresses: ckting@cs.ccu.edu.tw (C.-K. Ting), lcch97p@cs.ccu.edu.tw (C.-C. Liao).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.08.021

the constraints of full coverage and connectivity can be reduced to the full coverage constraint alone. This study adopts this result and therefore considers the full coverage constraint. The WSN *lifetime* is accordingly defined as the time span during which a WSN satisfies the full coverage constraint.

Scheduling sensor activity is an important approach for prolonging WSN lifetime. It involves the scheduling of sensors to alternate between *active* and *inactive* modes so that they can maintain coverage with reduced power requirements [34,36]. More specifically, this approach divides all sensors into disjoint sensor subsets, or *sensor covers*,[1] each of which must satisfy the full coverage constraint. At any time, only one sensor cover in active mode provides functionality, while the remaining sensor covers stay inactive to save energy. Once any sensor in the active sensor cover runs out of energy and thus cannot maintain full coverage, one of the inactive sensor covers is selected to enter active mode and continue the functionality. Therefore, identifying more sensor covers allows the WSN lifetime to be extended further. Recent studies [34,40] indicate that this approach not only reduces energy consumption, but also prolongs sensor network lifetime.

The problem of finding the maximum number of covers to extend WSN lifetime has been modeled as the SET K-COVER problem [38]. Provided K covers, the lifetime of WSNs can ideally be extended by a factor of K using the above approach under the coverage constraint. The SET K-COVER problem has been proven to be NP-complete. Under the assumption $NP \neq P$, no exact algorithm can solve this problem in polynomial time. Some heuristic algorithms have been presented, but they generally suffer from the trade-off between solution quality and running time. Recently, Lai et al. [23] proposed using genetic algorithm (GA) to deal with this trade-off. The GA achieves near-optimal solutions in acceptable time but requires information on the value of K or its upper bound, which is usually unobtainable. Additionally, such approaches rarely yield optimal solutions. A means of improving solution quality with a short running time is urgently needed.

This study develops a memetic algorithm (MA) to solve the SET K-COVER problem of extending WSN lifetime. Memetic algorithm is a blooming dialect of evolutionary algorithm (EA). In addition to *Darwinism*, MA adopts the *Lamarckian* theory that offspring can inherit the knowledge or characteristics that their parents acquire during their lifetime. The MA implements this idea by integrating a local enhancement, such as local search and repair operator, into the canonical EA, and making the enhancement inheritable. This integration significantly improves the exploitation ability of EA and has been widely shown to provide superior solution quality and high convergence speed [16,21,24,27,31–33,37]. The proposed MA is based on the order-based GA with the compact operator, which is a novel local enhancement operator that is designed to address the SET K-COVER problem. Furthermore, this study devises a fitness function based on the contribution of sensors to covers. A series of simulations is conducted to evaluate the performance of the proposed MA in terms of solution quality and running time, and to verify its superiority over several heuristic and evolutionary algorithms.

The rest of this paper is organized as follows. Section 2 reviews related work on WSN lifetime extension. Section 3 formulates the problem of extending the WSN lifetime. Section 4 sheds light on the proposed MA and its operators. Simulation results are presented in Section 5. Finally, conclusions are drawn in Section 6.

2. Related work

The many aspects of the WSN lifetime problem include sensor activity scheduling [8,38], network structure [7], data aggregation [17,20,22], data compression [26], and routing protocol [18,29,35]. (For a complete survey, see [3]). This study focuses on sensor activity scheduling. The problem of extending WSN lifetime by sensor activity scheduling was first modeled as the SET K-COVER problem by Slijepcevic and Potkonjak [38]. They proved the NP-completeness of this problem by reduction from the minimum cover problem [13]. To solve this problem, the authors proposed the most constrained-minimally constraining covering heuristic (MCMCC). This approach runs in polynomial time but often yields unsatisfactory results. Cardei and Du [8] formulated WSN lifetime extension as the disjoint set covers (DSC) problem, which is analogous to the SET K-COVER problem. They presented a heuristic algorithm, called maximum covers using mixed integer programming (MCMIP), to solve the DSC problem. Although the MCMIP method can find the optimal solution, its implicit exhaustive search requires exponential running time. This high computational cost limits its applicability for large-scale WSNs. Cardei et al. [9] further considered cases without the disjointedness constraint on sensor subsets. Additionally, Berman et al. [5] took the initial battery power and energy consumption rate into account. In their study, the constraint on disjointedness is relaxed so that each sensor can participate in different sensor covers as long as it still has energy. The authors designated this problem the sensor network lifetime problem (SNLP), which entails finding a monitoring schedule that maximizes the network lifetime. They developed a $(1 + \ln(1 - q)^{-1})$-approximation algorithm for the case in which a q-portion of the monitored area must be covered. To cover 95% of the monitored area, for example, their schedule ensures that the obtained lifetime is at most 3.99 times shorter than the optimal lifetime.

Instead of maximizing lifetime, Abrams et al. [1] formulated the problem of finding the maximum average coverage given the number of covers. They claimed that this problem is more natural than the SET K-COVER problem in that full coverage is less likely to be achieved when sensors are distributed randomly and heterogeneously. The authors devised three algorithms to solve this problem. The first is a randomized algorithm with an expected fraction $1 - \frac{1}{e}$ of the optimum. The second is a distributed greedy algorithm with a $\frac{1}{2}$-approximation ratio. The third is a centralized greedy algorithm, which is a derandomized version of the first algorithm and has a $(1 - \frac{1}{e})$-approximation ratio. Ai et al. [2] viewed the relationship between

[1] In this paper, the two terms 'sensor cover' and 'cover' are used interchangeably.

this problem and the optimal solutions as an approximate Nash equilibrium for graphical games. Their proposed DRACo method achieves near-optimal coverage on test problems.

The above approaches for extending WSN lifetime, however, suffer from the trade-off between solution quality and running time. For the SET K-COVER problem, the MCMCC takes only polynomial time but often yields unsatisfactory solutions. On the other hand, the MCMIP ensures optimal solutions but at the cost of exponential time complexity. To address this issue, Lai et al. [23] designed an integer-coded GA in which a gene $c_i \in \mathbb{N}$ indicates the group number assigned to sensor S_i, and a chromosome represents the group arrangement of all sensors for covers. In the course of evolution, the groups gradually form covers, *i.e.*, they satisfy the constraint of full coverage. Studies have demonstrated that this algorithm outperforms MCMCC in terms of the number of covers and is much faster than MCMIP. Nevertheless, one defect is that, owing to the integer representation of chromosomes, this GA requires an upper bound on the number of covers, which is usually unobtainable. Moreover, like most other GAs, the proposed GA rarely yields optimal solutions. Therefore, an algorithm is still needed to consistently deliver, within an acceptable running time, good activity schedules for extending WSN lifetime.

3. Problem formulation

This study addresses the SET K-COVER problem of WSN lifetime extension. Suppose n sensors $S_1, \ldots, S_n$ are deployed to monitor m targets $T_1, \ldots, T_m$. A target T_j is said to be *covered* if it lies within the sensing range of at least one sensor. Fig. 1 shows a WSN with five sensors and four targets. The relationship between sensors $S_1, \ldots, S_5$ and targets $T_1, \ldots, T_4$ is represented by a bipartite graph $G = (V, E)$, where $V = S \cup T$ and $e_{ij} \in E$ if S_i covers T_j. Fig. 2 presents the bipartite graph of the WSN in Fig. 1, where $S_1 = \{T_1\}$, $S_2 = \{T_1, T_2\}$, $S_3 = \{T_2, T_3, T_4\}$, $S_4 = \{T_3\}$, and $S_5 = \{T_4\}$. The maximum number K of disjoint covers in this example is two. They are $C_1 = \{S_1, S_3\}$ and $C_2 = \{S_2, S_4, S_5\}$.

The SET K-COVER problem is to find the maximum number of covers associated with the longest lifetime extension, which is equivalent to partitioning the set of sensors into the maximum number of covers. The problem is defined formally below.

Definition (*SET K-COVER Problem*). Given a collection $S = \{S_1, \ldots, S_n\}$ of subsets of a finite set $T = \{T_1, \ldots, T_m\}$, find the maximum number, K, of covers $C_1, \ldots, C_K \subseteq S$ with $C_i \cap C_j = \emptyset$ for $i \neq j$, such that every element of T belongs to at least one element of C_i.

This problem has been proven to be NP-complete [38]. Cardei and Du [8] also presented an equivalent formulation as the disjoint set covers (DSC) problem and proved its NP-completeness. This problem has been applied not only to extension of WSN lifetime, but also to protein and gene networks [6] and database systems [44].

4. Proposed memetic algorithm

This work proposes a novel MA to solve the SET K-COVER problem of maximizing WSN lifetime. The MA implements the GA scheme and additionally adopts the compact operator, which is a local enhancement operator that is designed specifically for the SET K-COVER problem. Algorithm 1 presents the framework of the proposed MA. Following the GA scheme, the MA encodes candidate solutions as *chromosomes*. The method of encoding chromosomes is referred to as *representation*, which is essentially related to the problem to be solved. The fitness function evaluates the quality (*fitness*) of candidate solutions (chromosomes). In maximization problems, a better solution corresponds to higher fitness. The EAs, such as GA and MA, manipulate a set (*population*) of chromosomes to search for the optimal solution.

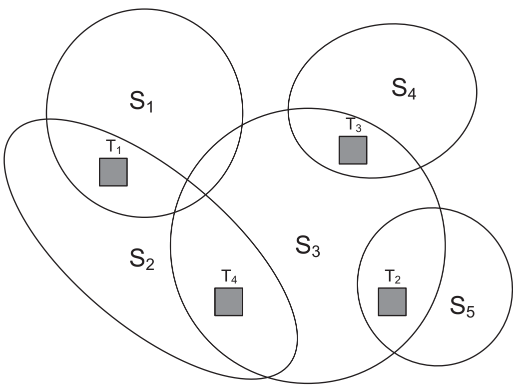

Fig. 1. Example deployment of WSN.

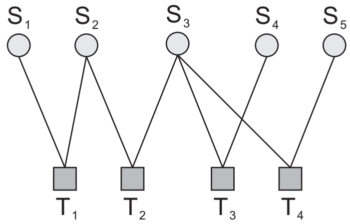

Fig. 2. Bipartite graph of a WSN.

Algorithm 1. Memetic algorithm

```
initialize population P;
evaluate P;
while (not terminated)
{
  Ps = Select (P);
  Pc = Crossover (Ps);
  Pm = Mutate (Pc);
  P' = LocalEnhance (Pm);
  evaluate P';
  P = Survival (P,P');
};
```

After initializing the population, MA embarks on the evolutionary process. First, the *selection* operator picks two chromosomes from the population to serve as *parents*. The *crossover* operator then exchanges the information between these two parents to produce their *offspring*. A predetermined crossover rate defines the probability of performing crossover. Analogously, *mutation* is performed with a probability, called mutation rate, to alter slightly some genes in the offspring. This study develops the *compact* operator as a local enhancement operator for the MA to deal with the SET K-COVER problem. This operator is applied to the offspring after the mutation phase.

The process of reproduction, selection–crossover–mutation–compact, is repeated until the offspring population is filled. Based on the Darwinian theory of "Survival of the Fittest", the *survivor* operator selects the fittest chromosomes from the offspring population with or without the parental population. The selected chromosomes constitute the next-generation population. As the evolution continues, the MA is expected to drive the search toward the global optima. The following subsections describe the elements of the proposed MA in further detail.

4.1. Representation

This study proposes the use of an order-based representation for chromosomes. In the SET K-COVER problem, partitioning the set of sensors into subsets for covers can be conceptualized as a process of accumulating covers. Accordingly, a gene c_i at locus i indicates that sensor S_{c_i} is collected in the ith *order*; a chromosome represents the sequence in which all sensors are collected to form covers. As Fig. 3 shows, S_3 initializes the first group G_1. When S_1 joins this group, it forms a cover since

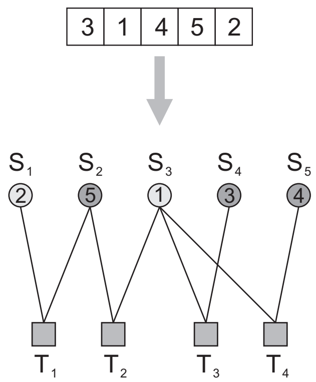

Fig. 3. Order-based chromosome. The number in the circle of sensor node S_i denotes the order in which S_i is collected.

group $G_1 = \{S_3, S_1\}$ includes all elements of T. Next, S_4 participates in a new group for another cover. This accumulative process continues until all sensors have been distributed to groups *in order*—the number of accumulated covers is therefore the number K of disjoint covers to be maximized. Like the integer representation used in [23], the order-based representation always satisfies the "disjointedness" of covers, in that each gene appears exactly once and can join only one group (cover). Moreover, a major advantage of the order-based representation is that, unlike integer representation, no upper bound or assumed number K is required.

4.2. Fitness function

The fitness function is vitally important to EAs because it explicitly or implicitly affects the search direction. An effective fitness function must render sufficient information about the search direction and clearly distinguish between good and bad candidate solutions. The fitness function is essentially problem-dependent. A fitness function for the SET K-COVER problem can simply treat the number of covers as the fitness value. Although this method has proven effective in integer-coded GA [23], it does not account for groups that include several sensors covering the same targets. This redundancy potentially reduces the number of obtained covers.

This study addresses this issue by considering how each sensor 'contributes' to a cover. More specifically, the fitness value of a chromosome is defined as the sum of contributions of all sensors, where the contribution of a sensor is quantified as the incremental change in the number of covers to which the sensor leads. For example, in the left part of Fig. 4, the contribution of S_3 is three, and that of S_5 is zero. Notably, the contribution of S_2 is two, since it covers two targets (T_1 and T_2) for a new group G_2. The fitness value of the chromosome is therefore 3 + 0 + 0 + 1 + 2 = 6. The proposed fitness function can distinguish between whether a sensor is contributive or redundant to the formation of a cover. This distinction regarding fitness helps in directing the evolutionary search. Section 5 will further examine the advantage of this fitness function.

Incidentally, the number of covers can be simply obtained by $\lfloor f(\boldsymbol{c})/|T| \rfloor$, where $f(\boldsymbol{c})$ denotes the fitness of chromosome $\boldsymbol{c}$. For instance, the chromosome in the left part of Fig. 4 has $\lfloor 6/4 \rfloor = 1$ cover.

4.3. Selection

The selection operators, including parent selection and survivor selection, follow the Darwinian principle of *survival of the fittest*. First, parent selection is ordinarily based on an alternative explanation of natural selection: fitter individuals should have a higher probability of reproducing. A common implementation of this principle is *fitness-proportionate selection*. This method is also called *roulette wheel selection* because it selects parents in a manner analogous to the spinning of a roulette wheel, in which the size of a pocket is proportional to the fitness of an individual. Its main drawbacks are its need for global fitness information and its sensitivity to the distribution of fitness in the population. The *k-tournament selection* eliminates these drawbacks by choosing the winner among k individuals that are drawn randomly from the population. The number k controls selection pressure: a higher k gives higher selection pressure.

Second, survivor selection genuinely applies the principle of survivor of the fittest. Only the fittest individuals are selected as parents for the next-generation. The methods of survivor selection can be classified according to the number of parents who compete for survival. The $(\mu + \lambda)$ survivor selection merges the parental and the offspring populations to compete for survival.

The proposed MA is not subject to selection operators: any GA selection operator can be applied to the MA. This study adopts tournament parent selection and $(\mu + \lambda)$ survivor selection in experiments because of their recognized good performance.

4.4. Crossover and mutation

The crossover of order-based chromosomes requires a special design to ensure the legality of an order, *i.e.*, containing no duplicate numbers in an order. Various crossover operators have been proposed for order-based representation. These

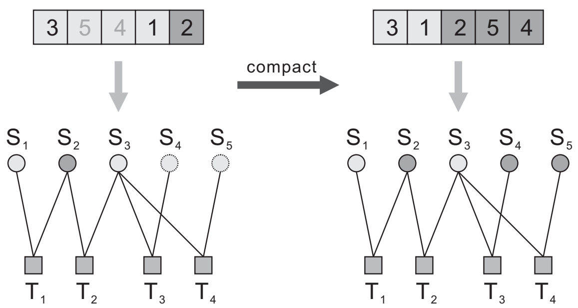

Fig. 4. Example of compact operation.

operators generally explore the relationships between the two parental orders and then use this information to generate legal orders. The two main relationships involved are mapping and adjacency. The mapping operation establishes a correspondence of genes at the same locus between parents in the crossover phase. The crossover operators using this technique include the well-known *partially mapped crossover* (PMX) [15], *order crossover* (OX) [11], and *cycle crossover* (CX) [30]. The adjacency information, however, is essential in combinatorial problems such as the traveling salesman problem (TSP). Crossover operators that adopt this information, such as the series of *edge recombinations* [28,39,41,43], are widely used to tackle the TSP, although they are applicable to other combinatorial problems.

The mutation of order-based representation also requires order legality. The bitwise and gene-wise mutation operators cannot fulfill this requirement since a legal move requires a change in at least two genes. Several mutation operators for order-based representation, *e.g.*, swap, insertion, scramble, and inversion, have been proposed to deal with the issue of order legality [12,14].

Any order-based crossover and mutation is applicable to the proposed MA. However, the crossover operators that focus on mapping relationships are preferred because the key information required for solving the SET K-COVER problem is the order of collecting sensors rather than adjacency. In the light of its virtue in preserving relative order, order crossover for order-based EAs is utilized herein. The widely used swap mutation is applied in the experiments. Additionally, an alternative definition [19] of mutation rate is employed. The original mutation rate for order-based representation is defined as the probability that a chromosome mutates once. Even the highest mutation rate 1.0 can result in only one swap in a chromosome, severely restricting the extent of mutation. To increase the extent of mutation, a Poisson-distributed random generator is adopted to determine the number of swaps needed to mutate a chromosome, where the mutation rate is given by the parameter λ in the Poisson distribution. Therefore, the number of swaps in a chromosome can exceed one to increase diversity.

4.5. Compact operator

In the proposed MA, sensor order is key to the cover collection process. A cover that contains redundant sensors inhibits from using them subsequently and therefore reduces the opportunities to form more covers. To avoid this situation, the compact operator is proposed to adjust the sensor order to increase the number of covers. This operator is applied to every offspring after mutation.

For an order-based chromosome, the compact operator checks the sensor groups successively. The redundant sensors of each cover are moved to the end of the chromosome such that they can later be collected again to form another cover. Restated, the compact operator rearranges the orders of redundant sensors to increase the number of covers. This operation continues checking the newly formed covers until all redundant sensors have been moved out. For example, the left part of Fig. 4 displays an offspring after mutation. The compact operator checks the contribution of each sensor, which is $\{3,0,0,1 \mid 2\}$ for the offspring with a single cover $C_1 = \{S_3,S_5,S_4,S_1\}$. Sensors S_5 and S_4 are made redundant from C_1; however, they are probably helpful for forming another cover. The compact operator then moves sensors S_5 and S_4 to the end of the offspring. As the right part of Fig. 4 shows, this operation helps the offspring to compact cover C_1 into $C'_1 = \{S_3,S_1\}$ and to form a new cover $C'_2 = \{S_2,S_5,S_4\}$. The contribution of each sensor becomes $\{3,1 \mid 2,1,1\}$, and the fitness increases accordingly from 6 to 8. This example illustrates the fact that the compact operator can distribute the participation of sensors among covers and can increase the number of covers.

5. Simulation results

A series of simulations was conducted to evaluate the performance of the proposed MA in extending WSN lifetime, in comparison with MCMCC [38], MCMIP [8], integer-coded GA (iGA) [23], and order-based GA. To assess the effect of the contribution fitness function, two order-based GAs were considered: oGA1 and oGA2. The former adopts the number of covers as fitness whereas the latter employs the contribution value that is defined in Section 4.2 as fitness. Table 1 lists the operators and related parameters used in these algorithms. Notably, the mutation rate for oGA1, oGA2, and MA is the parameter for the number of swaps in the Poisson-distributed random generator.

The simulation instances of WSN are generated randomly; specifically, sensors and targets are deployed at random over a 500×500 area. Each simulation setting includes 100 test instances and one run for each instance. The following subsections present and discuss the simulation results regarding the sensing range r, the number of targets $|T|$, and the number of sensors $|S|$. Here, two factors ρ_t and ρ_s are defined to characterize the test instances. Let $T(s)$ be the target set that is covered by sensor s, and let $S(t)$ be the sensor set that covers target t. The two factors are defined by

$$\rho_t = \frac{1}{|S|}\sum_{s\in S}|T(s)|, \tag{1}$$

$$\rho_s = \frac{1}{|T|}\sum_{t\in T}|S(t)|, \tag{2}$$

Table 1
Operators and parameters used in simulations.

	iGA	oGA1	oGA2	MA
Representation	Integer	Order	Order	Order
Fitness	#Covers	#Covers	Contribution	Contribution
Population size	100			
Parent selection	2-tournament selection			
Crossover	Uniform	Order	Order	Order
Crossover rate	1.0			
Mutation	Bit-flip	Swap	Swap	Swap
Mutation rate	0.01	1.0	1.0	1.0
Local enhancement	None	None	None	Compact
Survivor selection	$(\mu+\lambda)$			
Termination	1000 generations			

where $|\cdot|$ denotes the cardinality. An upper bound on the number of covers K that is used in iGA [23] is determined as

$$ub = \min_{t \in T} |S(t)|. \tag{3}$$

The value of ub gives the range of possible group numbers to which a target can be assigned, and thereby determines the problem space of a SET K-COVER problem of extending WSN lifetime. Notably, for randomly generated test instances, ρ_s and ub are related: a large ρ_s implies a large ub. The factor Δ is defined as the difference between ρ_s and ub:

$$\Delta = \rho_s - ub. \tag{4}$$

This measure represents the average number of sensors that should be excluded with respect to a single target when sensors are partitioned to maximize the number of covers. A larger value of Δ implies that the algorithm should pay more attention to picking out suitable sensors to maximize the number of covers.

Three measures are used for performance evaluation.

- Number of covers K: This number determines the maximum extension of WSN lifetime. The average and standard deviation are considered for the stochastic nature of test algorithms.
- Hit rate (HR): Since the maximum number, K, of covers in the test instances is unknown, the upper bound ub is used here as a basis for evaluating solution quality. The hit rate is defined as the percentage of runs that achieve ub covers:

$$\mathrm{HR} = \frac{\#\text{runs that achieve } ub \text{ covers}}{\#\text{runs}}, \tag{5}$$

where ub is decided by the smallest degree of target vertices in the bipartite graph of WSN.
- Running time: This time is measured on a simulation platform that uses C code on a Windows XP/Intel Core 2 Duo E6320 1.86 GHz machine.

5.1. Simulations with different sensing ranges

The purpose of the first simulation was to investigate the performance of the proposed MA for different sensing ranges. As Table 2 shows, extending the sensing range increases ρ_t, ρ_s, and ub. Notably, ρ_s is related to ub for randomly generated test instances. The simulation considers two sizes of WSN: the smaller WSN consists of 90 sensors and 10 targets, which size has also been used in [8]; the larger WSN consists of 300 sensors and 500 targets, and was simulated to examine scalability.

Table 3 compares the solution quality of the six test algorithms. The implicit exhaustive search ensures that the MCMIP always achieves the maximum number of covers; however, its exponential time complexity forbids experiments in which sensing range exceeds 300 in the smaller WSN and all sensing ranges in the larger WSN. Table 4 further presents the results of a one-tailed paired t-test of the numbers of covers obtained using the test algorithms.

On the obtained number of covers, this study explored the effects of four elements in the proposed MA, namely evolutionary scheme, order-based representation, contribution fitness, and local enhancement operator. First, the four EAs, *viz* iGA, oGA1, oGA2, and MA, outperformed MCMCC in terms of the number of covers obtained, except for iGA with $r \geqslant 300$ in both WSNs and oGA1 with $r \leqslant 400$ in the larger WSN. This superior performance of EAs over MCMCC shows the effectiveness of the evolutionary scheme in maximizing the number of covers. Second, the performance of iGA degrades as the sensing range r increases and is even worse than that of MCMCC for $r \geqslant 300$. This weakness of iGA is caused by the use of ub to indicate the range of potential values of genes, even though it may not equal the maximum number K, especially in large ρ_s WSNs. The error in estimating K incurs trivial representation space in iGA and a subsequent inefficient search for the optimal solution. The order-based representation used in oGA1, oGA2, and MA, nevertheless, overcomes this weakness by removing the need for ub, which is beneficial for large sensing ranges. Third, the number of covers achieved by oGA2 is larger than or at least comparable to that achieved by oGA1. This outcome confirms that the contribution fitness improves the performance of

Table 2
Average (Avg.) and standard deviation (S.D.) of ρ_t, ρ_s, ub, and Δ over 100 problem instances for different sensing ranges r with $|S|$ sensors and $|T|$ targets.

$\lvert S\rvert$	$\lvert T\rvert$	r	ρ_t		ρ_s		ub		Δ
			Avg.	S.D.	Avg.	S.D.	Avg.	S.D.	Avg.
90	10	100	1.04	0.14	9.32	1.26	4.13	1.61	5.19
		150	2.15	0.26	19.35	2.36	9.84	2.64	9.51
		200	3.44	0.32	30.94	2.90	16.77	4.47	14.17
		250	4.79	0.44	43.07	3.98	25.18	4.82	17.89
		300	6.12	0.54	55.09	4.85	34.78	6.13	20.31
		350	7.35	0.48	66.16	4.32	45.54	6.84	20.62
		400	8.43	0.37	75.89	3.30	58.31	6.55	17.58
		450	9.26	0.24	83.31	2.19	70.37	5.98	12.94
		500	9.74	0.12	87.64	1.07	80.53	4.20	7.11
300	500	100	52.52	0.93	31.51	0.56	8.69	2.17	22.82
		150	107.55	2.20	64.53	1.32	20.20	3.35	44.33
		200	172.69	3.75	103.61	2.25	36.01	4.38	67.60
		250	241.01	4.72	144.61	2.83	58.16	5.04	86.45
		300	309.15	5.34	185.49	3.20	84.71	5.81	100.78
		350	371.62	5.18	222.97	3.11	115.57	6.85	107.40
		400	425.36	4.34	255.22	2.60	151.43	6.73	103.79
		450	464.60	2.82	278.76	1.69	193.62	6.38	85.14
		500	487.67	1.20	292.60	0.72	238.08	6.24	54.52

Table 3
Average (Avg.), standard deviation (S.D.), and hit rate (HR) of the number of covers obtained for different sensing ranges (r) with $|S|$ sensors and $|T|$ targets. Boldface denotes the best result among the six test algorithms.

$\lvert S\rvert$	$\lvert T\rvert$	r	ub	MCMIP	MCMCC		iGA		oGA1		oGA2		MA	
				Avg. (S.D.)	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR
90	10	100	4.13	**4.13** (1.61)	**4.13** (1.61)	**1.00**	**4.13** (1.61)	**1.00**	4.12 (1.60)	0.99	**4.13** (1.61)	**1.00**	**4.13** (1.61)	**1.00**
		150	9.84	**9.84** (2.64)	9.78 (2.58)	0.95	**9.84** (2.64)	**1.00**	9.79 (2.54)	0.95	**9.84** (2.64)	**1.00**	**9.84** (2.64)	**1.00**
		200	16.77	**16.77** (4.47)	16.41 (4.15)	0.75	**16.77** (4.47)	**1.00**	16.67 (4.34)	0.92	**16.77** (4.47)	**1.00**	**16.77** (4.47)	**1.00**
		250	25.18	**25.18** (4.82)	24.55 (4.43)	0.66	**25.18** (4.82)	**1.00**	25.15 (4.79)	0.91	**25.18** (4.82)	**1.00**	**25.18** (4.82)	**1.00**
		300	34.78	–	33.97 (5.62)	0.61	34.75 (6.09)	0.97	34.64 (5.99)	0.89	34.75 (6.09)	**0.98**	**34.76** (6.10)	**0.98**
		350	45.54	–	44.67 (6.16)	0.64	45.12 (6.33)	**0.89**	45.20 (6.40)	0.85	45.27 (6.48)	0.88	**45.28** (6.49)	0.87
		400	58.31	–	56.70 (5.88)	0.47	56.64 (5.74)	0.66	57.26 (5.89)	0.62	57.29 (5.87)	0.62	**57.39** (5.89)	**0.67**
		450	70.37	–	69.12 (5.82)	0.55	68.54 (6.12)	**0.66**	69.37 (5.80)	0.64	69.36 (5.81)	0.64	**69.39** (5.81)	**0.66**
		500	80.53	–	80.03 (4.27)	0.73	79.75 (4.55)	0.75	**80.06** (4.27)	**0.76**	**80.06** (4.27)	**0.76**	**80.06** (4.27)	**0.76**
300	500	100	8.69	–	8.28 (1.93)	0.67	8.65 (2.11)	0.97	7.43 (1.08)	0.40	8.67 (2.13)	0.98	**8.69** (2.17)	**1.00**
		150	20.20	–	19.64 (3.01)	0.71	20.07 (3.19)	0.94	18.43 (1.98)	0.37	20.16 (3.31)	0.96	**20.20** (3.35)	**1.00**
		200	36.01	–	35.59 (4.15)	0.76	35.87 (4.21)	0.92	34.32 (3.20)	0.34	35.96 (4.29)	0.96	**36.00** (4.36)	**0.99**
		250	58.16	–	56.46 (4.42)	0.37	57.46 (4.32)	0.73	55.90 (3.59)	0.28	57.87 (4.76)	0.78	**58.15** (5.03)	**0.99**
		300	84.71	–	80.55 (4.01)	0.24	78.85 (4.04)	0.32	80.37 (3.68)	0.13	83.26 (4.75)	0.42	**84.59** (5.66)	**0.89**
		350	115.57	–	110.34 (4.40)	0.23	100.07 (5.57)	0.04	109.50 (4.07)	0.05	112.44 (4.74)	0.24	**114.92** (6.02)	**0.69**
		400	151.43	–	142.51 (4.09)	0.07	125.18 (7.02)	0.01	142.75 (4.37)	0.02	144.75 (4.53)	0.06	**148.74** (4.99)	**0.26**
		450	193.62	–	174.71 (4.41)	0.00	149.43 (8.04)	0.00	175.76 (4.13)	0.00	176.58 (4.11)	0.00	**178.66** (4.00)	**0.01**
		500	238.08	–	207.86 (5.29)	0.00	179.30 (8.73)	0.00	208.93 (5.26)	0.00	208.98 (5.35)	0.00	**209.20** (5.28)	0.00

order-based GA. Finally, the compact operator enables the MA to improve significantly oGA2 in terms of the number of covers and to yield the same number of covers as obtained using MCMIP. This demonstrates that the MA is highly capable of

Table 4
Results of one-tailed paired *t*-test of the numbers of covers obtained from *X* and *Y* algorithms (denoted by *X* vs. *Y*) for different sensing ranges *r* with $|S|$ sensors and $|T|$ targets. Positive *p*-values indicate that *X* is superior to *Y*, and vice versa. Boldface denotes that *X* is significantly better than *Y*. N/A indicates that *X* and *Y* yield identical results in all test instances.

$\|S\|$	$\|T\|$	*r*	iGA vs. MCMCC	oGA1 vs. MCMCC	oGA1 vs. iGA	oGA2 vs. MCMCC	oGA2 vs. iGA	oGA2 vs. oGA1	MA vs. MCMCC	MA vs. iGA	MA vs. oGA1	MA vs. oGA2
90	100	100	N/A	−1.60E−01	−1.60E−01	N/A	N/A	1.60E−01	N/A	N/A	1.60E−01	N/A
		150	**1.66E−02**	3.70E−01	−1.23E−02	**1.66E−02**	N/A	**1.23E−02**	**1.66E−02**	N/A	**1.23E−02**	N/A
		200	**3.00E−07**	**1.61E−05**	−3.45E−03	**3.00E−07**	N/A	**3.45E−03**	**3.00E−07**	N/A	**3.45E−03**	N/A
		250	**2.40E−07**	**1.51E−07**	−4.16E−02	**2.40E−07**	N/A	**4.16E−02**	**2.40E−07**	N/A	**4.16E−02**	N/A
		300	**7.20E−10**	**8.84E−09**	−3.50E−03	**1.41E−09**	5.00E−01	**3.50E−03**	**7.15E−10**	1.60E−01	**1.15E−03**	2.83E−01
		350	**8.29E−05**	**9.93E−08**	9.79E−02	**5.09E−08**	**1.07E−02**	**3.52E−02**	**1.15E−08**	**1.29E−02**	**5.35E−03**	3.28E−01
		400	−3.73E−01	**1.17E−07**	**8.14E−05**	**7.93E−08**	**2.89E−05**	2.35E−01	**8.96E−09**	**2.30E−06**	**3.00E−04**	**1.71E−03**
		450	−5.28E−04	**1.68E−05**	**6.35E−07**	**1.70E−05**	**8.47E−07**	−2.83E−01	**5.63E−06**	**2.95E−07**	**7.92E−02**	**4.16E−02**
		500	−9.46E−04	**4.16E−02**	**2.49E−04**	**4.16E−02**	**2.49E−04**	N/A	**4.16E−02**	**2.49E−04**	N/A	N/A
300	500	100	**9.87E−08**	−7.25E−12	−3.50E−15	**8.15E−09**	2.08E−01	**1.90E−15**	**4.52E−09**	**5.14E−02**	**4.69E−15**	**7.92E−02**
		150	**3.09E−05**	−4.33E−11	−2.71E−15	**8.22E−07**	7.52E−02	**1.51E−15**	**4.35E−07**	**2.09E−02**	**2.25E−15**	**2.25E−02**
		200	**1.53E−03**	−1.21E−11	−7.52E−16	**4.18E−06**	**4.16E−02**	**1.09E−15**	**1.17E−06**	**5.81E−03**	**1.94E−15**	**5.14E−02**
		250	**2.33E−07**	−2.28E−03	−3.80E−16	**2.63E−12**	**9.14E−04**	**5.63E−20**	**1.24E−14**	**2.77E−05**	**1.47E−19**	**1.39E−05**
		300	−3.96E−04	−2.49E−01	**8.53E−04**	**1.68E−14**	**6.92E−13**	**3.59E−24**	**5.76E−19**	**3.75E−14**	**3.06E−27**	**3.17E−13**
		350	−7.16E−25	−2.12E−03	**4.67E−23**	**1.04E−09**	**3.82E−27**	**3.45E−23**	**5.69E−19**	**5.42E−28**	**6.03E−32**	**2.25E−21**
		400	−5.53E−43	1.87E−01	**1.01E−41**	**1.63E−12**	**4.90E−44**	**1.47E−13**	**3.87E−35**	**3.19E−46**	**8.13E−42**	**9.82E−32**
		450	−1.45E−63	**1.48E−07**	**1.04E−62**	**1.54E−19**	**1.06E−64**	**9.89E−08**	**2.64E−44**	**4.43E−69**	**3.65E−38**	**3.44E−32**
		500	−2.44E−74	**6.84E−18**	**3.23E−74**	**1.19E−19**	**2.34E−75**	2.39E−01	**3.32E−26**	**5.46E−76**	**1.66E−06**	**3.32E−06**

achieving global optimal solutions. The MA substantially outperforms all other test algorithms except MCMIP in terms of the number of covers obtained. The advantage of oGA2 and MA over the other test algorithms increases with the WSN size.

In terms of hit rate, the comparative results somewhat differ between the smaller and larger WSNs. In the smaller WSN, the MA, iGA, and oGA2 perform best, followed by oGA1 with only a slightly poorer performance; MCMCC yields relatively low hit rates, especially for $200 \leqslant r \leqslant 450$. Fig. 5, which plots the variation of hit rate against sensing range, indicates that MA and iGA yield similar hit rates, while MCMCC has a low hit rate for most of the test sensing ranges. Notably, the low hit rates for $r \geqslant 400$ do not imply poor performance of the test algorithms. Rather, they reflect the significant difference between the optimal number *K* and the upper bound *ub*. Hence, given the definition of hit rate, the algorithm cannot have a high hit rate, even though it can achieve the maximum number of covers. In the larger WSN, the compact operator in the MA is very effective in increasing the hit rate, which fact is reflected by the apparent discrepancy between the HR of the MA and those of all other test algorithms. Moreover, oGA2 outperforms oGA1 and iGA. This outcome verifies the utility of the contribution fitness in increasing the number of covers obtained. The oGA1 performs poorly in the larger WSN, revealing the need for order-based representation to improve the fitness function or perform local enhancement, as in the proposed MA.

Table 5 lists the running time of the six test algorithms for different sensing ranges. For all test algorithms, the running time generally increases with sensing range. Owing to its implicit exhaustive search, MCMIP requires much longer running time than the other algorithms. MCMCC has the shortest running time. The EAs require somewhat more time than MCMCC, but much less time than MCMIP. The difference between the running time of oGA1 and that of oGA2 reflects the effect of the fitness function. At small sensing ranges ($r \leqslant 200$), oGA2 achieves shorter running time by using contribution fitness, and yet, as $r \geqslant 250$, the contribution fitness of oGA2 results in longer running time than oGA1. Concerning the cost of local

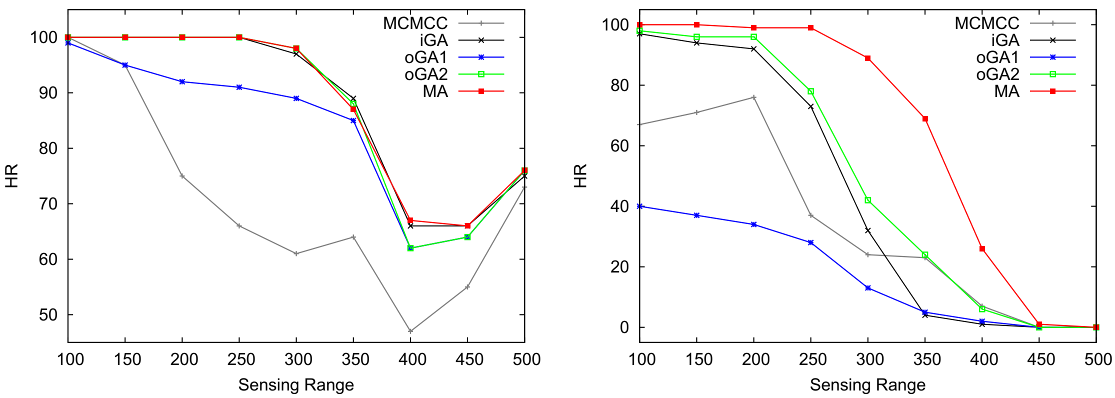

Fig. 5. Hit rates of MCMCC, iGA, oGA1, oGA2, and MA for different sensing ranges (*r*) with $|S| = 90$ and $|T| = 10$ (left) and with $|S| = 300$ and $|T| = 500$ (right).

Table 5
Running time of MCMIP, MCMCC, iGA, oGA1, oGA2, and MA for different sensing ranges (r) with $|S|$ sensors and $|T|$ targets.

r	$\|S\| = 90$, $\|T\| = 10$						$\|S\| = 300$, $\|T\| = 500$				
	MCMIP	MCMCC	iGA	oGA1	oGA2	MA	MCMCC	iGA	oGA1	oGA2	MA
100	0.33	0.00	2.13	1.69	1.47	2.16	2.20	32.08	158.99	115.51	170.39
150	13.99	0.00	3.47	1.87	1.74	2.46	8.09	58.46	166.48	165.66	225.80
200	208.17	0.01	4.49	2.01	1.94	2.54	16.20	90.23	185.26	212.93	236.75
250	1174.10	0.02	2.39	2.13	2.11	2.57	41.45	126.35	202.33	250.21	262.00
300	–	0.03	6.40	2.17	2.25	2.53	68.35	164.74	210.35	269.64	296.61
350	–	0.05	7.09	2.21	2.45	2.63	89.33	203.39	213.84	280.41	284.79
400	–	0.05	8.08	2.23	2.60	2.75	103.52	242.26	212.21	281.68	291.35
450	–	0.05	8.66	2.23	2.73	2.88	115.62	305.61	209.08	281.97	293.52
500	–	0.06	9.96	2.25	2.86	3.02	112.03	374.68	212.70	295.91	300.45

enhancement, *i.e.*, the compact operator, the MA requires more running time than oGA2 does. However, the increase in running time associated with compact operation gradually declines as the sensing range increases. The iGA has longer running time than MCMCC, order-based GAs, and the MA. Increasing the sensing range has a strongly negative effect on iGA running time since its search space depends on *ub*, which increases with the sensing range r.

5.2. Simulations with different numbers of targets

The second simulation tested the performance of the MA on WSNs with different numbers of targets. This simulation considered two settings for the number of sensors and sensing range of each. The first involved a WSN comprising 90 sensors with $r = 250$, which was used in [8]. The second involved $|S| = 300$ sensors with a sensing range of $r = 400$ to check the scalability of the test algorithms. Table 6 lists the characteristics of these problem instances for different numbers of targets, and demonstrates that adding targets in the WSN does not affect ρ_s, but does reduce *ub* and increase ρ_t and Δ. Increasing the sensing range from 250 to 400 also increases ρ_s. The effects of these changes are discussed below.

Table 7 compares the number of covers and hit rate obtained for different numbers of targets. The table shows only a few results for MCMIP because, as indicated by Table 9, its high cost limits the number of targets in the experiments. For the smaller sensing range ($r = 250$), the results in Table 7 demonstrate that iGA, oGA2, and MA perform very well for all tested numbers of targets. Notably, MA can achieve a number of covers that equals the upper bound *ub*. The t-test results in Table 8 reveal no significant variation among the results obtained using iGA, oGA2, and MA. Additionally, oGA1 generates significantly fewer covers than the above three algorithms. The difference between the simulation results for oGA1 and oGA2, moreover, validates the advantage of using contribution fitness. The four EAs (iGA, oGA1, oGA2, and MA) all outperform

Table 6
Average (Avg.) and standard deviation (S.D.) of ρ_t, ρ_s, *ub*, and Δ over 100 problem instances for different numbers $|T|$ of targets to be covered by $|S|$ sensors with sensing range r.

$\|S\|$	r	$\|T\|$	ρ_t		ρ_s		*ub*		Δ
			Avg.	S.D.	Avg.	S.D.	Avg.	S.D.	Avg.
90	250	10	4.79	0.44	43.07	3.98	25.18	4.82	17.89
		20	9.66	0.66	43.49	2.96	21.31	3.77	22.18
		30	14.52	0.87	43.56	2.60	21.13	3.91	22.43
		40	19.33	1.05	43.50	2.36	19.59	3.71	23.91
		50	24.12	1.19	43.41	2.15	19.41	3.10	24.00
		75	36.17	1.74	43.40	2.09	18.58	2.95	24.82
		100	48.35	2.06	43.52	1.86	17.85	2.83	25.67
		150	72.42	2.57	43.45	1.54	17.26	2.67	26.19
		200	97.11	3.30	43.70	1.48	16.55	2.61	27.15
		250	120.79	4.14	43.49	1.49	16.20	2.38	27.29
		500	241.52	7.94	43.47	1.43	15.54	2.26	27.93
300	400	10	8.45	0.38	253.45	11.38	193.67	21.10	59.78
		20	16.98	0.57	254.66	8.48	181.94	17.13	72.72
		30	25.59	0.73	255.88	7.34	178.43	17.01	77.45
		40	34.05	0.74	255.38	5.55	173.02	15.13	82.36
		50	42.51	0.94	255.09	5.62	167.79	13.82	87.30
		75	63.74	1.19	254.94	4.76	167.39	10.90	87.55
		100	84.73	1.32	254.20	3.97	162.87	9.58	91.33
		150	127.71	1.88	255.41	3.76	159.84	8.00	95.57
		200	170.14	2.25	255.21	3.38	157.96	9.09	97.25
		250	212.60	2.47	255.12	2.97	155.91	6.92	99.21
		500	425.36	4.34	255.22	2.60	151.43	6.73	103.79

Table 7
Average (Avg.), standard deviation (S.D.), and hit rate (HR) of the number of covers obtained for different numbers of targets (|*T*|) with |*S*| sensors and sensing range *r*. Boldface denotes the best result among the six test algorithms.

|*S*|	*r*	|*T*|	*ub*	MCMIP	MCMCC		iGA		oGA1		oGA2		MA	
				Avg. (S.D.)	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR
90	250	10	25.18	**25.18** (4.82)	24.55 (4.43)	0.66	**25.18** (4.82)	**1.00**	25.15 (4.79)	0.91	**25.18** (4.82)	**1.00**	**25.18** (4.82)	**1.00**
		20	21.31	**21.31** (3.77)	20.80 (3.45)	0.72	21.30 (3.76)	0.99	21.23 (3.69)	0.95	**21.31** (3.77)	**1.00**	**21.31** (3.77)	**1.00**
		30	21.13	**21.13** (3.91)	20.50 (3.39)	0.63	21.04 (3.73)	0.95	20.87 (3.59)	0.82	21.09 (3.85)	0.96	**21.13** (3.91)	**1.00**
		40	19.59	–	18.98 (3.22)	0.66	19.58 (3.69)	0.99	19.45 (3.56)	0.87	19.58 (3.69)	0.99	**19.59** (3.71)	**1.00**
		50	19.41	–	18.84 (2.68)	0.66	19.40 (3.09)	0.99	19.29 (2.94)	0.90	**19.41** (3.10)	**1.00**	**19.41** (3.10)	**1.00**
		75	18.58	–	18.04 (2.56)	0.65	18.52 (2.87)	0.96	18.40 (2.73)	0.88	18.57 (2.93)	0.99	**18.58** (2.95)	**1.00**
		100	17.85	–	17.40 (2.44)	0.69	17.78 (2.71)	0.95	17.67 (2.59)	0.87	**17.85** (2.83)	**1.00**	**17.85** (2.83)	**1.00**
		150	17.26	–	16.73 (2.32)	0.65	17.23 (2.64)	0.97	17.05 (2.46)	0.81	**17.26** (2.67)	**1.00**	**17.26** (2.67)	**1.00**
		200	16.55	–	16.03 (2.35)	0.67	16.52 (2.56)	0.99	16.33 (2.37)	0.82	16.54 (2.60)	0.99	**16.55** (2.61)	**1.00**
		250	16.20	–	15.82 (2.24)	0.71	**16.20** (2.38)	**1.00**	16.11 (2.28)	0.91	**16.20** (2.38)	**1.00**	**16.20** (2.38)	**1.00**
		500	15.54	–	15.17 (2.15)	0.71	15.51 (2.24)	0.97	15.42 (2.15)	0.89	**15.54** (2.26)	**1.00**	**15.54** (2.26)	**1.00**
300	400	10	193.67	–	189.23 (19.39)	0.44	188.16 (18.81)	0.56	189.76 (18.98)	0.46	189.93 (19.07)	0.43	**190.93** (19.34)	**0.59**
		20	181.94	–	176.72 (13.93)	0.41	173.40 (13.02)	0.43	176.62 (13.71)	0.38	177.15 (13.88)	0.49	**178.54** (13.99)	**0.55**
		30	178.43	–	171.56 (13.27)	0.26	167.25 (11.46)	0.26	171.79 (12.98)	0.20	172.38 (13.03)	0.24	**174.49** (13.26)	**0.47**
		40	173.02	–	165.69 (11.81)	0.21	160.38 (10.12)	0.11	166.08 (11.70)	0.18	167.07 (11.79)	0.20	**169.47** (12.04)	**0.38**
		50	167.79	–	160.78 (10.19)	0.22	154.94 (8.01)	0.15	160.98 (10.29)	0.13	162.09 (10.33)	0.18	**164.55** (10.72)	**0.37**
		75	167.39	–	158.77 (7.91)	0.11	148.73 (9.35)	0.09	158.71 (7.72)	0.07	160.08 (7.63)	0.10	**163.35** (8.20)	**0.29**
		100	162.87	–	154.59 (7.36)	0.11	144.30 (8.17)	0.03	154.83 (7.19)	0.03	156.19 (7.08)	0.08	**159.78** (7.42)	**0.35**
		150	159.84	–	150.54 (5.78)	0.12	137.20 (8.40)	0.04	151.05 (5.39)	0.05	152.45 (5.79)	0.08	**156.46** (6.38)	**0.18**
		200	157.96	–	148.98 (6.04)	0.09	133.93 (6.67)	0.02	148.65 (6.15)	0.02	150.55 (6.19)	0.09	**154.39** (6.76)	**0.25**
		250	155.91	–	147.01 (4.93)	0.10	131.10 (7.45)	0.01	147.01 (4.73)	0.02	148.42 (4.76)	0.03	**152.80** (5.15)	**0.24**
		500	151.43	–	142.51 (4.09)	0.07	125.18 (7.02)	0.01	142.75 (4.37)	0.02	144.75 (4.53)	0.06	**148.74** (4.99)	**0.26**

MCMCC in terms of the number of covers obtained and the hit rate. This superiority of EAs reconfirms the benefit of the evolutionary scheme in extending WSN lifetime with small sensing ranges.

As the sensing range r increases to 400, the hit rate of the test algorithms apparently decreases, and this decline is further intensified by the increase of targets. In the problem instances considered herein, the reduction in the hit rate corresponds to the increase in Δ, implying that more sensors should be excluded when sensors are arranged to cover a certain target. In addition, overestimating the maximum number K using ub still impairs the performance of iGA. The order-based EAs, including oGA1, oGA2, and MA, resolve this issue by eliminating the need for an upper bound on or an assumption about the maximum number of covers. Further, the proposed MA can achieve significantly more covers than the other test algorithms can, and this significance increases with the number of targets. Fig. 6 further shows that the MA achieves the highest hit rates for all tested numbers of targets. These experimental results demonstrate the excellent performance of the proposed MA in generating covers for extending the WSN lifetime.

Table 9 compares the running time of the six test algorithms in this simulation. The required running time of all test algorithms increases with the number of targets. The MCMCC is the fastest of all the test algorithms; however, its unsatisfactory solution quality detracts greatly from its efficiency. The comparison of test EAs indicates that iGA requires less running time than oGA1, oGA2, and MA for problem instances with a small sensing range (r = 250). Conversely, for problem instances with r = 400, iGA has longer running time than oGA1, oGA2, and even the MA, for all numbers of targets except $|T|$ = 500. This

Table 8
Results of one-tailed paired t-test of the numbers of covers obtained from X and Y algorithms (denoted by X vs. Y) for different numbers $|T|$ of targets with $|S|$ sensors and sensing range r. Positive p-values indicate that X is superior to Y, and vice versa. Boldface denotes that X is significantly better than Y. N/A indicates that X and Y have identical results in all test instances.

$\|S\|$	r	$\|T\|$	iGA vs. MCMCC	oGA1 vs. MCMCC	oGA1 vs. iGA	oGA2 vs. MCMCC	oGA2 vs. iGA	oGA2 vs. oGA1	MA vs. MCMCC	MA vs. iGA	MA vs. oGA1	MA vs. oGA2
90	250	10	**2.40E−07**	**1.51E−07**	−4.16E−02	**2.40E−07**	N/A	**4.16E−02**	**2.40E−07**	N/A	**4.16E−02**	N/A
		20	**6.40E−07**	**1.48E−06**	−2.59E−02	**6.57E−07**	1.60E−01	**1.59E−02**	**6.57E−07**	1.60E−01	**1.59E−02**	N/A
		30	**1.83E−09**	**1.52E−06**	−1.79E−03	**2.63E−09**	1.39E−01	**5.03E−05**	**2.37E−09**	**4.74E−02**	**3.80E−05**	**2.25E−02**
		40	**2.42E−08**	**1.65E−07**	−1.06E−04	**2.42E−08**	N/A	**1.06E−04**	**4.11E−08**	1.60E−01	**1.67E−04**	1.60E−01
		50	**5.49E−08**	**2.29E−07**	−3.50E−03	**9.55E−08**	1.60E−01	**2.08E−03**	**9.55E−08**	1.60E−01	**2.08E−03**	N/A
		75	**2.27E−08**	**2.51E−06**	−3.33E−03	**7.30E−09**	**2.92E−02**	**4.20E−04**	**6.23E−09**	**2.87E−02**	**5.92E−04**	1.60E−01
		100	**1.60E−07**	**1.51E−05**	−3.52E−04	**5.09E−08**	**1.71E−02**	**3.96E−04**	**5.09E−08**	**1.71E−02**	**3.96E−04**	N/A
		150	**1.32E−07**	**2.30E−05**	−3.72E−05	**2.16E−08**	**4.16E−02**	**6.14E−06**	**2.16E−08**	**4.16E−02**	**6.14E−06**	N/A
		200	**1.97E−08**	**3.15E−05**	−8.49E−05	**3.05E−08**	2.65E−01	**1.39E−05**	**3.60E−08**	1.60E−01	**1.57E−05**	1.60E−01
		250	**2.60E−07**	**8.60E−05**	−1.15E−03	**2.60E−07**	N/A	**1.15E−03**	**2.60E−07**	N/A	**1.15E−03**	N/A
		500	**1.80E−07**	**4.19E−05**	−5.91E−03	**2.89E−08**	**4.16E−02**	**5.38E−04**	**2.89E−08**	**4.16E−02**	**5.38E−04**	N/A
300	400	10	−1.48E−02	**1.10E−02**	**1.36E−04**	**2.34E−03**	**4.57E−05**	9.49E−02	**8.12E−10**	**3.30E−08**	**1.76E−11**	**6.50E−13**
		20	−8.46E−08	−2.50E−01	**1.56E−07**	**5.35E−03**	**1.03E−08**	**1.85E−05**	**1.98E−13**	**2.90E−12**	**3.92E−16**	**1.50E−13**
		30	−1.71E−08	1.29E−01	**2.97E−09**	**4.19E−04**	**4.14E−11**	**2.63E−04**	**2.39E−17**	**7.90E−16**	**4.04E−21**	**7.38E−19**
		40	−4.71E−12	**4.90E−02**	**9.93E−14**	**8.96E−09**	**1.07E−15**	**1.20E−07**	**3.91E−20**	**6.85E−21**	**5.47E−23**	**1.03E−21**
		50	−2.75E−13	2.32E−01	**5.32E−14**	**5.29E−07**	**4.00E−17**	**2.36E−09**	**4.37E−20**	**3.65E−21**	**3.50E−26**	**3.96E−22**
		75	−2.12E−20	−4.05E−01	**3.81E−20**	**5.88E−06**	**1.85E−22**	**3.13E−09**	**1.23E−28**	**3.21E−27**	**1.43E−34**	**1.25E−28**
		100	−4.95E−25	1.37E−01	**1.51E−23**	**2.79E−10**	**7.08E−27**	**1.80E−10**	**7.78E−31**	**7.94E−32**	**5.27E−40**	**1.22E−29**
		150	−6.59E−32	−2.56E−02	**1.13E−31**	**1.09E−09**	**1.21E−33**	**1.12E−09**	**6.09E−31**	**9.90E−36**	**6.83E−38**	**3.72E−28**
		200	−6.72E−36	−9.52E−02	**8.80E−34**	**2.83E−10**	**1.11E−38**	**2.00E−14**	**5.82E−33**	**2.36E−41**	**5.89E−40**	**4.95E−30**
		250	−1.32E−35	5.00E−01	**1.77E−34**	**8.59E−07**	**2.21E−37**	**4.09E−10**	**2.22E−32**	**4.18E−41**	**2.59E−41**	**7.30E−32**
		500	−5.53E−43	1.87E−01	**1.01E−41**	**1.63E−12**	**4.90E−44**	**1.47E−13**	**3.87E−35**	**3.19E−46**	**8.13E−42**	**9.82E−32**

outcome reveals that overestimation of K by upper bound in iGA degrades not only solution quality but also running time. The results in Table 9 also demonstrate that using contribution fitness or the compact operator increases running time. However, the merits of these two elements, *i.e.*, higher number of covers obtained and higher hit rate, still outweigh their slightly increased cost in running time.

5.3. Simulations with different numbers of sensors

The third simulation compared the performance of the MA across different numbers of sensors. This simulation used two WSN settings: one with ten targets and a sensing range of 250 and the other with 500 targets and a sensing range of 400. Similarly, the former was used in [8] and the latter was applied to examine scalability. Table 10 demonstrates, although ρ_t is unchanged, ρ_s, ub, and Δ increase with the number of sensors $|S|$. Extending the sensing range from 250 to 400 further increases ρ_s. Increasing the number of targets from 10 to 500 considerably augments ρ_t.

Table 11 shows that the EAs outperform MCMCC in terms of the obtained number of covers in all except the two cases of $|S| \geqslant 200$ with $|T| = 500$ and $r = 400$, in which iGA is inferior to MCMCC. These simulation results verify not only the effectiveness of the evolutionary scheme but also the advantage of the order-based representation over the integer

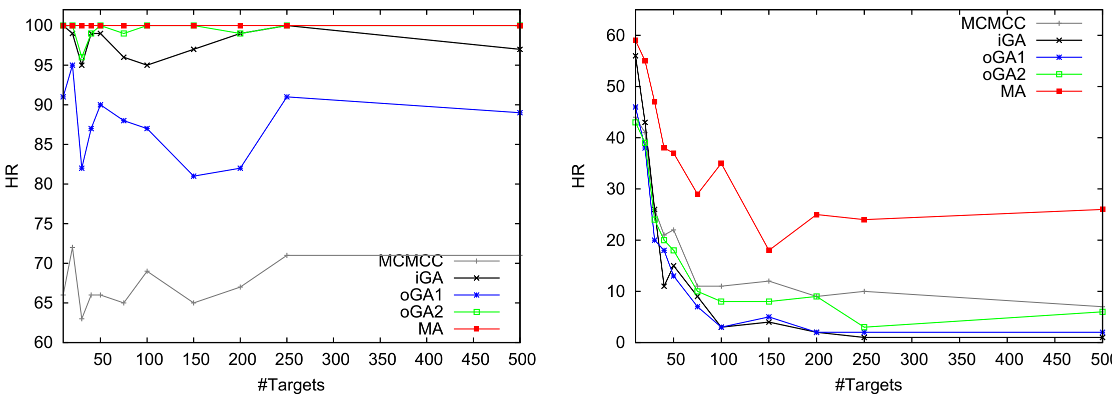

Fig. 6. Hit rates of MCMCC, iGA, oGA1, oGA2, and MA for different numbers of targets ($|T|$) with $|S| = 90$ and $r = 250$ (left) and with $|S| = 300$ and $r = 400$ (right).

Table 9
Running time of MCMIP, MCMCC, iGA, oGA1, oGA2, and MA for different numbers of targets ($|T|$) with $|S|$ sensors and sensing range r.

$\lvert T\rvert$	$\lvert S\rvert = 90$, $r = 250$						$\lvert S\rvert = 300$, $r = 400$				
	MCMIP	MCMCC	iGA	oGA1	oGA2	MA	MCMCC	iGA	oGA1	oGA2	MA
10	1234.63	0.02	2.39	2.13	2.11	2.54	1.85	53.41	7.04	8.44	8.82
20	1729.11	0.05	2.65	3.36	3.43	4.05	4.10	62.54	11.55	14.18	14.47
30	18024.71	0.08	3.01	4.60	4.87	5.55	6.35	69.43	16.45	19.96	20.23
40		0.11	3.30	5.75	6.22	7.26	8.27	74.28	20.20	25.58	25.95
50		0.14	3.64	7.01	7.73	8.72	10.45	77.61	24.44	30.89	31.56
75		0.20	9.25	9.94	11.46	12.69	15.59	87.46	35.36	45.44	46.09
100		0.27	11.24	12.80	15.17	16.79	20.42	100.92	46.00	58.95	60.27
150		0.42	14.96	18.61	22.62	25.90	31.51	134.72	66.82	86.39	89.19
200		0.51	17.03	24.05	29.84	33.86	41.05	143.55	87.92	114.26	117.67
250		0.63	20.65	34.93	37.31	41.60	52.47	149.51	110.67	141.98	144.75
500		1.24	36.60	63.20	76.96	84.77	103.52	242.26	212.21	281.68	291.35

Table 10
Average (Avg.) and standard deviation (S.D.) of ρ_t, ρ_s, ub, and Δ over 100 problem instances for different numbers $|S|$ of sensors with sensing range r to cover $|T|$ targets.

$\lvert T\rvert$	r	$\lvert S\rvert$	ρ_t		ρ_s		ub		Δ
			Avg.	S.D.	Avg.	S.D.	Avg.	S.D.	Avg.
10	250	90	4.79	0.44	43.07	3.98	25.18	4.82	17.89
		100	4.86	0.46	48.56	4.57	28.76	6.75	19.80
		200	4.84	0.47	96.76	9.41	57.98	11.89	38.78
		300	4.86	0.42	145.83	12.73	88.92	14.86	56.91
500	400	90	426.24	6.81	76.72	1.23	43.88	3.49	32.84
		100	426.13	6.73	85.23	1.35	48.63	3.47	36.60
		200	424.34	5.34	169.74	2.14	101.08	5.50	68.66
		300	425.36	4.34	255.22	2.60	151.43	6.73	103.79

Table 11
Average (Avg.), standard deviation (S.D.), and hit rate (HR) of the number of covers obtained for different numbers of sensors ($|S|$) with $|T|$ targets and sensing range r. Boldface denotes the best result among the six test algorithms.

$\lvert T\rvert$	r	$\lvert S\rvert$	ub	MCMIP	MCMCC		iGA		oGA1		oGA2		MA	
				Avg. (S.D.)	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR	Avg. (S.D.)	HR
10	250	90	25.18	**25.18** (6.01)	24.55 (4.43)	0.66	**25.18** (4.82)	**1.00**	25.15 (4.79)	0.91	**25.18** (4.82)	**1.00**	**25.18** (4.82)	**1.00**
		100	28.76	**28.76** (6.75)	27.94 (6.03)	0.66	28.73 (6.69)	0.97	28.57 (6.56)	0.88	28.75 (6.74)	0.99	**28.76** (6.75)	**1.00**
		200	57.98	–	55.86 (9.98)	0.57	57.87 (11.73)	0.93	57.27 (11.14)	0.73	57.78 (11.64)	0.89	**57.97** (11.88)	**0.99**
		300	88.92	–	85.78 (13.10)	0.47	88.34 (14.19)	0.85	87.57 (13.57)	0.64	88.29 (14.12)	0.78	**88.85** (14.74)	**0.96**
500	400	90	43.88	–	42.11 (2.47)	0.33	42.94 (2.68)	0.64	42.68 (2.64)	0.41	43.13 (2.87)	0.63	**43.57** (3.17)	**0.80**
		100	48.63	–	46.87 (2.61)	0.35	47.63 (2.80)	0.61	47.31 (2.68)	0.38	47.92 (2.89)	0.59	**48.28** (3.15)	**0.77**
		200	101.08	–	95.18 (3.98)	0.07	89.32 (4.81)	0.05	95.81 (3.88)	0.06	97.40 (3.77)	0.17	**99.57** (4.35)	**0.43**
		300	151.43	–	142.51 (4.09)	0.07	125.18 (7.02)	0.01	142.75 (4.37)	0.02	144.75 (4.53)	0.06	**148.74** (4.99)	**0.26**

representation. The statistical test results in Table 12 further indicate that the superiority of oGA2 and MA over oGA1 and oGA2 grows with the number of sensors. As the number of sensors in WSN increases, the effects of the contribution fitness adopted in oGA2 and the compact operator employed in MA become more significant.

Fig. 7 displays how hit rate varies with the number of sensors for all test algorithms except MCMIP. The hit rate declined as more sensors were added to WSNs. As mentioned above, this reduced hit rate does not necessarily imply a deteriorated algorithmic performance, but it does reflect the discrepancy between ub and the maximal number of covers K. The figure consistently demonstrates that MA > oGA2 > MCMCC. Similar to the results for the number of covers, iGA had high hit rates for $|T| = 10$ with $r = 250$ but very low hit rates for $|T| = 500$ with $r = 400$. However, MCMCC outperformed oGA1 and iGA for $|S| \geqslant 200$ in the latter case.

Table 12
Results of one-tailed paired *t*-test of the numbers of covers obtained from *X* and *Y* algorithms (denoted by *X* vs. *Y*) for different numbers $|S|$ of sensors with $|T|$ targets and sensing range *r*. Positive *p*-values indicate that *X* is superior to *Y*, and vice versa. Boldface denotes that *X* is significantly better than *Y*. N/A indicates that *X* and *Y* have identical results in all test instances.

$\|T\|$	*r*	$\|S\|$	iGA vs. MCMCC	oGA1 vs. MCMCC	oGA1 vs. iGA	oGA2 vs. MCMCC	oGA2 vs. iGA	oGA2 vs. oGA1	MA vs. MCMCC	MA vs. iGA	MA vs. oGA1	MA vs. oGA2
10	250	90	**2.40E−07**	**1.51E−07**	−4.16E−02	**2.40E−07**	N/A	**4.16E−02**	**2.40E−07**	N/A	**4.16E−02**	N/A
		100	**1.08E−08**	**6.44E−08**	−1.52E−03	**1.50E−08**	1.60E−01	**5.92E−04**	**1.86E−08**	**4.16E−02**	**3.65E−04**	1.60E−01
		200	**6.06E−09**	**2.84E−08**	−1.82E−06	**4.82E−09**	−6.44E−02	**7.97E−06**	**8.67E−09**	**5.84E−03**	**3.44E−06**	**3.58E−03**
		300	**2.10E−10**	**1.87E−08**	−1.67E−06	**4.21E−10**	2.97E−01	**2.64E−07**	**1.81E−10**	**2.14E−03**	**1.24E−07**	**1.16E−04**
500	400	90	**1.36E−10**	**5.82E−08**	−3.66E−03	**3.20E−15**	**1.82E−02**	**2.29E−07**	**6.42E−19**	**1.30E−07**	**1.89E−16**	**5.18E−07**
		100	**1.54E−08**	**3.37E−05**	−1.52E−03	**4.35E−16**	**2.67E−03**	**8.89E−10**	**1.74E−18**	**1.72E−08**	**9.33E−17**	**4.89E−07**
		200	−8.16E−20	**1.90E−03**	**2.44E−20**	**7.32E−21**	**1.73E−27**	**5.14E−16**	**8.54E−35**	**7.61E−30**	**5.84E−35**	**6.28E−24**
		300	−5.53E−43	1.87E−01	**1.01E−41**	**1.63E−12**	**4.90E−44**	**1.47E−13**	**8.93E−36**	**9.58E−47**	**1.18E−41**	**9.82E−32**

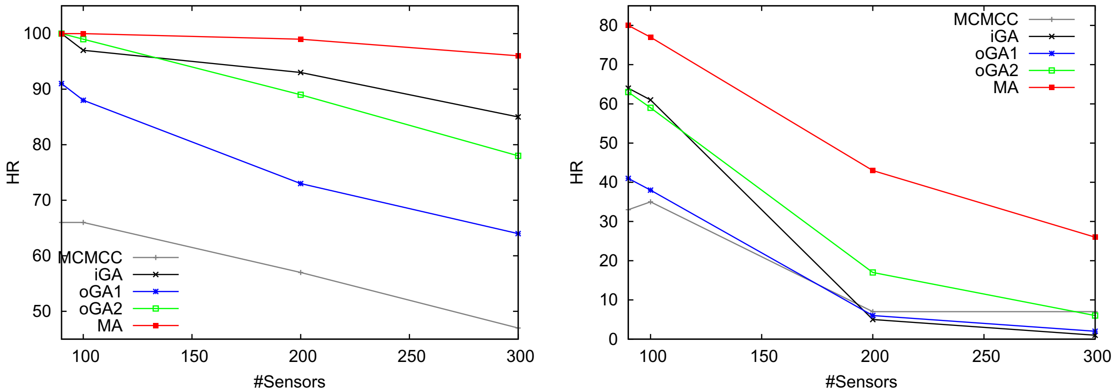

Fig. 7. Hit rates of MCMCC, iGA, oGA1, oGA2, and MA for different numbers of sensors $|(S)|$ with $|T| = 10$ and $|r| = 250$ (left) and with $|T| = 500$ and $|r| = 400$ (right).

Table 13
Running time of MCMIP, MCMCC, iGA, oGA1, oGA2, and MA for different numbers of sensors ($|S|$) with $|T|$ targets and sensing range *r*.

$\|S\|$	$\|T\| = 10$, $r = 250$						$\|T\| = 500$, $r = 400$				
	MCMIP	MCMCC	iGA	oGA1	oGA2	MA	MCMCC	iGA	oGA1	oGA2	MA
90	1234.63	0.02	2.39	2.13	2.11	2.57	3.01	71.72	71.17	90.31	92.39
100	4719.81	0.03	2.71	2.33	2.32	2.84	4.31	86.18	71.08	93.16	98.65
200	–	0.24	6.43	4.58	4.55	5.37	34.89	198.26	142.06	188.29	190.42
300	–	0.76	11.52	6.78	6.75	7.78	103.52	242.26	212.21	281.68	291.35

Table 13 indicates that running time generally increased with the number of sensors. Additionally, oGA1 was faster than iGA, confirming the advantage of order-based representation in terms of running time. A comparison with oGA1 reveals that the contribution fitness of oGA2 did not influence the running time for $|T| = 10$ with $r = 250$, but it increased the running time for $|T| = 500$ with $r = 400$. The use of the compact operator slightly increased running time, which is the cost of the excellent performance of MA in solution quality.

6. Conclusions

This study proposes an MA for extending WSN lifetime, formulated as the SET K-COVER problem. Specifically, the proposed MA follows the evolutionary scheme of GA. By viewing sensor arrangement as a process of collecting covers rather than one of partitioning, chromosomes in the MA are represented as collecting orders. This study presents a novel fitness function based on the contribution of each sensor to forming a cover. Furthermore, the compact operator is devised for local enhancement of the MA.

The proposed MA has the following advantages.

1. The evolutionary scheme of GA contributes to global search.
2. The order-based representation eliminates the need for, and therefore the sensitivity to, the upper bound on or assumptions about the maximum number of covers.
3. The contribution fitness enhances the differentiation of promising chromosomes.
4. The compact operator enhances the group composition to increase the number of covers.

To evaluate the proposed algorithm, this study conducted comprehensive simulations regarding sensing range, number of targets, and number of sensors in WSNs. The simulation results confirm the above-stated advantages of the MA in terms of the obtained number of covers, hit rate, and running time. In all of the problem instances, the proposed MA outperformed one heuristic and three other EAs in terms of the number of covers obtained and the hit rate. Notably, it achieved the same number of covers with much shorter running time than an exact algorithm requires. The scalability of the MA superiority was verified through simulations with different sensing ranges and numbers of sensors and targets. These preferable outcomes validate the effectiveness and efficiency of the proposed MA in extending WSN lifetime.

Future work may further consider different aspects of the proposed MA. First, given the above-stated advantages, the MA may be applied to extending the WSN lifetime with additional constraints and objectives, such as network connectivity for the communication range smaller than twice the sensing range and for point coverage [10]. Robustness and the dynamic performance as some sensors fail will also be important topics of future work. Second, further enhancing the search ability of the EA or the local enhancement operators will improve the performance of the MA in extending WSN lifetime.

Acknowledgments

The authors thank the editor and reviewers for their valuable comments and suggestions. This work was supported by the National Science Council of Taiwan, under contract NSC98-2221-E-194-047.

References

[1] Z. Abrams, A. Goel, S. Plotkin, Set k-cover algorithms for energy efficient monitoring in wireless sensor networks, in: Proceedings of the Third International Symposium on Information Processing in Sensor Networks, ACM Press, 2004, pp. 424–432.
[2] X. Ai, V. Srinivasan, C.K. Tham, DRACo: Distributed, robust an asynchronous coverage in wireless sensor networks, in: Proceedings of the Fourth Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 2007, pp. 530–539.
[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Computer Networks 38 (4) (2002) 393–422.
[4] B. Badrinath, J. Scholtz, M. Srivastava, K. Mills, V. Stanford, IEEE, IEEE Personal Communication, 2000 (Special issue on smart spaces and environments).
[5] P. Berman, G. Calinescu, C. Shah, A. Zelikovsky, Power efficient monitoring management in sensor networks, in: Proceedings of the Wireless Communications and Networking Conference, vol. 4, 2004, pp. 2329–2334.
[6] P. Berman, B. DasGupta, E. Sontag, On the complexities of some combinatorial problems in reverse engineering of protein and gene networks, Technical report, DIMACS, 2004.
[7] A.L. Buczak, Y. Jin, H. Darabi, M. Jafari, Genetic algorithm based sensor network optimization for target tracking, Intelligent Engineering Systems Through Artificial Neural Networks 9 (1999) 349–354.
[8] M. Cardei, D.Z. Du, Improving wireless sensor network lifetime through power aware organization, Wireless Networks 11 (3) (2005) 333–340.
[9] M. Cardei, M.T. Thai, Y. Li, W. Wu, Energy-efficient target coverage in wireless sensor networks, in: Proceedings of the IEEE INFOCOM, IEEE, 2005, pp. 1976–1984.
[10] M. Cardei, J. Wu, Energy-efficient coverage problems in wireless ad-hoc sensor networks, Computer Communications 29 (2006) 413–420.
[11] L. Davis, Applying adaptive algorithms to epistatic domains, in: Proceedings of the Ninth International Joint Conference on Artificial Intelligence, Morgan Kaufman, 1985, pp. 162–164.
[12] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Natural Computing, Springer-Verlag, 2003.
[13] M.R. Garey, D.S. Johnson, Computers and Intractability, Freeman, San Francisco, 1979.
[14] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, 1989.
[15] D.E. Goldberg, R. Lingle, Alleles, loci, and the traveling salesman problem, in: Proceedings of the First International Conference on Genetic Algorithms and Their Applications, Lawrence Erlbaum Associates, Publishers, 1985, pp. 154–159.
[16] W.E. Hart, N. Krasnogor, J.E. Smith (Eds.), Recent Advances in Memetic Algorithms, Springer, 2004.
[17] Z. He, B.S. Lee, X.S. Wang, Aggregation in sensor networks with a user-provided quality of service goal, Information Sciences 178 (9) (2008) 2128–2149.
[18] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, 2000, pp. 1–10.
[19] R. Hinterding, Mapping, order-independent genes and the knapsack problem, in: Proceedings of the First IEEE Conference on Evolutionary Computation, 1994, pp. 13–17.
[20] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: MobiCom '00: Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking, ACM, 2000, pp. 56–67.
[21] N. Krasnogor, J. Smith, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, IEEE Transactions on Evolutionary Computation 9 (5) (2005) 474–488.
[22] L. Krishnamachari, D. Estrin, S. Wicker, The impact of data aggregation in wireless sensor networks, in: Proceedings of the 22nd International Conference on Distributed Computing Systems Workshop, 2002, pp. 575–578, 2002.
[23] C.C. Lai, C.K. Ting, R.S. Ko, An effective genetic algorithm to improve wireless sensor network lifetime for large-scale surveillance applications, in: Proceedings of the 2007 Congress on Evolutionary Computation, 2007, pp. 3531–3538.
[24] M.N. Le, Y.S. Ong, Y. Jin, B. Sendhoff, Lamarckian memetic algorithms: local optimum and connectivity structure analysis, Memetic Computing Journal 1 (3) (2009) 175–190.
[25] D. Li, K.D. Wong, Y.H. Hu, A.M. Sayeed, Detection, classification, and tracking of targets, IEEE Signal Processing Magazine 19 (2002) 17–29.
[26] F. Marcelloni, M. Vecchio, Enabling energy-efficient and lossy-aware data compression in wireless sensor networks by multi-objective evolutionary optimization, Information Sciences 180 (10) (2010) 1924–1941.
[27] R. Meuth, M.H. Lim, Y.S. Ong, D.C. Wunsch II, A proposition on memes and meta-memes in computing for higher-order learning, Memetic Computing Journal 1 (2) (2009) 85–100.

[28] H.D. Nguyen, I. Yoshihara, M. Yasunaga, Modified edge recombination operators of genetic algorithms for the traveling salesman problem, in: Proceedings of IEEE International Conference on Industrial Electronics, Control, and Instrumentation, 2000, pp. 2815–2820.
[29] C. Ok, S. Lee, P. Mitra, S. Kumara, Distributed routing in wireless sensor networks using energy welfare metric, Information Sciences 180 (9) (2010) 1656–1670.
[30] I.M. Oliver, D.J. Smith, J.R.C. Holland, A study of permutation crossover operators on the traveling salesman problem, in: Proceedings of the Second International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum Associates, 1987, pp. 224–230.
[31] Y.S. Ong, N. Krasnogor, H. Ishibuchi, Special issue on memetic algorithms, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 37 (1) (2007) 2–5.
[32] Y.S. Ong, M.H. Lim, X.S. Chen, Research frontier: Memetic computation—past, present and future, IEEE Computational Intelligence Magazine 5 (2) (2010) 24–36.
[33] Y.S. Ong, M.H. Lim, N. Zhu, K.W. Wong, Classification of adaptive memetic algorithms: a comparative study, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 36 (1) (2006) 141–152.
[34] V. Raghunathan, C. Schurgers, S. Park, M.B. Srivastava, Energy-aware wireless microsensor networks, IEEE Signal Processing 19 (2) (2002) 40–50.
[35] P.S. Sausen, M.A. Spohn, A. Perkusich, Broadcast routing in wireless sensor networks with dynamic power management and multi-coverage backbones, Information Sciences 180 (5) (2010) 653–663.
[36] C. Schurgers, V. Tsiatsis, S. Ganeriwal, M. Srivastava, Optimizing sensor networks in the energy-latency-density design space, IEEE Transactions on Mobile Computing 1 (1) (2002) 70–80.
[37] A. Sinha, Y.P. Chen, D.E. Goldberg, Designing efficient genetic and evolutionary algorithm hybrids, in: W.E. Hart, N. Krasnogor, J. Smith (Eds.), Recent Advances in Memetic Algorithms, Studies in Fuzziness and Soft Computing, vol. 166, Springer, 2004, pp. 259–288.
[38] S. Slijepcevic, M. Potkonjak, Power efficient organization of wireless sensor networks, in: Proceedings of the IEEE international conference on communications, vol. 2, 2001, pp. 472–476.
[39] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, C. Whitley, A comparison of genetic sequencing operators, in: Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufman, 1991, pp. 69–76.
[40] J.A. Stine, G.D. Veciana, Improving energy efficiency of centrally controlled wireless data networks, Wireless Networks 8 (6) (2002) 681–700.
[41] C.K. Ting, Improving edge recombination through alternate inheritance and greedy manner, in: Evolutionary Computation in Combinatorial Optimization – EvoCOP 2004, LNCS, vol. 3004, Springer-Verlag, 2004, pp. 210–219.
[42] P. Varshney, Distributed detection and data fusion, Springer-Verlag, 1996.
[43] D. Whitley, T. Starkweather, D. Fuquay, Scheduling problems and traveling salesman: the genetic edge recombination operator, in: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufman, 1989, pp. 133–140.
[44] X. Yan, F. Zhu, P.S. Yu, J. Han, Feature-based similarity search in graph structures, ACM Transactions of Database Systems 31 (4) (2006) 1418–1453.
[45] H. Zhang, J.C. Hou, Maintaining sensing coverage and connectivity in large sensor networks, Ad Hoc & Sensor Wireless Networks 1 (2005) 89–124.