IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 3, 2020, accepted April 11, 2020, date of publication April 16, 2020, date of current version April 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2988368

A Hybrid Genetic Algorithm With Bidirectional Mutation for Maximizing Lifetime of Heterogeneous Wireless Sensor Networks

JINGJING LI, ZHIPENG LUO, AND JING XIAO
School of Computer Science, South China Normal University, Guangzhou 510631, China

Corresponding author: Jing Xiao (xiaojing@scnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Project 61872153 and Project 61432012, in part by the Ministry of Science and Technology of China through the Key Project of Science and Technology Innovation 2030 under Grant 2018AAA0101303, and in part by the Natural Science Foundation of Guangdong Province under Project 2018A030313318.

ABSTRACT Sleep scheduling is an effective mechanism to extend the lifetime of energy-constrained Wireless Sensor Networks(WSNs). It is often that the sensors are divided into sets with some constraints after plentiful sensors are deployed randomly, and then the sensors are scheduled to be activated successively according to the numbering of the sets. Many approaches divide the sensors into disjoint sets, which are not suitable for heterogeneous WSNs because of the waste of energy. In this paper, we propose a hybrid genetic algorithm which adopts greedy initialization and bidirectional mutation operations, termed BMHGA, to find a number of non-disjoint cover sets to prolong lifetime of heterogeneous WSNs, while subject to ensuring the full coverage of the monitoring area during the network lifetime. BMHGA adopts two-level structured chromosome to indicate the sensors and energy assignment to each set. A novel greedy method only uses little time to initialize the population that avoids time waste of random initialization. A new bidirectional mutation is proposed to keep multiplicity and global search. Through simulations, we show that the proposed algorithm outperforms the other existing approaches, finding the cover sets with longer lifetime by consuming less running time, especially in the large-scale networks. The experiment study also verifies the effectiveness of the proposed genetic operations and reveals the proper parameter settings for BMHGA.

INDEX TERMS Genetic algorithm, non-disjoint set cover problem, sleep scheduling, wireless sensor networks.

I. INTRODUCTION

Wireless sensor networks(WSNs) consist of numerous wireless sensor nodes that are densely deployed in target area to monitor the physical or environmental condition, including building monitoring [1]–[3], health monitoring [4]–[6], object tracking [7]–[9], intelligent transport system [10], [11], etc [12]. Each sensor has three basic modules: sensing module, processor module, and transmission module. The sensing module at first senses specific information surrounding the environment with certain sensing range and the transmission module transmits the signal to a data center, such as a sink node, through radio link after processing the sensed data into an electric signal by processor module. The above operations are powered by battery. Since WSNs are usually served for extensive and rugged circumstances, such as in the deep sea, arctic areas, and hazardous war zones, it is difficult to recharge batteries. Therefore, how to minimize the energy consumption of each sensor to extent the lifetime of the network is the foremost concern in WSNs.

Transmission module is a main unit contributing to the entire energy consumption of every sensor. In the module, different radio state consumes different energy. Minimum energy is consumed if the radio is in sleep state that there is no data to send or receive. Accordingly, sleep scheduling is motivated for the scenario as follows: a tremendous amount of sensors are randomly deployed such as throwing by aircraft. It is common as WSNs are used in unreachable area, the

The associate editor coordinating the review of this manuscript and approving it for publication was Gongbo Zhou.

sensors can not be configured artificially. Therefore, a portion of the sensors can already offer complete coverage and guaranteed connectivity of the area as dense deployment. The key point of sleep scheduling is to automatically and deliberately shut down subsets of nodes while remaining other nodes are alive with each given time interval [13]. The classic sleep scheduling methods usually divide the sensors into several disjoint sets with complete coverage and guaranteed connectivity, and the sets are activated successively [14]. In these methods, each sensor can only in one set that is alive with a given time. After the interval, even though there is remaining energy for the sensor, it can not be used anymore. The methods are suitable for homogeneous WSNs that assuming each sensor has same energy supply, the time interval of each disjoint sets is set as the lifetime of sensors. As a result, the whole lifetime of the network is the sum of periods of the disjoint sets. The sleep scheduling problem is transferred as finding the maximum number of disjoint sets named as disjoint set cover(DSC) problem.

However, in realistic world, many applications are designed as heterogeneous WSNs [15]–[17] in which different sensors involved have different data processing capability, transmission capability and energy supply. Energy heterogeneity means that the nodes have different initial energy which is the most important characters because both computational heterogeneity and link heterogeneity will consume more energy resource [18]. In this situation, it is the energy waste if the sensors are divided into disjoint sets. For example, in Fig.1(a), assuming a unit time is a, each sensor may have different initial energy that is different number of units time. The lifetime of s_1, s_2, s_3, s_4 is $1a$, $2a$, $2a$ and $1a$. Any two sensors can completely cover the target region. In disjoint sets cover problem, the network is divided into $S_1 = \{s_1, s_2\}$ and $S_2 = \{s_3, s_4\}$ for $1a$ time. The total lifetime of the network is $2a$ while there is still remaining energy $1a$ of s_2 and s_3 respectively.

In this paper, our objective is to find the optimal non-disjoint sets to maximize the lifetime of heterogeneous WSNs. In non-disjoint set cover(NDSC) problem, each sensor may be given different energy supplies and if the sensor has residual energy after a time duration in a set, it can be selected to other sets. As shown in Fig. 1(b), the non-disjoint sets are: $S_1 = \{s_1, s_2\}$ for $1a$ time, $S_2 = \{s_3, s_4\}$ for $1a$ time and $S_3 = \{s_2, s_3\}$ for $1a$ time. The lifetime of the network is $3a$ which is 33% increase compared to the disjoint sets.

Compared with DSC problem, NDSC problem is more complex since a sensor can be assigned to multiple sets. It is nondeterministic polynomial (NP)-complete which has been proved in [19]. NDSC problem can be solved by approximation algorithms, however, the obtained solutions are not always optimal. Heuristic algorithms can also be applied to NDSC problem. Gentic Algorithm(GA) is a heuristic search algorithm that obtains the optimized results through simulating biological evolution [20], which is suitable for multi-objectives problems, especially the resource-constraint scheduling problems [21] with large search space and small number of feasible solutions. Its population can process a set of solutions and each individual can involve different objectives and constraints. Since the NDSC problem defined in this paper includes two objectives and two constraints and the scale of networks is great while the number of possible solutions is small, we prefer GA to solve the problem. However, the searching speed of the traditional GA is slow and it is easy to trap into the local optima. Therefore, a modified genetic algorithm is proposed in this paper, termed the Bidirectional Mutation Hybrid Genetic Algorithm(BMHGA). Given different initial energy and position of each sensor, the algorithm can find the non-disjoint sets to cover the target area fully and the sum of the lifetime of the sets is maximized. Compared with the existing works, the contributions of BMHGA are as follow:

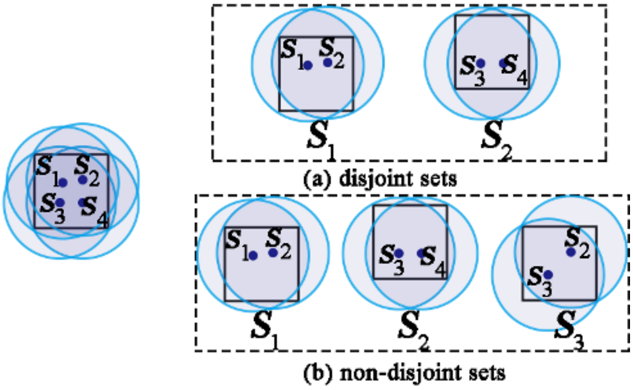


FIGURE 1. Sensors s_1, s_2, s_3, s_4 are deployed in target area. (a) The sensors are divided into two disjoint sets S_1, S_2. (b) The sensors are divided into three non-disjoint sets S_1, S_2, S_3.

1) For explicitly indicating the resource constraint and showing the schedules of each sensor, a two-level structured chromosome is designed in the paper. At a high level, each sensor is mapped to a gene. At a low level, there are genomes embodied in each 'high level' gene that is represented as a set of pairs, each specifying duration time for a different cover set.

2) An initial population of high quality can accelerate the population evolution of GA to reach a satisfactory solution rapidly [22]. Although the greedy algorithm makes a locally-optimal selection at each moment so that it cannot produce global optimized solutions, its advantages are their simplicity and efficiency [23]. Therefore, we initialize the chromosomes based on the feasible solutions that are obtained by a designed greedy algorithm to promote individual to evolve.

3) After greedy initialization, there are still a lot of redundant sensors in the individuals that need to be optimized. Thus, we propose a forward mutation that schedules the redundant sensors among the different cover sets without losing the coverage percentage of each cover set but increasing the total lifetime of the sets. GA is trend to the premature convergence to local optima. A backward mutation is presented to help skipping local optima when necessary.

4) Compared with the prior works, the proposed BMHGA can effectively extend the lifetime of the network with faster speed in different size of networks.

The BMHGA is executed by a central node that knows the number of sensors and their initial energy and location when deployment. The scheduling message is broadcast to the whole network after the scheduling scheme is computed. Then each sensor node would know when to sleep or activate. The scheduling can be based on the synchronization method proposed in [24], in which nodes are synchronized with their own clocks, sensors are able to interact with each other at the same physical time by periodical beacon broadcast.

The remainder of this paper is organized as follows. Section II describes the related works on this problem. The preliminaries and the problem definition are given in section III. Section IV illustrates the detailed implementations of the proposed algorithm. A series of experimental simulations are conducted and the results are analyzed in section V. Finally, section VI concludes the paper.

II. RELATED WORKS

The common approaches that sleep-scheduling protocols adopted for increasing the lifetime of the network are to divide the sensors into DSCs [25]. Cardei and Du [19] firstly proved the DSC problem is NP-complete. Deschinkel and Hakem [26] presented a near optimal algorithm with a lower bound. In [27], S. Henna constructed two disjoint cover sets in randomly deployed WSNs and presented an approximation algorithm. In [28], three kinds of algorithms named greedy game algorithms(GGAs) were proposed to solve the SET K-Cover problem in heterogeneous WSNs. However, these approximation approaches can not always find the optimal solution. Some researchers tried to use heuristic algorithms to find the optimal result. For example, Hu *et al.* [29] designed a modified genetic algorithm with forward encoding scheme(STHGA) and Mir *et al.* [30] adopted particle swarm optimization to maximize the lifetime of WSNs. The performance of the proposed approaches showed the heuristic algorithms are effective for solving the DSC problem.

In recent literatures, more sleep scheduling algorithms partition the network into NDSCs since NDSCs provide longer lifetime compared to disjoint cover sets [31], [32].

The methods of NDSC problem can be classified into three types: approximation methods, heuristic methods and bio-inspired search methods.

For the first type, Ding *et al.* [33] proposed a polynomial-time $(4 + \eta)$-approximation algorithm to extend the lifetime of WSNs based on Minimum Weight Sensor Coverage Problem. In [34], the target coverage with detection probability was considered and an approximation algorithm that has three main phases was presented. In the computation phase, the candidate coverage set of each target is calculated. In the selection phase, an independent coverage set is selected from every target candidate coverage set in turn. In the connection phase, a steiner tree algorithm is used to select some sensors as relay nodes that maintain the network connectivity. Paper [35] formulated the master problem and subproblem of NDSC problem. Consequently a column generation based exact algorithm is proposed to solve the master problem and branch-and-cut method is adopted to reduce the computation.

Since the results of above methods are not always optimal and NDSC problem is also a NP problem, the heuristic algorithms are preferred. B.Diop etc [36] proposed a greedy algorithm that distributes sensors among disjoint sets and each set satisfies full target coverage. In [37], a centralized connected target k-coverage algorithm(CCTCk) and a distributed connected target k-coverage algorithm(DCTCk) were proposed. Two algorithms minimize the total number of active sensor nodes and guarantee that each sensor node is connected to a sink. A novel Energy Efficient Connected Coverage (EECC) scheduling was proposed to maximize the lifetime of WSNs in [31]. EECC elevates the coverage and connectivity ability of each sensor, considering its remaining energy and the sensor nodes which monitor the crucial targets are handled carefully as well the redundant coverage at critical points is avoided. In [38]–[40], the clustering method is used for prolonging the network lifetime. The adjacent sensors are clustered by self-organizing, and some of them are changed to sleep status. When an activated sensor is run out of energy, the neighboring available sensors are waked up to cover the data loss. The methods have good fault tolerance and performance. Since the frequent computation among the neighboring sensors, they are suitable for small-scale WSNs.

Besides the heuristic algorithms, the bio-inspired search algorithms are usually used to solve the NDSC problem. Zhong and Zhang [41] proposed an ant colony optimization method(mc-ACO) based sleep scheduling. mc-ACO constructs a first layer set and multiple successor sets by selecting the sensors to cover the uncovered area according to the pheromone. However, mc-ACO is not suitable for large scale networks and sometimes achieves premature convergence. Liao and Ting [42] adopted integer representation and designed a new fitness function that considers both the number of covers and the contribution of each sensor to covers. A local improvement method was also developed to enhance the performance on Set k-Cover problem. However, the algorithm is only used for target coverage problem. In paper [32], a novel gene coding to solve NDSC problem by using genetic algorithm(NDSC-GA) was proposed. Each gene is an array that includes the time periods of the sensors in the set. The length of a gene is the number of sensors in the network. In order to prolong the lifetime of network, NDSC-GA adopts random crossover and probability mutation for improving gene and creating new set. It takes much time to execute traversal operation, hence, NDSC-GA is not efficient when the number of sensors is large.

III. PRELIMINARY AND PROBLEM DEFINITION

In this paper, we consider that each sensor has the same sensing range and communication range. The sensing area of each sensor is a circle with the radius R and the communication range is twice of sensing range that means a completely covered network is also connected [43], [44]. Therefore, the sink node can connect with each sensor through multihop communication in the completely covered network. It is responsible for scheduling the sensors and collecting the sensing data from sensors. We also assume the sensors execute fault-free operations in their lifetime. For evaluating whether the target field is completed covered, we use the grid-based method [45] to approximately estimate the sensing coverage of sensors. Assume the monitored area is divided into $L \times W$ virtual grids of uniform size, the sensing area of each sensor is calculated as the number of grids that are totally inside its sensing region. When a set of sensors are deployed in the target field, the whole sensing area of the set is D grids, the coverage ratio P of the set is approximately as in formula (1).

$$P = \frac{D}{L \times W} \tag{1}$$

If P = 100%, the target area can be covered by the set of sensors completely.

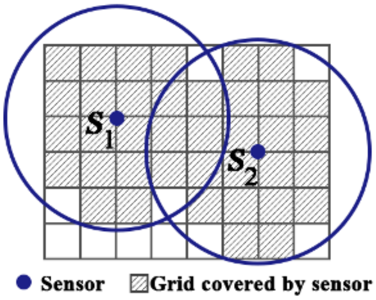


FIGURE 2. Coverage area of sensors s_1 and s_2.

For example, in Fig.2, two sensors are deployed in the area that consists of 48 grids and each sensor covers 20 grids. The overlapping area of two sensors are 4 grids. Therefore, the sensing range of the two sensors is 38 grids and the coverage ratio is $\frac{38}{48}$.

For monitoring the area, if all of the sensors in WSNs are activated, full coverage should be satisfied. Otherwise, the arrangement of sensors fails.

When the target field consists of $L \times W$ virtual grids, assume all the sensors are alive, each grid t_i is covered by $n(n \geq 1)$ sensors $s_{i_1}, s_{i_2}, \ldots, s_{i_n}$ with initial energy supply $e_{i_1}, e_{i_2}, \ldots, e_{i_n}$. The maximum time period l_i that grid t_i covered by the sensors can be computed by

$$l_i = \sum_{k=1}^{n} e_{i_k} \tag{2}$$

where $i \in [1, L \times W]$, e_{i_k} is the initial energy of sensor s_{i_k}. The grid with minimum value of l_i which is the critical field will firstly lose being monitored, meanwhile WSNs can not fully cover the target area. As a result, the upper limit lifetime T of network is decided by

$$T = min(l_1, l_2, \ldots, l_{L\times W}) \tag{3}$$

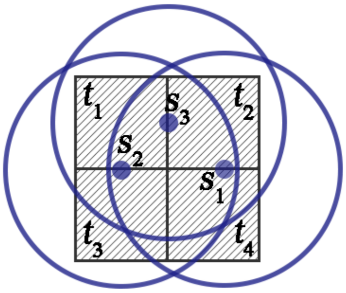


FIGURE 3. The target area that consists of 4 grids t_1,t_2,t_3,t_4 is covered by three sensors s_1,s_2,s_3.

For example, in Fig.3, the target area is divided into 4 grids t_1, t_2, t_3, t_4 and there are 3 sensors s_1, s_2, s_3. t_1 is covered by s_2, s_3, t_2 is covered by s_1, s_3, t_3 is covered by s_2, t_4 is covered by s_1. The initial energy of sensors s_1, s_2, s_3 is $e_1 = 2$, $e_2 = 3, e_3 = 4$. Therefore, the maximum time period of each grid is $l_1 = e_2 + e_3 = 7$, $l_2 = e_1 + e_3 = 6$, $l_3 = e_2 = 3$, $l_4 = e_1 = 2$. The upper limit lifetime of the network $T = min(l_1, l_2, l_3, l_4) = 2$.

There are still several simple concepts should be defined firstly before the problem is formally defined.

Definition 1 (Lifetime of Set): Let a set of sensors $S_k = \{s_1, s_2, \ldots, s_n\}$ provide energy $E = \{e_1, e_2, \ldots, e_n\}$. S_k is a complete cover set if the sensors in S_k cover the target area completely. S_k is an incomplete cover set if the sensors in S_k cannot fully cover the target area. The lifetime L_k of set S_k is

$$L_k = \begin{cases} min(e_1, e_2, \ldots, e_n) & P_k = 100\% \\ 0 & P_k < 100\% \end{cases} \tag{4}$$

Definition 2 (Residual Energy of Sensor): Suppose sensor s_i with initial energy e_i belongs to k sets $S_{i1}, S_{i2}, \ldots, S_{ik}$. The lifetime of the sets $S_{i1}, S_{i2}, \ldots, S_{ik}$ is $e_{i1}, e_{i2}, \ldots, e_{ik}$. Therefore the residual energy e_{ir} of sensor s_i is calculated by

$$e_{ir} = e_i - \sum_{k=1}^{j} e_{ik}. \tag{5}$$

Definition 3 (Lifetime of WSNs): The sensors in WSNs are divided into c sets $S_1, S_2, \ldots, S_c$. Lifetime L_T of WSNs is the sum of lifetime of sets.

$$L_T = \sum_{k=1}^{c} L_k \tag{6}$$

The NDSC problem in this paper is defined as follows:

Given a set $S = \{s_1, s_2, \ldots, s_N\}$ of sensors are randomly deployed in an $L \times W$ field. Assume each sensor s_i has initial energy e_i and the identical sensing range R. The objective is to find c successive scheduling sets $S_1, S_2, \ldots, S_c$ with the maximum lifetime $L_1 + L_2 + \ldots + L_c$ of WSNs, satisfying:

1) each set $S_k = \{s_{k1}, s_{k2}, \ldots, s_{k|S_k|}\}$ of sensors can completely cover the target area, where $|S_k|$ is the number of sensors that are assigned to the set S_k.

2) as to any sensor s_i, the total active time do not exceed its initial energy e_i, that is

$$e_i \geq \sum_{k=1}^{c} e_{iS_k} \tag{7}$$

where e_{iS_k} is the lifetime of set S_k if s_i is assigned to S_k. If s_i does not belong to set S_k, $e_{iS_k} = 0$.

IV. PROPOSED ALGORITHM

To solve the NDSC problem, in this paper, we propose a hybrid genetic algorithm(BMHGA) which uses bidirectional mutations. In this algorithm, initialization with greedy techniques instead of random initialization is adopted for each chromosome, then adaptive crossover and multiple mutation operations are designed to accelerate the solving procedure. The chromosomes with highest fitness value are selected to next generation. The flowchart of the proposed algorithm is shown in Fig.4, where f records the best fitness value of chromosomes in the previous generation, *i* counts the number of generations that the best fitness value remain unchanged. Table 1 tabulates the notations used in this paper. The proposed algorithm is performed by the sink node. According the obtained computing results, the sink node will wake up the sensors successively.

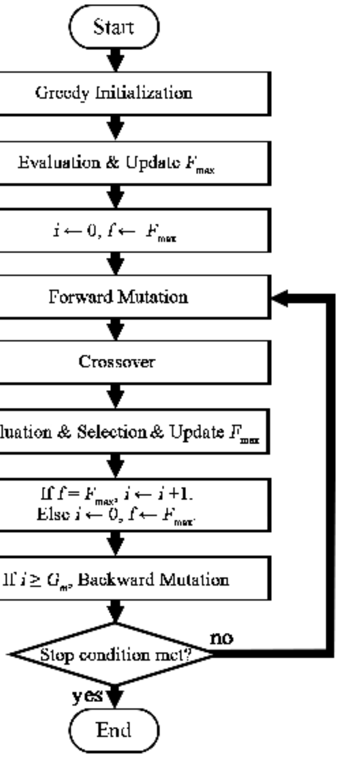

Start
Greedy Initialization
Evaluation & Update F_{max}
$i \leftarrow 0, f \leftarrow F_{max}$
Forward Mutation
Crossover
Evaluation & Selection & Update F_{max}
If $f = F_{max}$, $i \leftarrow i+1$.
Else $i \leftarrow 0$, $f \leftarrow F_{max}$.
If $i \geq G_m$, Backward Mutation
Stop condition met?
no
yes
End

FIGURE 4. Flowchart of the BMHGA.

A. REPRESENTATION OF CHROMOSOME

Since the multi-objectives and constraints are needed to be satisfied in the problem of the paper, the chromosome should include various information such as the cover sets, the corresponding sensors and their energy. The two-level structured chromosome is proposed in this paper. Since each sensor has a unique ID, at the high level of the chromosome, each gene is mapped to a sensor. Therefore each chromosome C_m at the high level in the population is represented as

TABLE 1. Notations.

Symbol	Descriptions
S_k	Set of sensors in the target area
s_i	ith sensor
e_i	Initial energy of sensor s_i
e_{ir}	Residual energy of sensor s_i
N	Total number of sensors in network
L	Length of the target area
W	Width of the target area
R	Sensing range of sensors
t_i	ith grid of target area
l_i	Maximum time period of grid t_i
T	Upper limit lifetime of network
η	Redundancy rate
L_k	Lifetime of set S_k
L_T	Total lifetime of network
M	Number of chromosomes in the population
C_h	hth chromosome
G_{hi}	ith gene of chromosome C_h
g_{hij}	jth value of gene G_{hi}
e_{hij}	Energy that sensor s_i consumes in set $S_{g_{hij}}$
c_h	Number of complete sets of chromosome C_h
P_k	Coverage percentage of set S_k
F_h	Fitness value of chromosome C_h
K	Number of genes selected in the forward mutation
G_m	Maximum fitness standstill generations of performing backward mutation
F_{max}	Maximum fitness value in current population
F_{min}	Minimum fitness value in current population
F_{init}	Fitness value of the initial population

$$C_m = \{G_{m1}, G_{m2}, \ldots, G_{mc}, G_{mN}\} \tag{8}$$

where $m = 1, 2, \ldots, M$. At the low level, each G_{mi} is represented as a genome shown in (9) that the number of sets of sensor s_i belongs to, and the specific sets with its lifetime are indicated.

$$G_i = \{(g_{i1}, e_{i1}), (g_{i2}, e_{i2}), \ldots, (g_{i|G_i|}, e_{i|G_i|})\} \tag{9}$$

where g_{ij} is the scheduling number that means the sensors with the same scheduling number will form a set, $|G_i|$ is the number of total sets sensor s_i is in.

The benefit of the hierarchical genomic structure is the lifetime of the total cover sets and the corresponding scheduling scheme are contained in the chromosome so that it is easy to be evaluated.

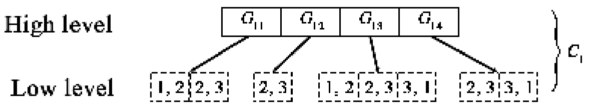


FIGURE 5. The two level structure of the example chromosome C_1.

For example, as shown in Fig.5, suppose a chromosome C_1 =$\{G_{11}, G_{12}, G_{13}, G_{14}\}$, $G_{11} = \{(1, 2), (2, 3)\}$,

$G_{12} = \{(2,3)\}$, $G_{13} = \{(1,2), (2,3), (3,1)\}$, $G_{14} = \{(2,3), (3,1)\}$ that means there are 4 sensors s_1, s_2, s_3, s_4 in the area, s_1, s_4 are scheduled in two different sets, s_2 is only activated in one set and s_3 is scheduled in three different sets. Accordingly, these sensors are divided into three sets $S_1 = \{s_1, s_3\}, S_2 = \{s_1, s_2, s_3, s_4\}, S_3 = \{s_3, s_4\}$. The lifetime of sets is 2,3,1 respectively.

B. GREEDY INITIALIZATION

Initially, in traditional GA, the individuals are usually generated randomly. However, in the case of this paper, we will use greedy policies for initialization. The first population is generated by a greedy algorithm. According to the method, at beginning, the grids in given field are marked as uncovered. There is a candidate set includes the whole sensors and the number of cover sets $c = 0$. Then the steps are executed as follows:

1)$c = c + 1$, and S_c is initialized as an empty set.

2)A sensor from the candidate set that can improve the coverage percentage of the set S_c is selected into the set S_c, meanwhile the corresponding grids are marked as covered.

3)The second step 2) is repeated until the total grids are marked. The set S_c is formed and the lifetime of the set S_c is calculated.

4)Accordingly, the rest lifetime of sensor is updated. If the sensor is exhausted, it is removed from the candidate set. And the total grids are marked as uncovered.

5)Execute the above four steps in a loop until the sensors in the candidate set cannot cover the whole area.

Finally, the remaining sensors in the candidate set will form the S_{c+1} set which is an incomplete set.

A series of successive cover sets are formed by the greedy algorithm, therefore, the chromosomes are initialized from the result of the algorithm. The proposed greedy algorithm ensures the initial population is not entirely random and thereby the convergence is accelerated. Since the complexity of the greedy algorithm is polynomial $O(c \times N)$, this initialization step takes little time to increase convergence speed significantly.

The whole initialization process is described as Algorithm 1, where function ClearGrids() marks all grids as uncovered, MarkGrids(s_{a_i}) marks the grids as covered that is covered by sensor s_{a_i}. Function GetenCoverGrids(s_n) returns the number of the uncovered grids that sensor s_n enables to cover. Function GetLifetime(S_c) calculates the lifetime of set S_c, and function GetRestEnergy(s_{a_n}) returns the remaining energy of sensor s_{a_n}.

After initialization, in each chromosome C_h, there are c_h complete cover sets and an incomplete set S_{h,c_h+1}. After executing mutation and crossover, the coverage percentage of set S_{h,c_h+1} rises higher. The lifetime of WSNs is increased when the incomplete set S_{h,c_h+1} becomes a complete set.

Algorithm 1 Greedy Initialization

```
Require: $N$ sensors $S = \{s_1, s_2, \dots, s_N\}$, $L \times W$ grids
Ensure: $M$ chromosomes $C_1, C_2, \dots, C_M$
c ← 0; $G_1, G_2, \dots, G_N \leftarrow \{\phi\}$;
repeat
    ClearGrids();
    $c \leftarrow c + 1$;$S_c \leftarrow \{\phi\}$;
    for $s_n$ in $S$ do
        if GetenCoverGrids($s_n$) > 0 then
            $S_c \leftarrow S_c + s_n$;
            MarkGrids($s_n$);
        end if
    end for
    /*Get the lifetime of the set $S_c$*/
    $L_c \leftarrow$ GetLifetime($S_c$)
    for $s_n$ in $S_c$ do
        $G_n \leftarrow G_n + (c, L_c)$
        /*Get the rest energy of sensor $s_n$*/
        $e \leftarrow$ GetRestEnergy($s_n$)
        if $e = 0$ then
            $S \leftarrow S - s_n$
        end if
    end for
until sensors in $S$ can not cover the whole area.
/*Remaining sensors form the last cover set*/
$S_{c+1} \leftarrow S$;
for $s_n$ in $S$ do
    $e \leftarrow$ GetRestEnergy($s_n$)
    $G_n \leftarrow G_n + (c, e)$;
end for
/*Copy $M$ chromosomes*/
for $m := 1 \rightarrow M$ do
    for $n := 1 \rightarrow N$ do
        $C_m \leftarrow C_m + G_n$
    end for
end for
```

C. EVALUATION & SELECTION

The chromosome indicates the full cover sets that WSNs are divided into and the lifetime of each set, therefore the lifetime of WSNs can be calculated. The fitness function of a chromosome C_h is defined as

$$F_h = \sum_{k=1}^{c_h} L_k + P_{c_h+1}. \tag{10}$$

The larger the lifetime of WSNs is, the higher the fitness of chromosome is. If the lifetime of WSNs for two chromosomes is equal, the chromosome with a greater coverage percentage of the incomplete set is the better one. The population is sorted by fitness values, and then the chromosomes with the higher fitness are selected to the next generation.

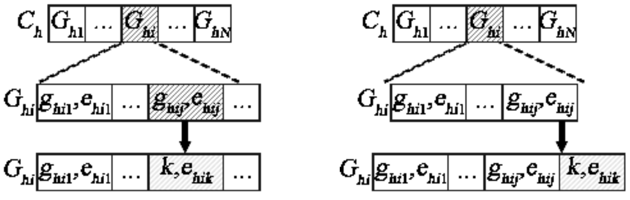
D. FORWARD MUTATION

To search a better chromosome, there are two kinds of sensors should be scheduled.

1) Coverage redundant sensor: if a sensor in a set that can be removed without changing the coverage percentage of the

set, we call the sensor is coverage redundant sensor. This kind of sensors should be rescheduled to other sets. Fig.6(a) shows the rescheduling procedure of coverage redundant sensors.

2) Lifetime redundant sensor: if the residual energy of a sensor is greater than 0, we call the sensor is a lifetime redundant sensor. It can be scheduled to other sets using the residual energy. Fig.6(b) describes the scheduling of lifetime redundant sensors.



(a)

(b)

FIGURE 6. (a)In chromosome C_h, sensor s_i is a coverage redundant sensor in set $S_{g_{hij}}$, then S_i is rescheduled to set S_k with energy e_{hik}. (b)In chromosome C_h, sensor s_i is a lifetime redundant sensor, then s_i can be scheduled to set S_k with energy e_{hik}, $k \in [1, c_h + 1]$, e_{hik} is residual lifetime of the sensor s_i.

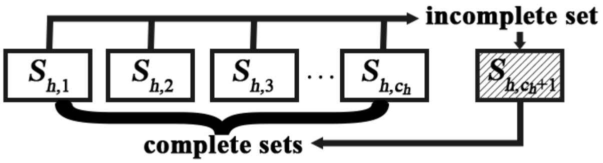


FIGURE 7. The change of the redundant sensors between complete cover sets and an incomplete cover set in the chromosome C_h.

In the mutation, K genes are checked randomly for each chromosome C_h. If the related sensors are coverage/lifetime redundant, they are assigned to other cover sets in two ways. Firstly, if the redundant sensor is in a complete cover set, it is scheduled to the $c_h + 1$th incomplete set. Secondly, if the redundant sensor is in the incomplete set S_{h,c_h+1}, it is scheduled to one of c_h complete sets, the specific number of the set is decided randomly. Fig.7 shows the redundant sensors in the complete sets are scheduled to the incomplete set S_{h,c_h+1}, and the redundant sensors in the incomplete set S_{h,c_h+1} are scheduled to one of complete sets. Since there is only one incomplete set for each chromosome, the first way of forward mutation can get the higher coverage percentage of incomplete set. The second way in forward mutation promotes the genetic diversity of chromosomes. This operation provides positive effect on evolutionary of individuals and thereby is termed forward mutation.

E. BACKWARD MUTATION WITH REPLACE OPERATION

When the highest fitness of chromosomes in populations gets stuck in a value, never to reach upper limit lifetime of network, the algorithm is trapped in the local optimum. The reason for this phenomenon is that there is no redundant sensor to be scheduled to the incomplete set. Thus, in this paper, backward mutation is designed to avoid the premature convergence. For the best chromosome in current population, if its fitness value is constant for G_m generations, backward mutation is performed under two scenarios.

When there exists an incomplete cover set in a chromosome, since the coverage percentage of incomplete cover set is less than 1, even to the best chromosome, there still are grids without being covered by sensors. Draw a circle with radius $2R$ centered on the center of a random uncovered grid t_u. For the incomplete cover set, the sensors within the circle are rescheduled back to random complete cover sets. Then replace operation is executed by rescheduling a coverage redundant sensor in a complete cover set that covering the grid t_u to the incomplete set.

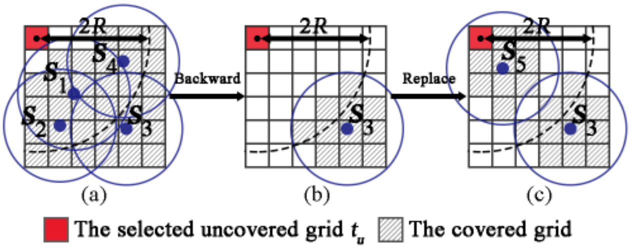
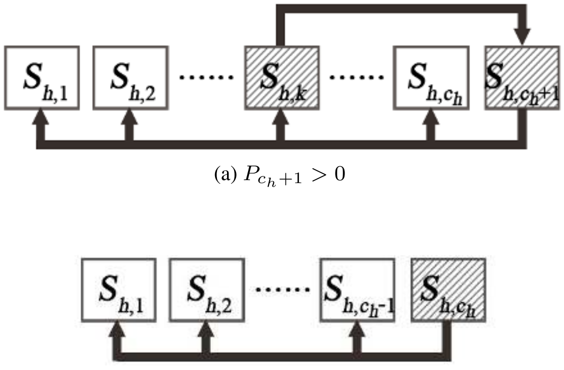


FIGURE 8. The change of the incomplete cover set during backward mutation.

For example, in Fig.8, there are 4 sensors s_1, s_2, s_3, s_4 in an incomplete cover set and an uncovered grid t_u. As shown in Fig.8(a), s_1, s_2, s_4 are in the circle that is centered on the center of grid t_u with radius $2R$. Then, s_1, s_2, s_4 are rescheduled back to the complete sets randomly and there is only one sensor s_3 in the incomplete set as shown in Fig.8(b). In Fig.8(c), after replace operation there are two sensors s_3 and s_5 in the incomplete cover set and s_5 can cover the grid t_u. s_5 is a coverage redundant sensor which is rescheduled from a complete cover set. Fig.9(a) illustrates the scheduling directions in the situation.

If there is no incomplete cover set in the best chromosome C_h that means $P_{c_h+1} = 0$. The backward mutation reschedules the sensors in the last complete cover set S_{h,c_h} back to other complete cover sets randomly, such as shown in Fig.9(b). Every sensor in set S_{h,c_h} of chromosome C_h has 50% chance to be rescheduled back. After backward mutation, the cover set S_{h,c_h} becomes an incomplete cover set.

In the first situation, the uncovered grid becomes being covered through the replace operation and the complete cover sets are changed by swapping the sensors. In the second situation, the combination of the complete cover sets that results in local optima is broken. As a result of backward mutation, all local optima can be avoided. Though the coverage percentage of the incomplete cover set or the last complete cover set may decrease after the mutation, that is also the reason why we call the mutation as backward mutation, the coverage percentage can rise when the forward mutation is executed again.



(a) $P_{c_h+1} > 0$

(b) $P_{c_h+1} = 0$

FIGURE 9. **(a)The specified sensors in the set S_{h,c_h+1} are rescheduled back randomly, and a specified redundant sensor in set $S_{h,k}$ is scheduled to the set S_{h,c_h+1}. (b)The sensors in set S_{h,c_h} are scheduled back to $c_h - 1$ sets randomly.**

The procedure of the backward mutation is shown in Algorithm 2, where function UncoveredGrid(C_m) picks an uncovered grid t_u from the incomplete cover set S_{m,c_m+1} in chromosome C_m, function Distance(s_n, t_u) calculates the distance between sensor s_n and grid t_u, function isRedundant($s_n, S_{m,g_{mni}}$) checks if sensor s_n is coverage redundant in set $S_{m,g_{mni}}$.

F. ADAPTIVE CROSSOVER

Crossover operation selects two individuals as parents to create new individuals. It aims to produce better chromosomes from old ones. The idea behind adaptive crossover [46] is that the crossover probability of each chromosome is determined by its genome and is evolved with it. Firstly, a chromosome C_i is selected randomly from current population as a parent. For each chromosome C_j, the matrix variance V_{ij} with chromosome C_i is calculated by

$$V_{ij} = \sum_{n=1}^{N} \sum_{k=1}^{max(|G_{in}|,|G_{jn}|)} (g_{ink} - g_{jnk})^2 \tag{11}$$

where G_{in} is the gene that belongs to chromosome C_i, $|G_{in}|$ is the number of sets that sensor s_i belongs to, g_{ink} is the set number.

The chromosome C_j with the greatest V_{ij} is considered as the other parent. According to the method, only two parent chromosomes with maximum difference between them are chosen so that the genetic diversity of their offsprings is guaranteed. If all the chromosomes in the population tend to have the same genetic coding, the algorithm tends to converge. Therefore, in this case, the crossover operation will be stopped.

For inheriting the genetic material from the better of two parent chromosomes, the crossover probability is calculated as

$$p_{better} = \frac{F_{better} - F_{min}}{F_{max} - F_{min}} \tag{12}$$

$$p_{other} = 1 - p_{better} \tag{13}$$

Algorithm 2 Backward Mutation

```
Require: $M$ chromosomes $C_1, C_2, \ldots, C_M$
for $m := 1 \rightarrow M$ do
    if $F_m = F_{max}$ and $P_{c_m+1} > 0$ then
        $t_u \leftarrow$ UncoveredGrid($C_m$)
        for $n := 1 \rightarrow N$ do
            if Distance($s_n, t_u$) $< 2R$ then
                for $i := 1 \rightarrow |G_{mn}|$ do
                    if $g_{mni} = c_m + 1$ then
                        $g_{mni} \leftarrow$ Rand(1,$c_m$)
                    end if
                end for
            end if
        end for
        /*Replace Operation*/
        for $n := 1 \rightarrow N$ do
            if sensor $s_n$ covers grid $t_u$ then
                for $i := 1 \rightarrow |G_{mn}|$ do
                    if isRedundant($s_n, S_{g_{mni}}$) then
                        $g_{mni} \leftarrow c_m + 1$
                    end if
                end for
            end if
        end for
    else if $F_m = F_{max}$ and $P_{c_m+1} = 0$ then
        for $n := 1 \rightarrow N$ do
            for $i := 1 \rightarrow |G_{mn}|$ do
                if $g_{mni} = c_m$ and Rand(0,1)=1 then
                    $g_{mni} \leftarrow$ Rand(1,$c_m - 1$)
                end if
            end for
        end for
    end if
end for
```

where p_{better} is the probability of selecting the gene from the better parent while p_{other} is the probability to select a gene from the other parent. F_{better} is the fitness of the better parent chromosome.

The formula (12) indicates the parent chromosome with higher fitness will maintain more genes to the offspring. If the parent chromosome is the best member in current population, the whole individual will become an offspring.

V. EXPERIMENTAL STUDY

In the simulation environment, N sensors are randomly deployed in a two-dimensional $L \times W$ area. Each node has its x and y location that randomly generated as integer values in [0,L] and [0,W]. The sensing range of each sensor is R and the initial energy e_i of sensor s_i is also generated randomly as an integer value in [1,E_{max}], E_{max} is the maximum value of initial energy of sensors in the network. All cases are run by a computer with an Intel i7-7700 3.60GHz CPU.

TABLE 2. Assumptions and goals of algorithms.

Algorithms	Assumptions					Goals
	Sensing Model	Sensing Range	Communication Range	Location Awareness	Initial Energy	
BMHGA	Boolean	Same	Same /2*Sensing Range	Yes	Different	Maximize the sum of lifetime of no DSCs/full area coverage
NDSC-GA	Boolean	Same	N/A	Yes	Same	Find maximum # of no DSCs/target coverage
mc-ACO	Boolean	Same	N/A	Yes	Different	Find maximum # of no DSCs/full area coverage
STHGA	Boolean	Same	Same /2*Sensing Range	Yes	Same	Find maximum # of DSCs/full area coverage
EXACT	Boolean	Different	Different	Yes	Different	Find maximum # of no DSCs/target coverage

A. COMPARISON WITH OTHER ALGORITHMS

In this section, we compare BMHGA with other algorithms, including STHGA [29], mc-ACO [41], NDSC-GA [32] and EXACT algorithm [35]. Table 2 shows a comparison in aspects of assumptions and goals among the algorithms above. We can find that the goal of BMHGA is different from other algorithms.

The algorithms are designed for NDSC problem except STHGA, which is a classic algorithm that is proposed to solve DSC problem under the assumption that each sensor has the same initial energy(a unit energy). However, STHGA can be modified easily for the problem defined in the paper through a preprocess, in which if a sensor with multiple units energy, it can be regarded as multiple sensors all with a unit energy. For example, a sensor with 3 units energy is regarded as 3 sensors with a unit energy at the same location. After the preprocess, the genetic operations of STHGA can function normally just the size of WSNs increases.

TABLE 3. Computational complexity of algorithms.

Algorithm	Time Complexity	Space Complexity
BMHGA	$O(c \times N) + O(J \times N)$	$O(E_{max} \times N)$
NDSC-GA	$O(J \times N)$	$O(N^2)$
mc-ACO	$O(N^2) + O(J \times N^2)$	$O(N)$
STHGA	$O(J \times N)$	$O(E_{max} \times N)$
EXACT	$O(N!)$	$O(N)$

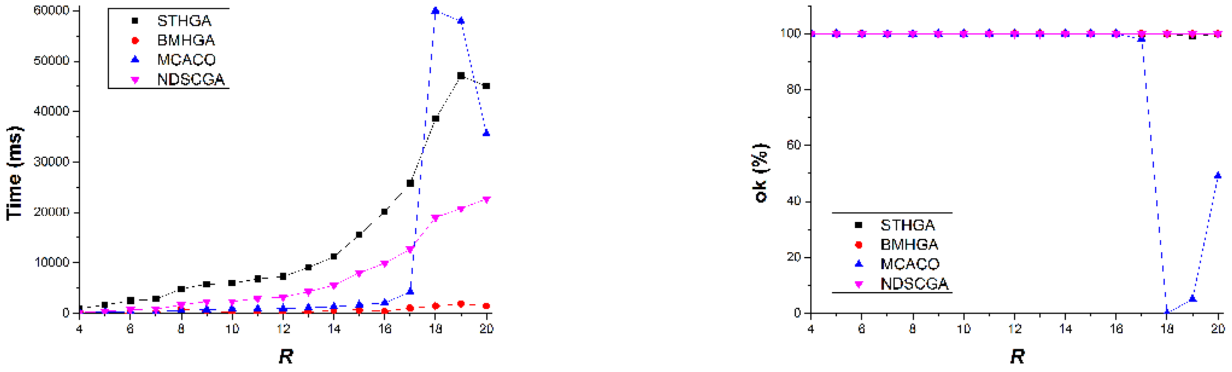
Table 3 investigates the time complexity and space complexity of above mentioned algorithms, where the J is the number of iterations for each algorithm. The running time is relevant to the network scale. The time complexity of traversing N sensors is $O(N)$. BMHGA is a hybrid algorithm consists of greedy initialization and genetic operators. The computational complexity of greedy algorithm in initialization step is $O(c \times N)$ that has been mentioned in section IV. In each generation, the running time is $O(K + N)$ that mainly consumed by the forward mutation and evaluation. Since the parameter $K \leq N$, the time complexity of each generation is $O(N)$. From the Table 3, we can find that, for BMHGA, NDSC-GA and STHGA, the time complexity of each generation is $O(N)$ respectively that the algorithm which executes less generations will be faster. mc-ACO also adopts the greedy method at initialization, so the time complexity is the sum of two parts. Each iteration of mc-ACO is solvable in exponential time. Besides, time complexity $O(N!)$ reflects that using the EXACT algorithm to get the best solution is the most costly. The space complexity of NDSC-GA is $O(N^2)$ since the specific structure of individual. The algorithms, mc-ACO and EXACT, yield an $O(N)$ space complexity. Because the size of chromosome is dynamically determined by the number of assigned sets for each sensor, BMHGA and STHGA is solved in $O(E_{max} \times N)$ space. Usually, E_{max} is smaller than N, the space complexity is $O(N)$ for the above two algorithms.

In the experiments, the parameter configurations of these algorithms are according to the corresponding references, which are shown in Table 4. For NDSC-GA, the population size M is set as 50 and the number of prescribed groups is set as $P = N/10$. The parameters of mc-ACO are the number of artificial ants M, exploitation rate q_0, pheromone decay coefficient φ, global pheromone update coefficient ρ and heuristic coefficient β. The population size M, scheduling coefficient K_1 and K_2, mutation probability μ and mutation generations g_m are set in STHGA. For the proposed BMHGA, if there is no other declaration, the population size M is fixed at 3, the interval for performing backward mutation G_m is 1, and the parameter $K = \frac{1}{10}N$. Since the processes and time complexity of these algorithms are extremely distinct, for a fair comparison, a maximum running time(maxRT) is used as the stopping criterion, except EXACT algorithm. If the lifetime of WSNs achieves T (the upper limit lifetime of the network), the algorithm also stops. All algorithms are executed 100 times for each case, with maxRT being 1 minute(60,000ms).

To evaluate the performance of different algorithms, firstly, a set of 13 cases with different number of sensors and different size of monitoring field in Table 5 are conducted. Table 6 lists the results obtained by the five algorithms on each case, including the average time(Time) in microsecond(ms) used for meeting the stop condition in each run, the success rate(ok) and the mean results(Mean). In all the cases, our proposed algorithm consistently reaches the superior performance than others, which implies that our algorithm can fast evolve to better chromosome and converge towards global optima quickly. The main reason is that our genetic operations especially the mutation operations can fully exploit the diversity of the chromosome and effectively avoid local optima. It is to be noted that, the proposed BMHGA produces

TABLE 4. Parameter configurations.

BMHGA			NDSC-GA		mc-ACO					STHGA				
M	K	G_m	M	P	M	q_0	φ	ρ	β	M	K_1	K_2	μ	g_m
3	$N/10$	1	50	$N/10$	10	0.9	0.5	0.5	2	3	$N/10$	5	0.5	100



(a) (b)

FIGURE 10. Comparison of running time and success rate with R increasing from 4 to 20 for the test cases that $N = 500$, $L = W = 50$.

TABLE 5. Specification of WSN Cases for experiment.

No.	N	$L \times W$	R	E_{max}	T
1	200	20×20	8	5	67
2	200	30×30	8	5	32
3	300	30×30	8	5	55
4	300	40×40	8	5	25
5	500	40×40	8	5	56
6	500	50×50	8	5	28
7	800	50×50	8	5	52
8	1000	50×50	8	5	78
9	1500	50×50	8	5	104
10	2000	50×50	8	5	122
11	3000	60×60	8	5	104
12	4000	80×80	8	5	80
13	5000	100×100	8	5	73

significantly better results than other algorithms on the cases $10 - 13$. These results verify that, even in large-scale WSNs, the proposed genetic operations can enhance the performance of GA powerfully.

We further compare BMHGA with the four evolutionary algorithms by varying the sensing range. Fig.10 depicts the experimental results including consumed time and success rate. Fig.10(a) reports that BMHGA outperforms other algorithms and its running time is not affected by the increasing sensing range. It can be observed that the running time of STHGA and NDSC-GA is increasing when using greater sensing range. This is principally because the lifetime of network is increasing with the increasing R that takes more generations to find out all the complete cover sets. For mc-ACO, the consumed time is in a relatively stable state that given $R \leq 17$, but the time is much larger than others when $R = 18, 19, 20$. From Fig.10(b), we can deduce the reason behind this is that, under the cases that $R = 18, 19, 20$, it is difficult for mc-ACO to find the optimal solution within the maxRT.

B. INVESTIGATION OF PARAMETERS

There is a main parameter K in the BMHGA, which is the number of sensors selected to be mutated in forward mutation. This section investigates the relationship among the number of sensors N, the number of selected sensors K in mutation operation and the consumed time of BMHGA. Furthermore, the suitable value of K is revealed here.

The running time of BMHGA under the search scenarios with the two varying parameters is reported in Fig.11. Comparing the running time with respect to different parameter values, it can be found that the time increases as N increases and K decreases. The BMHGA costs a lot of time to obtain the optimal result when $K = 5$, especially in the scenario that $N = 500$ and $K = 5$, the most time is consumed. This observation proves that the forward mutation plays an important role in searching optimal result. Meanwhile, from the figure, we can also find that the running time rises gradually when the number of sensors becomes larger. However, if the selected genes becomes more, the algorithm will be accelerated as more genes are changed in the chromosome without fitness value falling.

In addition, BMHGA is tested with $K \in [5, 400]$, respectively, with the other parameter N being fixed. Fig.12 illustrates that the plot of the running time is generally partitioned to three phases. At start there is a very sharply decline and the least time is reached when K is equal to a certain value, e.g. K_l. Then, the algorithm keeps a stable performance between the lowest point, e.g., K_l and a certain K value, e.g., K_i. When K exceeds the K_i value, the running time slowly increases as K rises. We further compare the performances of 8 cases and note that as N increases, greater the range of second phase becomes and much slower the raise in running time for third phase is. This indicates that setting K in the range of $[K_l, K_i]$ can provide a good performance

TABLE 6. Performance comparison of BMHGA with other algorithms.

CASE	BMHGA			NDSC-GA			STHGA			mc-ACO			EXACT	
No.	Time	*ok*	Mean	Time	*ok*	Mean	Time	*ok*	Mean	Time	*ok*	Mean	Time	Result
1	**56**	100	67	324	100	67	1064	100	67	137	100	67	4784	67
2	**331**	100	32	477	100	32	979	100	32	338	100	32	2953	32
3	**748**	100	55	1405	100	55	4426	100	55	10905	99	54.99	13922	55
4	**484**	100	25	1396	100	25	2061	100	25	1181	100	25	10746	25
5	**1830**	100	56	2750	100	56	9304	100	56	46275	39	55.39	37803	56
6	**2226**	100	28	4126	100	28	9941	100	28	36084	72	27.72	42730	28
7	**4346**	100	52	14296	100	52	35480	86	51.85	59704	2	50.26	79304	52
8	**6353**	100	78	15001	100	78	34749	100	78	60000	0	73.82	140378	78
9	**7622**	100	104	19068	100	104	46559	98	103.98	43671	56	103.54	224735	104
10	**12543**	100	122	27874	100	122	54908	82	120.89	55436	21	118.53	403466	122
11	**19735**	100	104	48963	100	104	57920	56	103.52	58754	12	102.84	624922	104
12	**27896**	100	80	57089	42	79.40	60000	0	75.12	60000	0	78.33	1063469	80
13	**44178**	100	73	60000	0	69.32	60000	0	64.93	60000	0	70.58	2583891	73

TABLE 7. Performance comparison with different genetic operations.

CASE	BMHGA			no_greedy_initialization			no_replace_operation			no_backward_mutation		
No.	Time	*ok*	Mean	Time	*ok*	Mean	Time	*ok*	Mean	Time	*ok*	Mean
1	**56**	100	67	257	100	67	62	100	67	13590	80	66.69
2	**331**	100	32	466	100	32	412	100	32	60000	0	28.67
3	**748**	100	55	1561	100	55	1169	100	55	60000	0	52.36
4	**484**	100	25	1131	100	25	703	100	25	60000	0	22.12
5	**1830**	100	56	3388	100	56	2338	100	56	60000	0	53.41
6	**2226**	100	28	3010	100	28	3940	100	28	60000	0	22.31
7	**4346**	100	52	10928	100	52	9308	100	52	60000	0	45.27
8	**6353**	100	78	18211	100	78	10872	100	78	60000	0	75.43
9	**7622**	100	104	25314	100	104	9466	100	104	10429	100	104

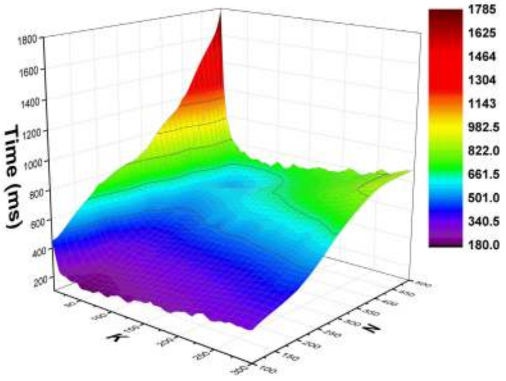


FIGURE 11. Comparison of the time used by BMHGA when *K* increases from 5 to 300 and *N* increases from 100 to 500 for the test cases that $L = W = 30$, $R = 8$.

for the algorithm, and meanwhile, the boundary values of the range, i.e., K_l and K_i, are depended on the number of sensors N. By observing these plots, approximately $K = \frac{1}{10}N$ is recommended in this paper.

Besides K, the interval of backward mutation G_m and the number of chromosomes M in a population are discussed. In Fig.13, the search speed of using different G_m on different K and N is depicted and compared, where the vertical axis is the executing time to obtain the optimal result. The parameter G_m decides the frequency that the backward mutation is performed. When using a large G_m, the algorithm can be trapped in a local optimum for a long time, which is not efficient. When using a small G_m, if the K is too small, the algorithm takes long time to achieve a local optimum and executes backward mutation frequently, which is also not efficient.

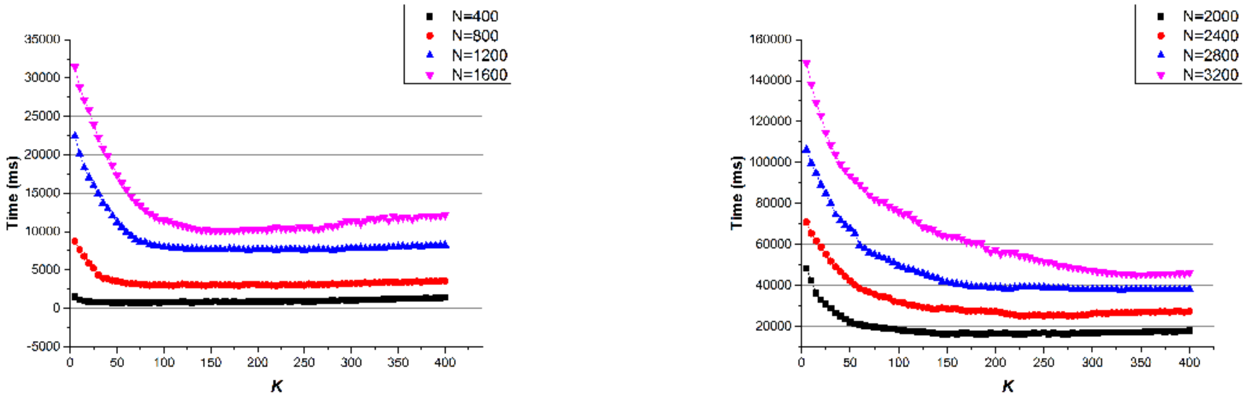
From Fig.13, it can be observed that, in different scale of WSNs, when the values of K are great enough such as $K \geq \frac{1}{10}N$, setting G_m as 1 can help improving the search efficiency of BMHGA. Therefore, $G_m = 1$ is recommended to be suitable setting.

The effect of different M on the performance of BMHGA is plotted in Fig.14, where the horizontal axis is M and the vertical axis is the running time(Time). It can be observed that the proper range of M is [2,4] for each case. A larger M will lead to more time needed in the process for each generation. When K is smaller than 20 and $M = 2$, since the diversity and the number of chromosomes of a population is decreased, more generations are required to obtain a optimal result. Hence, $M = 3$ is regarded as the proper value.

What's more, from the Fig.13 and Fig.14, we can find that the plots are almost overlapped in each case when $K > \frac{1}{10}N$. It demonstrates the correctness of the proper K setting that $K = \frac{1}{10}N$.

C. EFFECTS OF GENETIC OPERATIONS

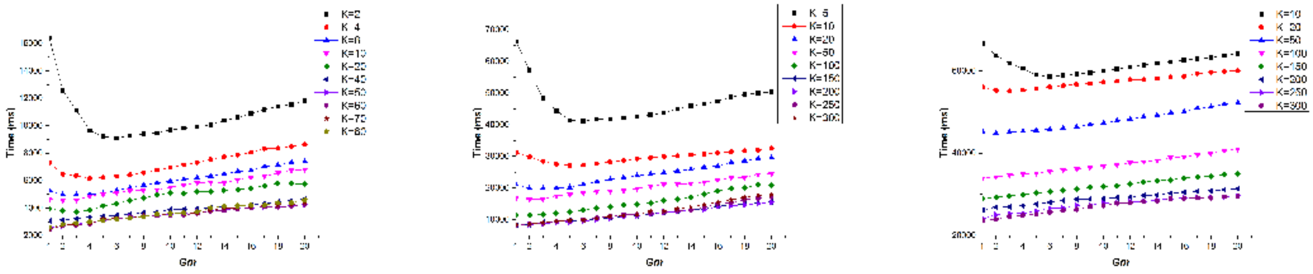
In BMHGA, greedy initialization is used for construction of an initial population that can speed up searching for better chromosomes. Forward mutation is the basic operation for improving the value of fitness. Backward mutation is able to escape from local optima, where the replace operation is



(a) $N = \{400, 800, 1200, 1600\}$

(b) $N = \{2000, 2400, 2800, 3200\}$

FIGURE 12. Comparison of the time used by BMHGA when *K* increases from 5 to 400 for the test cases that $G_m = 1$, $M = 3$ and $N \in \{400, 800, 1200, 1600, 2000, 2400, 2800, 3200\}$, $L = W = 30$, $R = 8$.

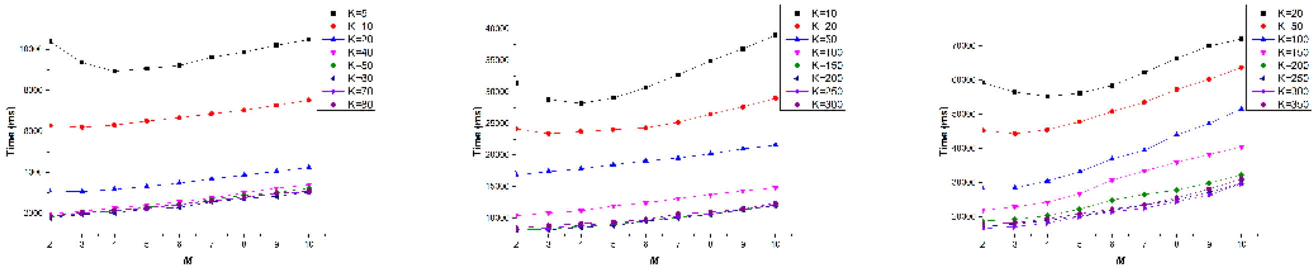


(a) $N = 500$

(b) $N = 1500$

(c) $N = 2500$

FIGURE 13. Comparison of the time used by BMHGA when G_m increases from 1 to 20 for the test cases that $N = \{500, 1500, 2500\}$, $L = W = 30$, $R = 8$.



(a) $N = 500$

(b) $N = 1500$

(c) $N = 2500$

FIGURE 14. Comparison of the time used by BMHGA with $G_m = 1$ and different *M*, *K* for the test cases that $N = \{500, 1500, 2500\}$, $L = W = 30$, $R = 8$.

designed for the incomplete cover set. In order to demonstrate the effectiveness of the proposed genetic operations, we compare the performances of BMHGA with default settings, without greedy initialization and without replace operation and without backward mutation that is shown in Table 7. The four algorithms are tested through executing the NO.1-9 cases in Table 5. Though the former three algorithms can gain the optimal results with 100% success rate, the convergence speed of BMHGA with default settings is always fastest. It validates that greedy initialization and replace operation can speed up the process of searching to get optimal results. The last algorithm obtains global optimal solutions in two cases. The success rates of case 1 and case 9 are 80% and 100% respectively. It is evident that backward mutation can jump out of local optima with great efficiency, and hence can support the global search.

To further observe the effect of the density of the network on the backward mutation and greedy initialization, we compare the fitness values of initial chromosomes and average executing times of backward mutation using vary redundancy

TABLE 8. Comparison of decreasing values of redundancy rate η.

CASE						BMHGA				
No.	N	$L \times W$	R	T	η	F_{init}	Backward	Time	ok	Mean
1	1000	20 × 20	8	100	5.02	98	1.61	711	100	100
2	900	20 × 20	8	100	4.52	98	2.19	734	100	100
3	800	20 × 20	8	100	4.01	97	2.49	744	100	100
4	700	20 × 20	8	100	3.51	92	4.41	1131	100	100
5	650	20 × 20	8	100	3.26	90	19.46	1604	100	100
6	600	20 × 20	8	100	3.01	88	22.74	3010	100	100
7	550	20 × 20	8	100	2.76	79	26.92	3442	100	100
8	500	20 × 20	8	100	2.51	78	233.42	18211	100	100
9	480	20 × 20	8	100	2.38	77	504.52	30600	100	100

rate. In the paper, the redundancy rate η is calculated by

$$\eta = \frac{\sum_{i=1}^{N} e_i}{T} \times \frac{\pi R^2}{LW} \tag{14}$$

where e_i is the initial energy of sensor s_i.

As illustrated in Table 8, the redundancy rate η decreases as the number of sensors N decreases with $L \times W = 20 \times 20, R = 8, T = 100$. The fitness value of the initial population F_{init}, the executing times of backward mutation(Backward) and the average running time(Time) are related to the redundancy rate η. Since the density of sensors in the area reduces, the number of complete cover sets obtained by greedy initialization decreases, whereas more backward mutation operations are needed that means the probability of being trapped in local optima increases. This observation proved that the complexity of the problem depends on the redundancy rate. A small η will make a hard search, which would result in local optima and hence increase the executing times of backward mutation(Backward). On the other hand, when η is a large value, the global optima results can be obtained quickly. Moreover, this conclusion can be further verified by comparing the running time(Time) of cases with different η.

VI. CONCLUSION

In the paper, we have designed a modified genetic algorithm BMHGA for extending the lifetime of heterogeneous WSNs. Different from the previous works, BMHGA uses the two-level structured chromosomes to represent solutions. The benefit of this chromosome is the specific scheduling of each set and the detailed energy assignment of each sensor are reflected in each chromosome. A greedy method is designed to produce an initial population. This method only costs $O(c \times N)$ time complexity but can accelerate the searching speed greatly. The novel mutation operations involving forward mutation and backward mutation are developed. The forward mutation can guarantee the diversity of chromosomes and the backward mutation can help jumping out of local optima effectively. Simulations results are verified that BMHGA is superior in the solution quality and computation speed to other methods on different size of networks, in particular, the advantages of BMHGA are even more obvious in large-scale networks.

As part of our future work, we will further investigate the heterogeneous WSNs. Since there is still possibility for reducing the time complexity of the greedy initialization, in the future, we will seek a method which has faster convergence speed so as to apply the proposed algorithm to larger scale network. Moreover, BMHGA may be modified to consider the sensors with different sensing ranges and different transmission range. And to be practical for more application scenarios, improving the fault tolerance of the algorithm is necessary. Another direction is based on BMHGA, to design a new algorithm for energy-efficient area coverage in 3D WSNs.

REFERENCES

[1] C. Tang, H. F. Rashvand, G. Y. Tian, P. Hu, A. I. Sunny and H. Wang, "Structural health monitoring with WSNs," in *Wireless Sensor Systems for Extreme Environments: Space, Underwater, Underground, and Industrial*. Hoboken, NJ, USA: Wiley, 2017, ch. 18, pp. 383–408.

[2] A. B. Noel, A. Abdaoui, T. Elfouly, M. H. Ahmed, A. Badawy, and M. S. Shehata, "Structural health monitoring using wireless sensor networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1403–1423, 3rd Quart., 2017, doi: 10.1109/COMST.2017.2691551.

[3] M. Z. A. Bhuiyan, G. Wang, J. Wu, J. Cao, X. Liu, and T. Wang, "Dependable structural health monitoring using wireless sensor networks," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 4, pp. 363–376, Jul. 2017, doi: 10.1109/TDSC.2015.2469655.

[4] A. Diaz-Ramirez, F. Bonino, and P. Mejia-Alvarez, "Human detection and tracking in healthcare applications through the use of a network of sensors," in *Human Behavior Understanding in Networked Sensing: Theory and Applications of Networks of Sensors*. Cham, Switzerland: Springer, 2014, pp. 171–190.

[5] S. Tennina, M. Santos, A. Mesodiakaki, P.-V. Mekikis, E. Kartsakli, A. Antonopoulos, M. Di Renzo, A. Stavridis, F. Graziosi, L. Alonso, and C. Verikoukis, "WSN4QoL: WSNs for remote patient monitoring in e-Health applications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.

[6] N. Dey, A. S. Ashour, F. Shi, S. J. Fong, and R. S. Sherratt, "Developing residential wireless sensor networks for ECG healthcare monitoring," *IEEE Trans. Consum. Electron.*, vol. 63, no. 4, pp. 442–449, Nov. 2017, doi: 10.1109/TCE.2017.015063.

[7] W. Jing, H. Zhao, X. Lin, and X. Shen, "Selectively iterative particle filtering and its applications for target tracking in WSNs," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Atlanta, GA, USA, Dec. 2013, pp. 504–509.

[8] S. M. Dehnavi, M. Ayati, and M. R. Zakerzadeh, "Three dimensional target tracking via underwater acoustic wireless sensor network," in *Proc. Artif. Intell. Robot. (IRANOPEN)*, Qazvin, Iran, Apr. 2017, pp. 153–157.

[9] J. Luo, Z. Zhang, C. Liu, and H. Luo, "Reliable and cooperative target tracking based on WSN and WiFi in indoor wireless networks," *IEEE Access*, vol. 6, pp. 24846–24855, 2018, doi: 10.1109/ACCESS.2018.2830762.

[10] L. Wu and T. Wang, "Design of wireless sensor networks-based equipment materials transport and storage environment monitoring system," in *Proc. Int. Conf. Netw. Inf. Syst. Comput.*, Wuhan, China, Jan. 2015, pp. 206–209.

[11] S. Choi, J. Eum, and S. Park, "Compatibility analysis between field disturbance sensors and intelligent transport system in 5.8 GHz adjacent bands," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Jeju, South Korea, Oct. 2017, pp. 1050–1053.

[12] P. Prasad, "Recent trend in wireless sensor network and its applications: A survey," *Sensor Rev.*, vol. 35, no. 2, pp. 229–236, Mar. 2015.

[13] Y. Wang, H. Chen, X. Wu, and L. Shu, "An energy-efficient SDN based sleep scheduling algorithm for WSNs," *J. Netw. Comput. Appl.*, vol. 59, pp. 39–45, Jan. 2016.

[14] W. Chao, C. Luo, L. Jia, Q. Zhang, and J. Yu, "Domatic partition in homogeneous wireless sensor networks," in *Proc. Int. Conf. Wireless Algorithm Syst. Appl. (WASA)*, 2015, pp. 508–517.

[15] M. Khan, B. N. Silva, and K. Han, "Internet of Things based energy aware smart home control system," *IEEE Access*, vol. 4, pp. 7556–7566, 2016, doi: 10.1109/ACCESS.2016.2621752.

[16] A. K. Kaushik and A. I. Khan, "An improved fuzzy-control based energy efficient heterogeneous wireless sensor network," in *Proc. 3rd Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Feb. 2016, pp. 610–615.

[17] B. T. Hartman, R. D. Tatum, and M. J. Bays, "Heterogeneous sensor-robot team positioning and mixed strategy scheduling," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 3137–3144.

[18] S. Sharma, R. K. Bansal and S. Bansal, "Energy efficient clustering techniques for heterogeneous wireless sensor networks: A survey," *J. Mobile Comput. Commun. Mobile Netw.*, vol. 3, pp. 1–9, Sep. 2016.

[19] M. Cardei and D.-Z. Du, "Improving wireless sensor network lifetime through power aware organization," *Wireless Netw.*, vol. 11, no. 3, pp. 333–340, May 2005.

[20] O. Kramer, *Genetic Algorithm Essentials*. Cham, Switzerland: Springer, 2017.

[21] X. Shu, Q. Su, Q. Wang, and Q. Wang, "Optimization of resource-constrained multi-project scheduling problem based on the genetic algorithm," in *Proc. 15th Int. Conf. Service Syst. Service Manage. (ICSSSM)*, Jul. 2018, pp. 1–6.

[22] J. Li, M. Zhou, Q. Sun, X. Dai, and X. Yu, "Colored traveling salesman problem," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2390–2401, Nov. 2015, doi: 10.1109/TCYB.2014.2371918.

[23] M. Poloczek, G. Schnitger, D. P. Williamson, and A. van Zuylen, "Greedy algorithms for the maximum satisfiability problem: Simple algorithms and inapproximability bounds," *SIAM J. Comput.*, vol. 46, no. 3, pp. 1029–1061, Jan. 2017.

[24] T. Ma, Z. Xu, M. Hempel, D. Peng, and H. Sharif, "Performance analysis of a novel low-complexity high-precision timing synchronization method for wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 9, pp. 4758–4765, Sep. 2014.

[25] R. Elhabyan, W. Shi, and M. St-Hilaire, "Coverage protocols for wireless sensor networks: Review and future directions," *J. Commun. Netw.*, vol. 21, no. 1, pp. 45–60, Feb. 2019, doi: 10.1109/JCN.2019.000005.

[26] K. Deschinkel and M. Hakem, "A near optimal algorithm for lifetime optimization in wireless sensor networks," in *Proc. 2nd Int. Conf. Sensor Netw-s (SENSORNETS)*, 2013, pp. 197–202.

[27] S. Henna, "Energy efficient fault tolerant coverage in wireless sensor networks," *J. Sensors*, vol. 2017, pp. 1–11, Apr. 2017.

[28] W. Yan, M. Cao, Y. Wu, and J. Zhang, "Greedy game algorithms for solving SET K-cover problem in HWSNs," *IEEE Access*, vol. 6, pp. 65604–65619, 2018, doi: 10.1109/ACCESS.2018.2871963.

[29] X.-M. Hu, J. Zhang, Y. Yu, H. S.-H. Chung, Y.-L. Li, Y.-H. Shi, and X.-N. Luo, "Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks," *IEEE Trans. Evol. Comput.*, vol. 14, no. 5, pp. 766–781, Oct. 2010, doi: 10.1109/TEVC.2010.2040182.

[30] A. K. Mir, M. Zubair, and I. M. Qureshi, "Lifetime maximization of wireless sensor networks using particle swarm optimization," *TURKISH J. Electr. Eng. Comput. Sci.*, vol. 24, no. 1, pp. 160–170, 2016.

[31] J. Roselin, P. Latha, and S. Benitta, "Maximizing the wireless sensor networks lifetime through energy efficient connected coverage," *Ad Hoc Netw.*, vol. 62, pp. 1–10, Jul. 2017.

[32] Y. E. E. Ahmed, K. H. Adjallah, R. Stock, I. Kacem, and S. F. Babiker, "NDSC based methods for maximizing the lifespan of randomly deployed wireless sensor networks for infrastructures monitoring," *Comput. Ind. Eng.*, vol. 115, pp. 17–25, Jan. 2018.

[33] L. Ding, W. Wu, J. Willson, L. Wu, Z. Lu, and W. Lee, "Constant-approximation for target coverage problem in wireless sensor networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Mar. 2012, pp. 1584–1592.

[34] A. Shan, X. Xu, and Z. Cheng, "Target coverage in wireless sensor networks with probabilistic sensors," *Sensors*, vol. 16, no. 9, p. 1372, 2016.

[35] F. Castaño, A. Rossi, M. Sevaux, and N. Velasco, "An exact approach to extend network lifetime in a general class of wireless sensor networks," *Inf. Sci.*, vols. 433–434, pp. 274–291, Apr. 2018.

[36] B. Diop, D. Diongue, and O. Thiaré, "Greedy algorithms for target coverage lifetime management problem in wireless sensor networks," *Int. J. Control Autom.*, vol. 8, no. 2, pp. 232–250, Feb. 2015.

[37] J. Yu, Y. Chen, L. Ma, B. Huang, and X. Cheng, "On connected target k-coverage in heterogeneous wireless sensor networks," *Sensors*, vol. 16, no. 1, pp. 104–124, 2016.

[38] J.-S. Lee and C.-L. Teng, "An enhanced hierarchical clustering approach for mobile sensor networks using fuzzy inference systems," *IEEE Internet Things J.*, vol. 4, no. 4, pp. 1095–1103, Aug. 2017.

[39] H. El Alami and A. Najid, "Fuzzy logic based clustering algorithm for wireless sensor networks," *Int. J. Fuzzy Syst. Appl.*, vol. 6, pp. 351–371, Apr. 2017.

[40] H. El Alami and A. Najid, "ECH: An enhanced clustering hierarchy approach to maximize lifetime of wireless sensor networks," *IEEE Access*, vol. 7, pp. 107142–107153, 2019.

[41] J.-H. Zhong and J. Zhang, "Energy-efficient local wake-up scheduling in wireless sensor networks," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, New Orleans, LA, USA, Jun. 2011, pp. 2280–2284.

[42] C.-C. Liao and C.-K. Ting, "A novel integer-coded memetic algorithm for the set k -cover problem in wireless sensor networks," *IEEE Trans. Cybern.*, vol. 48, no. 8, pp. 2245–2258, Aug. 2018, doi: 10.1109/TCYB.2017.2731598.

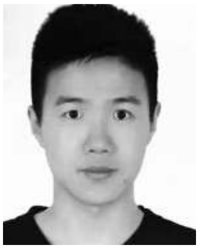
[43] D. Tian and N. D. Georganas, "Connectivity maintenance and coverage preservation in wireless sensor networks," *Ad Hoc Netw.*, vol. 3, no. 6, pp. 744–761, Nov. 2005.

[44] H. H. Zhang and J. C. Hou, "Maintaining sensing coverage and connectivity in large sensor networks," *Ad Hoc Sens. Wireless Netw.*, vol. 1, no. 2, pp. 89–124, 2005.

[45] S. Jannu and P. K. Jana, "A grid based clustering and routing algorithm for solving hot spot problem in wireless sensor networks," *Wireless Netw.*, vol. 22, no. 6, pp. 1901–1916, Aug. 2016.

[46] R. Kalkreuth, G. Rudolph, and J. Krone, "Improving convergence in Cartesian genetic programming using adaptive crossover, mutation and selection," in *Proc. IEEE Symp. Ser. Comput. Intell.*, Cape Town, South Africa, Dec. 2015, pp. 1415–1422.

JINGJING LI received the Ph.D. degree in computer science from The Hong Kong Polytechnic University, Hong Kong, in 2013. She is currently a Lecturer with the School of Computer Science, South China Normal University, Guangzhou, China. Her current research interests include evolutionary algorithm, energy efficient routing, and object tracking for wireless sensor networks.



ZHIPENG LUO received the B.E. degree in computer science and technology from South China Normal University, Guangzhou, China, in 2017, where he is currently pursuing the M.E. degree with the School of Computer Science. His research interests include heuristic algorithms and wireless networks.

JING XIAO received the B.S. and M.S. degrees in computer science from Wuhan University, Wuhan, China, in 1997 and 2000, respectively, and the Ph.D. degree from the School of Computing, National University of Singapore, Singapore, in 2005. She is currently a Professor with the School of Computer Science, South China Normal University, Guangzhou, China. Her research interests include intelligence computing, information retrieval, and data mining.