Randomized and Optimal Algorithms for $k$-Lifetime Dominating Set in Wireless Sensor Networks

WEIZHONG LUO$^{1}$, JUNBIN LIANG$^{2}$, AND TIANSHU WANG$^{3}$

$^{1}$School of Computer Science and Engineering, Huizhou University, Huizhou, Guangdong 516007, China
$^{2}$Guangxi Key Laboratory of Multimedia Communications and Network Technology, School of Computer and Electronics Information, Guangxi University, Nanning 530004, China
$^{3}$Neusoft Group Company Ltd., Nanning 530004, China

Corresponding author: Weizhong Luo (lwz@hzu.edu.cn)

This work was supported in part by the Ph.D. Startup Fund of the Natural Science Foundation of Guangdong Province, China, under Grant 2015A030310445; in part by the Ph.D. Research Startup Foundation of Huizhou University under Grant C513.0211; and in part by the Guangxi Key Research and Development Plan Project under Grant AB19259006.

ABSTRACT In wireless sensor networks, rotating dominating set is an efficient method for balancing the energy consumption of nodes, and thereby extending the network operational time. This method can be abstracted as $k$-Lifetime Dominating Set in bipartite graph, that partitions the set of graph vertices representing sensors into $k$ disjoint dominating sets. However, the considered problem has been proven to be NP-hard, and there is no hope of solving it in polynomial time unless $P = NP$. Existing studies mainly focus on developing approximation or heuristic algorithms, which usually cannot guarantee a solution for a given problem yes instance. In this study, we first propose a randomized algorithm that can generate a solution with guaranteed probability $1-\varepsilon$ ($0 < \varepsilon < 1$). Using the color coding method, we show that the randomized algorithm can be improved to guarantee the generation of a solution for a given problem yes instance in exponential time. Based on the idea of randomized partition, we further present a more practical centralized greedy algorithm, and then a distributed implementation. Simulation results indicate that the centralized algorithm can efficiently generate optimal solutions for almost all the given problem instances if the partition redundancy is above a certain limit. Compared with existing algorithm, the centralized algorithm increases the number of dominating sets by factors between 0% and 21%.

INDEX TERMS Dominating set, NP-hard, color coding, random partition, optimal algorithms.

I. INTRODUCTION

Wireless sensor networks provide new applications for environmental monitoring and military surveillance. In some cases, ground access to the monitored area is difficult or dangerous, and the only way to install the sensors is to deploy them from an aircraft. The number of sensors deployed is usually higher than required because of the lack of precise sensor placement.

The sensors are charged with a battery whose energy is limited and placed once. Hence, one of the main issues in sensor networks is prolonging the network lifetime by reducing the energy consumption of the sensor nodes [1]–[3].

Node activity scheduling is a common method for saving energy. This method schedules sensor activity so that for each sensor, the active state, in which it performs its monitoring task, alternates with a low-energy sleep state. Obviously, in the sleep state, the sensor consumes much less energy than in the active state.

The set of sensors is divided into disjoint sets, such that every set completely covers all targets. These disjoint sets are activated successively such that at any moment in time, only one set of sensors is active. The sensors from the active set are in active mode, and all other sensors are in sleep mode.

The approach above can be abstracted into $k$-Lifetime Dominating Set in bipartite graph. Given a bipartite graph, $G = (S \cup T, E)$, where $S$ represents the set of sensors and $T$ represents the set of targets. A dominating set $D \subseteq S$ is a subset of sensors $S$, such that every target $t \in T$ has
a neighbor in $D$. A $k$-lifetime dominating set is a partition $\{S_1, S_2, \ldots, S_k\}$ of $S$ such that every $S_i (1 \leq i \leq k)$ is a dominating set of $T$. $S_i$ is called the dominator and $k$ is called the network lifetime. By rotating each dominating set periodically, the energy consumption of the sensors can be greatly balanced and the network lifetime can be prolonged.

This problem has been proven NP-hard. In accordance with state-of-the-art computational complexity theory, we have to consider algorithms with exponential running time to solve it optimally. However, exponential growth quickly becomes prohibitive when algorithms are run in practice. Most previous studies investigated the considered problem from the perspective of centralized approximation or heuristics, which usually cannot generate the optimal solution (a partition $\{S_1, S_2, \ldots, S_k\}$ of $S$ such that every $S_i$ is a dominating set and $k$ is maximized), or guarantee a solution for a given problem yes instance. The study of efficiently generating an optimal solution for a given problem instance is much less developed. Moreover, in large-scale distributed systems, the use of centralized algorithms based on a global view of networks is infeasible. In this study, we mainly investigate the problem from an optimal algorithm perspective and develop algorithms that can find the optimal (not only approximate) solution practically.

First, we present a randomized R-LDSBG algorithm, where each sensor simply assigns itself to a dominator chosen uniformly at random from the set of all possible dominators. The effect of the algorithm is determined by the iteration times of the random assignment process, which can generate a solution for any given yes instance with probability $1 - \epsilon$ ($0 < \epsilon < 1$) if exponential iterations are permitted. Moreover, the randomized algorithm can be de-randomized and determinately generate a solution for any input yes instance. The de-randomization is obtained using the method of color coding and deterministic construction of the color coding scheme. To the best of our knowledge, this is the first time that the considered problem has been solved optimally.

The proposed de-randomized O-LDSBG algorithm guarantees an optimal solution in theory. However, this is unpractical because of its exponential iterations. Based on the idea of randomized partition, we further propose a more practical centralized greedy algorithm, C-LDSBG, to obtain the optimal solution. The assignment process in the R-LDSBG algorithm is completely random, which increases the assignment conflict and number of assignment iterations. To decrease assignment conflict, we propose two main methods. One method is to introduce an object function to evaluate the attractiveness of dominators. Each sensor assigns itself to the dominator with a probability that is positively proportional to the weight of the intersection between the set of targets monitored by the sensor and the set of targets monitored by the dominator. The intersection is weighted based on how likely it is to be monitored by another sensor during the assignment process and the size of the intersection. Another method is to conduct the assignment process in the order of the degree of sensors, from high to low. Simulation results indicate that the centralized greedy algorithm C-LDSBG can achieve optimal solutions efficiently for almost all the given instances when the redundancy degree for the partition task is more than 10. The average iteration times required to obtain the optimal solutions is approximately 450. Compared with existing algorithm, the C-LDSBG algorithm increases the number of dominators by factors between 0% and 21%.

Further, we present a distributed implementation of the centralized greedy algorithm. Each sensor simply assigns itself, in turn, to a dominator. The algorithm is completely distributed, and communication is only required to allow each sensor to know its neighbors.

The main contributions of this study and the novelty of the proposed methods are as follows.

- Using the method of color coding, we present an exact O-LDSBG algorithm that can find the optimal (not only approximate) solution for a given problem instance in theory. This problem can be solved optimally for the first time.
- Based on the idea of random partition, we develop a randomized algorithm C-LDSBG that can generate optimal solutions for almost all given problem instances in practice when the partition redundancy is above a certain limit.
- We further develop, for the first time, a distributed algorithm, D-LDSBG, for the considered problem, where each sensor can only communicate with its neighbors.

The remainder of this paper is organized as follows. In Section II, we provide an overview of relevant literature. In Section III, we describe our computation model and formally define the $k$-Lifetime Dominating Set problem in bipartite graph. Randomized and optimal algorithms are presented in Section IV. In Section V, a centralized greedy algorithm is presented. Section VI presents a distributed greedy algorithm. In Section VII, a simulation experiment is presented. Finally, Section VIII concludes this paper.

II. RELATED WORKS

The importance of energy efficiency in wireless sensor networks has led to a plethora of studies on $k$-Lifetime Dominating Set.

The optimal version of $k$-Lifetime Dominating Set is known as the domatic partition problem. Poon et al. showed that the 3-domatic partition problem is NP-complete on planar bipartite graphs, and the domatic partition problem is NP-complete on co-bipartite graphs [4].

Based on the graph coloring method, Mahjoub and Matula solved the domatic partition problem in random geometric graph and provided up to $(\ell + 1)$ disjoint $(1 - \epsilon)$ dominating sets on a large range of experimental graphs, where $\ell$ is the minimum degree of the graph [5]. They carried out further research by proposing a more practical solution to the distributed $(\ell + 1)$ domatic partition problem based on the localized graph coloring method [6].

Communication in wireless sensor networks is often modeled as the so-called unit disk graph (UDG). In UDG, there is
an edge between two nodes if their Euclidean distance is most
one. Pemmaraju and Pirwani proposed a method of uniform
partition and presented deterministic, distributed algorithms
for finding a k-domatic partition of size at least a constant
fraction of the largest possible (k-1)-domatic partition for
any k > 1 [7]. Pandit et al. first presented a constant-factor
distributed algorithm that can be implemented in O(log n)
rounds of communication on the UDG of order n [8]. Yu et al.
proposed another constant-factor approximation algorithm
using the skyline property of uniform radius disks [9]. Misra
and Mandal studied a distributed domatic partition based
scheme for energy-efficient clusterhead rotation in UDG [10].

Variations of domatic partition have also been studied
comprehensively. Liang studied the k-tolerant domatic part-
ition from both algorithmic complexity and graph theoretic
points of view. They showed that it is NP-complete to decide
whether the k-tolerant domatic number of a bipartite graph
is at least three, and presented a polynomial time algorithm
that approximates the k-tolerant domatic number of a graph of
order n within a factor of ((1/k + o(1)) ln n) [11]. Lee et al.
demonstrated that the total 3-domatic partition problem on
planar graphs is NP-complete [12]. Misra and Mandal stud-
yed the connected domatic partition problem and developed
a distributed algorithm to construct a connected domatic
partition with a guaranteed size [13]. Based on the connect-
ivity decomposition method, Censor-Hillel et al. Presented
a distributed algorithm for a generalization of the connected
domatic partition problem [14]. Pino et al. studied domatic
partition in wireless sensor networks, and introduced three
local search based algorithms to increase the network life-
time, where sensors can have different initial energies [15].

Maximizing the number of disjoint dominating sets to
increase the lifetime of networking while guaranteeing com-
plete coverage of the monitored targets was considered in [16]
and [17]. The problem can be abstracted into the k-Lifetime Dominating Set problem in bipartite graph. Both
studies focused on the design of heuristic algorithms and
did not provide any worst-case analysis or stringent bounds.
A √n-approximation algorithm for this problem was pro-
posed in [18], where n denotes the number of targets to be
monitored. Later, Pananjady et al. proposed a polynomial
time ln n-approximation algorithm through suitably defined
hypergraph coloring [19]. They further demonstrated a (1 +
ε)-approximation algorithm for the 2-dimensional geometric
case, where each sensor can monitor the circular area around
itself with a given radius. However, all the above heuristic or
approximate algorithms are centralized and hence inappropi-
ate in practice. Pananjady et al. proposed an online algorithm
with a competitive ratio of ln n [20] with prior knowledge of
the minimum degree of the targets. Emek et al. developed
an online algorithm for the problem that guarantees a poly-
logarithmic (O(ln² n)) competitive ratio [21] without prior
knowledge of the minimum degree of the targets.

Maximizing the number of non-disjoint dominating sets in
bipartite graph is another method to increase the lifetime of
the network. Berman et al. provided a 1 + ln n approximation
algorithm based on a ln n approximation to the minimum
weighted set cover problem [22]. Kasbekar et al. considered
a problem with an additional constraint that each sensor has
information only about its neighbors [23]. They provided a
distributed algorithm with an O(ln n · ln(nB)) approximation
ratio, where B is the maximum battery capacity of any sensor.
Recently, Ashwin et al. proved that the problem cannot be
approximated within a factor of less than ln n using any
polynomial time algorithm, where n is the number of tar-
gets [24]. Jia et al. considered a variant of the problem in
which sensors only sense directionally, and targets require
different coverage quality requirements [25]. They proposed
an efficient heuristic algorithm and obtained an upper bound
of the optimal solution.

Table 1 summarizes the main results for k-Lifetime Dom-
inating Set in bipartite graph. To make the running time
easier to read, we use the O*-notation allowing us to omit
the polynomial factors. The symbol C denotes centralized
algorithm, while D denotes distributed algorithm. The symbol
n’ denotes the number of sensors, and n denotes the number of
targets, and c denotes a constant.

III. SYSTEM MODEL
In this section, we describe the model and introduce the
notation used throughout this paper.

We model the network as a bipartite graph G = (S ∪ T, E)
with |S| = ns and |T| = nt. Herein, a bipartite graph is one
whose vertices can be divided into two independent sets, S
and T, and every edge of the graph connects one vertex in S
to one vertex in T. The nt targets t₁, t₂, ..., tnt with known
locations must be continuously observed, and the ns sensors
s₁, s₂, ..., sns are randomly deployed close to the targets. If a
target t ∈ T is within the sensing range of a sensor s ∈ S,
then there is an edge (s, t) ∈ E between s and t. We call

| Type       | Time       | Ratio | Ref. |
|------------|------------|-------|------|
| Heuristic+C| O(n²)      |       | [16] |
| Approximate+C | unknown | √n   | [18] |
| Approximate+C | unknown | ln n | [19] |
| Online     | O*(c⁵)    | 1/2   | [21] |
| Heuristic+C| O(n²n)    |       |      |
| Heuristic+D| O(n²n²)   |       |      |

| Symbol | Description       |
|--------|-------------------|
| | the number of elements in the set S |
| N(s) | the set of vertices adjacent to the vertex s |
| w(t) | the weight of vertex t |
| N(S) | the set of vertices adjacent to some vertex in S |
s can dominate (monitor) t. Let D ⊆ S be a subset of S. If every vertex of T is dominated by some vertex of D, then D dominates T.

As we assume that the number of sensors deployed in the field is much greater than the optimum required to perform the monitoring task, an important method of prolonging the network lifetime consists in scheduling the sensor node activity to alternate between active mode and sleep mode [16]–[18]. Assume that each sensor can be active for a unit time of 1. That is, if all sensors are continuously active, the network lifetime is 1. Assume that all sensors initially have the same amount of power, and their energy depletes at the same rate. To achieve a lifetime of k, the sensors are organized into k disjoint groups (dominators) S1, S2, ..., Sk with S1 ⊆ S and S1 ∪ S2 ∪ ... ∪ Sk = S such that every group (dominator) Si can dominate all targets in T. Herein, disjoint demands that S1 ∩ Si = ∅ (1 ≤ i, j ≤ k). At any moment in time, only one such group is active for monitoring the targets and consumes energy, while the other groups are in sleep mode with no energy consumption.

The formal definition of the considered problem is described as follows.

k-Lifetime Dominating Set in bipartite graph

Input: An undirected bipartite graph G(S ∪ T, E), nonnegative integer k;

Question: Is there a partition (S1, S2, ..., Sk) of S such that every dominator Si (1 ≤ i ≤ k) can dominate T?

If a partition (S1, S2, ..., Sk) of S exists such that every dominator Si (1 ≤ i ≤ k) can dominate T, then the given problem instance is called a yes instance. (S1, S2, ..., Sk) is called a k-partition of S. Finding a partition (S1, S2, ..., Sk) of S such that every dominator can dominate T and k is maximized is called the optimal version of the considered problem.

If |E| − k * |T| > 0, then partition redundancy exists for the partition task and the redundancy degree is |E| − k * |T|.

Suppose that a sensor s is assigned to dominator Si. If N(s) ∩ N(Si − {s}) ≠ ∅, then we say that there exists assignment conflict in the assignment for s, and the conflict degree is |N(s) ∩ N(Si − {s})|.

See the problem instance of k-Lifetime Dominating Set in bipartite graph in Fig. 2. The instance has eight sensors S = {s1, s2, s3, s4, s5, s6, s7, s8}, and two targets T = {t1, t2}. Suppose that k = 3. Suppose that s1, s2, and s3 are assigned to dominator S1, s4 and s5 are assigned to dominator S2, and s6, s7, and s8 are assigned to dominator S3. Because every dominator Si (1 ≤ i ≤ k) can dominate T, (S1, S2, S3) is a 3-partition of S and this problem instance is a yes instance. The number of edges |E| is nine, and the number of targets |T| is two. Because |E| − k * |T| = 9 − 3 * 2 = 3 > 0, partition redundancy exists for the partition task, and the redundancy degree is 3. Because s1 is assigned to the dominator S1 and N(s1) ∩ N(S1 − {s1}) = {t1} ≠ ∅, there exists assignment conflict in the assignment for s1, and the conflict degree is 1.

Herein, we are concerned only with the sensor node scheduling mechanism and do not address the problem of selecting which protocol is used for data gathering or node synchronization.

IV. RANDOMIZED ALGORITHM

In this section, we propose a simple and randomized algorithm for the k-Lifetime Dominating Set problem in bipartite graph, and then show how to de-randomize it.

The randomized R-LDSBG algorithm shown in Fig. 1 just assigns each sensor to a dominator chosen uniformly at random. It makes few assumptions about the networks and is simple enough to be implemented. In addition, the expected performance can be guaranteed because the performance of the algorithm is proportional to the iteration times \(lt\) of the random partition process. Let \(lt\) be \(Nk^{k|T|}\), with \(N = \lceil\ln(1/ε)\rceil\) (0 < ε < 1).

**Theorem 4.1:** Given yes instance \(Y = (G(S ∪ T, E), k)\) of k-Lifetime Dominating Set in bipartite graph, the algorithm R-LDSBG generates a solution for \(Y\) with probability \(1 - ε\).

**Proof:** Let \((S_1, \ldots, S_k)\) be a partition of the set of sensors \(S\) such that every dominator \(S_i\) dominates \(T\). Note that for every \(S_i\), there exists a subset \(B \subseteq S_i\) such that \(|B| ≤ |T|\) and \(B\) dominates \(T\). Suppose \(D_i\) is a subset of \(S_i\), with \(|D_i| ≤ |T|\) that dominates \(T\). Let \(D = D_1 ∪ D_2 ∪ \ldots ∪ D_d\). We say that the sensors in \(D\) are properly partitioned by a k-partition \((P_1, \ldots, P_k)\) of \(S\) if for two sensors \(v_1, v_2 \in D\), the following conditions hold:

1. If \(v_1, v_2 \in D_i\), then \(v_1, v_2 \in P_{i′}\) (1 ≤ i, i′ ≤ k);
2. If \(v_1 \in D_i\) and \(v_2 \in D_j\) with \(i ≠ j\), then \(v_1 \in P_i\) and \(v_2 \in P_j\) with \(i′ ≠ j′\) (1 ≤ i, i′, j, j′ ≤ k).

If we partition the sensors in \(S\) into \(k\) disjoint subsets randomly, then the probability that the sensors in \(D\) are properly partitioned is not less than \(\frac{k!}{k^{|T|}}\).

**Algorithm R-LDSBG**

Input: bipartite graph \(G(S ∪ T, E),\) nonnegative integer \(k\)

Output: a partition \((S_1, \ldots, S_k)\) of \(S\) such that every dominator \(S_i\) dominates \(T\), or report "no".

1. initialize \(lt\)
2. for \(i = 1\) to \(lt\) do
   2.1 for every sensor \(s \in S\), choose a random number \(i \in \{1, \ldots, k\}\) and assign \(s\) to \(S_i\)
   2.2 if every \(S_i\) dominates \(T\) then return \((S_1, \ldots, S_k)\)
3. return "no".

**FIGURE 1. Randomized algorithm R-LDSBG.**

The R-LDSBG algorithm, shown in Fig. 1, is based on these ideas. According to the discussion above, each random k-partition of \(S\) has a probability of at least \(\frac{k!}{k^{|T|}}\) to obtain a solution. Because Step 2 loops \(Nk^{k|T|}\) times with a probability of at least \(1 - (1 - \frac{k!}{k^{|T|}})^{Nk^{k|T|}}\), the partition constructed in Step 2.1 is a solution. Since \(\lim_{n→+∞}(1 + 1/n)^n = e\), \(1 - (1 - \frac{k!}{k^{|T|}})^{Nk^{k|T|}} < e^{-N}\). Note that \(N = \lceil\ln(1/ε)\rceil\). It follows that \(1 - (1 - \frac{k!}{k^{|T|}})^{Nk^{k|T|}} > 1 - ε\).

In the following, we illustrate how the R-LDSBG algorithm works with the problem instance shown in Fig. 2.
The instance has eight sensors $S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$, and two targets $T = \{t_1, t_2\}$. Suppose that $k$ is set to three, and the iteration time $l$ is 1. In Step 2.1, each sensor is randomly assigned to some dominator. Suppose that $s_1, s_2,$ and $s_3$ are assigned to dominator $S_1$, $s_4$ and $s_5$ are assigned to dominator $S_2$, and $s_6, s_7,$ and $s_8$ are assigned to dominator $S_3$. Since $S_1$ cannot dominate all the targets $T$, the algorithm returns “no”.

We further show that the random partition process can deterministically generate a solution for a given problem yes instance by deterministic construction of the $h$-color coding scheme. We first briefly introduce this technique. Given a set $B$ of elements, the $h$-coloring of $B$ is a function from $B$ to $\{1, 2, \ldots, h\}$. A subset $B'$ of $B$ is properly colored by coloring $f$ if no two elements in $B'$ are assigned the same color by $f$. Let $F$ be a collection of $h$-coloring of $B$. If for any size $h$ subset $B'$ of $B$, there is a $h$-coloring $f$ in $F$ that colors $B'$ properly, then $F$ is a $h$-coloring scheme for $B$.

**Lemma 4.2:** [26] For any finite set $B$ and any integer $h$, there is a $h$-coloring scheme $C$ for the set $B$.

Based on Lemma 4.2, we obtain the optimal algorithm O-LDSBG in Fig. 3.

**Algorithm O-LDSBG**

Input: bipartite graph $G(S \cup T, E)$ and nonnegative integer $k$.

Output: a partition $(S_1, \ldots, S_k)$ of $S$ such that every $S_i$ dominates $T$, or report “no”.

1. construct a $k|T|$-color coding scheme $F$ for $S$
2. for each $k|T|$-coloring $f$ in $F$
   1. let $S' = \{s_1, \ldots, k|T|\}$
   2. for each $k$-partition $P = (P_1, \ldots, P_k)$ of $S'$
   3. let $S_i$ be the subset of $S$ assigned colors by $f$ from $P_i$
   4. if every $S_i$ dominates $T$ then return $(S_1, \ldots, S_k)$
3. return “no”

**FIGURE 3.** Optimal algorithm O-LDSBG.

**Theorem 4.3:** Given yes instance $Y = (G(S \cup T, E), k)$ of k-Lifetime Dominating Set in bipartite graph, the algorithm O-LDSBG generates a solution for $Y$.

**Proof:** Suppose $(G(S \cup T, E), k)$ is a yes instance. Let $(S_1, \ldots, S_k)$ be a partition of the set of sensors $S$, such that every $S_i$ dominates $T$. As shown in Theorem 4.1, there exists a set $D$ of size at most $k|T|$ in $S$ and a partition $(D_1, \ldots, D_k)$ of $D$ such that $D_i \subseteq S_i$, $|D_i| \leq |T|$, and $D_i$ dominates $T$. Note that by the definition of $h$-color coding scheme, at least one $k|T|$-coloring $f_0$ in $F$ properly colors the sensors in $D$. Consider the execution of Steps 2.1-2.4 on this particular $k|T|$-coloring $f_0$. Given a $k$-partition $P = (P_1, \ldots, P_k)$ of the $k|T|$ colors, we say that $P$ is a proper partition of $k|T|$ colors if, for any two sensors $v_1, v_2 \in D$, the following conditions hold:

1. if $v_1, v_2 \in D_i$ then $f_0(v_1), f_0(v_2) \in P_i'$
2. if $v_1 \in D_i, v_2 \in D_j$ with $i \neq j$, then $f_0(v_1) \not\in P_j, f_0(v_2) \not\in P_j$ with $i \neq j$.

Note that for any two sensors $v_1, v_2 \in D$, $f_0(v_1) \neq f_0(v_2)$. Moreover, Step 2.2 enumerates all $k$-partition of the $k|T|$ colors. Therefore, at least one partition $P' = (P_1', \ldots, P_k')$ of the $k|T|$ colors from Step 2.2 is a proper partition. Suppose now we have the proper partition $P' = (P_1', \ldots, P_k')$. Then, the corresponding $k$-partition of $S$ constructed in Step 2.3 is obviously a solution for the considered problem and is returned by Step 2.4 of O-LDSBG algorithm in Fig. 3.

In the following, we show how the O-LDSBG algorithm works using an example in Fig. 2. As shown in the figure, there are eight sensors $S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$, and two targets $T = \{t_1, t_2\}$, $k$ is set to 3. Suppose $S_1 = \{s_1, s_2, s_3, s_4\}, S_2 = \{s_5, s_6\}$ and $S_3 = \{s_7, s_8\}$. Clearly, $(S_1, S_2, S_3)$ is a solution for the given instance. Observe that for every $S_1$, there exists a subset $B \subseteq S_1$ such that $|B| \leq |T|$ and $B$ dominates all the targets $T$. Let $D_1 = \{s_1, s_3\}, D_2 = \{s_5, s_6\}$ and $D_3 = \{s_7, s_8\}$. Every $D_i$ dominates all the targets. We demonstrate that the O-LDSBG algorithm returns a solution $(S_1', S_2', S_3')$ with $D_i \subseteq S_i'$. Let $D = D_1 \cup D_2 \cup D_3$. In Step 1, a 6-color coding scheme $F$ for $D$ is constructed. According to Lemma 4.2, at least some 6-color coding $f_0$ in $F$ colors the sensors in $D$ properly: for any two sensors in $D$, they will be assigned different colors under $f_0$. Suppose that Step 2 is conducted under the 6-color coloring $f_0$. Now, we are ready to find a proper partition of the 6 colors into $k = 3$ disjoint groups: for any two sensors $s_i, s_j$ in $D$ with $i \neq j$ and $1 \leq i, j \leq 8$, if $s_i, s_j$ belong to the same dominator in $\{D_1, D_2, D_3\}$, then the color of $s_i$ and the color of $s_j$ belong to the same group, otherwise ($s_i, s_j$ belong to different dominators), the color of $s_i$ and the color of $s_j$ belong to different groups. Note that Step 2.2 enumerates all the 3-partition of the 6 colors. Hence, at least one 3-partition $P' = (P_1', \ldots, P_3')$ of the 6 colors is a proper partition. Suppose that the execution of Step 2.2 is on this proper partition $P'$. Subsequently, the corresponding 3-partition $(S_1', S_2', S_3')$ of $S$ constructed in Step 2.3 satisfies $D_i \subseteq S_i'$ with $1 \leq i \leq 3$. Clearly $(S_1', S_2', S_3')$ is a solution for the given instance.

Noth that algorithm O-LDSBG takes exponential time and is infeasible in practice.

**V. CENTRALIZED GREEDY ALGORITHM**

In this section, based on the idea of randomized algorithm, we provide a more practical centralized and randomized algorithm to obtain the optimal solution for a given instance.
of the \( k \)-Lifetime Dominating Set problem in bipartite graph.

Assume that each sensor has location determination capabilities and can determine which targets of interest it will be capable of monitoring. The sensors send to the base station (BS) their ID numbers and the targets monitored. Then, the BS executes the sensor scheduling algorithm and broadcasts the schedule information. Finally, every sensor schedules itself for active/sleep modes.

The pseudocode of the greedy algorithm, C-LDSBG, is shown in Fig. 4. This algorithm is based on the idea of the random algorithm shown in Fig. 1. The random algorithm indicates that the probability of obtaining a solution is proportional to the iteration times of the partition process. If exponential iterations of the partition process are permitted, a solution can be determinately constructed. But, to generate a solution determinately, the iteration times must be huge, which is unpractical. In the greedy algorithm, we use the iteration times \( lt \) as the input. The value of \( lt \) is balanced between the running times and solution effectiveness. Moreover, the assignment process in the random algorithm is completely random, which increases the number of iterations required to obtain the right partition. In the greedy algorithm, an object function is introduced to evaluate the attractiveness of each dominator to unassigned sensors. The key point is that an unsigned sensor is favored to be assigned to a dominator achieving covering more sparsely uncovered targets.

The algorithm takes as the input parameters \( S \) — the set of sensors, \( T \) — the set of targets, \( k \) — the number of dominators, and \( lt \) — the iteration times of the random partition process. The greedy algorithm returns the set of dominators \((S_1, \ldots, S_k)\).

At the beginning of each recursion of the algorithm, all the sensors in \( S \) are selected into \( C \), the set of available sensors, and every dominator \( S_i \) is set to be empty. The greedy algorithm iteratively executes the process of random partition from Step 1.2. Once a right partition is obtained, the algorithm exits the recursion and returns the set of dominators in Step 1.3. One sensor is assigned to some dominator from the set of sensors \( S \) in each iteration of Step 1.2. All sensors from \( S \) are assigned to some dominator \( S_i \) after the iteration of Step 1.2.

The assignment process for sensors is conducted in the order of the degree of sensors, from high to low, which is helpful for decreasing the assignment conflict.

At each iteration of Step 1.2, a sensor \( s \in C \) whose degree is the highest in \( C \) is assigned to some dominator \( S_i \). The sensor with the highest degree is found at the beginning of each iteration of Step 1.2. Every variable \( a[i] \) is set to zero, which is used to store a value representing the contribution of the unassigned sensor \( s \) to the dominator \( S_i \). In Step 1.2.3, for all dominators \( S_i \) and the given unassigned sensor \( s \), the values of the objective function \( f \) are calculated. The objective function for dominator \( S_i \) and sensor \( s \) is given by

\[
f(s, S_i) = k^{\left| N(s) - N(S_i) \right|} \cdot \sum_{t \in N(s) \cap N(S_i)} w(t).
\]

**Algorithm C-LDSBG**

**Input:** bipartite graphs \( G(S \cup T, E) \), nonnegative integer \( k \), nonnegative integer \( lt \).

**Output:** a partition \((S_1, \ldots, S_k)\) of \( S \) such that every \( S_i \) dominates \( T \), or report “no such partition exists”.

1. for \( i = 1 \) to \( lt \) do
2. for \( i = 1 \) to \( k \) do \( a[i] = 0 \)

1.2 while \( C \neq \emptyset \) do

1.2.1 Find \( s \in C \) such that \( s \)'s degree is highest in \( C \)

1.2.2 for \( i = 1 \) to \( k \) do \( a[i] = 0 \)

1.2.3 for each \( S_i \) do \( a[i] = f(s, S_i) \)

1.2.4 \( \text{sum} = 0 \); for \( i = 1 \) to \( k \) do \( \text{sum} = \text{sum} + a[i] \)

1.2.5 for \( i = 2 \) to \( k \) do \( a[i] = a[i - 1] \)

1.2.6 for \( i = 1 \) to \( k \) do \( a[i] = a[i]/\text{sum} \)

1.2.7 \( r = \text{random}(0,1) \)

1.2.8 if \( i = 1 \) then \( k \) do if \( a[i] > r \) then \( j = i \) break

1.2.9 \( S_j = S_j \cup s, C = C - \{s\} \)

1.3 if every \( S_i \) dominates \( T \) then return \((S_1, \ldots, S_k)\)

2. return “no such partition exists”.

**FIGURE 4. Centralized algorithm C-LDSBG.**

Herein, \( N(s) \) denotes the set of targets dominated by \( s \), and \( N(S_i) \) the set of targets dominated by the dominator \( S_i \). Hence, \( |N(s) - N(S_i)| \) denotes the contribution of \( s \) to \( S_i \) — the number of targets dominated by \( s \) but not dominated by \( S_i \). Every target \( t \) is weighted by \( w(t) = 1/d(t) \), where \( d(t) \) is the number of sensors dominating \( t \) that have not yet been assigned to a dominator. The incentive of such an objective function is to (1) favor dominators that dominate a high number of unassigned targets by the value \( k^{|N(s) - N(S_i)|} \) and (2) favor dominators that dominate more sparsely dominated targets by the value \( \sum_{t \in N(s) \cap N(S_i)} w(t) \).

From Step 1.2.4 to Step 1.2.9, the sensor \( s \) is assigned to some dominator \( S_i \) according to the probability using the roulette method.

In Step 1.3, if every dominator \( S_i \) can dominate all targets, then the algorithm returns the right partition \((S_1, \ldots, S_k)\).

If Step 1 cannot generate a right partition, then we assume that the input instance is a no instance, and return “no such partition exists”.

For given \( s \) and \( S_i \), \( f(s, S_i) \) can be calculated in time \(|S||T|\).

Hence, Step 1.2.3 can be finished in time \( k|S||T| \). It follows that Step 1.2 can be finished in time \( k|S|^2|T| \). Therefore, the C-LDSBG algorithm requires \( O(lt \cdot k \cdot ns^2 \cdot nt) \) time, where \( ns \) is the number of sensors, and \( nt \) is the number of targets.

**VI. DISTRIBUTED GREEDY ALGORITHM**

In this section, we give a distributed implementation of the centralized greedy algorithm.

The distributed algorithm makes some assumptions about the network:

1. Sensor synchronization has been realized by MAC protocol;
2. Every sensor knows the parameter \( k \);
3. Every sensor can determine the set of targets it is able to monitor and recognize them by geographic coordinate;
4. Every sensor has a unique ID number taken from the set of integers \( \{1, 2, \ldots, ns\} \), where \( ns \) is the number of sensors;
5. Every sensor can communicate with its neighbors. For two sensors \( s_1, s_2 \in S \), if \( s_1 \) and \( s_2 \) dominate a common target, then \( s_1 \) is the neighbor of \( s_2 \) and vice versa.

VOLUME 10, 2022
Algorithm Initialization (sensor $s$)
1. while $tm \leq s$ do
   1.1 if $tm < s$ then
      1.1.1 If message is received from neighbor node
            about its ID number and targets monitored
            then
      1.1.2 renew Table II
   1.2 if $tm = s$ then
   1.2.1 send its ID number, and targets monitored
            to neighbors

Algorithm D-LDSBG (sensor $s$)
1. while $tm \leq s$ do
   1.1 if $tm < s$ then
      1.1.1 If message is received from neighbor node
            that some
            sensor will be assigned to a dominator
            then
      1.1.2 renew Table II and Table I
   1.2 if $tm = s$ then
   1.2.1 find $i$ such that $f(s, S_i)$ is largest
   1.2.2 assign itself to dominator $S_i$
   1.2.3 send assignment information to neighbors

FIGURE 5. Initialization at sensor $s$.

Each sensor $s$ stores two tables. Table 1 is a matrix of $k \times |N(s)|$, and Table 2 is a matrix of $1 \times |N(s)|$. Herein, $N(s)$ denotes the set of targets that are dominated by $s$.

Table 1 for sensor $s$ stores the domination information of every target $t \in N(s)$. If the value of entry in row $i \in \{1, \ldots, k\}$ and column $j \in \{1, \ldots, |N(s)|\}$ is 1, then it indicates that the target $j$ is dominated by the dominator $S_i$. All entries are initialized to 0 and refreshed in the D-LDSBG algorithm (Fig. 6).

Table 2 stores the degree $d(t)$ of every target $t$ monitored by $s$. All entries are set to 0 and then updated by algorithm Initialization in Fig. 5. At Step 1.1.1 in the algorithm Initialization, the sensor $s$ receives message from its neighbor sensor $s_s$. The message includes the ID number of sensor $ss$, and the targets monitored by $ss$. According to the message, the sensor $s$ renews its Table 2: the value of the entry in column $i$ is added by one.

The D-LDSBG algorithm is initiated at time $tm = 0$. When $tm < s$ indicating that it is not the time slot of $s$ for assignment, sensor $s$ receives a message from its neighbor $ss$. The message includes the ID number of sensor $ss$, targets monitored by $ss$, and dominator $j$ assigned to $ss$. According to the message, sensor $s$ renews its Table 2: the value of the entry in column $i$ is subtracted by one. Table 1 is renewed using the following method: change the entry in row $j$, column $i$, from 0 to 1. When it is the time slot of sensor $s$ for assignment, for all dominators $S_i$ the values of the objective function $f$ are calculated. The objective function for the dominator $S_i$ is given by

$$ f(s, S_i) = k^{|N(s)\cap N(S_i)|} \cdot \sum_{t \in N(s)\cap N(S_i)} w(t), $$

where $w(t) = 1/d(t)$. The sensor $s$ is assigned to the dominator $S_i$ such that $f(s, S_i)$ is the largest over all dominators $(S_1, \ldots, S_k)$.

Note that $d(t)$, $N(s)$, and $N(S_i)$ can be obtained directly from Table 2 or Table 1. Hence, it is easy to see that the algorithm requires $|S|^2|T|^2$ time.

After the assignment phase, every sensor $s$ is assigned to some dominator $S_i (1 \leq i \leq k)$ and knows the number $i$. Sensor $s$ will schedule itself for the active mode from time $i$-1 to time $i$ and communicate with other sensors by broadcasting.

VII. SIMULATION

In this section, we evaluate the performance of our centralized greedy algorithm, C-LDSBG, for $k$-Lifetime Dominating Set in bipartite graph. The evaluation is conducted on a PC with a 2.67GHz Inter Core i5 CPU and 8GB main memory.

The problem instances are created as follows. Given the number of sensors $|S|$, number of targets $|T|$, and number of edges $|E|$, a bipartite graph $G(S \cup T, E)$ is created, where the edges are chosen uniformly at random from all possible sensor-target pairs. A sensor $s$ can dominate a target $t$ if $s$ has an edge connecting it with $t$. If the degree of some vertex $s \in S$ is zero, then the topology $G$ is discarded, and a new topology is generated. We choose this approach for generating topology as opposed to an approach where nodes are in Euclidean space and sensor nodes sense target nodes within a radius of their location because the latter limits the variety of applications.

In the simulation, we consider the following tunable parameters:

- $ns$, the number of sensor nodes;
- $nt$, the number of target nodes;
- $md$, minimum degree of sensor nodes;
- $m$, the number of edges;
- $k$, the preset number of dominators for a given problem instance;
- $lt$, the iteration times.

A. PERFORMANCE IN ACHIEVING OPTIMAL SOLUTION

The effectiveness of the proposed algorithm is proportional to the number of iterations. Thus, we focus on showing how the iteration times needed to generate one solution are affected by different parameters and the average iteration times for different network instances.

For any given instance, we set the number of dominators $k$ equal to the minimum degree of the target nodes. Because the optimum number of dominators for any problem instance cannot be larger than the minimum degree of the target nodes, our algorithm actually returns the optimal solution for any given instance in the simulation.

In the first experiment, we provide an intuition of the iteration times needed to obtain an optimal solution in 50 network instances. The number of sensor nodes $ns$ is 100, number of target nodes $nt$ is 10, and number of edges $m$ is set to 250. Fig. 7 plots the iteration times when $k$ does not exceed 22. As can be observed, the iteration times vary between 1 and 7. In most cases, our algorithm can return the optimal partition with iteration time of 1. When $k$ is 23, as shown
in Fig. 8, the iteration times increase. In most instances, the iteration times are more than 1, and the average iteration times is approximately 8. In the worst case, the iteration times are 49. Fig. 9 shows the distribution of iteration times when $k$ is 24. It can be saw that the iteration times increase dramatically. The average iteration times is approximately 450, and 3160 iteration times are required to obtain the optimal partition for the hardest instance.

Fig. 10 shows the impact of iteration times $lt$ on the probability of getting an optimal solution successfully. For every value of the iteration times, we repeat the experiment 20 times and count the times of returning an optimal solution successfully. As we can observe, the probability of generating an optimal partition increases as the iteration times of partition process increase. When the iteration time is bounded by 1, three instances return optimal solutions, and the average probability of getting one solution is 0.15. When the iteration times are bounded by 50, only one instance cannot return one solution.

In the following set of experiments, several factors are studied determining the minimum iteration times guaranteeing an optimal solution for a given instance. For each parameter setup, we generate ten network topologies and report the average results.

Fig. 11 plots the iteration times when the number of dominators $k$ is changed. As we can see, the iteration times increase slowly when the value of $k$ is no more than 22. But after $k$ exceeds 22, the iteration times needed to generate an optimal solution increase rapidly. The increase in iteration times is proportional to $k$. This is expected, since with the increase of the value of $k$, the algorithm has less partition redundancy that can be utilized to get the right partition for given instance.

In Fig. 12, we present the average iteration times when the number of sensors changes between 90—110 in increments of 5. As shown in Fig. 6, the decrease in iteration times is proportional positively to the number of sensor nodes. With an increase in the number of sensor nodes, the average degree of the sensor nodes decreases. Thus, the probability of having assignment conflict decreases, which makes it easier for the random partition procedure to obtain a right partition.

In Fig. 13, we measure the iteration times when the number of targets varies between 8 and 12. The iteration times increase as the number of targets increases. With an increase in the number of target nodes, for each dominator, more sensor nodes are required to realize full domination. Thus, the probability of having assignment conflict increases, which makes it more difficult for the random partition process to obtain a right partition.

Fig. 14 shows the convergence of the iteration times with the number of edges in the network instances. As we can see, the iteration times decrease with the increase of number of edges. The reason is that, with the increase of edges, the
algorithm has more partition redundancy that can be utilized to get the right partition.

Fig. 15 plots the iteration times when the minimum degree of the sensors changes. As we can see, the iteration times increase with the increase of minimum degree of sensors. This is can be expected, since with the increase of minimum degree of the sensors, the probability of having assignment conflict is increased. Consequently, it is more difficult for the random partition procedure to get a right partition.

B. PERFORMANCE COMPARED WITH R-LDSBG
The performance of the centralized greedy algorithm C-LDSBG in obtaining the problem solution is compared with the performance of the random algorithm R-LDSBG. Because the iteration times \( t \) are decisive for the effectiveness of obtaining a solution for both C-LDSBG and R-LDSBG, we use iteration times to evaluate the performance.

We provide an overview of the iteration times \( t \) required to obtain the problem solution in 200 network instances. The number of sensors is set to 100, and the number of targets is set to 10, and number of edges is set to 250. The minimum degree of the target nodes \( m_t \) is set to 15, and the number of dominators \( k \) is set to \( m_t - 5 \). The upper bound on iteration times is set to 5000. For every instance, Algorithm C-LDSBG returns the problem solution successfully with iteration time of 1. The R-LDSBG algorithm successfully returns problem solutions for 53 instances, but fails to get problem solutions for the other 147 instances. For the 53 instances, the average iteration times required to obtain the solution is 2774.

C. BENCHMARKING THE HEURISTIC ALGORITHM MCCH
Our algorithm, C-LDSBG, serves for benchmarking the heuristic algorithm MCCH by Slijepcevic and Potkonjak [16], indicating the quality of their solutions. The MCCH approach favors to select a sensor into current dominator that dominates the largest number of sparsely dominated targets. The heuristic algorithm MCCH is effective and easy to be implemented. It has time complexity \( O(n^2) \), where \( n \) is the number of sensors in the network. Compared to other heuristic algorithms.
for $k$-Lifetime Dominating Set in bipartite graph, MCCH has lower execution time.

For a given problem instance, we compare the number of dominators produced by the MCCH algorithm with the maximum number of dominators (optimal solutions) produced by our C-LDSBG algorithm. We provide an intuition of the quality of the solutions produced by the MCCH algorithm in 250 network instances. For a given instance, the parameter $k$ is set to the minimum degree of the target nodes. Hence, the proposed C-LDSBG algorithm returns the optimal solution for any given instance. The results are shown in Fig. 16. For almost 30 percent of the 250 problem instances, the heuristic algorithm MCCH returns optimal solutions. In the worst case, the number of dominators computed by the heuristic algorithm MCCH is four less than the maximum number of dominators.

VIII. CONCLUSION

We concentrate on the design of efficient algorithms that generate optimal solutions determinately for the $k$-Lifetime Dominating Set problem in bipartite graph both in theory and in practice. We provide a simple algorithm that just partitions the sensors into groups randomly. We prove that the random algorithm can guarantee a solution using color coding. However, it takes exponential time, which is unpractical. Based on the idea of random partition, we then present a more practical centralized greedy algorithm which keep a balance between the running time and solution effectiveness. We further present a distributed version of the centralized algorithm where each sensor can only communicate with its neighbors. To the best of our knowledge, this is the first distributed algorithm for the considered problem. Simulation findings demonstrate that the centralized algorithm can achieve optimal solutions for almost all given instances efficiently when the redundancy degree is no less than 10. Finally, our algorithms also serve for benchmarking heuristic algorithms. However, the proposed centralized algorithm is unpractical when partition redundancy is below a certain limit.

One challenge is to efficiently achieve optimal solutions for problem instances when partition redundancy is below a certain limit. Another interesting direction for further work is to conduct research on the application of evolutionary computation algorithms [27], [28] to the considered problem.

REFERENCES

[1] X. Cheng, D. Ciunzo, and P. S. Rossi, “Multibit decentralized detection through fusing smart and dumb sensors based on rao test,” IEEE Trans. Aerosp. Electron. Syst., vol. 56, no. 2, pp. 1391–1405, Apr. 2020.
[2] R. Niu and P. K. Varshney, “Performance analysis of distributed detection in a random sensor field,” IEEE Trans. Signal Process., vol. 56, no. 1, pp. 339–349, Jan. 2008.
[3] D. Ciunzo, P. S. Rossi, and P. K. Varshney, “Distributed detection in wireless sensor networks under multiplicative fading via generalized score tests,” IEEE Internet Things J., vol. 8, no. 11, pp. 9059–9071, Jun. 2021.
[4] S. H. Poon, C. K. Yen, and C. T. Ung, “Domatic partition on several classes of graphs,” in Proc. Int. Conf. Combinat. Optim. Appl., 2012, pp. 245–256.
[5] D. Mahjoub and D. W. Matula, “Building (1-ε)-dominating sets partition as backbones in wireless sensor networks using distributed graph coloring,” in Proc. DCOSS, 2010, pp. 57–144.
[6] D. Mahjoub and D. W. Matula, “Employing (1-ε)-dominating sets partitions as backbones in wireless sensor networks,” in Proc. ALHENX, 2011, pp. 98–111.
[7] S. V. Penmmaraju and I. A. Pirwani, “Energy conservation via domatic partitions,” in Proc. ACM MobilC06, 2006, pp. 54–143.
[8] S. Pandit, S. V. Penmaraju, and K. Varadarajan, “Approximation algorithms for domatic partitions of unit disk graphs,” in Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (Lecture Notes in Computer Science), vol. 5687. Berlin, Germany: Springer, 2009.
[9] J. Yu, Q. Zhang, D. Yu, C. Chen, and G. Wang, “Domatic partition in homogeneous wireless sensor networks,” J. Netw. Comput. Appl., vol. 37, pp. 186–193, Jan. 2014.
[10] R. Misra and C. Mandal, “Efficient cluster head rotation via domatic partition in self-organizing sensor networks,” Wireless Commun. Mobile Comput., vol. 9, no. 8, pp. 1040–1058, 2009.
[11] H. Liang, “The algorithmic complexity of k-domatic partition of graphs,” in Theoretical Computer Science, vol. 7804. Berlin, Germany: Springer, 2012, pp. 240–249.
[12] C. M. Lee, “Total k-domatic partition on some classes of graphs,” in Proc. Int. Comput. Symp. (ICS), 2016, pp. 74–79.
[13] R. Misra and C. Mandal, “Rotation of CDS via connected domatic partition in ad hoc sensor networks,” IEEE Trans. Mobile Comput., vol. 8, no. 4, pp. 488–499, Apr. 2009.
[14] K. Sensor-Hillel, M. Ghaffari, and F. Kuhn, “Distributed connectivity decomposition,” in Proc. ACM Symp. Princ. Distrib. Comput. (PODC), 2014, pp. 156–165.
[15] T. Pino, S. Choudhury, and F. Al-Turjman, “Dominating set algorithms for wireless sensor networks survivability,” IEEE Access, vol. 6, pp. 17527–17532, 2018.
[16] S. Sljepcevic and M. Potkonjak, “Power efficient organization of wireless sensor networks,” in Proc. IEEE INFOCOM, Jun. 2001, pp. 276–472.
[17] M. Cardei and D.-Z. Du, “Improving wireless sensor network lifetime through power aware organization,” Wireless Netw., vol. 11, no. 3, pp. 333–340, 2005.
[18] S. Henna and T. Erlebach, “Approximating maximum disjoint coverage in wireless sensor networks,” in Proc. Int. Conf. Ad-Hoc Netw. Wireless Ad-Hoc, Mobile, Wireless Netw., 2013, pp. 148–159.
[19] A. Pananjady, V. K. Bagaria, and R. Vaze, “Optimally approximating the coverage lifetime of wireless sensor networks,” Comput. Sci., vol. 2, no. 11, pp. 1–23, 2013.
[20] A. Pananjady, V. K. Bagaria, and R. Vaze, “The online disjoint set cover problem and its applications,” in Proc. IEEE Conf. Comput. Commun. (INFOCOM), Apr. 2015, pp. 1221–1229.
[21] Y. Emek, A. Goldbriehl, and E. Kantor, “Online disjoint set cover without prior knowledge,” in Proc. 27th Ann. Eur. Symp. Algorithms (ESA), 2019, pp. 44:1–44:16.
[22] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky, “Power efficient monitoring management in sensor networks,” in Proc. IEEE Wireless Commun. Netw. Conf., Mar. 2004, pp. 2329–2334.
[23] G. S. Kasbekar, Y. Bejerano, and S. Sarkar, “Lifetime and coverage guarantees through distributed coordinate-free sensor activation,” IEEE/ACM Trans. Netw., vol. 19, no. 2, pp. 470–483, Apr. 2011.
[24] A. Pananjady, V. K. Bagaria, and R. Vaze, “Optimally approximating the coverage lifetime of wireless sensor networks,” IEEE/ACM Trans. Netw., vol. 25, no. 1, pp. 98–111, Feb. 2017.
[25] J. Jia, C. Dong, X. He, D. Li, and Y. Yu, “Sensor scheduling for target coverage in directional sensor networks,” Int. J. Distrib. Sensor Netw., vol. 13, no. 6, pp. 1–12, Jun. 2017.
[26] J. Chen, S. Lu, S. H. Sze, and F. Zhang, “Improved algorithms for path, matching, and packing problems,” in Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA), 2007, pp. 298–307.
[27] Z.-G. Chen, Y. Lin, Y.-J. Gong, Z.-H. Zhan, and J. Zhang, “Maximizing lifetime of range-adjustable wireless sensor networks: A neighborhood-based estimation of distribution algorithm,” IEEE Trans. Cybern., vol. 51, no. 11, pp. 5433–5444, Nov. 2021.
[28] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, and J. Zhang, “Adaptive granularity learning distributed particle swarm optimization for large-scale optimization,” IEEE Trans. Cybern., vol. 51, no. 3, pp. 1175–1188, Mar. 2021.

WEIZHONG LUO received the Ph.D. degree in computer science from Central South University, China. His research interests include network optimization, combinatorial optimization, and parameterized algorithms.

JUNBIN LIANG received the B.S. and M.S. degrees from Guangxi University, Nanning, China, in 2000 and 2005, respectively, and the Ph.D. degree from Central South University, Changsha, China, in 2010. He is currently a Professor with Guangxi University. His research interests include wireless sensor networks and distributed computing.

WEIZHONG LUO received the Ph.D. degree in computer science from Central South University, China. His research interests include network optimization, combinatorial optimization, and parameterized algorithms.

TIANSHU WANG is a Senior Research and Development Expert with Neusoft Group Company Ltd.

***