Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2010, Article ID 926075, 17 pages
doi:10.1155/2010/926075

Research Article

Perimeter Coverage Scheduling in Wireless Sensor Networks Using Sensors with a Single Continuous Cover Range

Ka-Shun Hung and King-Shan Lui

Department of Electrical and Electronics Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong

Correspondence should be addressed to Ka-Shun Hung, kshung@eee.hku.hk

Received 21 September 2009; Revised 27 January 2010; Accepted 24 February 2010

Academic Editor: Yu Wang

In target monitoring problem, it is generally assumed that the whole target object can be monitored by a single sensor if the target falls within its sensing range. Unfortunately, this assumption becomes invalid when the target object is very large that a sensor can only monitor part of it. In this paper, we study the *perimeter coverage problem* where the perimeter of a big object needs to be monitored, but each sensor can only cover a *single* continuous portion of the perimeter. We describe how to schedule the sensors so as to maximize the network lifetime in this problem. We formally prove that the perimeter coverage scheduling problem is NP-hard in general. However, polynomial time solution exists in some special cases. We further identify the sufficient conditions for a scheduling algorithm to be a 2-approximation solution to the general problem, and propose a simple distributed 2-approximation solution with a small message overhead.

1. Introduction

Wireless sensor networks have caught lots of attention in recent years. One of the major problems is the coverage problem. Traditionally, coverage problems concern whether a certain target area or a certain target object can be fully monitored by the sensors collaboratively. Instead of considering whether a certain area or a certain target object is completely covered, we focus on a specific coverage problem called the *perimeter coverage problem*. In this problem, we want to monitor the perimeter of a big target but each sensor can only monitor part of the perimeter. One typical application scenario is to monitor the coastline of a large lake so as to ensure that no people can go through its perimeter intentionally or accidentally. Another application scenario is to monitor the wall of a prison so as to ensure that no criminal can escape easily by digging holes through the wall. Therefore, perimeter monitoring is an important problem.

In [1–4], we studied how to identify a set of sensors to cover the perimeter of a large object with the minimum size and minimum cost. Since sensors are battery-powered and the battery is unlikely to be rechargeable, every sensor network has a functional network lifetime. In this work, we are particularly interested in how to schedule the sensors so as to maximize the network lifetime in the perimeter coverage problem. To the best of our knowledge, the lifetime maximization issue in the perimeter coverage problem was first studied in [5]. In [5], we adopted several heuristic energy-related scheduling objectives as cost metrics. Different sets of sensors are identified using these objectives to monitor the target at different times. The energy-related objectives include the minimization of energy of each sensor set found, the minimization of the battery cost which is defined as the reciprocal of the remaining battery capacity of the sensor, the hybrid algorithm, and so forth. The heuristic scheduling algorithms developed based on the adoption of different energy-related objectives are then compared through extensive simulations.

Unlike that of [5], in this paper, we study the problem from the theoretical perspective and make the following contributions: (1) We formally prove that the perimeter coverage problem is NP-hard in general, but polynomial time solution exists in some special cases. (2) We identify the sufficient conditions for a perimeter coverage scheduling algorithm to be a 2-approximation solution. (3) We develop a distributed scheduling algorithm that generates O(size of the minimum cover) number of messages. The schedules generated must have a lifetime that is at least half of the

optimal one. (4) We simulate the proposed algorithm and compare it with the optimal schedules.

The paper is organized as follows. Section 2 discusses the related work on the other coverage problems. Section 3 describes the notations, problem statement, and the properties of the problem. Section 4 studies the special cases in which polynomial time optimal solutions exist. A formal proof of NP-hardness of this problem in general is given in Section 5. The sufficient condition for a perimeter coverage scheduling algorithm to be a 2-approximation solution is discussed in Section 6. Our proposed distributed 2-approximation solution which requires O(size of the minimum cover) number of messages is also described. Section 7 presents the simulation results of our proposed algorithm by comparing it with the optimal schedules. Finally, we conclude our work with some of the future directions in Section 8.

2. Related Work

Extensive research has been carried out to extend the lifetime of sensor networks [6–8]. In this study, network lifetime is defined as the length of the time period that the target area or target objects are being covered continuously. One way to extend network lifetime is to schedule the sensor nodes' activities to alternate between active and sleep modes so that only a minimal number of sensors are turned on at any time [9]. Tian and Georganas [10] propose a distributed scheduling algorithm which turns off the sensors with a cover area completely replaceable by the cover areas of their neighboring nodes. Yan et al. [11] adopt a similar concept. However, instead of considering whether the sensing area of a sensor is covered by its neighbors, the sensor considers a certain number of grid points inside the area. Later, Hsin and Liu [12] propose a randomized algorithm and a coordinated algorithm so as to schedule the sleeping times of the nodes. In the randomized algorithm, a sensor will sleep with a certain probability. In the coordinated algorithm, a sensor will sleep only when its sensing area is completely covered by others. On the other hand, Huang and Tseng [13] study the criteria for determining whether the sensing area of a sensor is covered by k different sensors simultaneously. Huang et al. [14] propose several decentralized energy-conserving and coverage-preserving protocols so as to solve the k area coverage problem. Other than those suggested above which aim at full area coverage, the fractional coverage problem, in which only a certain fraction of the target area has to be covered, has also been considered in [15]. In [15], Ye et al. first calculate the ideal density for providing a certain fraction of coverage. Then, each sensor is activated with a probability. This probability is determined based on the ratio between the current density and the ideal density calculated. On the contrary, Wang and Kulkarni [16] investigate similar problem in another direction, and they show that sacrificing a certain fraction of the target area can significantly increase the network lifetime. Ren et al. [17] study the relationship between the fraction of coverage and the quality of the object detection capability. As a result, the network can be deployed with a fraction of coverage that can achieve an acceptable quality.

Cardei et al. [18] study the target coverage problem where a target object has to be monitored by at least one sensor at any moment and each sensor can monitor multiple target objects if the objects fall within the sensor's sensing area. They formulate this target coverage problem as a set coverage problem, and propose both centralized approximation and distributed heuristic solutions. In [19], Liu et al. study similar problem. However, they assume that each sensor can only monitor one target object at any moment even if multiple targets fall within the sensing area of a sensor. Unfortunately, the cover formed by using these approaches may not be connected as some nodes in the cover may not have any route back to the sink node. Therefore, Zhao and Gurusamy [20] further investigate the connected target coverage problem. They prove that this problem is NP-complete, and develop a centralized approximation solution and a distributed heuristic solution. Thai et al. [21] propose an $O(\log N)$-approximation distributed algorithm to solve the target coverage problem, where N denotes the number of sensors in the network. They organize the sensors into nondisjoint cover sets in which each set can cover all the target objects with high probability. They formally show that their solution is an $O(\log N)$ approximation solution if the initial battery capacity of all the sensors is the same, but the approximation ratio is worsen if the initial battery capacity of the sensors is different. On the other hand, Calines and Ellis [22] suggest an $O(1+\epsilon)$-approximation algorithm. Their solution achieves the same approximation ratio no matter the initial battery capacity of all the sensors is the same or not. Unfortunately, these algorithms cannot guarantee that all the targets are covered all the time.

Another related problem is the barrier coverage problem. In [23], Kumar et al. consider the k barrier coverage problem. They assume that an intruder can be detected by a sensor if it falls within its sensing area. They consider the scenario in which an intruder will be detected by at least k sensors if she goes through a thin strip of area (known as belt) no matter where her start point and her end point are. In [23], Kumar et al. propose a polynomial time mechanism to determine whether there exists k barrier coverage. On the other hand, Kumar et al. [24] study the optimal sleeping schedule on the k barrier coverage problem. They transform the problem to a maximum flow problem so that it can be solved optimally in a centralized manner. Chen et al. further propose a localized algorithm in [25]. They first study the critical conditions for the existence of k barrier coverage locally. The sensor can then turn into sleep mode when it determines that k barrier coverage can be provided by its neighbors. Chen et al. further study how to find and measure the quality of k barrier coverage accurately in [26]. Later, Liu et al. [27] discover that it may not be possible to guarantee k barrier coverage by using the approach in [25] under some special situations. They identify the critical conditions for the existence of a strong barrier coverage and devise an efficient technique to guarantee k coverage. Other than that, Saipulla et al. [28] study the barrier coverage problem where the sensors are dropped from air. They suggest that the sensors are deployed along a line with a certain offset by using this deployment strategy. They show that the

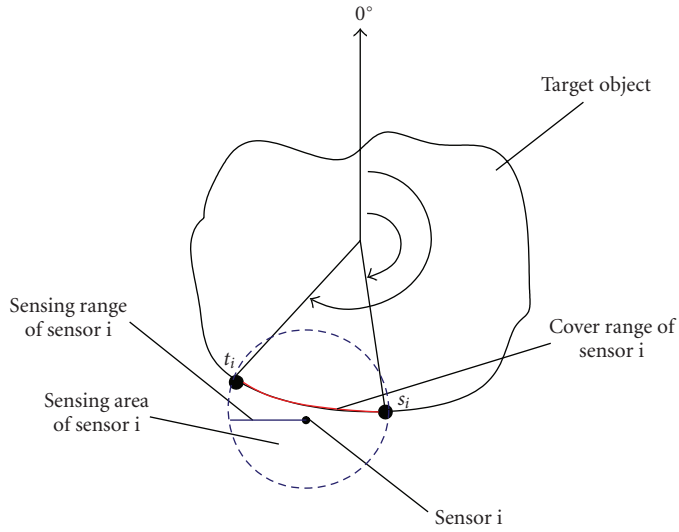


Figure 1: The cover range of sensor i.

performance achievable by using this method is much better than that of using the random deployment strategy which is generally assumed in most of the literatures.

3. Notations, Problem Statement, and Properties

We assume that the object is a very big one and we want the whole perimeter to be monitored continuously. Each sensor has a certain sensing range. It can monitor a point such that the distance between the point and the sensor is less than or equal to the sensing range. The area that the sensor can monitor, called sensing area, is a circle. The object that falls within the sensing area of a sensor is said to be monitored by the sensor. It is assumed that each sensor can only monitor a single continuous portion of the perimeter, and no sensor can monitor the whole perimeter as shown in Figure 1. Sensors are randomly distributed around the big object. The set of sensors that can cover a portion of the target object is denoted as S. Also, we use N to denote the number of sensors in S, that is, $N = |S|$.

3.1. Cover Range, Proper Set of Sensors, and General Set of Sensors. *Cover range* is defined as the portion of the perimeter of the target object covered by the sensing area of a sensor node. In this paper, we represent the cover range in terms of angular measurement for the ease of discussion. It is worth noting that the perimeter of the target object can be in any irregular shape as long as it forms a loop, a sensor only has a single continuous cover range, and the sensor can determine its cover range. Note that under some extreme conditions, the assumption that each sensor only has a single continuous cover range may not be valid, such as, the target object contains a sharp convex or concave shape on the perimeter. In this case, the sensor has multiple cover ranges instead [29]. This makes the problem a lot more complicated, and we leave it as future work. On the other hand, how a sensor determines the range is application dependent, and is outside the scope of this paper. Interested readers are referred to [30–32].

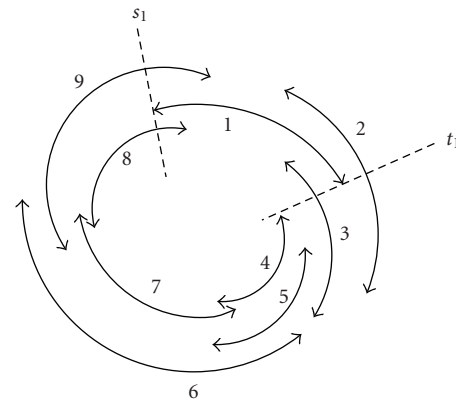


Figure 2: Example of covers.

We denote the *cover range* of a sensor node $i \in S$ as $[s_i, t_i]$ as shown in Figure 1. We refer to s_i as the start angle and t_i as the end angle. $s_i < t_i$ for most $i \in S$. Sensor j that covers 0° would have $s_j > t_j$. We denote the set of sensors that cover 0° as S_0.

S is a *proper* set of sensors if $\forall i \in S, \nexists j \in S$, such that $s_j \leq s_i < t_i \leq t_j$ or $s_i \leq s_j < t_j \leq t_i$. In other words, S is a *proper* set of sensors if none of the sensors has cover range containing or contained inside another sensor. Otherwise, S is known as a *general* set of sensors. As an example, the set of sensors shown in Figure 4 is a *proper* set of sensors. On the other hand, the set of sensors shown in Figure 2 is a *general* set of sensors.

3.2. Cover, Proper Cover, and Mutually Disjoint Cover Set. A set $D \subseteq S$ is a *cover* if for each angle $\gamma \in [0°, 360°]$, there exists a sensor i in D such that $\gamma \in [s_i, t_i]$. In other words, $\bigcup_{i \in D}[s_i, t_i] = [0°, 360°)$. Figure 2 illustrates a scenario of 9 sensors surrounding a target object. Each arrow represents

the cover range of a node. $\{1,3,5,7,8\}$, $\{1,2,3,5,6,9\}$, and $\{1,3,5,7,9\}$ are all covers.

A *proper cover* is a cover that every sensor is essential for the coverage. That is, if D is a proper cover, then $D \setminus \{i\}$ is not a cover for any $i \in D$. For instance, $\{1,3,5,7,8\}$ and $\{1,3,5,7,9\}$ are *proper covers* in Figure 2.

Suppose D_1 and D_2 are two covers. We say D_1 and D_2 are *mutually disjoint* if $D_1 \bigcap D_2 = \emptyset$. A set of covers $\digamma$ is a *mutually disjoint cover set* if any arbitrary pair of covers inside $\digamma$ are mutually disjoint. Formally, for any $D_1, D_2 \in \digamma$, $D_1 \bigcap D_2 = \emptyset$. Refering to Figure 4, $\digamma$ can be $\{\{0,3,6,9\},\{1,4,7,10\},\{2,5,8,11\}\}$.

3.3. Sensor Lifetime—Uniform and Nonuniform, Schedule, Network Lifetime, and Problem Statement. We adopt the cycle-based scheduling mechanism as in [33, 34]. In other words, a cover is selected to monitor the target object in each cycle, and the duration of each cycle is the same. The *sensor lifetime* of a sensor i, $B(i)$, is the number of cycles it can be turned on. If $B(i) = B$ for all $i \in S$, we say that the sensors have *uniform* battery. Otherwise, they have *nonuniform* battery.

In each cycle, a set of sensors is turned on to monitor the target. Those sensors that are not in the set can switch to sleep mode to conserve energy. A *schedule* defines the cycles in which a sensor has to be turned on or off. We use a matrix SC of size $L \times |S|$ to represent the schedule. $SC(l,i) = 1$ means Sensor i should be turned on at cycle l; $SC(l,i) = 0$ otherwise. With the definition of the *schedule*, *network lifetime* L refers to the number of cycles that the perimeter of the target can be monitored continuously according to the schedule SC. Note that different schedules will result in different network lifetimes. An *optimal scheduling algorithm* finds a schedule $SC^S_{\max}$ such that the maximum network lifetime $L^S_{\max}$ on a set of sensors S can be achieved. In other words, no scheduling algorithm can find a schedule SC which can achieve a network lifetime $L^S > L^S_{\max}$ on S.

Therefore, the objective of the scheduling problem is to find a schedule SC to monitor the target object continuously so that the lifetime L^S is maximized. Formally, we would like to find an SC which maximizes L^S, such that $\forall l \le L^S$, $\bigcup_{i\in\chi_l}[s_i,t_i] = [0^\circ, 360^\circ)$, where $\chi_l = \{i \mid SC(l,i) = 1\}$.

3.4. Neighbors, Backward Neighbors, and Forward Neighbors. If two nodes' cover ranges overlap, they are neighbors. Formally, i and j are neighbors if $s_i < s_j < t_i$ or $s_i < t_j < t_i$ (This applies when both i and j do not cover 0°. The definition can be extended easily to ranges that cover 0° but we leave it out for the ease of discussion.). We assume that a node can only communicate with its neighbors. It is possible that $[s_j,t_j]$ completely contains $[s_i,t_i]$, such as Sensors 9 and 8 in Figure 2. When two sensors have overlapping cover ranges and none of them is contained in the other, one of them is a *backward neighbor* and the other is a *forward neighbor*. i is a backward neighbor of j and j is a forward neighbor of i if $s_i < s_j < t_i$. Refer to Figure 2, Sensors 2 and 3 are forward neighbors of Sensor 1, while Sensors 9 and 8 are backward neighbors of Sensor 1.

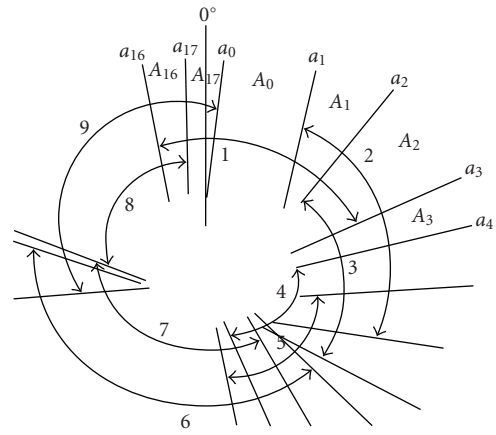


Figure 3: Example of perimeter segments.

3.5. Perimeter Segment and Properties of the Problem. An endpoint can be any start angle s_i or end angle t_i of any sensor $i \in S$. Suppose all the endpoints are distinct, there are $2N$ endpoints formed by N sensors in the network. We denote each endpoint as a_j where $0 \le j \le 2N-1$ in ascending order. Figure 3 illustrates the endpoints of the sensors in Figure 2. The *perimeter segment* is the portion of the perimeter of the target object formed by a pair of consecutive endpoints $[a_j, a_{j+1}]$, where $0 \le j < 2N-1$. When $j = 2N-1$, the perimeter segment is $[a_{2N-1}, a_0]$. We denote A_j as the segment $[a_j, a_{j+1}]$ as shown in Figure 3. Now, we use Λ to denote the set of all perimeter segments formed by consecutive endpoints, that is, $\Lambda = \{[a_0,a_1],[a_1,a_2],\ldots,[a_{2N-1},a_0]\} = \{A_0,\ldots A_j\ldots,A_{2N-1}\}$. Since $\bigcup_{0\le j\le 2N-1} A_j = [0^\circ, 360^\circ)$, if $D \subseteq S$ covers A_j, $\forall 0 \le j \le 2N-1$, then D is a cover. Similarly, if D is a cover, it covers all $A_j \in \Lambda$.

Since D covers all $A_j \in \Lambda$ if and only if D covers $[0^\circ, 360^\circ)$, an upper bound of the lifetime of covering $[0^\circ, 360^\circ)$ can be derived. To start the discussion, we further define some terms as follows and we use the example in Figure 3 to explain them. For the ease of discussion, we use $B = B(i) = 2$, $\forall i \in S$ in the examples.

(i) H^S_j denotes the set of sensors in S which cover A_j. Formally, $H^S_j = \{i \mid i \in S \text{ covers } A_j\}$. For instance, $H^S_1 = \{1,2\}$. We further let ρ^S_j denote $|H^S_j|$. For instance, $\rho^S_1 = 2$.

(ii) q^S_j denotes the total number of energy cycles of the sensors in H^S_j. Formally, $q^S_j = \sum_{i\in H^S_j} B(i)$. For instance, $q^S_1 = 4$ since $B(1)+B(2) = 4$.

(iii) $A^S_{\min}$ denotes the A_j with the minimum q^S_j. Formally, $A^S_{\min} = \{A_j \mid \min_{0\le j\le 2N-1} q^S_j\}$, for example, $A^S_{\min} = A_0$.

(iv) $H^S_{\min} = \{i \mid i \in S \text{ covers } A^S_{\min}\}$, $q^S_{\min} = \sum_{i\in H^S_{\min}} B(i)$, and $\rho^S_{\min} = |H^S_{\min}|$. For instance, $H^S_{\min} = \{1\}$, $q^S_{\min} = 2$, and $\rho^S_{\min} = 1$.

We now prove that the maximum achievable lifetime of sensor set S ($L^S_{\max}$) is upper bounded by $q^S_{\min}$.

Property 1. $L^S_{\max} \leq q^S_{\min}$.

Proof. Suppose $L^S_{\max} > q^S_{\min}$. In each cycle, at least one sensor $x \in H^S_{\min}$ is selected to cover $A^S_{\min}$. Since each sensor $x \in H^S_{\min}$ can be activated at most $B(x)$ units of time, $A^S_{\min}$ is at most covered $q^S_{\min} = \sum_{x \in H^S_{\min}} B(x)$ units of time. Afterwards, $A^S_{\min}$ cannot be covered by any sensor. Therefore, this leads to the contradiction that $L^S_{\max} > q^S_{\min}$. □

When all the sensors have the same initial battery B, $q^S_{\min} = \rho^S_{\min} \times B$. Therefore, in the uniform battery case, Property 1 can naturally be extended to Property 2.

Property 2. If $B(i) = B \ \forall i \in S$, then $L^S_{\max} \leq (\rho^S_{\min} \times B)$.

4. Proper Set of Sensors

In this section, we describe how to find a schedule when S is a proper sensor set, such that there is no sensor whose cover range is contained inside another sensor. We consider two scenarios: uniform battery scenario and nonuniform battery scenario. In the following, we drop the superscript S in notations for simplicity if the context is clear.

4.1. Uniform Battery Scenario. In this scenario, by Property 2, the maximum lifetime $L_{\max} \leq (\rho_{\min} \times B)$. Hence, if there exists a mutually disjoint cover set $\mathcal{F}$ where $|\mathcal{F}| = \rho_{\min}$, an optimal schedule can be found by activating each cover $D \in \mathcal{F}$ for B cycles. We consider two cases: (1) $N \bmod \rho_{\min} = 0$; (2) $N \bmod \rho_{\min} > 0$.

(1) $N \bmod \rho_{\min} = 0$: in this case, we can find a mutually disjoint cover set $\mathcal{F}$ with exactly $\rho_{\min}$ covers and each cover has exactly $N/\rho_{\min}$ number of sensors. To find these covers, we first label the nodes in S in the clockwise manner where the most anti-clockwise sensor in $H_{\min}$ is labeled as n_0. In Figure 4, $H_{\min} = \{0, 1, 2\}$ and so n_0 is Sensor 0. Let $D_0, D_1, \ldots, D_{\rho_{\min}-1}$ be subsets of S where $D_i = \{n_j \mid j \bmod \rho_{\min} = i\}$. On the other hand, a collection of D_i, where $0 \leq i \leq \rho_{\min} - 1$, is denoted as $\mathcal{F}$. Refering to Figure 4, a mutually disjoint cover set $\mathcal{F} = \{D_0 = \{0, 3, 6, 9\}, D_1 = \{1, 4, 7, 10\}, D_2 = \{2, 5, 8, 11\}\}$ with $\rho_{\min} = 3$ covers is formed. We show that $\mathcal{F} = \{D_i \mid 0 \leq i \leq \rho_{\min} - 1\}$ is a mutually disjoint cover set with $\rho_{\min}$ covers by the following property and lemma.

Property 3. If S is proper, n_i is a backward neighbor of $n_{i+\rho_{\min}}$.

Proof. Let the start angle and end angle of sensor n_i be s_i and t_i, respectively. Let $j = i + \rho_{\min}$. Suppose $n_i \in S$ is not the backward neighbor of $n_j \in S$ where $i < j$. This implies that $s_i < t_i < s_j < t_j$ as n_i and n_j do not overlap in terms of sensing range. Since $A_{\min}$ is covered by $\rho_{\min}$ sensors, we know that the range $[t_i, s_j]$ (which may be a union of several perimeter segments) must be covered by at least $\rho_{\min}$ sensors. In this case, $[t_i, s_j]$ is covered by Sensors $n_{i+1}, \ldots, n_{j-1}$. It implies that $[t_i, s_j]$ is only covered by $\rho_{\min} - 1$ sensors. This contradicts to the definition of $\rho_{\min}$. Therefore, n_i is a backward neighbor of $n_{i+\rho_{\min}}$. □

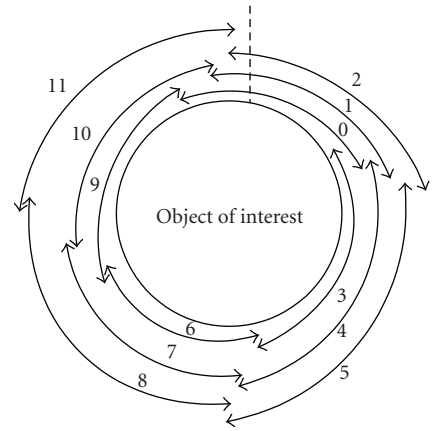


Figure 4: Example of $N \bmod \rho_{\min} = 0$.

Lemma 1. *If $N \bmod \rho_{\min} = 0$, then $\mathcal{F} = \{D_i \mid 0 \leq i \leq \rho_{\min} - 1\}$ forms a mutually disjoint cover set with $\rho_{\min}$ covers.*

Proof. By Property 3, we know that $n_i \in D_i$ is the backward neighbor of $n_{i+\rho_{\min}} \in D_i$. Similarly, $n_{i+\rho_{\min}} \in D_i$ is the backward neighbor of $n_{i+2(\rho_{\min})} \in D_i$, and so on. Since N is divisible by $\rho_{\min}$, for each D_i, the last member is $l_i = n_{i+(N/\rho_{\min}-1)*\rho_{\min}}$. By Property 3, l_i is the backward neighbor of $n_i \in D_i$. This implies that every D_i, where $0 \leq i \leq \rho_{\min} - 1$, forms a cover. As a result, $\rho_{\min}$ covers are formed. On the other hand, a sensor in S falls into exactly one of the D_i, where $0 \leq i \leq \rho_{\min} - 1$. Therefore, $\mathcal{F} = \{D_i \mid 0 \leq i \leq \rho_{\min} - 1\}$ forms a mutually disjoint cover set with $\rho_{\min}$ covers. □

The direct consequence of Lemma 1 is that the maximum lifetime schedule can be achieved by activating each cover $D_i \in \mathcal{F}$, where $0 \leq i \leq \rho_{\min} - 1$, for B units of time and the corresponding lifetime is $B \times \rho_{\min}$.

(2) $N \bmod \rho_{\min} > 0$: in this case, Lemma 1 cannot be directly applied to S. Refering to Figure 5, if we set $D_i = \{n_j \mid j \bmod \rho_{\min} = i\}$, then $D_0 = \{0, 3, 6, 9, 12\}$, $D_1 = \{1, 4, 7, 10, 13\}$, and $D_2 = \{2, 5, 8, 11\}$. However, D_2 is not a cover. To solve the problem, one possible way is to split S into S_1 with N_1 sensors and S_2 with N_2 sensors such that $N_1 \bmod \rho^{S_1}_{\min} = 0$ and $N_2 \bmod \rho^{S_2}_{\min} = 0$ and $\rho^S_{\min} = \rho^{S_1}_{\min} + \rho^{S_2}_{\min}$. Then, Lemma 1 can be applied to S_1 and S_2 individually to form two corresponding mutually disjoint cover sets, denoted as $\mathcal{F}_1$ and $\mathcal{F}_2$, with $\rho^{S_1}_{\min}$ covers and $\rho^{S_2}_{\min}$ covers, respectively. Refering back to Figure 5, two mutually disjoint cover sets with 3 covers in total do exist. Suppose the broken arcs are sensors in S_1, that is, $\rho^{S_1}_{\min} = 2$. On the other hand, solid arcs are sensors in S_2, that is, $\rho^{S_2}_{\min} = 1$. In this case, we set $D^{S_1}_i = \{n_j \mid j \in S_1 \bmod \rho^{S_1}_{\min} = i\}$ in S_1 and $D^{S_2}_i = \{n_j \mid j \in S_2 \bmod \rho^{S_2}_{\min} = i\}$ in S_2. Hence, $\mathcal{F}_1 = \{D^{S_1}_0, D^{S_1}_1\}$ where $D^{S_1}_0 = \{1, 4, 7, 9, 12\}$ and $D^{S_1}_1 = \{2, 5, 8, 11, 13\}$. $\mathcal{F}_2 = \{D^{S_2}_0\}$ where $D^{S_2}_0 = \{0, 3, 6, 10\}$. Finally, $\mathcal{F} = \mathcal{F}_1 \bigcup \mathcal{F}_2$ forms a mutually disjoint cover set with 3 covers in total.

However, let us consider Figure 6. In this example, the maximum network lifetime is 3 if $\forall i \in S$, $B(i) = B = 2$ units. This can be achieved by activating the sets $\{1, 3, 5, 7, 9\}$,

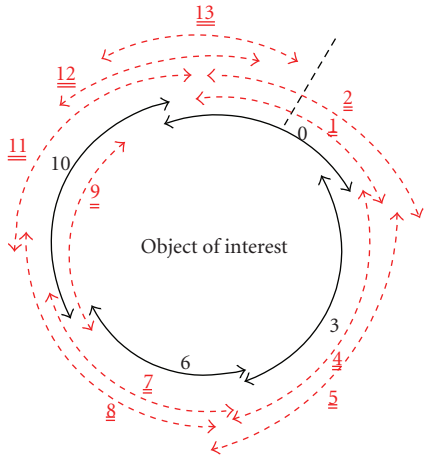


Figure 5: Example of N mod $\rho_{\min} > 0$.

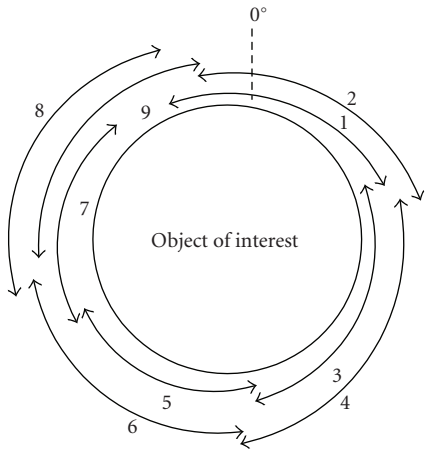


Figure 6: Example of non-existence of a mutually disjoint cover set with $\rho_{\min}$ covers.

$\{2, 4, 6, 7, 9\}$, and $\{1, 3, 5, 6, 8\}$, each for 1 time unit. In this example, although we know that $\rho^S_{\min} = 2$, there is no way for us to find F_1 and F_2 such that $\rho^{S_1}_{\min} + \rho^{S_2}_{\min} = 2$. Instead, we can only identify a single mutually disjoint cover set that contains one cover. By activating this only cover, the achievable network lifetime is 2 units. Hence, we can conclude that when we cannot find a mutually disjoint cover set with exactly $\rho^S_{\min}$ covers using the sensor partitioning method, we cannot be sure we have an optimal solution. This implies that even if we can find a mutually disjoint cover set with $\rho^S_{\min} - 1$ covers, the set found may not be an optimal solution as shown in the example in Figure 6. The sensor partitioning problem aforementioned can be transformed to a shortest path problem and solved with the time complexity of $O(N^2)$. We refer readers for details in [35, 36].

4.2. Nonuniform Battery Scenario. In the nonuniform battery scenario, it is possible to transform S into a sensor set S' such that all sensors have the same initial battery. For every $i \in S$ with $B(i)$ units of battery and with the cover range $[s_i, t_i]$, we put $B(i)$ sensors in S', each with 1 unit of battery and cover range $[s_i, t_i]$. Then, the maximum achievable lifetime on S is the same as the maximum achievable lifetime on S'. However, S' is no longer a proper set since all the sensors we put in S' based on $i \in S$ share the same cover range. Figure 7 illustrates an example of the transformation with $B(i) = B = 2\ \forall i \in S$. The next section studies how to find a schedule when the sensor set is not proper.

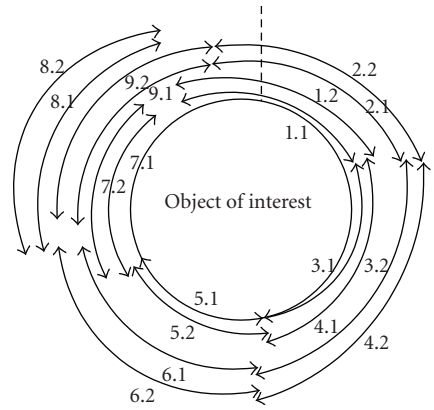


Figure 7: Example of constructed instance set S' from S.

5. General Set of Sensors

In this section, we would like to prove that the perimeter coverage scheduling problem is NP-hard if S is a *general* set of sensors. We first consider the situation where $\forall i \in S$, $B(i) = 1$. To prove the perimeter coverage scheduling problem is NP-hard, a reduction is constructed from the k-coloring problem on a circular-arc graph, which was first shown to be NP-hard in [37], to the perimeter coverage scheduling problem. Here, $\langle S, L_{\max} \rangle$ denotes a perimeter scheduling problem instance with a set of sensors S, and the maximum lifetime achievable on S is $L_{\max}$.

Before we go into details about the transformation, we first describe the k-coloring problem on a circular arc graph G. A circular arc i in G is denoted as $[\alpha_i, \beta_i]$, where α_i denotes the anti-clockwise endpoint, and β_i denotes the clockwise endpoint of a circular arc. In other words, a circular arc is open on the clockwise endpoint. We further assume that all the endpoints are distinct. Arcs i and j are neighbors if the arcs overlap. In the k-coloring problem, we would like to determine whether k colors are enough to assign a color to each arc such that no two neighbor arcs share the same color.

We further introduce some notations so as to define the problem formally. Here, we use N to denote the number of circular arcs in G. Therefore, there are $2N$ endpoints in G. We sort the endpoints in ascending order, and denote them as a_j, where $0 \le j \le 2N - 1$. A pair of consecutive endpoints $[a_j, a_{j+1})$ are denoted as A_j. We use H_j to denote the set of arcs covering A_j, that is, $\forall i \in H_j$, $\alpha_i \le a_j < a_{j+1} \le \beta_i$. Let C_λ denote the set of circular arcs with Color λ. Since the sensors in H_j are neighbors of each other, at least $\max_{0 \le j \le 2N-1} |H_j|$ colors are needed.

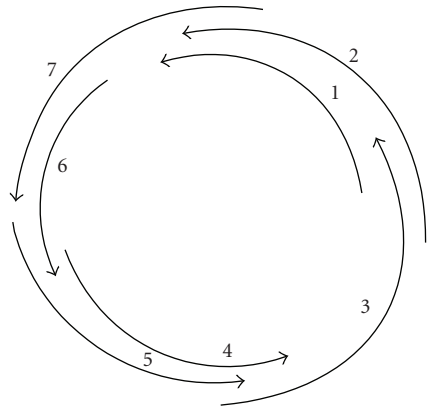


Figure 8: Example of a set of circular arcs G.

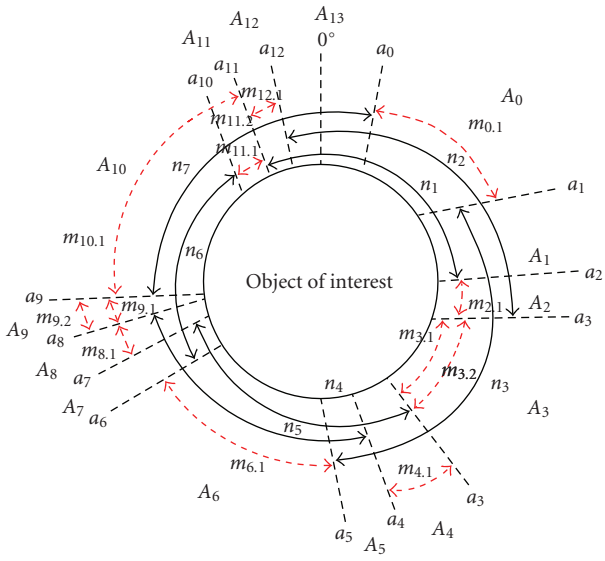


Figure 9: Example of the constructed set of sensors S from G.

Given any instance $\langle G, k\rangle$ of the k-coloring problem in the circular arc graph, where k is an integer larger than or equal to $\max_{0\le j\le 2N-1}|H_j|$, an instance $\langle S, L_{\max}\rangle$ of the perimeter coverage scheduling problem can be constructed from G as follows.

(1) For each $i \in G$, put Sensor n_i in S and the cover range of n_i is $[\alpha_i, \beta_i]$.

(2) $\forall 0 \le j \le 2N-1, k-|H_j|$ sensors with cover range of $[a_j, a_{j+1}]$ are put in S. These sensors are named with $m_{j.e}$, where $1 \le e \le k-|H_j|$.

(3) $\forall x \in S, B(x) = 1$.

Consider Arcs 1,...,7 in Figure 8; Sensors n_1,...,n_7 are put into S according to (1) in Figure 9. Suppose $k = 3$. Consider the range $[a_0, a_1]$; according to (2), Sensor $m_{0.1}$ is put in S. Similarly, Sensors $m_{2.1}$, $m_{3.1}$, $m_{3.2}$, $m_{4.1}$, $m_{6.1}$, $m_{8.1}$, $m_{9.1}$, $m_{9.2}$, $m_{10.1}$, $m_{11.1}$, $m_{11.2}$, and $m_{12.1}$ are put into S as shown in Figure 9. As a result, we know that $\rho_{\min} = k$. Since $\forall i \in S$, $B(i) = B = 1$. Therefore, $L_{\max} \le k$.

Lemma 2. *G is k-colorable if and only if the maximum lifetime on S is $L_{\max} = k$.*

Proof. "⇒" part: if G is k-colorable, we can put the arcs into k different partitions $C_1, \ldots, C_k$ according to their colors. We show that k covers $D_1, \ldots, D_k$ can be formed in S by using the following approach:

(1) if $i \in C_\lambda$, $n_i \in D_\lambda$.

(2) $\forall 1 \le \lambda \le k$, $\forall 0 \le j \le 2N-1$, if there is no sensor in D_λ which can cover A_j, then we can randomly select a sensor $m_{j.e} \in S$, where $1 \le e \le k-|H_j|$, with cover range $[a_j, a_{j+1}]$ into D_λ to cover A_j. Recall that there are $|H_j|$ arcs in G covering A_j and these arcs must fall into $|H_j|$ different colors, that is, different partitions. Since we have constructed $k-|H_j|$ sensors with cover range $[a_j, a_{j+1}]$ in S, if there is no sensor in D_λ which can cover A_j, then we can always find a sensor $m_{j.e} \in S$, where $1 \le e \le k-|H_j|$, with cover range $[a_j, a_{j+1}]$ to put into D_λ.

The corresponding D_λ formed, $\forall 1 \le \lambda \le k$, are covers, because $\forall 0 \le j \le 2N-1$; there is a sensor in D_λ covering A_j. At the same time, $B(x) = B = 1$, $\forall x \in S$. Therefore, $D_\lambda \bigcap D_\kappa = \emptyset$, for any $1 \le \lambda \ne \kappa \le k$. Since $\rho_{\min}$ is k and $B = 1$, by Property 2, the maximum lifetime $L_{\max}$ is upper bounded by k. Hence, the maximum lifetime $L_{\max}$ can be achieved by activating each cover for 1 unit of time. This results in $L_{\max} = k$. Therefore, if G is k-colorable, then the maximum lifetime on S is $L_{\max} = k$.

"⇐" part: If the maximum lifetime $L_{\max}$ on S is k and the corresponding schedule is $SC_{\max}$, then there must exist a mutually disjoint cover set with k covers since each sensor has a battery lifetime of 1 unit. Without loss of generality, these k covers are $D_1, \ldots, D_k$. In other words, $D_\lambda = \{x \mid x \in S \text{ and } SC_{\max}(\lambda, x) = 1\}$, where $1 \le \lambda \le k$.

Hence, with the existence of k covers, k partitions $C_1, \ldots, C_k$ on G can be formed by the following approach:

> $\forall 1 \le \lambda \le k$, $\forall n_i \in D_\lambda$, $i \in C_\lambda$. Note that we do not need to consider the sensors $m_{j.e}$ in D_λ, where $0 \le j \le 2N-1$ and $1 \le e \le k-|H_j|$.

Since S is constructed in such a way that every A_j, where $0 \le j \le 2N-1$, is exactly covered by k sensors, if there exist k covers, then every A_j is exactly covered by 1 sensor in each cover. Therefore, by constructing C_λ, $\forall 1 \le \lambda \le k$, according to the above description, neighbor arcs must fall into different color partitions. As a result, C_λ, $\forall 1 \le \lambda \le k$, forms a color partition and so G is k-colorable if the maximum lifetime on S is $L_{\max} = k$. □

By Lemma 2, Theorem 1 can then be developed.

Theorem 1. *The perimeter coverage scheduling problem is NP-hard.*

6. Approximation Solutions

Since the perimeter coverage problem is NP-hard, it is not likely to have a polynomial time solution. The problem can be formulated as a mixed integer programming problem,

and solved by some existing software. However, a centralized approach is not appropriate. Therefore, we develop a distributed approximation solution with a small overhead. Before describing our algorithm, we first show that any scheduling algorithm that selects a proper cover in each cycle is a 2-approximation algorithm.

6.1. Properties of Proper Covers

Property 4. In a proper cover, there are at most two sensors covering y for each $y \in [0^\circ, 360^\circ)$.

Proof. Suppose $x, y, z \in D$ all cover range $[a, b]$, and the cover ranges of x, y, and z are $[s_x, t_x]$, $[s_y, t_y]$, and $[s_z, t_z]$, respectively. Without loss of generality, we assume $s_x \leq s_y \leq s_z$. In this case, $s_x \leq s_y \leq s_z < a < b < t_x \leq t_y \leq t_z$. This contradicts to the definition of proper cover as y is redundant. □

Property 4 says that each portion on the perimeter is at most covered by 2 sensors in a proper cover. Lemma 3 shows that any scheduling algorithm which selects a proper cover to monitor the target object in each cycle is a 2-approximation solution.

Lemma 3. *Suppose SC is the schedule which selects a proper cover in each cycle. SC has a lifetime $L \geq L_{\max}/2$, where $L_{\max}$ is the maximum lifetime on S.*

Proof. We use $V(i,t)$ to denote the remaining energy of Sensor i after cycle t. At the beginning, $V(i,0) = B(i)$, $\forall i \in S$. We further use the term $q_j(t)$ to denote $\{\sum V(i,t) \mid i \text{ covers } A_j\}$. That is, $q_{\min}(t) = \{\sum V(i,t) \mid i \text{ covers } A_{\min}\}$ and $q_{\min}(0) = q_{\min}$. By Property 4, at most 2 sensors cover $A_{\min}$ in cycle t. In fact, this implies that $q_{\min}(t) = q_{\min}(t-1) - 2$ if 2 sensors are required to cover $A_{\min}$ in each cycle t, $\forall 1 \leq t \leq L$. Note that this is the worst possible case for the selection of a proper cover in each cycle as $A_{\min}$ is covered by sensors with the least total amount of energy cycle. Here, we assume $q_{\min}(t) = q_{\min} - 2t$. Therefore, $q_{\min}(\lfloor q_{\min}/2 \rfloor) \geq 0$ if $q_{\min}$ is even and $q_{\min}(\lfloor q_{\min}/2 \rfloor) \geq 1$ if $q_{\min}$ is odd. In case $q_{\min}$ is odd, all the sensors can be activated in the last unit of time. This is possible as $q_j(\lfloor q_{\min}/2 \rfloor) \geq q_{\min}(\lfloor q_{\min}/2 \rfloor) \geq 1$. Since $L_{\max} \leq q_{\min}$, $L \geq \lceil L_{\max}/2 \rceil$. □

All the heuristic algorithms we proposed in [5] identify a proper cover in each cycle. By Lemma 3, we know that all of them are 2-approximation solutions if no message transmission cost is considered. Unfortunately, these approaches have very high message overheads. It may require $O(q_{\min} \times$ minimum size of cover $\times$ size of $S_0)$ number of messages during the whole network lifetime, where S_0 denotes the set of sensors with cover range passing through 0°.

6.2. Our Proposed 2-Approximation Algorithm. To reduce message overhead, our approximation algorithm identifies as many proper covers as possible by circulating messages around the sensors once. This can be done by using a similar method suggested in [3], which will be discussed later in this section. In the uniform battery case, we try to find a mutually disjoint cover set with as many proper covers as possible. Then, each proper cover can be activated for B units of time. On the other hand, in the nonuniform battery scenario, each sensor with $B(i)$ units of battery is transformed to $B(i)$ sensors with 1 unit each similar to that discussed in Section 4.2, and thus the battery of the transformed set of sensors becomes uniform. Then, we try to find a mutually disjoint cover set with as many proper covers as possible and each cover is activated for 1 unit of time.

Note that although the uniform battery case is discussed here, the suggested algorithm can be extended to the nonuniform battery case easily. Basically, our proposed algorithm works as follows. A message is passed around the sensors in the clockwise direction. A sensor passing through 0° creates the message. The message contains information for constructing mutually disjoint covers. When a sensor receives the message, it is responsible for filling in the information and passes it to a neighbor sensor. When the message is received by another sensor passing through 0°, mutually disjoint covers are identified.

Suppose S_0 denotes the set of sensors with cover range passing through 0°. The sensor with the largest ending angle in S_0, denoted as q, initiates the algorithm. It first puts its neighbors which are in S_0 into $|S_0|$ different sets. Then, for each set, it identifies some remaining neighbors to be put in the set, such that the sensors in the set can cover the range $[0^\circ, t_q]$. The detailed procedure is as follows: q keeps track of a sorted order of the sets based on the ending angles of the cover ranges in the sensors of the sets. It takes the set with the smallest ending angle, and identifies a sensor that can extend its cover range. After putting the sensor in the set, the ending angle is updated. The process ends when there is no more sensor to extend the cover range of the set. It is possible that not all the sets can cover the whole range $[0^\circ, t_q]$ after all the neighbors of q are put in the sets. In this case, these sets are removed and would not be put in the message. We argue that the removal of these sets will not affect the approximation ratio of the algorithm. Suppose no neighbor can extend the cover range of a set without a gap. Since this is the set with the smallest ending angle among all the sets and there are at least $\rho^S_{\min}$ sensors covering the gap, this implies that $\rho^S_{\min}$ sensors that can cover the gap must fall into $\rho^S_{\min}$ different sets. Therefore, there are still $\rho^S_{\min}$ different sets remaining even the aforementioned one is removed.

Refering to Figure 10, Sensor 3 is the sensor in S_0 with the largest end angle, so q is Sensor 3. Sensor 3 puts Sensors 1, 2, and 3 into 3 different sets, denoted as Set 1, Set 2, and Set 3, respectively. Since Sensor 1 ends before Sensor 2 and Sensor 3 do, Sensor 3 tries to put a neighbor into Set 1 to extend its cover range. Hence, Sensor 4 is put into Set 1 by Sensor 3. Now, Sensor 3 updates the ending angle of each set and finds that Set 1 still ends before the others. Unfortunately, other neighbors of Sensor 3 cannot be put into Set 1 to extend the range of Set 1. Set 1 will then be removed. At this moment, only Set 2 and Set 3 are remaining. Sensor 3 puts Sensor 6 into Set 2, and Sensor 7 into Set 3. Note that Sensor 5 is not put in either set because it cannot extend the cover ranges of both sets. Now, Set 2 consists of $\{2, 6\}$, while Set 3 consists of $\{3, 7\}$.

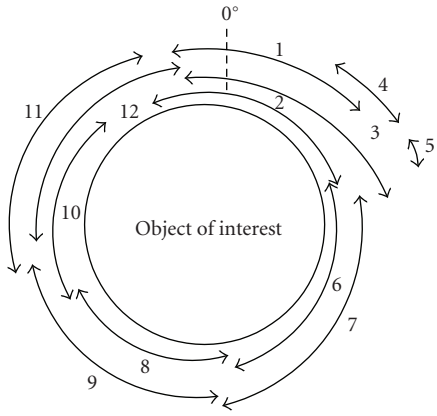


Figure 10: An example to illustrate the proposed algorithm.

After the sets are identified, q puts the information in a message. The message format is $\langle\langle 1, q_1, M_1\rangle, \ldots, \langle j, q_j, M_j\rangle, \ldots, \langle L, q_L, M_L\rangle\rangle$, where each tuple ¡$j, q_j, M_j$¿ contains the information of one set. The first element represents the identifier of the set. The second element represents the first member in the set j. Note that this member must be a sensor in S_0 and can later be used for determining whether the set j forms a cover or not. Without loss of generality, we assume the tuple in the message is sorted according to the ending angle of $q_j \in S_0$. The third element represents the list of members selected to be included in the set j by the node sending the message. The message is sent to the only *greedy forward neighbor* of q. The *greedy forward neighbor* of q, denoted as $GFN(q)$, is the forward neighbor of q with the largest end angle. Refering back to the example in Figure 10. Since Sensor 3 selects $\{2, 6\}$ into Set 2 and $\{3, 7\}$ into Set 3, it sends out $\langle\langle 2, q_2 = 2, M_2 = \{2, 6\}\rangle, \langle 3, q_3 = 3, M_3 = \{3, 7\}\rangle\rangle$ to Sensor $GFN(3) = 7$.

When $GFN(q)$ receives the message, it tries to extend the cover ranges of the sets by putting its neighbors in them. The procedure is similar to what q did earlier. Refering back to the example in Figure 10, when Sensor 7 receives $\langle\langle 2, q_2 = 2, M_2 = \{2, 6\}\rangle, \langle 3, q_3 = 3, M_3 = \{3, 7\}\rangle\rangle$, it finds that the ending angle of Set 2 is smaller than that of Set 3. Therefore, it puts Sensor 8 into Set 2. Then, it further puts Sensor 9 into Set 3, and then informs its own greedy forward neighbor to continue the set construction.

We let the node with the smallest start angle in S_0 be q'. Since the message has been passed in the clockwise manner, it must be received by one of the backward neighbors of q', say x. After x has performed the set construction, it sends the message to q' instead of its greedy forward neighbor. After receiving the message, q' continues the set construction for those sets which have not yet formed a cover. After considering all the neighbors that can extend the cover range of the sets, if there still exist sets which have not yet formed covers, q' reorganizes those sets. To avoid ambiguity, we now use covers to denote those sets which have formed covers. q keeps track of a sorted order of the sets based on the ending angles of the cover ranges in the sensors of the sets. It starts to form a cover from the set with the smallest ending angle. If no neighbors can be identified to form a cover, the sensors in the set with the largest identifier will be used, and the set with the largest identifier will no longer be considered for forming a cover. We argue that there will be at least $\rho^S_{\min}/2$ covers formed at the end of the process. As discussed before, when no covers have been formed yet, $\rho^S_{\min}$ sets are remaining. However, the worst case occurs when half of the sets are removed by q' so that the sensors can be moved to fill up the gaps to form the covers in the other half. Note that each perimeter segment is covered by at most 2 sensors in a proper cover, and so it is not possible to remove more than $\rho^S_{\min}/2$ sets.

Let us consider the example in Figure 10 again. Since Sensor 7 selects $\{8\}$ into Set 2 and $\{9\}$ into Set 3, it sends $\langle\langle 2, q_2 = 2, M_2 = \{8\}\rangle, \langle 3, q_3 = 3, M_3 = \{9\}\rangle\rangle$ to $GFN(7) = 9$. Sensor 9 sends $\langle\langle 2, q_2 = 2, M_2 = \{10\}\rangle, \langle 3, q_3 = 3, M_3 = \{11\}\rangle\rangle$ to $GFN(9) = 11$. Sensor 11 further puts Sensor 12 into Set 2 and then removes Set 3. Therefore, it sends $\langle\langle 2, q_2 = 2, M_2 = \{12\}\rangle\rangle$ to Sensor $q' = 2$ which is the forward neighbor of Sensor 11. Now, Sensor 2 finds that Sensor 12 is the backward neighbor of $q_2 = 2$. As a result, a cover is formed in Set 2.

It is worth mentioning that all other sensors which do not involve in the circulation of the message can overhear the message transmission. Therefore, they know the set which they have been selected in. When q' completes the covers formation process, it can construct the activation schedule. Then, q' announces the activation schedule of different covers. The message about the activation schedule can be circulated through the whole network in the same manner as the cover identification process. By overhearing this message, all the sensors know their schedules. The whole process can then be terminated when the message circulates back to q' again. The pseudocodes of Sensors $GFN(q)$ and q' who receives the cover identification message are provided in Algorithms 1 and 2, respectively.

In fact, we can further optimize the message overheads in terms of the number of messages required by removing the activation schedule circulation process. To do so, we can reorganize the sets when we find that the cover range of a set cannot be extended by putting a neighbor into the set. Here, we always use the sensors in the set with the largest identifier to extend the cover range of a set, and the set with the largest identifier will no longer be considered for forming a cover. By doing so, we know that if L covers are formed finally, they will have set identifiers $1, \ldots, L$. Since the sensors can overhear in which set they are selected in, they can activate automatically following the time slot allocated to the cover with a specific set identifier without the need of circulating the activation schedule.

7. Simulation Results

For comparison, we have implemented several algorithms. The first algorithm is the optimal solution found by transforming the maximum lifetime scheduling problem to the Multicommodity Network Flow problem, and solved by AMPL-CPLEX [38], which is denoted as *Optimal* in our figures. The second algorithm is the one proposed in this

1: M_i is a set of sensors selected into available set i by c. Initially, the last member of M_i' is put into M_i.
2: l_i is the last member in M_i.
3: m is the index of the set which has the smallest ending angle among all l_i, where $1 \le i \le L$.
4: T is the next hop target for the message.
5: **If** q' is forward neighbor of c **then**
6: T is q'.
7: **else**
8: T is $GFN(c)$.
9: **end if**
10: /* c selects neighbors to extend the cover range of the sets. */
11: **for** each neighbor j **do**
12: **If** j cannot extend the cover range of set m **then**
13: Remove the Set m.
14: **end if**
15: **if** j is the forward neighbor of l_m **then**
16: Put j into M_m and $l_m = j$.
17: **end if**
18: Find the set with the smallest end angle and denote it as m.
19: **end for**
20: Send the message $\langle\langle i, q_i, M_i\rangle\rangle$, for $i = 1$ to L and set i has not been removed, to T.

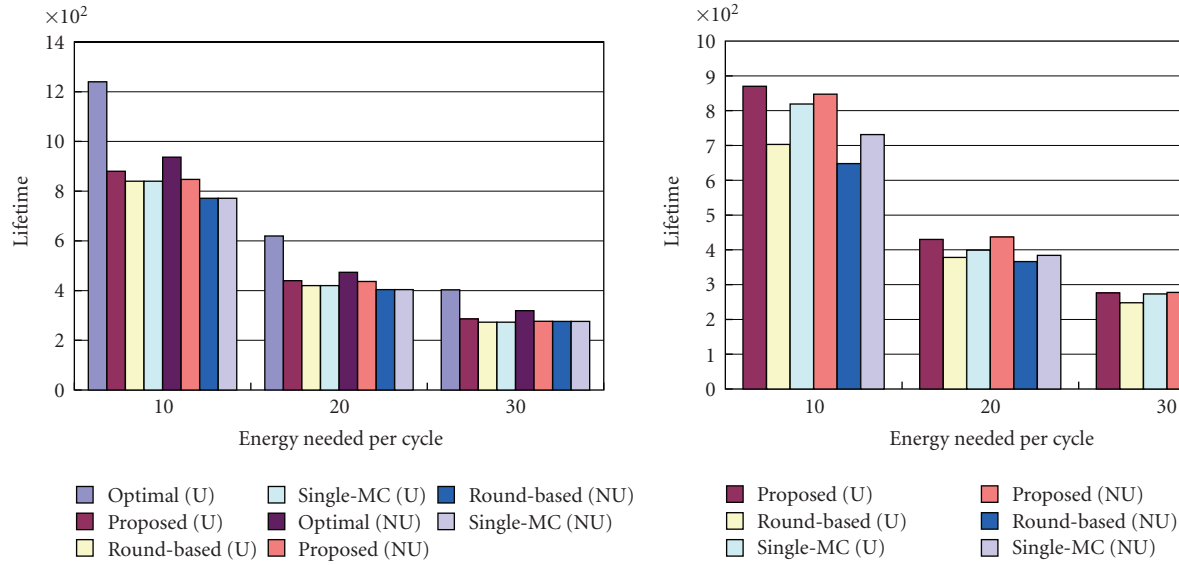
Algorithm 1: Pseudocode of $c = GFN(q)$ receives $\langle\langle 1, q_1, M_1'\rangle, \ldots, \langle L, q_L, M_L'\rangle\rangle$ sent by q.

1: *Unselected* is a set of sensors which has originally been selected into a set but later removed by c.
2: M_i is a set of sensors selected into available set i by c. Initially, the last member of M_i' is put into M_i.
3: l_i is the last member in M_i.
4: m is the index of the available set which has the smallest ending angle among all l_i and has not yet formed a cover, where $1 \le i \le L$.
5: /* c selects neighbors to extend the cover range of the sets. */
6: **for** each set i **do**
7: **while** (set i does not form a cover) **do**
8: Select sensors in *Unselected* which can cover $[t_{l_i}, s_{q_i}]$ to M_j.
9: **if** Set i still does not form a cover **then**
10: /* Remove the available set with the highest index number, that is, L */
11: Move sensors in M_L, M_L', and q_L to *Unselected*.
12: $L = L - 1$.
13: **end if**
14: **end while**
15: **end for**
16: Announce $\langle\langle i, q_i, M_i\rangle\rangle$, for $i = 1$ to L, together with the activation schedule for each cover.

Algorithm 2: Pseudocode of $c = q'$ receives message $\langle\langle 1, q_1, M_1'\rangle, \ldots, \langle L, q_L, M_L'\rangle\rangle$.

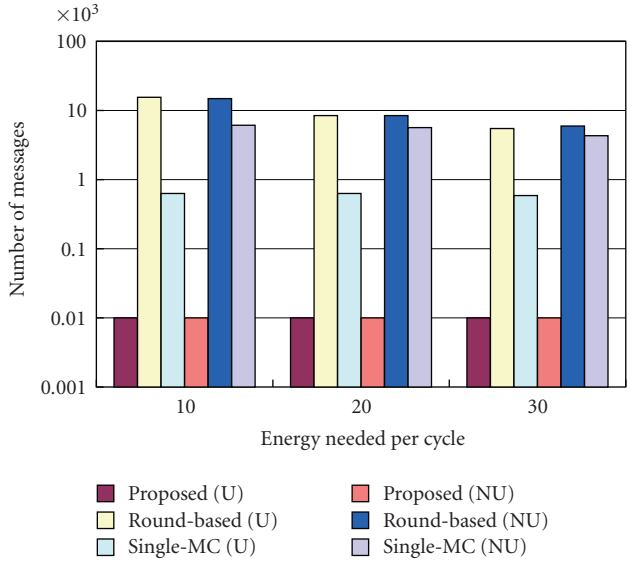
paper, which is denoted as *Proposed* algorithm in the figures. The third algorithm is the one proposed in [5], which is denoted as *Round-based* algorithm in the figures. In this algorithm, a minimum size cover, known as *MC*, is found to monitor the target object in each monitoring cycle until no more MC can be found. The fourth algorithm is denoted as *Single-MC* in the figures. In this algorithm, an MC is found and this MC is activated until a sensor is running out of energy. Afterwards, another MC is found and activated again. This process continues until no more MC can be found.

In our simulations, we considered an area of 50 grids $\times$50 grids. Each grid takes up an area of 1 unit^2 and has a certain probability of containing a sensor. A grid that is partially/completely occupied by the target does not have any sensor. This probability is known as the *sensor availability probability*, and we use p_a to denote it. If the grid contains a sensor, the sensor is located at the center of the grid. This is similar to the simulation environment considered in [5]. Each sensor has a sensing range r_s; The target object is located at the center $(25, 25)$ with the object radius of r_o. For uniform initial battery case, each sensor has B units of energy. In the nonuniform battery case, each sensor has a mean value B units of energy. The uniform case is denoted as (U), while the nonuniform case is denoted as (NU) in the figures. Each monitoring cycle is assumed to take up e_m units of energy, and each message takes up 1 unit of energy if transmission cost is counted.

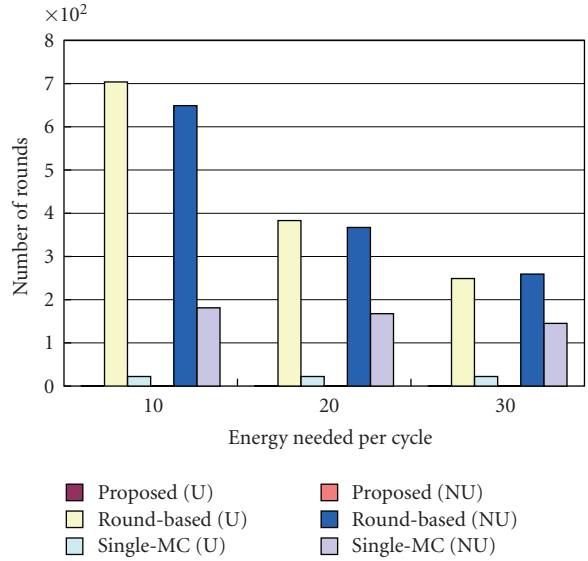


(a) Network lifetime versus energy needed per cycle without transmission cost

(b) Network lifetime versus energy needed per cycle with transmission cost



(c) Number of messages versus energy needed per cycle with transmission cost

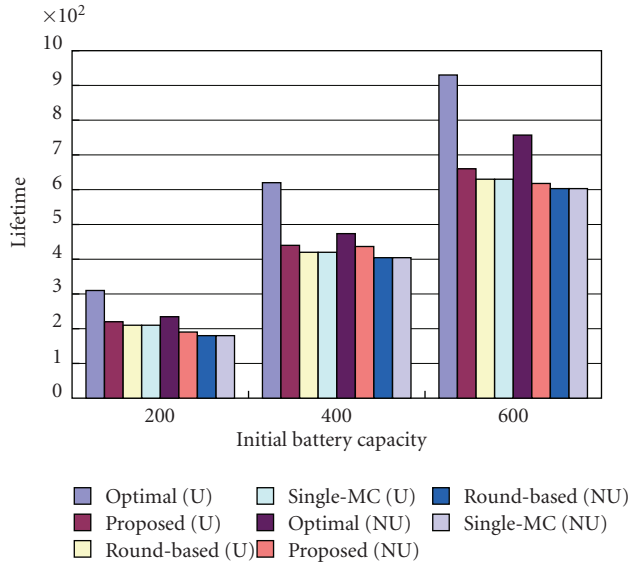


(d) Number of rounds versus energy needed per cycle with transmission cost

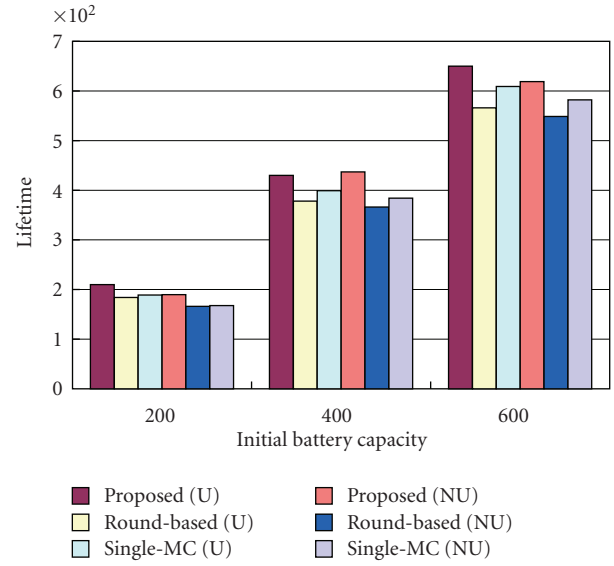
Figure 11: Varying energy needed per cycle scenario.

7.1. Effect of Energy Needed Per Cycle. First of all, we simulated the scenario in which $p_a = 1$, $r_o = 12.5$ units, $r_s = 4.5$ units, and $\overline{B(i)} = B = 400$ units. In the simulations, we vary e_m from 10 to 30 units per cycle. Figure 11(a) illustrates the network lifetime achieved by various algorithms under different energy needed per cycle without transmission cost. The figure shows that all the algorithms have shorter lifetimes when each monitoring cycle requires more energy. On the other hand, our proposed algorithm achieves similar network lifetime with that of *Single-MC* and *Round-based* algorithm as all the algorithms are 2-approximation solutions if no transmission cost is counted. All of them achieve around 70% of the maximum lifetime under the uniform battery environment. This is mainly because many sensors have never been chosen in any proper cover. In the nonuniform situation, the lifetimes of the algorithms are about 80% of the optimal one. This is mainly due to the reason that more nodes are involved.

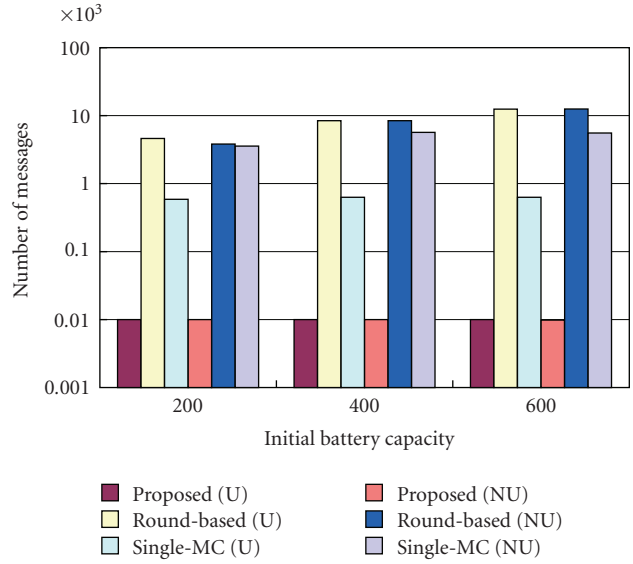
Figure 11(b) shows the performance under different energy needed per cycle with transmission cost. It can be shown that *Round-based* algorithm is affected the most by the transmission cost as an MC needs to be found every cycle, while our proposed algorithm is affected by the transmission cost the least. Therefore, our purposed algorithm outperforms the others. Figures 11(c) and 11(d) illustrate the number of messages and the number of scheduling rounds



(a) Network lifetime versus initial battery capacity without transmission cost



(b) Network lifetime versus initial battery capacity with transmission cost

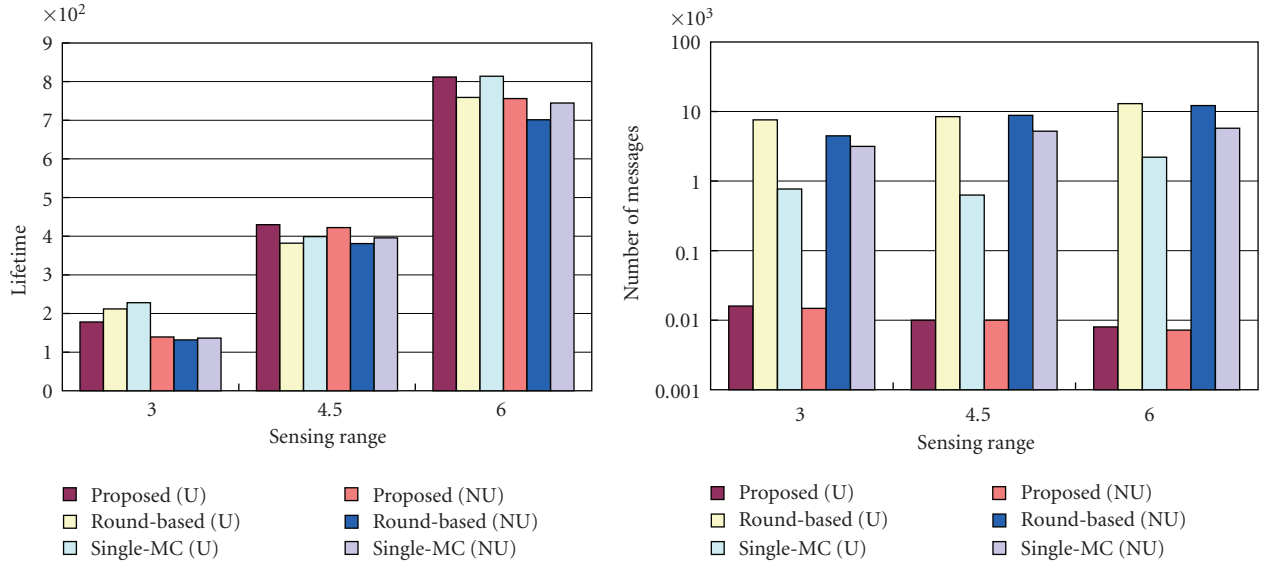


(c) Number of messages versus initial battery capacity with transmission cost

Figure 12: Varying initial battery capacity scenario.

required by various algorithms throughout the network lifetime. As expected, *Round-based* algorithm requires the most number of messages and rounds, while our algorithm requires the least. It is worthwhile to state that our proposed algorithm requires only one scheduling round with O(size of the MC) and this contributes to the good performance shown in Figures 11(c) and 11(d). On the other hand, the figures also show that our proposed algorithm and the *Round-based* algorithm require similar amount of messages and rounds no matter the initial battery capacity of a sensor is uniform or not. However, there is a large discrepancy between uniform and nonuniform battery for the *Single-MC* algorithm. It is because *Single-MC* algorithm finds an MC and then it is activated until a sensor in this MC fails. In the uniform battery scenario, nearly all the sensors in an MC runs out of energy simultaneously. However, this is not the case in nonuniform battery scenario.

7.2. Effect of Initial Battery Capacity. In Figures 12(a) and 12(b), we simulated the scenario where $r_o = 12.5$ units, $r_s = 4.5$ units, and $e_m = 20$ units per cycle. In these simulations, we vary the mean value B from 200 to 600 units. Figures 12(a) and 12(b) illustrate the network lifetime achieved by various algorithms under different initial battery capacity



(a) Network lifetime versus sensing range with transmission cost

(b) Number of messages versus sensing range with transmission cost

Figure 13: Effect of sensing range in homogenous sensing range scenario.

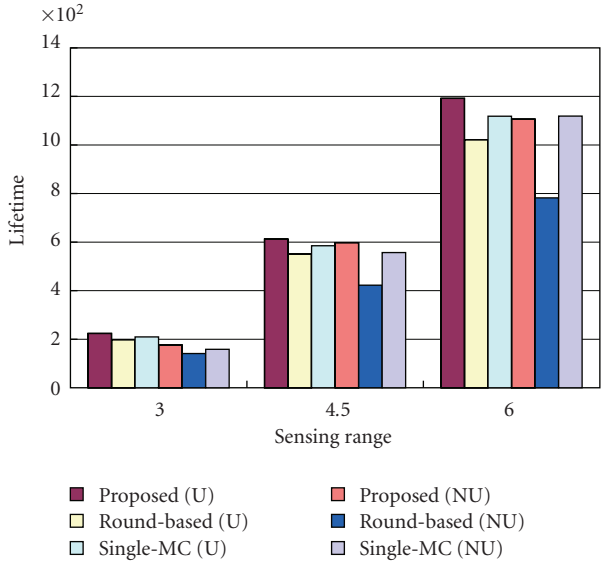
settings without transmission cost and with transmission cost, respectively. The figure shows that all the algorithms have longer lifetimes when the initial battery capacity increases. Similar to the results in Figure 11(a), all the 2-approximation solutions achieve about 70% and over 80% of the optimal solution under the uniform and nonuniform scenarios, respectively. These results show that the algorithms considered generally achieve good performance although they are 2-approximation solutions. Similar to that in Figures 11(a) and 11(b), our proposed algorithm performs similarly to the *Round-based* and *Single-MC* algorithms in case no transmission cost is counted. However, our proposed algorithm outperforms them when transmission cost is considered. This is mainly due to the reasons that more messages are required in other algorithms as shown in Figure 12(c) where the number of messages required grows with the increasing initial battery capacity.

In both the scenarios considered in Sections 7.1 and 7.2, the algorithms achieve longer lifetime when uniform initial battery is considered. It is because any portion of the perimeter is covered by sensors with approximately the same amount of total energy. Unfortunately, this is not the case in the nonuniform initial battery scenario where some portions have significantly higher total energy than others.

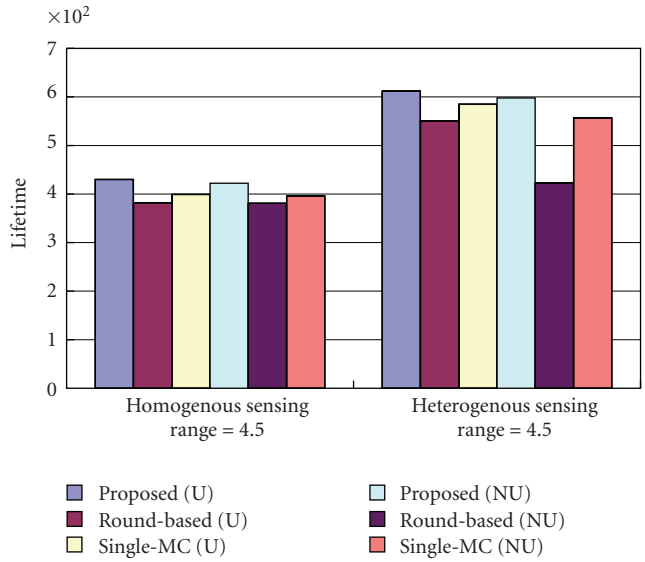
7.3. Effects of Sensing Range. In this section, we have simulated the scenario in which $p_a = 1$, $r_o = 12.5$ units, $e_m = 20$ units, and mean value $B = 400$ units. In the simulations, we vary r_s from 3 to 6 units. Here, we study two scenarios. First, each sensor has the same sensing range r_s, we denote this as the *homogenous sensing range scenario*. Second, different sensors may have different sensing ranges. In other words, each sensor has the sensing range with the mean value r_s. We denote this scenario as the *heterogenous sensing range scenario*.

Let us start with the homogenous sensing range scenario. Generally speaking, the longer the sensing range, the fewer the sensors needed to cover the perimeter of the target object. This results in longer network lifetime in various algorithms as shown in Figure 13(a). On the other hand, Figure 13(b) illustrates that our proposed algorithm requires fewer messages when the sensing range increases. This is mainly due to the reason that the larger the sensing range the smaller the size of the minimum cover is. Recall that our proposed algorithm requires O(size of the minimum cover) number of messages to find all the schedules. However, *Round-based* and *Single-MC* algorithms require more messages when the sensing range increases. On one hand, the longer the sensing range, fewer messages are needed to find an MC. On the other hand, the longer the sensing range, less sensors are required to cover a target object. This results in a larger number of scheduling rounds. Since the second factor outweighs the first factor, the number of messages increases in both the *Round-based* and *Single-MC* algorithms.

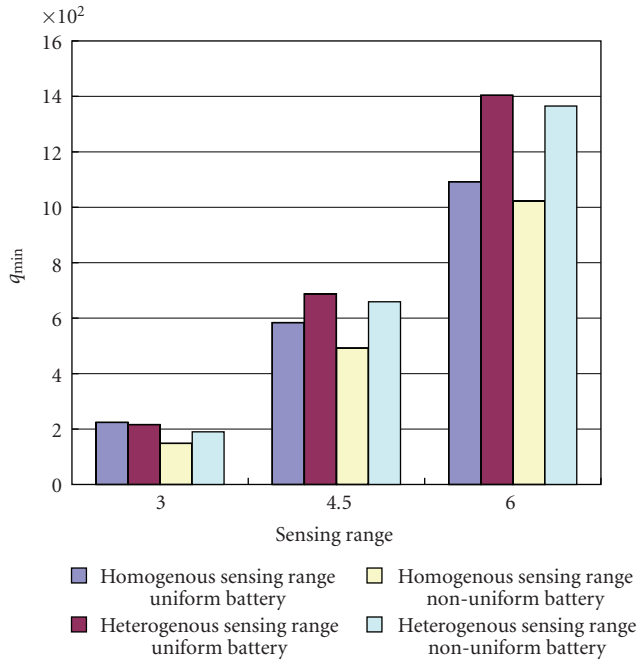
We further consider the heterogenous sensing range scenario in Figure 14(a). Figure 14(a) exhibits similar trend as that of the homogenous sensing range scenario in Figure 13(a) due to the same reason aforementioned. However, when we compare the homogenous sensing range and the heterogenous sensing range scenarios with the mean value $r_s = 4.5$ in Figure 14(b), we can find that the lifetime achievable by the heterogenous scenario is better than that of the homogenous case. In fact, this phenomenon can be explained with the help of Figure 14(c). As we know that the network lifetime achievable is upper bounded by $q_{\min}$, we compare $q_{\min}$ of the homogenous and the heterogenous sensing range scenarios in Figure 14(c). In both scenarios, we can observe that $q_{\min}$ of the uniform initial battery capacity case is higher than that of the nonuniform initial battery case. These agree with the results shown in Figures 13(a) and 14(a)



(a) Network lifetime versus sensing range with transmission cost (heterogenous sensing range scenario)



(b) Comparison between homogenous and heterogenous sensing range scenarios in terms of lifetime

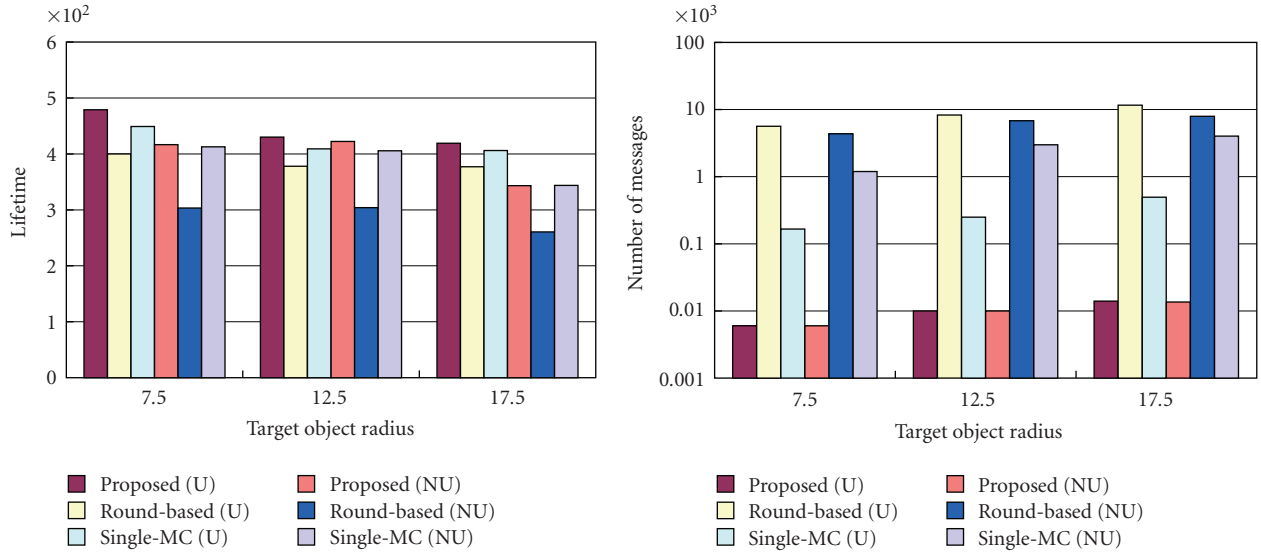


(c) Comparison between homogenous and heterogenous sensing range scenarios in terms of $q_{\min}$

Figure 14: Comparison between homogenous and heterogenous sensing range scenarios.

that various algorithms achieve better lifetime in the uniform initial battery case than that in the nonuniform initial battery case. On the other hand, we can observe that $q_{\min}$ of the homogenous sensing range scenario is generally smaller than that of the heterogenous sensing range scenario no matter the initial battery capacity of the sensor is uniform or not. This is mainly due to the reason that some sensors which are too far to cover any portion of the perimeter of the target object in the homogenous sensing range scenario may cover a certain portion of the perimeter in the heterogenous sensing range scenario. This also accounts for the better performance shown in heterogenous sensing range scenario.

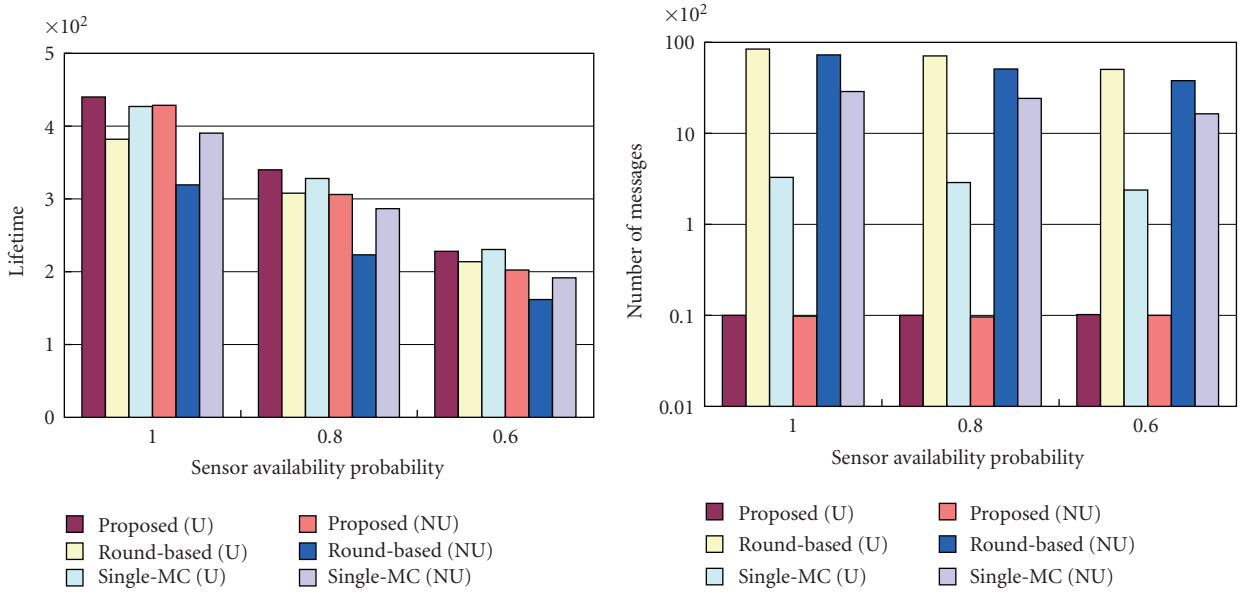
7.4. Effect of Object Size. In this section, we have simulated the scenario in which $p_a = 1$, $r_s = 4.5$, $e_m = 20$ units, and mean value $B = 400$ units. In the simulations, we vary r_o from 7.5 to 17.5 units. Generally speaking, the larger the target radius, the larger the number of sensors needed to cover the perimeter of the whole target object. This results in smaller network lifetime in various algorithms as shown in



(a) Network lifetime versus target object radius with transmission cost

(b) Number of messages versus target object radius with transmission cost

Figure 15: Varying target object radius scenario.



(a) Network lifetime versus sensor availability probability with transmission cost

(b) Number of messages versus sensor availability probability with transmission cost

Figure 16: Varying sensor availability probability scenario.

Figure 15(a). On the other hand, Figure 15(b) illustrates that various algorithms require larger amount of messages when the target radius increases. This is mainly due to the reason that the larger the target radius the larger the size of the minimum cover is. In both figures, we can observe that our proposed algorithm generally achieves better performance than the others.

7.5. Effect of Network Density. In this section, we have simulated the scenario in which $r_s = 4.5$ units, $r_o = 12.5$ units, $e_m = 20$ units, and mean value $B = 400$ units. In the simulations, we vary p_a from 1 to 0.6 units. Generally speaking, the smaller the sensor availability probability, the fewer the sensors exist in the network. This results in smaller network lifetime in various algorithms as shown in Figure 16(a). On the other hand, Figure 16(b) illustrates that our proposed algorithm requires similar amount of messages no matter what the value of p_a is. However, *Round-based* and *Single-MC* require smaller amount of messages when p_a becomes smaller. This is mainly due to the reason

that fewer sensors exist in the network. Therefore, the total number of rounds required to find the *MCs* decreases in both cases. Hence, this results in the smaller number of messages. Nevertheless, we still observe that our proposed algorithm generally achieves the best performance.

8. Conclusion and Future Work

In this paper, we study the *perimeter coverage scheduling* problem. We found that this problem is solvable in polynomial time under some special sensor configurations. However, this problem is NP-hard in general. We realize that the selection of a proper cover in each cycle leads to a 2-approximation solution. We then propose a simple distributed 2-approximation solution with O(size of the minimum cover) number of messages, and we demonstrate its effectiveness through simulations.

In the future, we plan to study the perimeter coverage scheduling issues on the scenario where a sensor can monitor multiple continuous portions of the perimeter on the target object instead of a single continuous portion. In [29], we have proposed a distributed O(maximum number of cover ranges per sensor) approximation solution on the problem of finding the minimum size and minimum cost set of sensors to cover the perimeter of the whole target object. To the best of our knowledge, no approximation solution has been developed on the scheduling problem up to now and the approximation ratio is likely to be larger than $O(2\times$ maximum number of cover ranges per sensor). Therefore, we would like to develop a distributed approximation solution to tackle this issue in the coming future.

References

[1] K.-Y. Chow, K.-S. Lui, and E. Y. Lam, "Maximizing angle coverage in visual sensor networks," in *Proceedings of IEEE International Conference on Communications (ICC '07)*, pp. 3516–3521, June 2007.

[2] K.-Y. Chow, K.-S. Lui, and E. Y. Lam, "Achieving 360Â° angle coverage with minimum transmission cost in visual sensor networks," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '07)*, pp. 4115–4119, March 2007.

[3] K.-S. Hung and K.-S. Lui, "On perimeter coverage of wireless sensor networks," Tech. Rep. TR-2008-004, Department of Electrical and Electronic Engineering, The University of Hong Kong, September 2008.

[4] K.-S. Hung and K.-S. Lui, "On perimeter coverage of wireless sensor networks with minimum cost," Tech. Rep. TR-2008-006, Department of Electrical and Electronic Engineering, The University of Hong Kong, December 2008.

[5] K.-Y. Chow, K.-S. Lui, and E. Y. Lam, "Wireless sensor networks scheduling for full angle coverage," *Multidimensional Systems and Signal Processing*, vol. 20, no. 2, pp. 101–119, 2009.

[6] Q. Dong, "Maximizing system lifetime in wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 13–19, April 2005.

[7] C. Schurgers and M. B. Srivastava, "Energy efficient routing in wireless sensor networks," in *Proceedings of IEEE Military Communications Conference (MILCOM '01)*, vol. 1, pp. 357–361, October 2001.

[8] I. Dietrich and F. Dressler, "On the lifetime of wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 5, no. 1, article 5, 2009.

[9] C.-F. Huang, L.-C. Lo, Y.-C. Tseng, and W.-T. Chen, "Decentralized energy-conserving and coverage-preserving protocols for wireless sensor networks," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '05)*, vol. 1, pp. 640–643, May 2005.

[10] D. Tian and N. D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications*, pp. 32–41, September 2002.

[11] T. Yan, T. He, and J. A. Stankovic, "Differentiated surveillance for sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 51–62, November 2003.

[12] C.-F. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: random and coordinated sleep algorithms," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN '04)*, pp. 433–442, April 2004.

[13] C.-F. Huang and Y.-C. Tseng, "The coverage problem in a wireless sensor network," in *Proceedings of the 2nd ACM International Workshop on Wireless Sensor networks and Applications (WSNA '03)*, pp. 115–121, September 2003.

[14] C.-F. Huang, L.-C. Lo, Y.-C. Tseng, and W.-T. Chen, "Decentralized energy-conserving and coverage-preserving protocols for wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 2, pp. 182–187, 2006.

[15] M. Ye, E. Chan, G. Chen, and J. Wu, "Energy efficient fractional coverage schemes for low cost wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS '06)*, July 2006.

[16] L. Wang and S. S. Kulkarni, "Sacrificing a little coverage can substantially increase network lifetime," *Ad Hoc Networks*, vol. 6, no. 8, pp. 1281–1300, 2008.

[17] S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang, "Design and analysis of sensing scheduling algorithms under partial coverage for object detection in sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 3, pp. 334–350, 2007.

[18] M. Cardei, M. T. Thai, Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM '05)*, vol. 3, pp. 1976–1984, March 2005.

[19] H. Liu, P. Wan, and X. Jia, "Maximal lifetime scheduling for sensor surveillance systems with K sensors to one target," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 12, pp. 1526–1536, 2006.

[20] Q. Zhao and M. Gurusamy, "Lifetime maximization for connected target coverage in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 6, pp. 1378–1391, 2008.

[21] M. T. Thai, Y. Li, D.-Z. Du, and F. Wang, "O (log n)-localized algorithms on the coverage problem in heterogeneous sensor networks," in *Proceedings of the 27th IEEE International Performance Computing and Communications Conference (IPCCC '07)*, pp. 85–92, April 2007.

[22] G. Calines and R. B. Ellis, "Monitoring schedules for randomly deployed sensor networks," in *Proceedings of the 5th ACM International Workshop on Foundations of Mobile Computing (DIALM-POMC '08)*, pp. 3–11, August 2008.

[23] S. Kumar, T. H. Lai, and A. Arora, "Barrier coverage with wireless sensors," *Wireless Networks*, vol. 13, no. 6, pp. 817–834, 2007.

[24] S. Kumar, T. H. Lai, M. E. Posner, and P. Sinha, "Optimal sleep-wakeup algorithms for barriers of wireless sensors," in *Proceedings of the 4th International Conference on Broadband Communications, Networks and Systems (BroadNets '07)*, pp. 327–336, September 2007.

[25] A. Chen, S. Kumar, and T. H. Lai, "Designing localized algorithms for barrier coverage," in *Proceedings of the 13th Annual International Conference on Mobile Computing and Networking (MOBICOM '07)*, pp. 63–74, September 2007.

[26] A. Chen, T. H. Lai, and D. Xuan, "Measuring and guaranteeing quality of barrier-coverage in wireless sensor networks," in *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, pp. 421–430, April 2008.

[27] B. Liu, O. Dousse, J. Wang, and A. Saipulla, "Strong barrier coverage of wireless sensor networks," in *Proceedings of the 9th International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, pp. 411–419, May 2008.

[28] A. Saipulla, C. Westphal, B. Liu, and J. Wang, "Barrier coverage of line-based deployed wireless sensor networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '09)*, pp. 127–135, April 2009.

[29] K.-S. Hung and K.-S. Lui, "Perimeter coverage made practical in wireless sensor networks," in *Proceedings of the 9th International Symposium on Communications and Information Technology (ISCIT '09)*, pp. 87–92, September 2009.

[30] H. Lee and H. Aghajan, "Vision-enabled node localization in wireless sensor networks," in *Proceedings of the Cognitive Systems with Interactive Sensors (COGIS '06)*, March 2006.

[31] C. McCormick, P.-Y. Laligand, H. Lee, and H. Aghajan, "Distributed agent control with self-localizing wireless image sensor networks," in *Proceedings of the Cognitive Systems with Interactive Sensors (COGIS '06)*, March 2006.

[32] N. Tezcan and W. Wang, "Self-orienting wireless multimedia sensor networks for maximizing multimedia coverage," in *Proceedings of IEEE International Conference on Communications (ICC '08)*, pp. 2206–2210, May 2008.

[33] M. Cardei, J. Wu, and M. Lu, "Improving network lifetime using sensors with adjustable sensing range," *International Journal of Sensor Networks*, vol. 1, pp. 41–49, 2006.

[34] J. Deng, Y. S. Han, W. B. Heinzelman, and P. K. Varshney, "Balanced-energy sleep scheduling scheme for high-density cluster-based sensor networks," *Computer Communications*, vol. 28, no. 14, pp. 1631–1642, 2005.

[35] M. A. Bonuccelli, "Dominating sets and domatic number of circular arc graphs," *Discrete Applied Mathematics*, vol. 12, no. 3, pp. 203–213, 1985.

[36] J. B. Orlin, M. A. Bonucceli, and D. P. Bovet, "An $o(n^2)$ algorithm for coloring proper circular arc graphs," *SIAM Journal on Algebraic Discrete Methods*, vol. 2, no. 2, pp. 88–93, 1981.

[37] M. R. Garey, D. R. Johnson, G. L. Miller, and C. H. Paradimitriou, "The complexity of coloring circular arcs and chords," *SIAM Journal on Algebraic and Discrete Methods*, vol. 1, no. 2, pp. 216–223, 1980.

[38] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, Cole Publishing Company, 2nd edition, 2002.