On Target Monitoring in Directional Sensor Networks by jointly considering Network Lifetime and Fault Tolerance

Suman Saha
Dept. of Comp. Scie. & Engg.
Bangladesh Univ. of Engg. & Tech. (BUET)
Dhaka, Bangladesh
sumancsecu04@gmail.com

Abdullah Al Zishan
Dept. of Computing Science
University of Alberta
Edmonton, AB, Canada
azishan@ualberta.ca

Ashikur Rahman
Dept. of Comp. Scie. & Engg.
Bangladesh Univ. of Engg. & Tech. (BUET)
Dhaka, Bangladesh
ashikur@cse.buet.ac.bd

***Abstract*—The target coverage in directional sensor networks (DSNs) faces two major challenges:–prolonging network lifetime, and providing fault tolerance. To tackle the first challenge, sensors are divided into multiple set covers each of which is capable of monitoring all the targets and schedule them in a round-robin fashion. Therefore, maximizing the number of set covers can lead to a longer network lifetime. Interestingly, the number of set covers is increased when the sensors are allowed to overlap within the set covers. Thus in one extreme, one can attain maximum lifetime by allowing the sensors to overlap boundlessly. Besides lifetime, fault tolerance is another important aspect which has been overlooked by most of the research works. Practically, sensor nodes are very much prone to failure and difficult to be replaced. If a particular sensor dies out, then all set covers containing the faulty sensor get affected. This indicates another extreme: increasing the sensor overlap within set covers decrease fault tolerance of the network. In this paper, we address these two antithetical approaches and provide a solution to resolve this problem. At first, we formulate the problem as a Linear Programming (LP) problem which gives the optimal solution. As LP formulation is computationally expensive, we develop two approximation greedy algorithms which are solvable in polynomial time. We also investigate the performance of our proposed heuristics through rigorous simulation experiments.**

***Index Terms*—Wireless sensor network, Target coverage, LP, Network lifetime, Fault tolerance**

I. Introduction

Wireless sensor networks (WSNs) have drawn remarkable interests of research community due to its extensive usage in various applications such as environmental monitoring, traffic control, object tracking, and smart spaces creation [1] [2] [3], to name a few. Coverage problem in WSNs measures how well an area of interest (AOI) or a set of targets can be monitored by a set of randomly deployed (directional) sensors. Generally, there are two types of coverage: area coverage and target coverage. In this paper, we focus only on target coverage for *directional* sensors.

For the target coverage problem, sensors monitor all the targets continuously until their battery gets depleted. Since sensor nodes are equipped with limited battery power and cannot be easily replaced or recharged, the primary goal of the target coverage problem is to prolong the network lifetime using the least resource consumption. When a set of sensors are used to cover a set of targets in sensor networks, it is expected that some sensors share their common sensing region. Therefore, in one version of target coverage problem, sensors are to be divided into multiple sets where each set covers all the targets. These sets are called cover sets and this type of problem is known as coverage scheduling which extends the network lifetime. In real scenarios, sensors may run out of power or die over time which causes the network to be detached at any time [4]. Therefore, besides coverage, fault tolerance mechanism is also needed in designing a robust and secure network.

In this paper, we construct an energy efficient fault-tolerant solution for target coverage problem in DSNs and generate maximum possible number of set covers within given bound on sensor overlaps such that each set completely covers all the targets. No sensors are allowed to participate within the set covers more than the given bound. This strategy of binding the sensor overlap within set covers improves the network lifetime better than disjoint set covers scheme as well as provides reliable fault-tolerant networks compared to an unbounded scheme.

The major contributions of this paper are summarized below:

i. We study the target coverage problem and generate maximum possible number of the set covers by activating a subset of sensors within a bounded overlap.
ii. We develop an exact solution of the problem as linear programming (LP) formulation to understand the optimal solution.
iii. As LP formulation is not feasible for large problem instances, we propose a couple of polynomial time solvable greedy heuristics which provides maximum possible number of the set covers (approximately).
iv. Finally, we perform sensitivity analysis concerning different performance criterion such as the number of generated set covers, network lifetime, and fault tolerance.

II. Related Works

Over the last few years, a number of research works have been carried out on several fundamental issues of wireless sensor networks (WSNs). Only a some of them have worked with directional sensors. The earliest research work related to coverage for *directional* sensor networks (DSNs) was introduced by Ai and Abuzeid in [5]. Here, the authors proposed a Maximum Coverage with Minimum Sensors (MCMS) problem in which randomly deployed targets should be given single coverage by minimizing the number of sensors to be activated at any instant. However, they did not consider the fault tolerance aspect of the network which is an important issue. As the sensor nodes are very much prone to failure, Fusco and Gupta [6] proposed k coverage problem where targets are covered more than once. Sakib et al. [7] extend the basic k-coverage for achieving a *balanced* k-coverage. Although both schemes provide more fault tolerance, the network life time aspect was completely ignored in their works.

For coverage scheduling, Cardei et al. [8] proposed Maximum Disjoint Set Cover (MDSC) problem by arranging the sensors into disjoint subsets. The proposed work increases network lifetime. In another research work, Munishwar and Abu-Ghazaleh [9] introduced target based heuristic for maximizing the coverage. However, both research works have been conducted for omni-directional sensors. Shahla et al. proposed Maximum Disjoint Set Covers (MDSC) problem for directional sensors [3] in which maximum number of pairwise disjoint set cover are generated using minimal number of sensors to enhance the network lifetime and provide secure connectivity by designing a fault tolerant network. They did not consider sensor overlaps. On the other hand, MDSC problem for omni-directional sensors was further extended in [10], where sensors could become member in more than one set and sensors are divided into multiple non-disjoint set covers. This type of problem is known as Maximum Set Cover (MSC) problem. Cardei et. al showed that in [10], allowing sensor overlap within the covers can increase the network lifetime compared to the pairwise disjoint schemes. Rossi et al. [11] extend the foregoing work for directional sensors. However, none of these research works jointly consider the fault tolerance issue while extending the network lifetime.

III. Preliminaries

In this section, we give an overview of the system model and describe the main motivation of the problem.

A. Directional Sensing Model and Parameters

Unlike the omnidirectional sensors which can sense 360 degrees at a time, directional sensors have a limited angle of sensing in a particular direction with fixed angle of view. Formally, a directional sensor can be represented by five parameters $(M, R_s, E, \overrightarrow{d_{ij}}, \theta)$ where M is the location of the sensor in 2-D plane; R_s is the maximum sensing range beyond which a target can not be detected with reasonable accuracy; E is the energy or battery power of a sensor; $\overrightarrow{d_{ij}}$ is the unit vector which cuts the sensing sector into half representing the orientation of sensor s_i towards pan p_j; θ is the maximum sensing angle of the sensor called angle of view (AOV). Formally, θ and R_s define the field of view (FOV) of a sensor. If FOV of a sensor is defined as $\theta = \pi/4$ then a sensor has eight discrete non-overlapping pans. Figure 1 depicts all of these parameters. We assume that every sensor is modeled as if it has finite number of non-overlapping pans and a sensor can choose only one pan at any instance. In Figure 1, $\overrightarrow{MM1}$ is the distance vector in the direction from sensor s_i located at M to a target g_t located at M1.

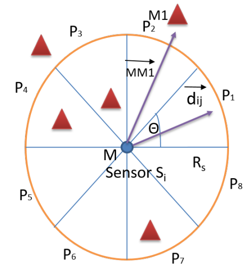


Figure 1: A directional sensor with eight discrete pans

B. Target in Sector (TIS) test

Using TIS test, one can verify whether a sensor pan pair covers a target or not [3] [5] [7]. Generally, a target is coverable by a sensor if it is located within a sensor's FOV and the distance between target and sensors is not greater than the sensor's maximum sensing range. Therefore, in TIS test, following two conditions are tested:

i. $d(M, M1) \leq R_s$ where $d(M, M1)$ is the distance between location of sensor at M and target at M1.
ii. $\delta_{it} \leq \frac{\theta}{2}$ where δ_{it} is the angle between a sensor's orientation $\overrightarrow{d_{ij}}$ of pan p_j and the distance vector $\overrightarrow{MM1}$. The angle δ_{it} is calculated by the equation as follows:

$$\delta_{it} = cos^{-1}\left(\frac{\overrightarrow{d_{ij}}.\overrightarrow{MM1}}{|\overrightarrow{d_{ij}}|.|\overrightarrow{MM1}|}\right)$$

If both conditions are met, then the result of TIS test is true; sensor s_i covers target g_t in pan p_j. Running this test over every pan p_j of sensor s_i and for all targets, we can easily construct a binary coverage matrix $A^m_{n\times q}$ of the network comprising of m targets and n sensors with q pans such that an element of this matrix can be computed as:

$$\alpha^t_{ij} = \begin{cases} 1, & \text{if sensor, pan pair}(s_i, p_j) \text{ covers target } g_t \\ 0, & \text{otherwise} \end{cases}$$

C. Related Definitions

Definition 1 : The set of targets covered by j^{th} orientation of i^{th} sensor is denoted by Φ_{ij}.

$$\Phi_{ij} = \{g_t \mid \alpha^t_{ij} = 1 \text{ i.e., } (s_i\ ,\ p_j) \text{ covers } g_t\}$$

Definition 2 : A set of sensor, pan pairs covering a target g_t is denoted by $\Phi^{-1}(g_t)$.

$$\Phi^{-1}(g_t) = \{(s_i, p_j) | \alpha^t_{ij} = 1\}$$

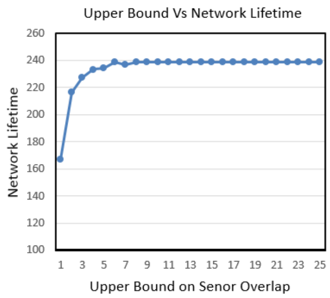


Figure 2: Effect of overlap bound over lifetime

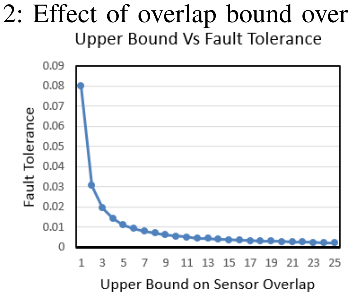


Figure 3: Effect of overlap bound over fault tolerance

Definition 3 (Critical targets and sensors) : A target covered by minimum number of sensors is called critical targets and sensors covering the critical target are called critical sensors.

Definition 4 (Cardinality of target) : The number of sensor orientation pairs covering a target g_t is called cardinality of the target g_t and is denoted by D(g_t).

$$D(g_t) = \|\Phi^{-1}(g_t)\|$$

D. *Motivation*

In literature, there are two threads of work that deal with the coverage scheduling in WSNs. In one thread, pairwise disjoint sets are generated [3]. Here, each set of sensors cover all the targets and no sensor participates more than one set. In another thread, sensors are allowed to participate in any number of cover sets. Here, the main goal is to maximize the lifetime of the network [10]. However, from the Figure 2, it is evident that allowing the sensors to overlap unboundedly will give us a maximum lifetime. On the other hand, Figure 3 indicates that fault tolerance decreases as upper bound value is increased. As a result, the network becomes more faulty as the bound increases. Note that, fault tolerance is calculated by the inverse of summing the membership count of the sensors in different set covers.

In summary, disjoint sets provide maximum fault tolerance but underperform in maximizing network lifetime. On the other hand, by allowing sensor overlap, we can attain a maximum lifetime by suppressing fault tolerance. Our work, in one sense, fit in the middle of these two antithetical approaches. We select the maximum possible number of set covers with a given upper bound on sensor overlap; no sensor can participate more than this given upper bound value. In [12], Kamrul and Selim address a similar problem where their goal is to maximize the number of covers by minimizing the maximum membership count of the sensors. But they do not provide any mathematical formulation and sensors are omnidirectional whereas this research work deals with the directional sensors.

IV. Optimum Solution of the Problem

In this section, we provide a linear programming (LP) formulation of the problem that maximizes the number of set covers by activating a subset of sensors keeping membership count of the sensors is bounded.

A. *LP Formulation*

The set of parameters used in this formulation can be described as follows. n: the number of smart directional sensors; m: number of stationary targets; q: number of non overlapping pans available for each directional sensor; and z: upper bound on sensor overlap. Given that, a collection of all possible set covers C = $\{C_1, C_2, ..., C_r\}$ by n sensors each of which covers all the targets. x_{ij}^l is a selection variable that indicates whether sensor, pan pair $(s_i, p_j) \in C_l$ or not.

$$x_{ij}^l = \begin{cases} 1, & \text{if sensor, pan pair } (s_i, p_j) \in C_l \\ 0, & \text{otherwise} \end{cases}$$

Decision variables used in the formulation can be defined as follows.

$$y_t^l = \begin{cases} 1, & \text{if target } g_t \text{ is covered in } l^{th} \text{ set cover} \\ 0, & \text{otherwise} \end{cases}$$

$$\gamma_l = \begin{cases} 1, & \text{if } l^{th} \text{ set cover is selected} \\ 0, & \text{otherwise} \end{cases}$$

Then goal of the LP formulation is to:

maximize:

$$\sum_{l=1}^{r} \gamma_l \tag{1}$$

subject to constraint:

$$\sum_{l=1}^{r} \sum_{j=1}^{q} x_{ij}^l \leq z; \quad \forall i \tag{2}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{q} [g_t \in \tau_{ij}] x_{ij}^l \geq 1; \quad \forall l, \forall t \tag{3}$$

$$\sum_{t=1}^{m} y_t^l \geq m; \quad \forall l \tag{4}$$

$$\sum_{j=1}^{q} x_{ij}^l \leq 1; \quad \forall l, \forall i \tag{5}$$

where:

$$x_{ij}^l = 0 \, or \, 1; \quad \forall l, \forall i, \forall j \tag{6}$$

$$y_t^l = 0 \, or \, 1; \quad \forall l, \forall t \tag{7}$$

The objective function of the LP formulation in (1) is to maximize the number of set covers. If $\gamma_l = 0$ then it means, the set cover, C_l is not selected. Constraint (2) assures that maximum membership count of a sensor cannot exceed the given upper bound on sensor overlap. Constraint (3) indicates that each target is covered by at least one direction of a sensor in a set cover. Constraint (4) provides the guarantee that each set completely covers all the targets. Constraint (5) imposes the condition that only one pan of a sensor can be activated at a time in a set cover. Constraint (6) and Constraint (7) state the binary integer requirements on the decision variables.

V. Centralized Greedy Heuristic for the problem

Although the LP formulation described in section IV gives the optimal number of set covers, for large problem instances it does not scale well since the amount of time required for solving the mentioned problem of target coverage using LP formulation is computationally high. Therefore, we propose two greedy heuristics which are solvable in polynomial time that provide sub-optimal solutions. They are :– (i) Sensor Oriented Greedy Heuristic (SOGH), and (ii) Target Oriented Greedy Heuristic (TOGH)

A. *Sensor Oriented Greedy Heuristic (SOGH)*

The basic idea behind the sensor oriented greedy heuristic (SOGH) is to greedily choose a sensor pan pair which provides maximum coverage (of uncovered targets) among least used sensors in previously formed set covers. The pseudocode of this heuristic has been given in Algorithm 1. This algorithm iteratively constructs the set covers from line 4 $\sim$ 33. At each step, at first it selects the sensors which cover at least one uncovered target(s) and create a set $S^{'}$ (line 11 $\sim$ 18). Next, sensors having minimum count value are selected from $S^{'}$ and make another set $S^{''}$ (line 22 $\sim$ 28). After that, it selects the sensor pan pair from $S^{''}$ which maximizes coverage of the uncovered targets (line # 29). Ties are broken arbitrarily. The newly selected sensor pan pair is included in the current set cover C_k (line # 30). Next, all the covered targets by newly selected sensor pan pair are removed from TARGETS (line # 31) and the newly selected sensor has been removed from SENSORS (line # 32). Steps 9 $\sim$ 33 continue until TARGETS set is empty. When all the targets are covered, a new set cover C_k has been formed. Once a set cover C_k has been obtained, the newly created set cover is optimized (line # 34) to remove any redundant sensors from the set cover which did not contribute during coverage.

The algorithm used for optimization at this step is provided in Algorithm 2 in details. Next, priority of each sensor in optimized set cover has been updated (# 36). When priority of a sensor reaches to the given bound z, it has been removed from S and will not be available for further set cover formation (line 37 $\sim$ 39). This algorithm terminates when no sensors are available to cover at least one uncovered target in TARGETS. Finally, algorithm returns the collection of set covers $C = \{C_1, C_2, ..., C_k\}$.

Algorithm 1 Sensor Oriented Greedy Heuristic

Input: S = set of sensors; G = set of targets; P = set of discrete pans; z = upper bound value on sensor overlap
Output: Collection of set covers C= $\{C_1, C_2, ..., C_k\}$

```
1: function MaximizeTheNumberOfSetCovers(S,G,z)
2:     set count_i ← 0; 1 ≤ i ≤ n
3:     k = 0
4:     while (S ≠ φ) do
5:         k = k + 1
6:         C_k = φ
7:         TARGETS = G
8:         SENSORS = S
9:         while (TARGETS ≠ φ) do
10:            S' = φ
11:            for each sensor s_i ∈ SENSORS do
12:                for each pan p_j ∈ P do
13:                    if (| Φ_ij ∩ TARGETS | ≥ 1) then
14:                        S' = S' ∪ {s_i}
15:                        break
16:                    end if
17:                end for
18:            end for
19:            if (S' = φ) then
20:                goto marker
21:            end if
22:            mincount = min_{s_i ∈ S', 1≤i≤n} count_i
23:            S'' = φ
24:            for each s_j ∈ S' do
25:                if (count_j == mincount) then
26:                    S'' = S'' ∪ { s_i}
27:                end if
28:            end for
29:            (s_i, p_j)←argmax_{s_i ∈ S'', 1≤j≤q} Φ_ij∩TARGETS
30:            C_k = C_k ∪ {(s_i, p_j)}
31:            TARGETS = TARGETS \ {Φ_ij}
32:            SENSORS = SENSORS \ {s_i}
33:        end while
34:        C_k = SetCoverOptimization (C_k, G)
35:        for each sensor s_i ∈ C_k do
36:            count_i=count_i + 1
37:            if (count_i == z) then
38:                S= S \ { s_i }
39:            end if
40:        end for
41:    end while
42:    marker: return the collection of set covers C={ C_1, C_2, ......, C_k}
43: end function
```

Let us analyze the time complexity of SOGH. The algorithm has two while loops. For each iteration of the inner while loop (line # 9), algorithm needs to perform two major tasks. First task (line 11 $\sim$ 18) takes $(m+1)nq$ operations. For the second task, (line # 29), again algorithm executes $(m+1)nq$ operations. Since, inner while loop runs over all the targets which requires at most m iterations. So the cost of the inner while loop is $(2(m+1)mnq)$. Besides, set cover optimization takes $(m + 1)mnq$ operations in its calculation which is discussed later. Thus the total cost of the inner while loop becomes $3(m + 1)mnq$. The outer while loop (line # 4) runs over all sensors at most nz times. Therefore, the worst case time complexity of SOGH is $O((m+1)mn^2qz)$ which eventually becomes $O(m^2n^2qz)$.

Set Cover Optimization: Once the coverage requirement is fulfilled, set covers are needed to be optimized to eliminate the redundant sensors within set cover. A problem may arises during set cover formation in both algorithms 1 and 3 is that it does not require activating all the sensors for coverage. Therefore, a sensor may not contribute to the coverage after set cover formation (rest of the sensors may already cover all the targets) could be dropped. Pseudocode for set cover optimization is shown in Algorithm 2. By optimizing the newly obtained set cover, the proposed algorithm minimizes the usages of sensors where possible. Initially, it creates an empty set cover C_k^* where all the sensors in C_k are assigned and the set cover C_k is made empty. For each iteration, it selects a sensor pan pair from C_k^* that covers maximum number of uncovered targets. The selected sensor pan pair is included in the set cover C_k, and targets covered by the newly selected sensor pan pair are removed from G. When G is empty (i.e., no more uncovered targets), the newly optimized set cover C_k has been formed and all the redundant sensors within set cover are eliminated. The run time of this algorithm is $O(m^2nq)$. Because the while loop (line # 3) runs for m iterations in worst case . The major step in while loop is (line # 4) which requires mnq operations. Thus worst case time complexity becomes $O(m^2nq)$.

Algorithm 2 Set Cover Optimization

Input: C_k, a set cover consists of sensor pan pairs covering all the targets; G=set of targets; P=set of discrete pans
Output: optimized set covers, C_k

1: **function** SETCOVEROPTIMIZATION(C_k ,G)
2: C_k^* = C_k, C_k = ϕ
3: **while** (G $\neq \phi$) **do**
4: $(s_i, p_j) \leftarrow$ arg $max_{(s_i,p_j)\in C_k} \Phi_{ij} \cap G$
5: $C_k = C_k \cup (s_i, p_j), G = G - \Phi_{ij}$
6: $C_k^* = C_k^* \setminus \{(s_i, p_j)\}$
7: **end while**
8: return C_k
9: **end function**

B. *Target Oriented Greedy Heuristic (TOGH)*

Target Oriented Heuristic (TOGH) differs from SOGH in a sense that, it considers targets first while selecting a sensor. In other words, it determines critical target(s) first (i.e. the target(s) with least coverage) and selects sensor(s) that cover critical target(s). Pseudocode of this heuristic is shown in Algorithm 3. At each iteration, at first it finds the critical targets and makes a set $G^{'}$ (line # 15). Then it selects the sensor that covers at least one critical target and makes a set $S^{'}$ (line 16 $\sim$ 20). Since more than one sensor may cover critical targets, so sensors having minimum count value have been selected from $S^{'}$ to create another set $S^{''}$ (line 23 $\sim$ 27). If two sensors have the same priority, then the sensor pan pair with highest contribution (i.e., covering maximum uncovered targets) from $S^{''}$ (line # 29) is selected (line # 28). Ties are broken by randomly choose one of them. Once the sensor pan pair is selected, it has been added to the current set cover C_k (line # 29). All the additional targets covered by the sensor pan pair are removed from the TARGETS (line # 30) and newly select sensor is excluded from SENSORS (line # 31). The above procedure (line 9 $\sim$ 33) is continued until all of the targets are covered. When all the targets are covered, a new set cover C_k has been formed. When a set cover is created, the newly created set cover is optimized using Algorithm 3 like SOGH. Rest of procedures are same as Sensor Oriented Greedy Heuristic (SOGH).

Let's investigate the time complexity of TOGH. Like SOGH, TOGH also has two while loops. Inner while (line # 9) loop consists of three major tasks: firstly it takes mnq operations for finding the targets with minimum cardinality (line # 15). The second task is to find the sensor pan pairs covering the critical targets which also takes mnq operations (line 16 $\sim$ 20). And for the third task, it takes $mnq + nq$ operations to find out a sensor pan pair which covers maximum uncovered targets (line # 28). Since inner while loop (line # 9) is bounded by the number of targets (m) therefore the cost of inner while loop is $(m(3m + 1)nq)$ which is eventually O(m^2nq). Since set cover optimization requires O(m^2nq) operations (line # 34) and outer while loop (line # 4) is bounded by n number of sensors and runs at most nz times. Thus the overall time complexity of TOGH in the worst case is $O(m^2n^2qz)$.

Algorithm 3 Target Oriented Greedy Heuristic

Input: S= set of sensors; G=set of targets; P=set of discrete pans; z=upper bound value of sensor overlap
Output: Collection of set covers C= $\{C_1, C_2, ..., C_k\}$

1: **function** MAXIMIZETHENUMBEROFSETCOVERS(S,G,z)
2: set $count_i \leftarrow 0; 1 \leq i \leq n$
3: $k = 0$
4: **while** (S $\neq \phi$) **do**
5: $k = k + 1$
6: $C_k = \phi$
7: TARGETS = G
8: SENSORS = S
9: **while** (TARGETS $\neq \phi$) **do**
10: find D_{min} = min D(g_t): $g_t \in$ G
11: **if** ($|D_{min}|$=0) **then**
12: goto marker
13: **else**
14: $S^{'}$ = ϕ
15: find targets having cardinality equals D_{min} and create a set $G^{'}$
16: **for** each $g_t \in G^{'}$ **do**
17: **for** each $(s_i, p_j) \in \Phi^{-1}(g_t)$ **do**
18: $S^{'}$ =$S^{'} \cup \{(s_i, p_j)\}$
19: **end for**
20: **end for**
21: mincount = $min_{s_i \in S^{'}, 1 \leq i \leq n}$ $count_i$
22: $S^{''}$ = ϕ
23: **for** each $s_j \in S^{'}$ **do**
24: **if** ($count_j$ == $mincount$) **then**
25: $S^{''}$ = $S^{''} \cup$ { s_i}
26: **end if**
27: **end for**
28: (s_i, p_j)←arg$max_{s_i \in S^{''}, 1 \leq j \leq q} \Phi_{ij}$∩TARGETS
29: C_k = $C_k \cup \{(s_i, p_j)\}$
30: TARGETS =TARGETS $\setminus \{\Phi_{ij}\}$
31: SENSORS = SENSORS $\setminus \{s_i\}$
32: **end if**
33: **end while**
34: C_k = SetCoverOptimization (C_k, G)
35: **for** each sensor $s_i \in C_k$ **do**
36: $count_i$=$count_i$ + 1
37: **if** ($count_i$==z) **then**
38: S= S $\setminus$ { s_i }
39: **end if**
40: **end for**
41: **end while**
42: **marker**: return the collection of set covers C = { $C_1, C_2, ..., C_k$ }
43: **end function**

VI. COVERAGE SCHEDULING

Both of the heuristics SOGH and TOGH generate multiple set covers $C_1, C_2, \ldots, C_k$ each of which is capable of monitoring all the targets. As the sensor energy is consumed only when the sensor is turned on (we ignore sleep energy), a scheduling scheme can be applied where set covers are switched from one cover to another in a round-robin fashion such that only a subset of sensors consume energy at a time. Needless to say that this scheme extends the network lifetime. Algorithm 4 is used to compute the activation time (i.e. operational time) of each set cover. The algorithm works as follows. Let us assume a homogeneous system where each sensor has same maximum usable energy that can run each sensor for a maximum of E time unit. Suppose, l_i indicates the remaining energy of sensor s_i and t_i is the activation time of a set cover C_i. $flag[i]$ is a Boolean variable indicating whether a set cover C_i can be considered for assigning further activation time or not. For example, if $flag[i] = 1$ then the set cover C_i will not be considered for assigning further activation time. At each iteration the algorithm finds a set cover C_i with $flag[i] = 0$ (line # 5). Next, minimum remaining lifetime of all the sensors in a set cover C_i is determined (line 8 $\sim$ 12). Let it be $minlife$. If $minlife$ is 0 then the set cover C_i can not be used further; consequently $flag[i]$ is set to 1 and the algorithm resumes from step 5 to check another set cover (line 13 $\sim$ 15). Otherwise, it calculates the minimum of $minlife$ and E/z (line 17 $\sim$ 21) and sets this minimum value to $duration$. Note that the maximum assigned time duration for a set cover in a single iteration is E/z. After that, activation time of the set cover C_i (line # 22) and the remaining energy (in time unit) of each sensor in the set cover C_i are updated (line 23 $\sim$ 25). Iteration stops when $flag[i]$ is 1 for all set covers in which case no more set covers could be chosen for activation. The total lifetime of the network is calculated by summing up the activation time of all set covers (line # 28).

Algorithm 4 Lifetime Calculation

Input: a set of k set covers, C= $C_1, C_2, ..., C_k$; z = upper bound value on sensor overlap; E = initial energy of a sensor
Output: activation time of k set covers t = { $t_1, t_2, ..., t_k$ }

1: **function** SETCOVERACTIVATIONTIME(C, z, E)
2: set $l_i \leftarrow$ E ; $1 \leq i \leq n$
3: set $t_i \leftarrow 0$; $1 \leq i \leq k$
4: $flag[i] = 0$; $1 \leq i \leq k$
5: find a set cover C_i such that $flag[i] = 0$; $1 \leq i \leq k$
6: if no such set cover C_i is found then go to step 28
7: $minlife = E, duration = 0$
8: **for** each sensor $s_j \in C_i$ **do**
9: **if** ($l_j < minlife$) **then**
10: $minlife = l_j$
11: **end if**
12: **end for**
13: **if** ($minlife = 0$) **then**
14: $flag[i] = 1$
15: go to step 5
16: **else**
17: **if** ($minlife \leq (E/z)$) **then**
18: $duration = minlife$
19: **else**
20: $duration = (E/z)$
21: **end if**
22: $t_i = t_i + duration$
23: **for** each sensor $s_j \in C_i$ **do**
24: $l_j = l_j - duration$
25: **end for**
26: **end if**
27: go to step 5
28: $LT_z = \sum_{i=1}^{k} t_i$
29: return activation time of k set covers t = $\{t_1, t_2, ..., t_k\}$
30: **end function**

VII. FAULT TOLERANCE CALCULATION

In non-disjoint set cover scheme, sensors are allowed to participate in more than one set covers. The more a sensor participate in set covers the network becomes more error-prone. Algorithm 5 shows how to calculate the network fault tolerance. The algorithm calculates the membership count of each sensors within set covers (line 5 $\sim$ 9). At the end, the network fault tolerance is calculated by summing the inverse of membership count of each sensors (line # 10).

Algorithm 5 Calculation of Network Fault Tolerance

Input: a set of k set covers, C= $\{C_1, C_2, ..., C_k\}$; and S = set of sensors
Output: fault tolerance, F

1: **function** FAULTTOLERANCECALCULATION(C, S)
2: **for** each sensor $s_i \in$ S **do**
3: $count_i$= 0
4: **end for**
5: **for** each setcover $C_i \in$ C **do**
6: **for** each sensor $s_j \in C_i$ **do**
7: $count_j$=$count_j$+1
8: **end for**
9: **end for**
10: F = 1/($\sum_{i=1}^{n} count_i$)
11: return F
12: **end function**

VIII. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we discuss the experimental setup and compare the performance of the solutions provided by LP and other proposed schemes through extensive simulations.

A. Simulation Setup

Consider a stationary network in a 2-D plane. Assume that there are m number of targets randomly deployed in the network over an area of 100 $\times$ 100 units followed by n number of sensors to cover the targets. In our simulations, tunable parameters are as follows. n: number of sensors which is varied from 10 to 80 to understand the impact of sensors' density over network performance; m: number of coverable targets which is fixed to 10; z: upper bound value on sensor overlap which is varied from 1 to 15 to observe the impact of sensor overlap in network performance; R_s: maximum sensing range of a sensor, R_s is also varied from $15m$ to $25m$ with $q = 8$. All the algorithms were programmed in Java programming language using JDK 1.7 in Eclipse IDE and LP formulation was solved using cplex optimizer library [13].

Scenario Creation: We simulated a fixed stationary network consisting of randomly deployed sensors and targets in a sensing field of $100m \times 100m$. Initial battery power, E of each sensor is set to 100 time unit. n is varied ranging from 10 to 80 with an increment of 10 from one scenario to another with $q = 8$ while keeping the number of targets, m fixed at 10. Here, new sensors were combined with the sensors from the previous scenario in a way such that the former scenario is a subset of the new one. This assures a consistent evaluation of the result of the extended group of sensors by preserving all the features of the former scenario and simply making it better in terms of coverage. R_s is also varied from 15 to 25 with an increment of 2 at each step. We also vary $z = 1$

to 15 with an increment of 1. For a particular value of n, m, R_s, and z, a simulation network has been generated. LP formulation and the other proposed algorithms are simulated over this network. Results are reported as an average of 5 instances for each scenario.

B. Performance Metrics

Three metrics were used to analyze the performance of the proposed heuristics. They are:– i) number of set covers, ii) network lifetime, and iii) fault tolerance. These performance metrics are defined below.

1) Number of Set Covers: The number of set covers is an indicator of how many sensor sets capable of monitoring all the targets are discovered by a heuristic. It is always advantageous to create more and more set covers because then the heuristic gets more options to choose from. The bounding parameter z plays a major role here. When $z = 1$, only disjoint set covers are found and this is the minimum possible number of set covers. The number of set covers (i.e., non-disjoint) increases when z is increased. Thus allowing the sensors to overlap unboundedly one can avail maximum possible number of set covers. Consequently, the number of set covers, $k \propto z$.

2) Network Lifetime: The network lifetime is a metric that indicates how long the network remains operative. Longer network lifetime is always expected.As the proposed heuristics partition the sensors into different set covers to schedule them in a round robin fashion for monitoring the targets, consequently the lifetime of the network depends on the lifetime of the individual set covers. Assume that all sensors initially have similar energy level. Additionally, assume that available power holds a sensor to run for t time unit. If all the sensors are activated at the same time to cover all the targets, then the lifetime of the network will become $L = t$. Since, sensors are partitioned into different set covers which are scheduled in different time slots, partitioning extends the network lifetime. Therefore, maximizing the number of set covers also maximizes the network lifetime. Since number of set covers, $k \propto z$, so the lifetime, $L \propto z$. If there are k disjoint set covers and each set is being active for t time unit, then the network lifetime becomes, $L = kt$. On the other hand, if there are k overlapping set covers and activation time for a set cover C_i is t_i then network lifetime becomes, $L = \sum_{i=1}^{k} t_i$.

3) Fault Tolerance: In reality, sensors within a set cover may become faulty or damaged over the time. For disjoint set covers, when a sensor malfunctions only the cover set containing the faulty sensor will be no longer available for providing coverage. Sensors in all other set covers still might continue to monitor the targets. Therefore, increasing number of disjoint set covers will provide maximum fault tolerance in such case. The scenario is different when we allow overlaps within set covers:–the fault tolerance of the network gets suppressed. Since when a sensor becomes faulty or it dies out, all the set covers containing the faulty sensor get affected which notably decrease the network performance. Therefore, the network becomes more faulty as we increase the overlapping upper bound z. Hence, fault tolerance, $F \propto \frac{1}{z}$. Assume that, n sensors are deployed to cover m targets and k non-disjoint set covers have been generated using the overlap upper bound z. And if membership count of a sensor s_i is m_i then the fault tolerance of the network can be formally calculated as:

$$F = \frac{1}{\sum_{i=1}^{n} m_i}$$

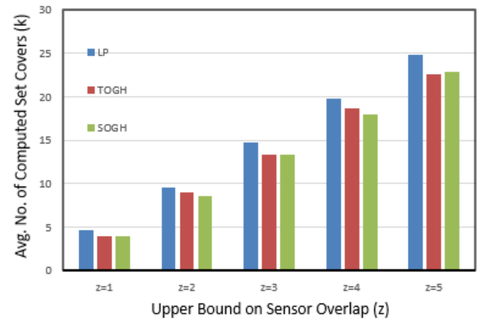


Figure 4: Average number of computed set covers by varying bound value on sensor overlap

C. Simulation Results

1) Number of Set covers Evaluation: In this experiment, we compare the average number computed set covers (k) generated by TOGH and SOGH with the optimal number of set covers given by LP. To do this comparison, we generate maximum possible set covers using LP and other two heuristics TOGH, and SOGH by keeping m = 10; n = 25; $q = 8$; and R_s = 25 unit. We vary the upper bound value, z from 1 to 5 with an increment of 1. Figure 4 demonstrates this comparison results. From Figure 4, it is evident that the number of set covers is increased as overlap bound (z) is increased. However, LP formulation maximizes the number of set covers and outperforms other two heuristics TOGH and SOGH. On the other hand, solutions provided by TOGH and SOGH are very much close to LP. Among TOGH and SOGH, TOGH provides a better result than SOGH in most cases and SOGH gives the worst performance. Although, LP formulation provides the best result but the computational time of LP is much higher compared to TOGH and SOGH. Whereas the proposed greedy heuristics TOGH and SOGH are solvable in polynomial time and much more scalable than LP.

2) Network Lifetime Evaluation: In Figure 5, we present the network lifetime (L) which is proportionate to the number of set covers (k). Figure 5a shows the network lifetime in terms of varying the number of sensors n between 10 to 80 with an increment of 10 while keeping $m = 10$; R_s = 25 unit; $q = 8$; and $z = 3$. Initially battery power of each sensor set to 100 unit. Clearly, lifetime of the network (L) increases with the number of sensors (n). This is because when more sensors participate to cover the targets, probability of generating more set covers also increases which enhance the network lifetime. On the other hand, network lifetime w.r.t varying sensors sensing range are shown in figure 5b. Here, we vary sensing range R_s from 15 to 25 with an increment of 2 while keeping n = 25; m = 10; q = 8; and z = 3. Like Figure 5a, network lifetime is also increased when we increase the sensors' sensing range as the coverage region of a sensor is enhanced by increasing sensors sensing range. In figure 5c, we plot the network lifetime w.r.t overlapping bound z of the sensors within set covers. Figure 5c shows the simulation result for 25 sensors covering 10 targets while keeping R_s = 25 and q = 8. z is varied from 1 to 10 with an increment of 1. From the Figure 5c, it is evident that disjoint set covers (i.e., z = 1) provide the least network lifetime and lifetime of the network increases as the bound value z is increased. Therefore, increasing sensor overlap within set covers enhances the network lifetime. From Figure 5a , 5b, and 5c, it is observed that performance of both heuristics TOGH and SOGH are very close to each other. However, TOGH has better network lifetime than SOGH in most of the cases as TOGH generates more number of set covers than SOGH.

3) Network Fault Tolerance Evaluation: In Figure 6, we plot the network fault tolerance (F) w.r.t. number of sensors, sensing range and sensors overlap bound (z). From the figures it is clearly evident that fault tolerance is inversely proportional to the number of overlapping set covers. For instance, in Figure 6a, we vary the number of sensors, n from 10 to 80 with an increment of 10 while in Figure 6b, sensing range is varied ranges from 15 to 25 with an increment of 2. In both cases we set m = 10; q = 8, and z = 3. In both figures, network fault tolerance decreases when either the number of sensors or sensors' sensing range is increased. This is because, if we increase number of sensors or sensors' sensing range, the number of participating sensors within the set covers also gets increased. With more sensor overlaps the fault tolerance gets lowered. Figure 6c shows the network fault tolerance (F) when the sensors overlap bound (z) is varied. Here, z is varied from 1 to 15 with the increment of 1 while keeping n = 25; m = 10 ; q = 8; and z = 3. Clearly, fault tolerance decreases when (z) is increased. Disjoint scheme provides maximum fault tolerance, and allowing more sensor overlap within the set covers, the network becomes more error-prone. Another thing, from the Figure 6a, 6b, and 6c, it is evident that fault tolerance of SOGH is higher than TOGH in most scenarios. As TOGH generates more number of (overlapping) set covers than SOGH, naturally the fault tolerance value gets lowered.

4) Determination of Optimal Bound Value: To determine the optimal upper bound (z^*) on sensor overlap, we have introduced a metric called Goodness Index, GI, which denotes how "good" a network is when both lifetime and fault tolerance is considered together. The particular value of z for which GI is maximum is defined as optimal upper bound (z^*). Goodness Index (GI) is defined as follows:

$$GI = \alpha L_z + (1-\alpha) F_z$$

Here, F_z denotes normalized value of the fault tolerance at upper bound z. A lower value of F_z index indicates the network is more error prone. L_z denotes normalized value of lifetime index at bound value z. α is a tunable parameter between 0 to 1. When α = 1, only lifetime is considered; when α = 0, only fault tolerance is considered. But for α = 0.5, both lifetime and fault tolerance is considered in equal proportion to evaluate GI.

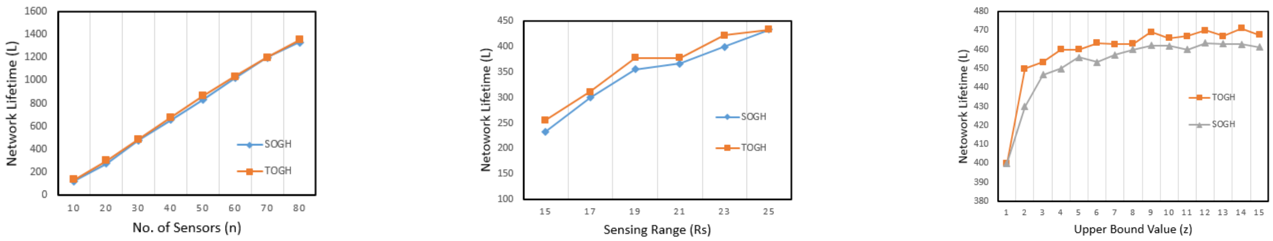
Table I: Sample network lifetime and fault tolerance for sensors' overlap bound

	Lifetime	Fault Tolerance	Normalized Lifetime (L_z)	Normalized Fault Tolerance (F_z)
1	225	0.082921245	0.5	1
2	256.25	0.037087912	0.784090909	0.714392882
3	266.25	0.023990544	0.875	0.63277714
4	275	0.016951588	0.954545455	0.588914154
5	276.25	0.013562312	0.965909091	0.567794015
6	277	0.011499494	0.972727273	0.554939643
7	279	0.009712885	0.990909091	0.543806458
8	279	0.008493301	0.990909091	0.536206666
9	279	0.00753325	0.990909091	0.530224144
10	279.5	0.00672801	0.995454545	0.525206322
11	279.5	0.006097674	0.995454545	0.521278411
12	279.50	.005610307	0.995454545	0.518241401
13	279.5	0.00522225	0.995454545	0.51582324
14	280	0.004790717	1	0.513134156
15	279.5	0.00447434	0.995454545	0.511162665

Now consider Table I which shows the network lifetime and fault tolerance of a sample network scenario for overlap bound. The value of network lifetime and fault tolerance at column # 2 and 3 have been normalized between 0.5 to 1 using min-max normalization procedure and results are shown at column # 4 and 5 respectively. Based on this table I, we draw three plots of goodness index (GI) w.r.t upper bound on sensor overlap (z) as shown in Figure 7. From Figure 7a, it is evident that, when we set α = 1, only lifetime will be considered and (GI) increases as z is increased. In that case, largest value of z indicates the optimal bound value (z^*). In Figure 7b, we set α = 0, i.e. only fault tolerance is considered and GI decreases as z is increased. In that case, disjoint scheme (i.e., z = 1) indicates the optimal bound value (z^*). Finally in Figure 7c, we set α = 0.5, i.e., both the lifetime and fault tolerance are considered together. In this case, some upper bound between 1 to ∞ becomes the optimal bound value (z^*). Hence, it can be concluded that during target coverage we should walk in the middle, neither disjoint scheme nor unbounded overlap will give us the best network performance.

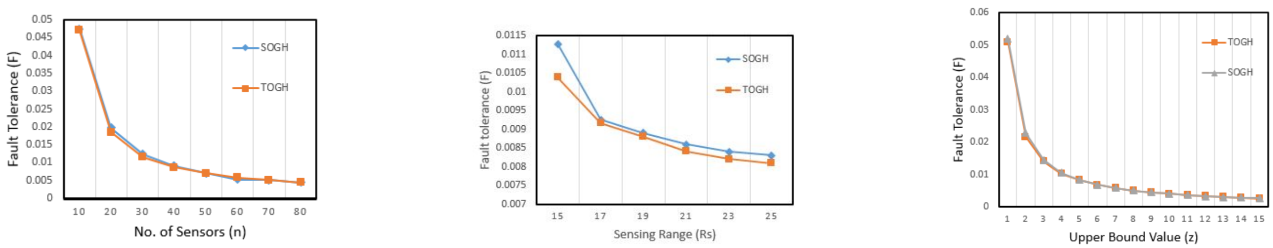
IX. Conclusion

In this paper, we have studied the target coverage problem for directional sensors in WSNs and constructed energy-efficient target monitoring algorithms considering both network lifetime and fault tolerance. At first, we have formulated an exact solution of the problem as linear programming (LP) formulation by adding necessary constraints to minimize power consumption. Then we have developed two greedy heuristics to find the maximum possible set covers each of which individually covers all the targets. Additionally, we have developed:– i) a method to schedule the set covers such that the network can avail maximum lifetime by considering sensor membership value and residual energy, and ii) a method for measuring network fault tolerance. Performance of the proposed algorithms have been demonstrated by conducting extensive simulations. Simulation results reveals the fact that



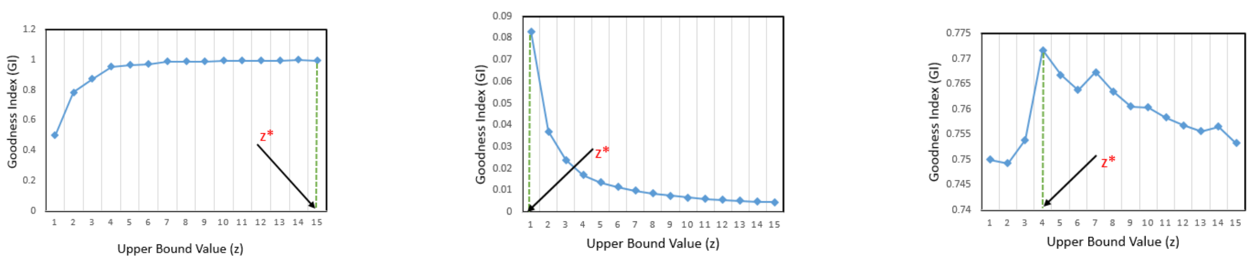
(a) R_s = 25, $q = 8$ and $z = 3$ (b) $n = 25$, $q = 8$ and $z = 3$ (c) R_s = 25, $n = 25$, $q = 8$

Figure 5: Network lifetime with respect to varying (a) number of sensors (b) sensing range (c) bound on sensor overlap



(a) R_s = 25, $q = 8$ and $z = 3$ (b) $n = 25$, $q = 8$ and $z = 3$ (c) R_s = 25, $n = 25$, $q = 8$

Figure 6: fault tolerance with respect to varying (a) number of sensors (b) sensing range (c) bound on sensor overlap



(a) α = 1 (b) α = 0 (c) α = 0.5

Figure 7: Measure the goodness index for bounded sensor overlap when (a) $\alpha = 1$ (b) $\alpha = 0$ (c) $\alpha = 0.5$

there is an inevitable trade-off between prolonging the network lifetime and providing fault tolerance by increasing sensor overlaps within set covers. As a solution of this trade-off, we have used a performance metric called goodness index which helps in determining the optimal bound on sensor overlap by considering both network lifetime and fault tolerance.

References

[1] J.-M. Gil and Y. H. Han, "A target coverage scheduling scheme based on genetic algorithms in directional sensor networks," *Sensors*, vol. 11, no. 2, pp. 1886–1906, 2011.

[2] S. Mini, S. K. Udgata, and S. L. Sabat, "Sensor deployment and scheduling for target coverage problem in wireless sensor networks," *IEEE Sensors Journal*, vol. 14, no. 3, pp. 636 – 644, 2014.

[3] S. Farzana, K. A. Papry, A. Rahman, and R. Rab, "Maximally pair-wise disjoint set covers for directional sensors in visual sensor networks," in *Wireless Days (WD), Toulouse, France, March 23-25*, 2016.

[4] D. Bein, W. W. Bein, and S. Malladi, "Fault tolerant coverage model for sensor networks," in *5th International Conference on Computational Science, Atlanta, GA, USA, May 22-25*, pp. 535 – 542, 2005.

[5] J. Ai and A. A. Abouzeid, "Coverage by directional sensors in randomly deployed wireless sensor networks," *Journal of Combinatorial Optimization*, vol. 11, no. 1, pp. 21–41, 2006.

[6] G. Fusco and H. Gupta, "Selection and orientation of directional sensors for coverage maximization," in *6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), Rome, Italy, June 22-26*, 2009.

[7] S. M. B. Malek, M. M. Sadik, and A. Rahman, "On balanced k-coverage in visual sensor networks," *Journal of Network and Computer Applications*, vol. 72, no. 1, pp. 72–86, 2016.

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

[9] V. P. Munishwar and N. Abu-Ghazaleh, "Target-oriented coverage maximization in visual sensor networks," in *9th MOBIWAC ACM International Symposium on Mobility Management and Wireless Access · MOBIWAC, Miami Beach, FL, USA, October 31-November 04*, 2011.

[10] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad hoc sensor networks," *Computer Communications*, vol. 29, no. 4, pp. 413–420, 2006.

[11] A. Rossi, A. Singh, and M. Sevaux, "Lifetime maximization in wireless directional sensor network," *European Journal of Operational Research*, vol. 231, no. 1, pp. 229–241, 2013.

[12] K. Islam and S. G. Akl, "A localized algorithm for target monitoring in wireless sensor networks," in *International Conference on Ad-Hoc, Mobile and Wireless Networks, Murcia, Spain, September 22-25*, 2009.

[13] "LP solver." https://www.ibm.com/analytics/cplex-optimizer. [Online; accessed 22-January-2019].