Connected coverage in WSNs based on critical targets

Dimitrios Zorbas and Christos Douligeris
jim@students.cs.unipi.gr

Abstract

One of the recent challenges in wireless sensor networks is the design of efficient algorithms to monitor a set of discrete targets lying at a field. A set of active nodes must cover all the available targets and at the same time retain connectivity with the sink. Such a set can remain active until one active node depletes its battery. In this paper, we analyze the problem of finding the proper sensor scheduling in order to maximize the total network lifetime. We present OCCH (Optimized Connected Coverage Heuristic) an efficient algorithm that is based on a general connected coverage methodology. This methodology takes into account the association of the sensors with the poorly covered targets that set an upper bound on the overall computed lifetime. Two solutions are presented to efficiently manage the battery life of these sensors followed by other minor improvements that prolong the network lifetime. Extensive simulation results are presented that show that our solution outperforms other known algorithms found in the literature in terms of achievable network lifetime.

1 Introduction

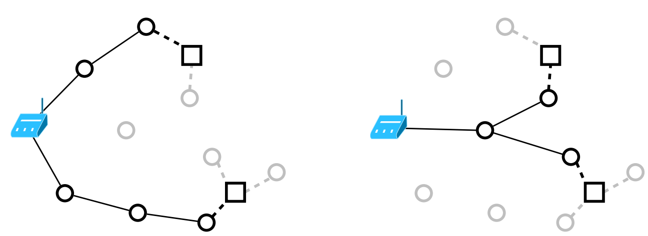
A Wireless Sensor Network (WSN) can be used in a variety of applications, such as in environmental monitoring or in battlefields surveillance in military applications [1]. A WSN consists of hundreds or thousands of most often randomly deployed sensors. Each sensor can receive data from nodes that lie in a usually small area that it is in its sensing range. We say that the sensor provides *coverage* to this area. A sensor collects data periodically or continuously depending on the nature of the application and forwards the data to a node called the Base Station (BS) which provides the necessary connections to infrastructure networking. A sensor node is equipped with a radio device that supports *connectivity* between two nodes or between a node and the BS.

The problem of coverage in wireless sensor networks has been studied from many different aspects. In [2, 3, 4], the coverage problem is described as a quality of service problem, where the objective is to find how well, in terms of the quality of monitored data, the field is monitored by the sensors. In [5, 6, 7, 8, 9], the problem is to compute the appropriate node scheduling and the separation of the available nodes in active and inactive nodes per time unit in order to extend the achievable network lifetime.

Two major types of scheduling coverage have been described in the literature, the *area* coverage and the *point* or *target* coverage [10]. In the former, the whole area must be monitored by the sensors, while in the latter the sensors must cover a set of points (targets) lying in the field. [6, 11, 12, 13] deal with the area coverage problem. This paper focuses on the target coverage problem, but several concepts of this work can be applied on area coverage problems as well, since as described in [5] there are applications where the two problems are equivalent.

A very important issue in WSNs is the availability of connectivity between nodes and the BS, since the monitored data must be forwarded to the BS for further processing. In single-hop networks, the sensors communicate directly with the BS, but in multi-hop networks, a path (i.e. a set of communicating nodes) that connects the two sides must exist throughout the monitoring process.

The most important challenge in a WSN is to efficiently manage the battery consumption of the sensors, since they are characterized by limited energy resources and low computational capabilities. Managing the energy consumption in an efficient way can lead to an extension of the total network lifetime. This energy management takes advantage of the ability of a sensor to put certain parts of its hardware into "sleep mode" and, thus, to consume less energy whenever it is not needed to perform monitoring or, most often, to participate in relaying tasks. Sensors can be divided into sets, called *cover sets*, whereas each cover set can monitor all the available targets. Thus, only one set must be active at any time, while the rest of the sensors can be in sleep mode. Figure 1 illustrates two connected cover sets that monitor two targets. A sensing node can cover a target (square) if it is in its sensing ranges (dashed line) and retain connectivity (normal line) with the BS via relay nodes. Some nodes may remain in sleep mode (gray color) in each cover set.



(a) (b)

Figure 1: Two connected cover sets

The initial works on target coverage address the problem of finding the maximum number of cover sets. Since each cover set operates for a predefined time length the solution to this problem maximizes the network lifetime. Cardei et al. [7] prove that this is a NP-Complete problem and propose linear programming and greedy-based algorithms to provide approximate solutions. This approach is extended in [8], where the sensors are allowed to participate in more than one cover sets, an approach that may extend the produced number of cover sets. In [14], the authors present a detailed methodology of how a greedy target coverage algorithm works and how it is possible to maximize the number of sets by efficiently managing the poorly covered targets. Two heuristics are proposed based on a complex cost function that evaluates the available nodes according to their coverage status, their association to the poorly covered targets and their remaining energy. It must be noted though that these works as well as some other similar works ([15, 16]) do not take into account the connectivity requirement that are present in a multi-hop sensor network environment.

More recent works in the literature take into account the connectivity requirements that appear in multi-hop networks. The issue to be addressed concerns the finding of the maximum number of cover sets, while every node in each cover set in multi-hop networks remains connected with the BS. This problem often translates to the computation of paths with the minimum possible cost since the consumed energy rises with the distance between two nodes. Specifically, in [17], Cardei et al. propose centralized and distributed algorithms for the computation of the connected cover sets. They use a breadth first search algorithm to discover the node-path to the BS through a centralized algorithm, while a minimum spanning tree algorithm is used in a distributed version of the algorithm. In [18], Jaggi et al. propose another connected cover set generation algorithm in order to extend the lifetime of the network. They consider that all the cover sets are disjoint and they try to maximize their number, while they compute a shortest path tree to select the relay nodes that manage to retain connectivity in the network. These works use a simplified energy consumption model, where the energy consumed for communication is predefined for all sensors and it does not depend on the distance between the nodes, which is far from true in a real networks environment. It is, also, assumed that each sensor consumes the same amount of energy, regardless of the number of targets it covers. In real-time WSNs the consumed energy increases with the distance between the nodes, while the amount of the transmitted data depends on the size of the packets and the degree of the data aggregation that is used.

The work most close to ours is that of [19], where the connected target coverage problem is modeled as a maximum cover tree problem. The authors present a theoretical analysis of the problem, prove that it is NP-Complete problem and they propose an approximation algorithm as well as a greedy one, called CWGC, with lower computation cost. Connectivity, coverage and a practical energy consumption model are taken into account. The greedy algorithm applies weights on the edges of the graph of nodes in order to select nodes with high remaining energy and low communication cost. However, CWGC requires a recomputation of all the weights of the graph, each time a new cover set is generated and no policy is applied about the poorly covered targets.

Since critical targets are closely related to the upper bound of the network lifetime, efficient techniques must be applied in order to minimize the energy consumed by a sensor that covers a critical target. In this paper, we present a generic methodology for the computation of cover sets that retain connectivity.

In addition, we propose an efficient greedy heuristic that takes into account the monitoring capabilities of a sensor, its remaining lifetime, the Euclidean distance between the nodes, as well as a number of optimizations related to the critical targets for further network prolonging.

2 Problem Description

In this section, we formulate the connected target coverage problem, we analyze a generic model of how a connected target coverage algorithm works, and, finally, we present optimization problems that an algorithm should solve in order to maximize the network lifetime.

Let $T_0 = \{t_1, t_2, \ldots, t_k\}$ be the set of targets and $S_s = \{s_1, s_2, \ldots, s_n\}$ the set of sensing nodes in a WSN. We assume that each target in T_0 is covered by at least one sensor node in S_s. Moreover, the set S_r contains the sensors that cannot cover any target in T_0, but they can be used as relay sensors in the connected cover sets. Sensors in S_s can be used for sensing or for relaying when it is necessary. The nodes of S_s and S_r, as well as the BS, are part of a undirected connected graph $G = (V, E)$, corresponding the vertices of graph (i.e. $S_s \cup S_r \cup \{BS\} \in V$). Two nodes-vertices $s_j, s_{j'}$ are connected with an edge in G, if and only if their Euclidean distance is lower or equal to the communication range R_c (i.e. $jj' \in E$, iff $d_{j,j'} \leq R_c$). Nodes in S_s and S_r have an initial energy equal to l_0 and they may spend this energy while in active mode in one or more cover sets.

The energy consumption of a sensor is mainly attributed to the energy consumed by the sensing and communication operations [20]. It is commonly assumed that a node spends a constant amount, α_3, of energy in order to "sense" a data bit. Concerning the communication cost, a node spends an energy amount of E_{tx} in order to transmit a bit at distance d and an energy amount of E_{rx} in order to receive a bit, where:

$$E_{tx} = \alpha_{11} + \alpha_2 d^{\alpha} \text{ and } E_{rx} = \alpha_{12}. \tag{1}$$

α_{11} is the energy/bit consumed by the transmitter electronics, α_2 accounts for the energy dissipated in the transmit op-amp, α_{12} is the energy consumed by the receiver electronics and α is the loss exponent of the signal.

A series $C = \{C_1, \ldots, C_m\}$ of m connected cover sets is generated, where each cover set C_p is a subset of the available sensors ($C_p \subseteq S_s \cup S_r$). Each cover set p operates for a period of time denoted as τ_p. This is the time interval until one active node in a cover set depletes its battery. The amount of energy $E(s_j, C_p, \tau_p)$ that a node s_j consumes when it is active in a cover set C_p depends on the sensing operation, on the number of targets that s_j monitors and the number of data that the node may forward (if s_j is used for relaying).

S_{r_p} contains all the nodes that will operate as relay nodes in the cover set C_p, thus, it may consist of nodes from S_s and/or S_r. S_{s_p} contains nodes that are used only for sensing, i.e. $S_{s_p} \subseteq S_s$. rel_data_j denotes the data produced by the descendant sensing nodes of node s_j in C_p. The set of descendant nodes of s_j is symbolized with D_j. s_j must forward these data to the next node. P_j contains the targets that a sensing node s_j can cover, while *data_rate* the rate at which events occur at each target. The value *packet* is the size of a data packet produced in a sensing node. Depending on whether data aggregation is used or not, *packet* may contain data produced by a single target or by multiple targets. If no data aggregation is used, rel_data_j is given by:

$$rel\_data_j = \sum_{j'=1}^{|D_j|} |P_{j'}| \lceil \frac{data\_rate}{packet} \rceil \; packet, \; j' \in D_j, \; j' \in S_s, \; s_j \in C_p.$$

If data aggregation is used, rel_data_j is given by:

$$rel\_data_j = \lceil \frac{\sum_{j'=1}^{|D_j|} |P_{j'}| \; data\_rate}{packet} \rceil \; packet, \; j' \in D_j, \; j' \in S_s, \; s_j \in C_p.$$

Similar to rel_data_j, $sent\_data_j$ denotes the data that s_j transmits to the next node in the path towards the BS. If no data aggregation is used, $sent\_data_j$ is given by:

$$sent\_data_j = (\sum_{j'=1}^{|D_j|} |P_{j'}| + |P_j|) \lceil \frac{data\_rate}{packet} \rceil \; packet, \; j' \in D_j, \; j' \in S_s, \; s_j \in C_p.$$

If data aggregation is used, rs_j is given by:

$$sent\_data_j = \lceil \frac{(\sum_{j'=1}^{|D_j|} |P_{j'}| + |P_j|) \; data\_rate}{packet} \rceil \; packet, \; j' \in D_j, \; j' \in S_s, \; s_j \in C_p.$$

$$E(s_j, C_p, \tau_p) = \begin{cases} (\alpha_3 \; |P_j| \; data\_rate + E_{tx} \; sent\_data_j) \; \tau_p & \text{if } s_j \in S_{s_p}, s_j \notin S_{r_p}, \\ (\alpha_3 \; |P_j| \; data\_rate + E_{rx} \; rel\_data_j + & \\ +E_{tx} \; sent\_data_j)\tau_p & \text{if } s_j \in S_{s_p}, s_j \in S_{r_p}, \\ (E_{rx} \; rel\_data_j \; + \; E_{tx} \; sent\_data_j) \; \tau_p & \text{if } s_j \notin S_{s_p}, s_j \in S_{r_p}, \\ 0 & \text{if } s_j \notin S_{s_p}, s_j \notin S_{r_p}. \end{cases} \tag{2}$$

The objective of a connected target coverage algorithm is to maximize the total surveillance time of the network $\sum_{p=1}^{m} \tau_p$, subject to:
(a) $\sum_{p=1}^{m} E(s_j, C_p, \tau_p) \leq l_0, \; \forall \; s_j \in S_s \cup S_r$ and
(b) for each $C_p \in C, \; \exists \; s_j \in C_p : \; l_j - E(s_j, C_p, \tau_p) = 0$. As l_j is defined the remaining energy of the node s_j throughout the monitoring process.

Authors of [19] propose a model where a cover set may remain active for a predefined period of time (i.e. τ) except if a node exhausts each battery. This model is equivalent to ours considering that $\tau \to \infty$. In Section 4 we compute the maximum possible time interval that a cover set may remain active.

3 A generic framework to compute cover sets

Because of the NP-Completeness of the connected target coverage problem [19], suboptimal solutions are proposed in order to provide results in a relatively short amount of time. In this subsection we describe a model that includes the steps followed by a general greedy heuristic algorithm in order to generate connected cover sets.

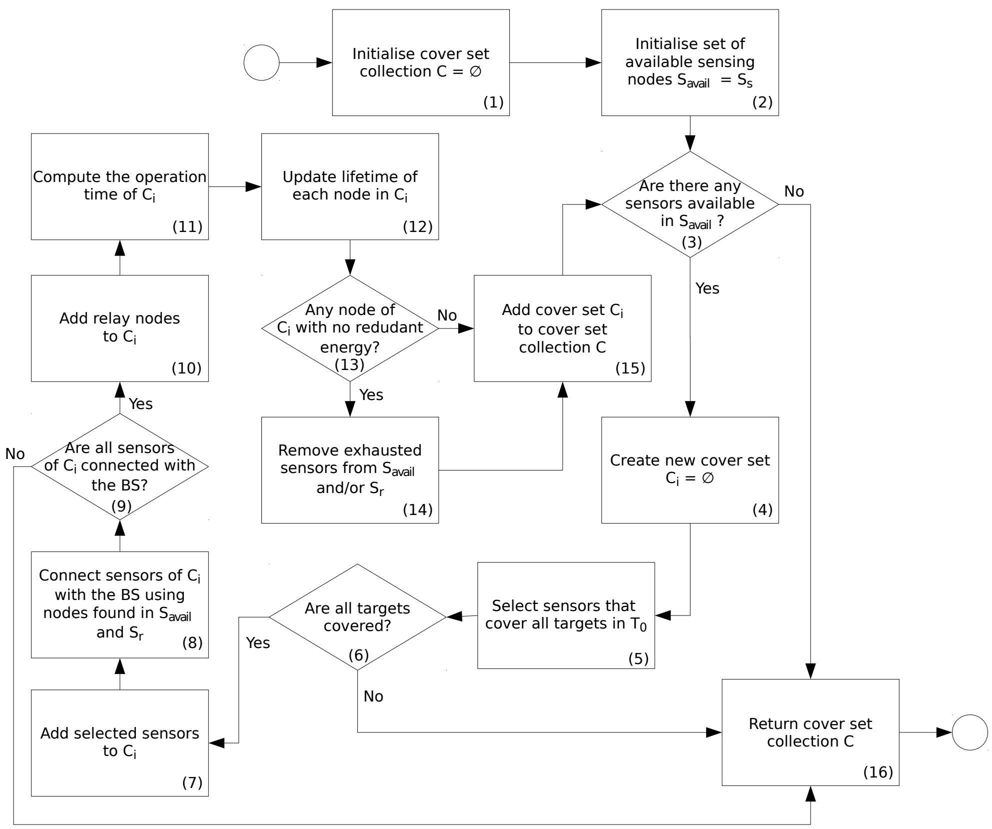


Figure 2: Flowchart of a general greedy heuristic algorithm for the connected target coverage problem

Figure 2 shows the general structure of a greedy connected coverage algorithm. First, the cover sets collection C is initialized (step 1) along with the set of the available sensing nodes S_{avail} (step 2). This set includes only nodes that have not depleted their battery. If S_{avail} is not empty (step 3) a new cover set is initialized (step 4) and a number of nodes that fully covers all the targets of T_0 is selected (step 5). If the nodes of S_{avail} cannot provide full coverage, the generation process is terminated and the algorithm returns the cover set collection C (step 6). The selected sensors are added to the current cover set

C_i (step 7). Next, a path from each node of C_i to the base station must be computed (step 8). In order to find such a path, all the nodes can be considered as part of a connected graph, where the vertices of the graph correspond to the nodes and the edges to the physical connections between the nodes. Since the consumed energy increases as a function of the distance, a connected tree with the minimum cost must be computed (step 8). If no path can be found for at least one sensor in C_i (i.e. there are not enough relay nodes or the graph is not transitive), the cover set collection C is returned (step 9). All relay nodes are added to C_i in step 10. This cover set can operate until the battery of one node of C_i is depleted or for a certain predefined time. Depending on the time length of the cover set the lifetime of the nodes of C_i is updated (step 12). Sensors with no remaining energy are excluded from S_{avail} and/or S_r (steps 13-14). Finally, the current cover set C_i is added to the cover set collection C and new cover sets can be computed using the remaining sensors in S_{avail} and S_r.

3.1 Lifetime maximization issues

In previous works, where a simplified energy model was assumed, the maximum operation time of the network was bounded by the number of sensors that cover the most sparsely covered target(s), called *critical target(s).* It was often assumed that all the sensors consume the same amount of energy and each node can participate in a predefined number of sets. This means that all the cover sets can operate for the same fixed amount of time. In our work, we assume that the consumed energy is proportional to the distance between two nodes and, thus, the sensors consume different amounts of energy in a cover set. Therefore, the number of sensors is not a representative factor in order to define a critical target. Thus, we can make the following definitions:

Definition 1 *We call a target critical, if and only if the sum of the energy of the sensors covering this target is less than or equal to the sum of the energy of the sensors covering each of the other targets in the network.*

At the beginning of the cover set generation process all the sensors have initially the same amount of energy. Thus, the number of critical targets may be more than one.

Definition 2 *We call a sensor critical, if and only if it covers one or more critical targets.*

The maximum operation time of the network is bounded by the amount of energy of the critical sensors, if no communication bottleneck occurs (i.e. a sufficient number of relay nodes exists). Thus, this amount of energy sets an upper bound on the network lifetime, so an energy aware generation process is required.

Next, we describe solutions in order to flexibly manage the critical target(s) and maximize the network lifetime. Optimizations can be done in two phases: the coverage phase where a double target coverage must be avoided, and the connectivity phase where a shortest path selection and all operations about critical and low energy sensors must be applied.

Avoidance of double coverage. The first is related to the coverage phase, where a number of sensors must be selected in order to cover all the available targets. These sensors must provide coverage with as few target overlappings as possible, avoiding the monitoring of the same targets and, thus, reducing the spare traffic in the network. As described in Section 1, many authors have proposed solutions to temper this problem. Our solution is described in Section 4.

Shortest path selection. Many improvements can be done in the connectivity phase, where relay nodes are selected to connect the sensing nodes with the sink. Since the energy consumption is proportional to the distance between two sensors, it is efficient to select nodes that are on the shortest path to the BS.

Avoidance of critical sensors. A number of optimizations can be done about the critical sensors, since they set an upper bound on the network lifetime. Figure 3 illustrates an indicative example.

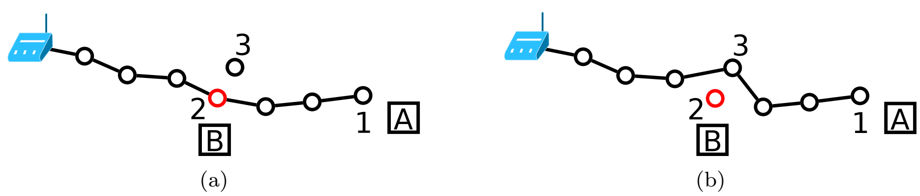


Figure 3: Avoid traversing a node that covers a critical target

We assume that sensor “1” covers target “A” and sensor “2” covers target “B”, where “B” is a critical target. A path from sensor “1” to the BS must be found to forward the monitoring data to the sink. A number of relay nodes is used on the shortest path to the sink. However, this path may include a node that covers critical target “B”, as shown in Figure 3a. This could decrease the energy associated with the critical target, decreasing the overall computed network lifetime. Hence, an algorithm must include mechanisms to choose alternative paths as shown in Figure 3b. Selecting node “3” a bypass is created, avoiding the critical sensors and permitting their usage in future cover sets.

Avoidance of low energy nodes. Furthermore, an important issue is to avoid selecting nodes with very low remaining energy. A possible selection of a such a node could lead to cover sets with a short operation time and, thus, a low overall network lifetime. Hence, a coverage algorithm must take into account not only the distances between the sensors but their remaining energy as well.

Avoidance of nodes with a double role. Finally, an additional improvement is related to the time that a cover set operates. Choosing the same sensor as a relay node in the same cover set must be avoided in the sensing node selection process of the coverage phase. Choosing a sensing node as a relay node as well in the same cover set, could quickly exhaust its battery and it could lead to a cover set with a lower operation time. Sensing nodes in a cover set can be considered as critical sensors, similar to node "2" of Figure 3.

4 Optimized Connected Coverage Heuristic (OCCH)

The proposed algorithm solves the connected target coverage problem by taking into account the issues described in the previous section that manage to considerably extend the network lifetime in most practical cases. The algorithm is called optimized connected coverage heuristic (OCCH) and it operates in a greedy manner. However, OCCH has been designed to operate as a low cost protocol since it is based on node-neighboring information. In this section, we analyze how OCCH works and we provide two ways in order to efficiently manage the critical targets. The advantages and disadvantages of the two methods are explained.

The input of OCCH consists of the following elements:

- the set of targets T_0,
- the set of sensing nodes S_s,
- the set of relay nodes S_r,
- the sets N_i that contain the sensors that cover target t_i, $\forall t_i \in T_0$,
- the sets P_j that contain the targets that a sensor s_j can cover, $\forall s_j \in S_s$,
- the connected graph G,
- the initial battery capacity of a sensor l_0, and
- the size of a data packet $packet$.

OCCH follows the connected coverage model described in Section 3 and consists of three nested loops (see Algorithm 1). The outer loop generates one connected cover set per execution and it incorporates two main operations. During the first operation all the available targets in T_0 are covered (first inner loop) by evaluating and selecting certain sensors (second inner loop). The sensors selected by the first operation are connected to the sink during the second operation. The length of time that the selected sensors will remain active is, also, computed during the second operation. The algorithm outputs the cover set collection C that contains the set of tuples of the form: $C = \{(C_1, \tau_1), ..., (C_m, \tau_m)\}$.

Next, we present in more detail each step of OCCH. At the beginning of the algorithm the cover set collection C is initialized, along with the set of the

Algorithm 1: Optimized Connected Coverage Heuristic

require: $S_s \neq \emptyset$, $S_r \neq \emptyset$, $N_i \neq \emptyset$, $\forall\ t_i \in T_0$, $P_j \neq \emptyset, b_j > 1\ \forall\ s_j \in S_s$, $T_0 \neq \emptyset$, $l_0 > 0$, $packet > 0$, $data\_rate > 0$, G

`// Initialization and Setup`

$C = \emptyset$;

$S_{avail} = S_s$;

foreach $s_j \in S_s \cup S_r$ **do** $l_j := l_0$;

;

;

foreach *edge* $jj' \in E$ **do** set $w_{j,j'}$ according to Formula (4);

;

;

`// Outer loop`

while $|S_{avail}| > 0$ **do**

- $C_{cur} = \emptyset$;
- $S_{cur} = S_{avail}$;
- $T_{cur} = T_0$;
- $\tau_{cur} = 0$;
- update weights according to Formula (6) or (8);
- `// First inner loop`
- **while** $|T_{cur}| > 0$ **do**
  - $selected := none$;
  - $max\_CF := 0$;
  - `// Second inner loop`
  - **foreach** $s_j \in S_{cur}$ **do**
    - compute CF according to Formula (3);
    - **if** $CF > max\_CF$ **then**
      - $max\_CF := CF$;
      - $selected := s_j$;
  - **if** $selected = none$ **then** **return** C;
  - ;
  - ;
  - $T_{cur} = T_{cur} - P_{selected}$;
  - $S_{cur} = S_{cur} - \{selected\}$;
  - $C_{cur} = C_{cur} \cup \{selected\}$;
  - **foreach** $s_{j'} \in neighbors_{selected}$ **do** $w_{selected,j'} \to \infty$;
  - ;
  - ;
- compute the SPT;
- **foreach** $s_j \in C_{cur}$ **do**
  - $C_{cur} = C_{cur} \cup \{$sensors on path from s_j to BS$\}$;
- compute τ_{cur} of the cover set C_{cur};
- **foreach** $s_j \in C_{cur}$ **do**
  - update l_j;
  - **if** $l_j = 0$ **then**
    - delete vertex s_j from G;
    - $S_{avail} = S_{avail} - \{s_j\}$;
- restore weights affected by lines' 10, 23 statements to their previous state;
- **foreach** $s_j \in C_{cur}$ **do**
  - update weights between s_j and its neighbors using Formula (10);
- $C = C \cup \{(C_{cur}, \tau_{cur})\}$;

return C;

available sensors S_{avail} and the current remaining energy of the nodes $l_j, \forall\, j \in S_s \cup S_r$ (lines 1–3).

The outer loop starts by creating an empty cover set C_{cur} and by initializing the time length of this set to τ_{cur} (lines 6–7). S_{cur} and T_{cur} contain the sensing nodes as well as the targets that take part in the next two nested loops respectively (lines 8–9). At each iteration of the first inner loop (lines 11–23) all the available targets in T_{cur} must be covered by sensors of S_{cur}. The cardinality of T_{cur} gradually decreases as sensors are selected for inclusion in the current cover set by the second inner loop. In each execution of the second loop (lines 14–18), the nodes of S_{cur} are evaluated according to a cost function and the top scored node is selected. The targets that the selected sensor covers are removed from T_{cur}, while the remaining ones will be covered in the next iterations. If it is not possible to cover a target, the algorithm returns the current collection of the cover sets C (lines 19–23).

The cost function CF that evaluates the available sensing nodes is given by:

$$CF(T_{cur}, P_j, l_j) = \frac{uncovered}{covered + 1} + \frac{l_j}{l_0} = \frac{|P_j \cap T_{cur}|}{|\{T_0 - T_{cur}\} \cap P_j| + 1} + \frac{l_j}{l_0}, \quad (3)$$

where $|P_j \cap T_{cur}|$ denotes the number of uncovered targets that s_j covers and $|\{T_0 - T_{cur}\} \cap P_j|$ the number of already covered targets that s_j covers (we add one to avoid division by zero). Using this cost function we promote the inclusion in C_{cur} of nodes that cover as many uncovered targets as possible and at the same time as few already covered targets as possible. Moreover, the cost function gives an advantage to the candidates that have high remaining energy. The range of CF is:

$$CF(T_{cur}, P_j, l_j) \in (\frac{1}{|T_0|}, |T_0| + 1].$$

In the second part of the outer loop (lines 24–36), the sensors in C_{cur} must be connected to the BS. A *Shortest Path Tree (SPT)* is computed starting from the BS-vertex. The intermediary nodes on the shortest path from each selected sensing node to the BS are added once to C_{cur}. The current cover set C_{cur} can operate until one active node depletes its battery. Assuming that $\alpha_3 \geq \alpha_{12}$ and that the minimum distance between two sensors is d_{min}, the maximum possible operation time of a cover set, τ_{max}, regardless of the number of the targets or the network topology, is bounded by:

$$\tau_{max} = \frac{l_0}{(\alpha_{11} + \alpha_2 d_{min}^{\alpha}) \lceil \frac{data\_rate}{packet} \rceil packet + \alpha_3 data\_rate}.$$

OCCH terminates whenever it runs our of sensing nodes (i.e. $S_{avail} = \emptyset$) or whenever it runs out of relay nodes (i.e. no path from at least one sensing node to the BS can be discovered), or whenever a target cannot be covered by any sensor.

4.1 Maximizing the lifetime

OCCH incorporates several optimizations in order to maximize the network lifetime taking into account the issues described in Section 3.1. Specifically, three types of weights are considered in the solutions that are not related to each other; weights based on distance, weights based on criticality of the nodes and weights based on residual energy. The weights based on distance are completely different than the other weights. Moreover, two methods are presented in order to manage the critical targets. In the first method, the most critical target is computed based on the residual energy of the sensors it is covered by. However, the residual energy is not included in the computation of the weights, since these weights are considered as infinite. The residual energy is only taken into account in the definition of the most critical target. In the second method, neither the computation of the parameter that controls the critical sensors, nor the computation of the weights include the residual energy of the nodes. Since our purpose is to use nodes with high remaining energy during each phase of the algorithm, it is comprehensible to use the residual energy during the node selection process as well as during the connectivity phase. This notion is used in [19] as well.

Weights based on distance. Initially, OCCH applies weights, $w_{j,j'}$, on each edge jj' of graph G given by (line 10):

$$w_{j,j'} = \frac{\alpha_{11} + \alpha_2\ d^{\alpha}_{j,j'}}{\alpha_{11}},\ jj' \in E. \tag{4}$$

The longer the distance between two nodes, the higher the weight assigned to the edge. $\alpha_{11} + \alpha_2\ d^{\alpha}_{j,j'}$ depicts the energy consumed by s_j in order to send a data bit to $s_{j'}$ and vice versa. We divide by α_{11} in order to avoid the assignment of very small weights. It follows that:

$$w_{j,j'} \in (1, \frac{\alpha_2\ R^{\alpha}_c}{\alpha_{11}}]. \tag{5}$$

Initially, all the nodes have the same amount of energy, thus, the selection of the nodes is achieved using Formula (4) (except of the sensors that are critical).

Weights based on critical sensors. In order to face the issue related to the critical targets, described in Section 2, we present two methods that can be incorporated into OCCH. The **first method** finds the critical targets and the sensors, and it puts an extra weight as a penalty to the edges of the graph between each critical sensor and its neighbors (line 10). Since, as it is presented in 6, the cost of computing this penalty weight is high, this method considers the weights between a critical sensor s_j and a neighbor $s_{j'}$ as infinite:

$$[w_{j,j'}]_{upd} \to \infty,\ jj' \in E,\ s_j \in C_{cur}. \tag{6}$$

The advantage of this method is that only the weights between the critical sensors and their neighbors are updated, but the algorithm needs to find the critical sensors at each execution of the outer loop resulting in an increase into the total number of exchanged messages and, thus, the protocol cost.

The **second method** uses an attribute called *badness* [14]. This parameter is computed once for all the sensing nodes during the setup phase. Badness is a measure of how many poorly covered targets a sensor monitors. This attribute is given by:

$$b_j = \sum_{i=1}^{|P_j|} (M - |N_i| + 1)^3 ,$$

where N_i contains the sensors that cover target t_i and M is the maximum number of sensors that cover a target, i.e. $M = \max(|N_1|, \ldots, |N_k|), k = |T_0|$. The range of b_j is given as follows:

$$b_j \in [1,\ |S_s|^3(|T_0| - 1) + 1]. \tag{7}$$

This attribute is low-valued for sensors that cover none or a few poorly covered targets, while it increases for sensors that cover many poorly covered targets. In [14] this attribute was used in order to avoid the selection of two critical sensors in the same cover set. This attribute can be used to affect the weights between the sensing nodes and their neighbors, increasing the weights according to the badness of each node:

$$[w_{j,j'}]_{upd} = \begin{cases} [w_{j,j'}]_{prev}\ b_j & \text{if } s_j \in S_s,\ s_{j'} \notin S_s,\ jj' \in E \\ [w_{j,j'}]_{prev}\ b_{j'} & \text{if } s_j \notin S_s,\ s_{j'} \in S_s,\ jj' \in E \\ [w_{j,j'}]_{prev}\ \max(b_j, b_{j'}) & \text{if } s_j, s_{j'} \in S_s,\ jj' \in E \\ [w_{j,j'}]_{prev} & \text{if } s_j, s_{j'} \notin S_s,\ jj' \in E \end{cases} \tag{8}$$

In practice, badness substantially increases the weights of nodes that cover the most poorly covered targets in the network. On the other hand, it leaves unaffected the weights of nodes that monitor sufficiently covered targets and, thus, a possible choice of them as relay nodes will not cause an adverse impact in the total surveillance time of the network. Badness can adequately protect the critical sensors since it causes a uniform increase on the weights of the graph, giving the highest priority to paths that do not include any sensing node.

According to Formula (11), the maximum value of badness must be larger than $\frac{w_{max}}{\lambda_1+\lambda_2}$ that is upper bounded by $\frac{|S_s|(|S_s|+|S_r|-1)}{2}$. Considering a uniform node distribution it holds true that $\frac{|S_s|}{|S_r|} = \frac{area\_covered\_by\_targets}{terrain\_size - area\_covered\_by\_targets}$. As *area_covered_by_targets* is defined the area of the field, where if we place a node everywhere within it, the node will cover at least one target in T_0. For example, in case of two targets, this area is given by $2\pi R_s^2$. Hence, $\frac{w_{max}}{\lambda_1+\lambda_2}$ is upper bounded by:

$$\frac{|S_s|^2\ terrain\_size}{4\pi R_s^2} - \frac{|S_s|}{2}. \tag{9}$$

It is comprehensible that the exponent "3" in (7) is used to theoretically increase the maximum value of badness and make it larger than the value calculated by

(9) for deployments with over 30 sensing nodes and large terrain sizes up to 40,000m^2.

Table 1 presents the maximum values of $\frac{w_{max}}{\lambda_1+\lambda_2}$, as well as the badness value of the critical sensors for different terrain sizes and sensing node populations. In practice, even in the case where the terrain is large and the number of sensing nodes low, badness is much higher than the maximum value of $\frac{w_{max}}{\lambda_1+\lambda_2}$ that appeared after 10 executions of the algorithm.

Table 1: badness in comparison to l_0 and to the number of sensors

terrain size (acres)	**number of sensing nodes**	**Maximum value of $\frac{w_{max}}{\lambda_1+\lambda_2}$**	**critical sensor's badness**
10	10	3.11	54
	30	3.06	99
	80	3.66	351
20	10	4.44	128
	30	4.16	250
	80	5.03	547
40	10	6.71	54
	30	5.14	216
	80	7.09	343
60	10	8.46	64
	30	9.02	164
	80	10.50	1331

Avoidance of double role nodes. As explained in Section 3.1, all the selected sensing nodes must be considered as critical sensors in order to avoid their selection as relay nodes in the same cover set. OCCH increases the weights of a selected sensing node with its neighbors, considering these weights as infinite (line 23).

Weights based on remaining energy. At the end of each cover set the weights between a selected node and its neighbors are further updated using Formula 10 (lines 34–35). Sensors with a high remaining energy will have only a small increase, but the increase will be high for nodes with a very low remaining energy. The minimum energy value $w_{j,j'}$ between two neighbors $s_j, s_{j'}$ is applied as it is not predefined which node will be the sender during the next iteration.

$$[w_{j,j'}]_{upd} = [w_{j,j'}]_{prev} + \frac{l_0}{min(l_j, l_{j'})},\ jj' \in E, s_j \in C_{cur}. \quad (10)$$

Nodes with almost no redundant energy cause a significant increase on the corresponding weights and, gradually, the remaining energy becomes the dominant term for selecting a node in future cover sets.

In conclusion, OCCH tries to maximize the total network lifetime by varying the weights of G using the following policy:
(a) Formula (4) contributes the selection of relay nodes that are on the shortest possible path to the BS.
(b) Formula (6) or Formula (8) is used in order to avoid traversing a critical sensor during the SPT computation.

(c) The weights between every selected sensing node and its neighbor are considered as infinite in order to avoid selecting a sensing node as a relay node in the same cover set. A possible selection could decrease the time interval that the current set remains active.
(d) The weights affected by (b) and (c) are restored to their previous state at the end of the operation of the cover set, in order to avoid the indefinite increase of the weights.
(e) Formula (10) is applied in order to select paths with as high redundant energy as possible in future cover sets.

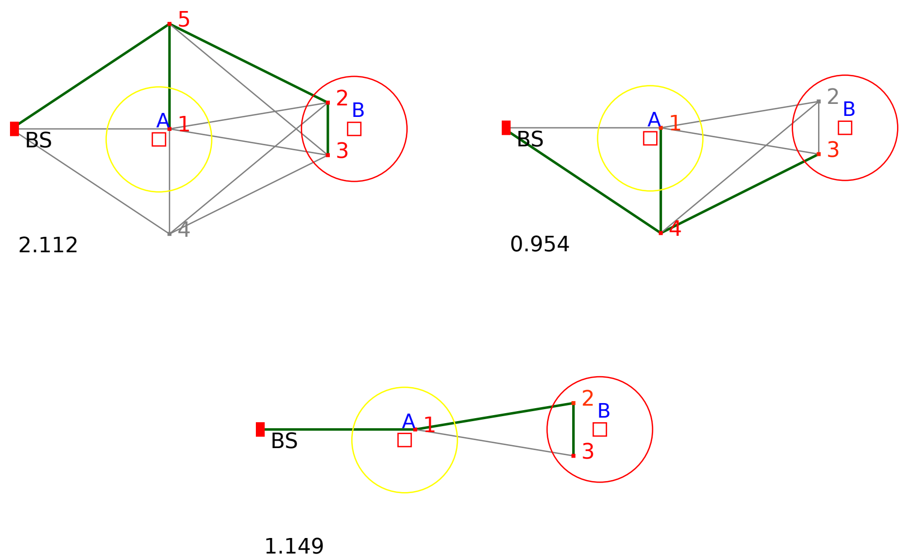


Figure 4: Cover sets generated avoiding critical sensors

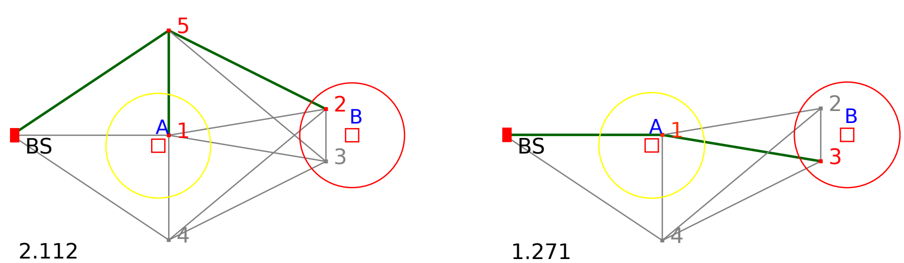


Figure 5: Cover sets generated without avoiding critical sensors

Figures 4 and 5 illustrate the cover sets produced by two approaches. The first one takes into account the critical sensors using the schemes that we mentioned above, while the second does not. The red squares denote the targets and the surrounding circles denote the sensing neighborhood of each target. The circles of the critical targets (e.g. A) are drawn in yellow. "A" is covered by node "1", while "B" is covered by nodes "2" and "3". The active nodes are drawn in red color and the active links in green. The inactive nodes (sleep mode nodes) and the inactive links are shown in gray color. A node with no redundant energy along its links is not shown in the graph. The time duration of each cover set is shown at the left bottom of each graph. The first approach generates three cover sets bypassing critical sensor "1", when it is possible. The total lifetime of the three cover sets is 4.215 hours. The second approach generates two cover sets. Since no action is taken for the critical sensors, node "1" is used as sensing and relay node in the second cover set resulting in a reduction of the overall produced network lifetime (3.383 hours).

Table 2: Minimum and maximum number of messages needed per node

Number of sensors	**critical scheme**		**badness scheme**		**update of all the weights**	
150	325.94	790.36	22.10	39.77	38.74	179.93
200	420.81	1,368.50	31.74	58.60	80.17	649.50
250	849.45	2,942.28	61.64	84.85	178.86	1,120.75
300	699.75	5,270.78	71.23	111.96	332.33	743.48
350	1,532.29	7,114.66	90.21	165.69	308.81	2,545.32
400	911.26	9,303.13	99.79	185.29	254.28	2,118.51

Table 2 shows the number of exchanged messages needed per node for a network with a variable number of sensors, 30 targets and a terrain size of 22.5 acres. Three schemes are presented. In the first scheme (critical), all the sensing nodes broadcast the number of targets they cover along with their remaining battery lifetime. When a sensing node takes this information from all the other sensing nodes in the network, it can decide if it will be a critical sensor during this cover set and updates its weights with its neighbors. The broadcasting process takes place in the beginning of each cover set. On the other hand, in the second scheme (badness), the broadcasting process takes place only once. All the sensing nodes can compute the value M as well as their badness value. An indicative measure of how many exchanged messages are required to update all the weights of the graph is shown in third column (this scheme is used by [19]). The results show that the critical scheme requires a higher number of messages compared to the badness scheme. Furthermore, as the number of sensors increases, the number of messages for the first scheme becomes prohibitive in terms of the protocol cost. One could decrease the number of messages by finding the critical sensors for a few times only during the generation process (than calculating them in each cover set). However, even in the case where the critical sensors are calculated every ten cover sets the number of messages still remains high.

5 Performance evaluation

In this section, we simulate OCCH and compare its performance to previously proposed connected target coverage algorithms, namely CWGC [19], Greedy-CSC [17] and GIECC [18]. We have slightly modified Greedy-CSC and GIECC in order to use the energy consumption model described by (1) and (2). We present results for the two flavors of OCCH according to how it deals with the critical sensors. The first flavor is called *OCCH-critical* and it uses the first scheme presented in Section 4.1 and the second one is called *OCCH-badness* and uses the second scheme[1].

5.1 Terrain generator

Our terrain generator is capable of producing 2D square topologies with uniformly deployed sensors and targets. The targets are not allowed to be positioned in a distance lower than R_s (sensing range) from the borders of the terrain. The minimum allowed distance between two sensors is 0.1 meters. Targets not covered by any sensor are ignored. Sensors that are not part of the graph component that contains the BS are also ignored. The generator script produces (a) the sets N_i that contain the sensors that cover target t_i, $\forall\, t_i \in T_0$, (b) the coordinates of the sensors, the BS and the targets, and (c) graph G.

Figure 6 illustrates a solution assuming a network that consists of 250 sensors and 15 targets. The squares denote the targets and the surrounding circles denote the neighboring sensors that cover each target. The color of each circle varies depending on the number of the neighboring sensors and their available energy. The sensors that are in sleep mode have gray color, while the active ones are with red color. The semicircle denotes the communication range of the BS. The shortest path tree is shown in black color.

5.2 Simulation results

Our evaluation consists of four experiments, where we assess the impact of a single parameter of the problem on the network lifetime. The parameters we vary are: (a) the terrain size, (b) the number of targets, (c) the number of sensors, and (d) the position of the base station. We, also, present results that include data aggregation and results that do not include data aggregation. We run each simulation scenario 50 times, with random target and sensor deployments, and for each scenario we compute the average network lifetime of these 50 runs, along with the 95% confidence intervals. The communication range of the nodes is $50m$ and their sensing range is $10m$. The position of the base station is fixed at $(0, y/2)$, except of the last experiment where we move the BS to various positions. We assume that all the nodes have the same hardware capabilities. Concerning the energy consumption model, we use the following parameters [21]: $\alpha_3 = 100nJ/bit$, $\alpha_{11} = 50nJ/bit$, $\alpha_{12} = 100nJ/bit$, $\alpha_2 = 100pJ/bit/m^2$,

[1]The proposed algorithms have been implemented in Perl programming language and they can be found at http://rainbow.cs.unipi.gr/projects/sensors

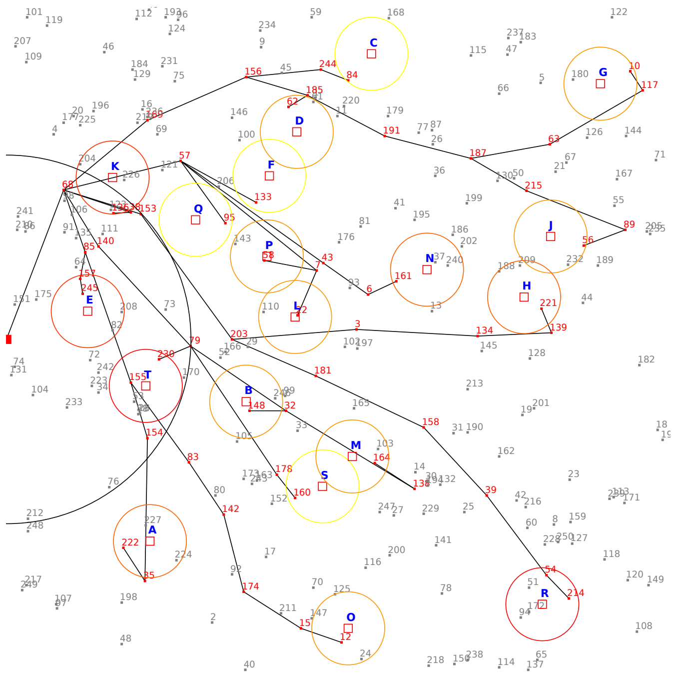


Figure 6: A connected cover set of a WSN with 250 sensors and 15 targets

$l = 20J$, $packet = 500bytes$. Each target generates constant traffic with a data rate of 1 Kbps.

Figure 7 illustrates the performance of the algorithms when the terrain size increases but the number of sensors and targets is kept constant. We separate the results into two figures for presentation purposes. The reduction of the terrain size affects the density of the network and lengthens the distances between the nodes. Both flavors of OCCH present the best results, outperforming CWGC by 10 to 30% and the other two algorithms by 100-200%. OCCH has a more clear advantage in sparse node deployments where the weight assignments play a critical role, since in this environment the nodes consume much more energy.

In the second experiment the terrain size as well as the number of sensors are kept fixed and we vary the number of targets. Here, we assess the impact of the generated traffic on the network lifetime. As it shown in Figure 8, CWGC performs close (10-15% difference) to OCCH in deployments with a few targets, but OCCH performs even better when the number of targets is large (30-35% difference). This considerable improvement occurs due to the fact that OCCH minimizes the probability of having two sensors in the same cover set, when many targets are deployed, resulting in many sensor overlappings.

In our next experiment we vary the number of sensors, keeping a fixed terrain size and a constant number of targets. The results are shown in Figure 9. Both

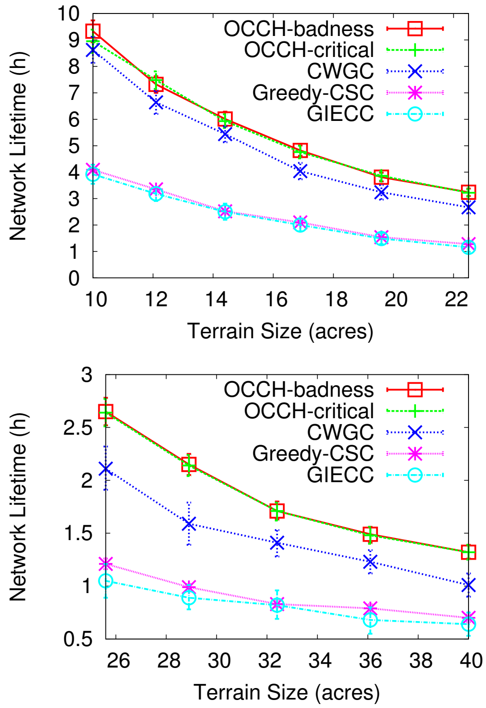


Figure 7: 300 sensors, 30 targets, variable terrain size, no data aggregation

flavors of OCCH perform from 9 to 36% better than CWGC and up to 165% better than the other two algorithms. Table 3 presents the energy consumed per hour by the five algorithms. It is obvious that OCCH-badness consumes the most energy due to the fact that it tries to bypass sensing nodes, resulting in an increase of the total node path length. OCCH-critical is second, as it avoids only the critical sensors.

Figures 10,11 and 12 present the corresponding measurements for the case when data aggregation is used. The total produced lifetime can be doubled, while both flavors of OCCH retain the lead.

Finally, in our last experiment we assess the impact of the BS position to the network lifetime. We have placed the BS in four different positions and we vary the terrain size keeping constant the number of sensors (300) and targets (30). These four positions are: (a) the center of the field $(x/2, y/2)$, (b) the middle of the left side of the square field $(0, y/2)$, (c) the upper left corner of the field $(0, 0)$ and finally in (d) the BS is located at least $R_s + R_c$ distance away from each target, so that none of the sensing nodes will be able to directly communicate with the BS. Figure 13 illustrates the corresponding results for these four cases. The results show that as the BS is located more centrally in the field, CWGC is more close to OCCH. This considerable situation occurs

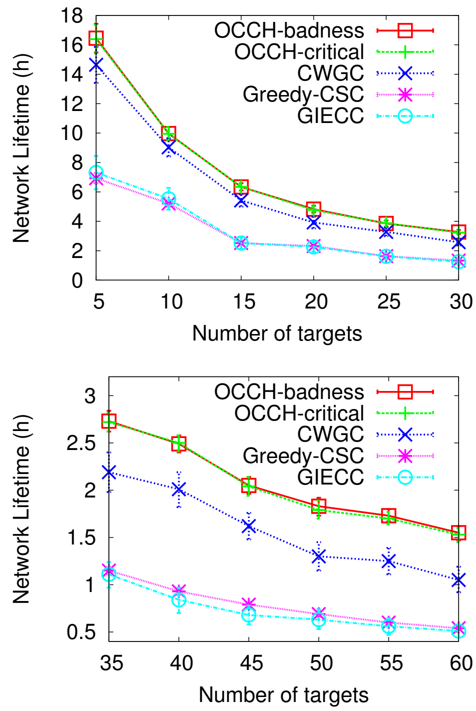


Figure 8: 300 sensors, variable number of targets, 22.5 acres terrain size, no data aggregation

Table 3: Energy consumed per hour (J) – variable number of sensors, 30 targets, 22.5 acres terrain size

Number of sensors	**OCCH badness**	**OCCH critical**	**CWGC**	**Greedy CSC**	**GIECC**
150	598.95	554.44	441.75	465.48	472.71
200	634.08	580.25	478.55	493.42	496.70
250	656.70	595.37	508.26	510.81	513.05
300	666.40	596.55	520.66	518.90	526.88
350	671.00	600.07	525.44	511.25	521.56
400	682.35	598.93	526.45	521.24	528.04
450	670.35	590.11	522.32	515.08	520.95
500	679.85	597.18	534.41	526.74	536.64
550	673.42	593.55	532.55	518.62	522.59
600	675.84	593.97	530.70	512.87	519.40

due to the fact that the probability of having sensors that directly communicate with the BS is higher in topologies where the BS is located at the middle of the field, while the opposite holds true when the sensing nodes are many hops

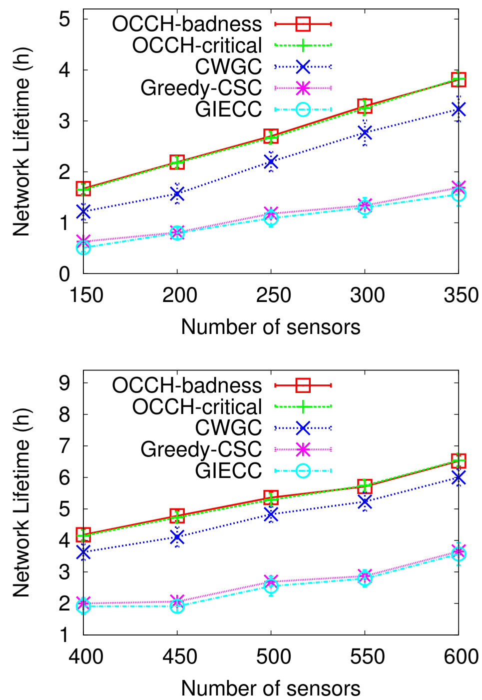


Figure 9: variable number of sensors, 30 targets, 22.5 acres terrain size, no data aggregation

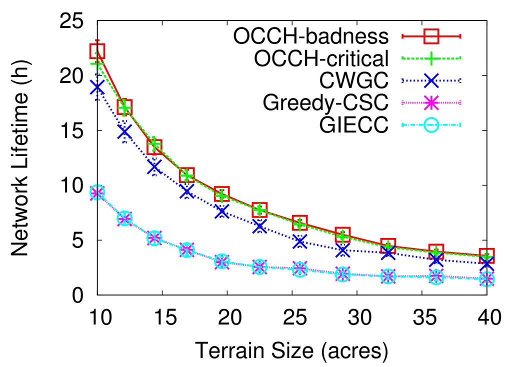


Figure 10: variable terrain size, fixed number of sensors and targets, data aggregation is used

away from the BS. Since many sensors directly communicate with the BS, the probability of traversing a critical sensor on the path to the BS is low. Thus, the problem is transformed to the calculation of how many sensors are in a single

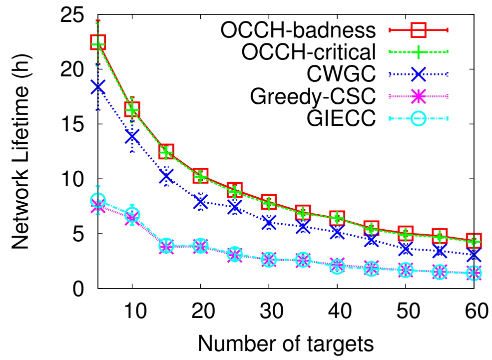


Figure 11: variable number of targets, fixed terrain size and number of sensors, data aggregation is used

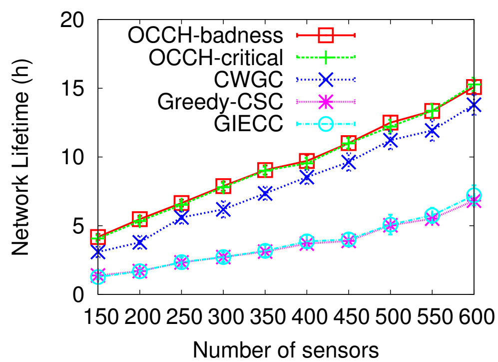


Figure 12: variable number of sensors, fixed terrain size and number of targets, data aggregation is used

hop away from the BS, than to how avoid in an efficient way the critical sensors. In the last case, where all the sensing nodes are multiple hops away from the BS, all the algorithms produce the same amount of network lifetime (for large terrain sizes), since only a few sensors can directly communicate with the BS. The energy of these sensors depletes quite quickly, becoming a bottleneck for the network.

6 Conclusions and future work

In this paper, we have presented OCCH, an efficient algorithm that produces connected sets of sensors in order to cover a set of discrete targets. The proposed solution is based on a general greedy methodology and tries to avoid double covering targets reducing the total generated traffic. Moreover, it avoids the

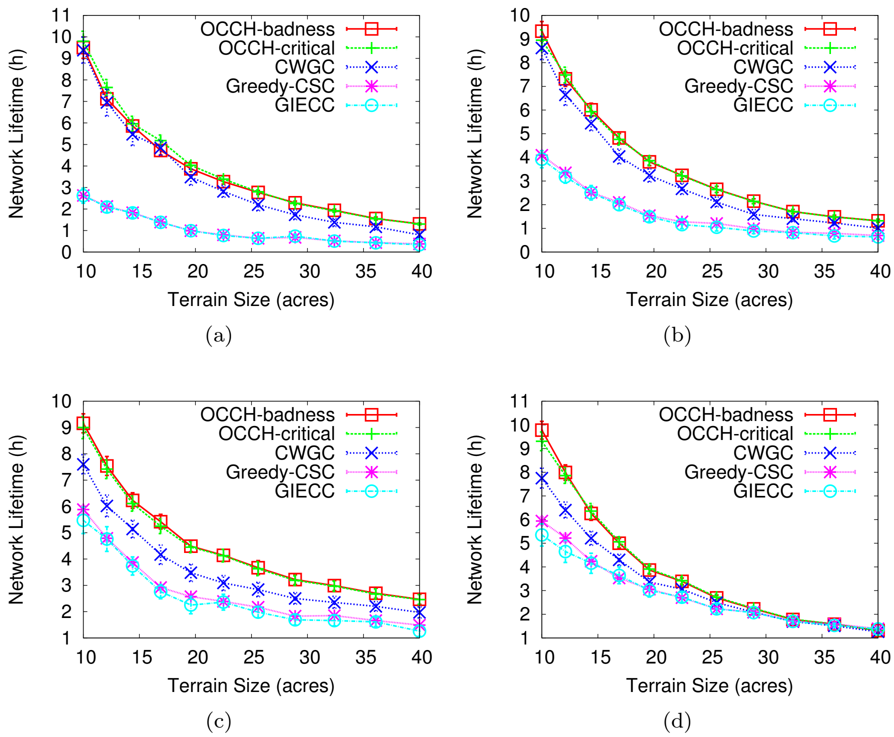


Figure 13: 300 sensors, 30 targets, variable terrain size, variable base station's position

traversing of the critical sensors when the sensing nodes forward the monitored data to the base station. Two different schemes have been presented for this purpose. The simulation results show that OCCH outperforms the existing solutions for all sensor and target deployments. CWGC is about 10-30% below OCCH, as it lacks a critical sensor management policy. On the other hand, Greedy-CSC and GIECC have been developed to work with a simplified energy consumption model and, thus, no weights have been applied on the network graph. A part of our future work is to assess OCCH in a distributed environment or to assess the impact of a non-uniform deployment and use it in other types of coverage problems such as the area coverage or the *k*-coverage problem.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Computer Networks 38 (4) (2002) 393–422.

[2] S. Megerian, F. Koushanfar, M. Potkonjak, M. B. Srivastava, Worst and best-case coverage in sensor networks, IEEE Transactions on Mobile Computing 4 (2005) 84–92.

[3] X.-Y. Li, P.-J. Wan, O. Frieder, Coverage in wireless ad-hoc sensor networks, in: Communications, 2002. ICC 2002. IEEE International Conference on, Vol. 5, 2002, pp. 3174 – 3178.

[4] Y. Gu, Y. Ji, J. Li, B. Zhao, Qos-aware target coverage in wireless sensor networks, Wirel. Commun. Mob. Comput. 9 (12) (2009) 1645–1659.

[5] S. Slijepcevic, M. Potkonjak, Power efficient organization of wireless sensor networks, in: Proc. of International Conference on Communications (ICC'01), IEEE, 2001, pp. 472–476.

[6] P. Berman, G. Calinescu, C. Shah, A. Zelikovsky, Power efficient monitoring management in sensor networks, in: Proc. of Wireless Communications and Networking Conference, Vol. 4, IEEE, 2004, pp. 2329–2334.

[7] M. Cardei, D.-Z. Du, Improving wireless sensor network lifetime through power aware organization, ACM Wireless Networks 11 (3) (2005) 333–340.

[8] M. Cardei, M. Thai, Y. Li, W. Wu, Energy-efficient target coverage in wireless sensor, in: Proc. of INFOCOM 05, Vol. 3, IEEE, 2005, pp. 1976–1984.

[9] D. Zorbas, D. Glynos, P. Kotzanikolaou, C. Douligeris, B{GOP}: An adaptive coverage algorithm for wireless sensor networks, in: Proc. of the 13th European Wireless Conference, Paris, 2007.

[10] M. Cardei, J. Wu, Energy efficient coverage problems in wireless ad hoc sensor networks, Computer Communications 29 (4) (2006) 413–420.

[11] L. Wang, S. S. Kulkarni, Sacrificing a little coverage can substantially increase network lifetime, Ad Hoc Netw. 6 (8) (2008) 1281–1300.

[12] F. Y. Zhong, G. S. Lu, L. Zhang, Peas: a robust energy conserving protocol for long-lived sensor networks, in: Proc. of 10th IEEE International Conference on Network Protocols, 2002, pp. 200–201.

[13] H. Zhang, J. C. Hou, Maintaining sensing coverage and connectivity in large sensor networks, Ad Hoc & Sensor Wireless Networks 1 (1-2).

[14] D. Zorbas, D. Glynos, P. Kotzanikolaou, C. Douligeris, Solving coverage problems in wireless sensor networks using cover sets, Ad Hoc Networks 8 (2010) 400–415.

[15] Z. Hongwu, W. Hongyuan, F. Hongcai, L. Bing, G. Bingxiang, A heuristic greedy optimum algorithm for target coverage in wireless sensor networks, Pacific-Asia Conference on Circuits, Communications and Systems 0 (2009) 39–42.

[16] Y. Gu, J. Li, B. Zhao, Y. Ji, Target coverage in wireless sensor networks: A column generation based approach, in: Proc of IEEE MASS 2009, 2009.

[17] I. Cardei, M. Cardei, Energy efficient connected coverage in wireless sensor networks, Int. J. Sen. Netw. 3 (3) (2008) 201–210.

[18] N. Jaggi, A. A. Abouzeid, Energy-efficient connected coverage in wireless sensor networks, in: 4th Asian International Mobile Computing Conference (AMOC), 2006, pp. 77–86.

[19] Q. Zhao, M. Gurusamy, Lifetime maximization for connected target coverage in wireless sensor networks, IEEE/ACM Trans. Netw. 16 (6) (2008) 1378–1391.

[20] M. Bhardwaj, T. Garnett, A. Chandrakasan, Upper bounds on the lifetime of sensor networks, in: Communications, 2001. ICC 2001. IEEE International Conference on, Vol. 3, 2001, pp. 785 –790 vol.3.

[21] Z. Hu, B. Li, On the fundamental capacity and lifetime limits of energy-constrained wireless sensor networks, in: RTAS '04: Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium, IEEE Computer Society, Washington, DC, USA, 2004, p. 2.

Appendix A

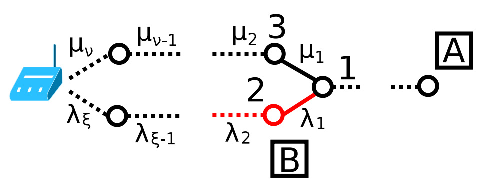


Figure 14: Applying a penalty weight to the links between the critical sensor and its neighbors

Figure 14 shows a general example of how it is possible to avoid the critical sensors by increasing the weights with their neighbors. We assume that target "A" is covered and the monitored data must be forwarded to the BS. Node "1" must decide where it will route the data. Two alternative paths exist. Path $\lambda_1, \lambda_2, ..., \lambda_\xi$ that contains critical sensor "2" and path $\mu_1, \mu_2, ..., \mu_\nu$. We can increase the weights λ_1 and λ_2 by multiplying them by a factor $\rho > 1$ in order to avoid the selection of node "2". The new weights are symbolized with λ'_1 and λ'_2. Hence, it must hold true that:

$$\lambda'_1 + \lambda'_2 + \sum_{i=3}^{\xi} \lambda_i > \sum_{j=1}^{\nu} \mu_j \Leftrightarrow$$

$$\rho(\lambda_1 + \lambda_2) > \sum_{j=1}^{\nu} \mu_j - \sum_{i=3}^{\xi} \lambda_i \Leftrightarrow$$

$$\rho > \frac{\sum_{j=1}^{\nu} \mu_j - \sum_{i=3}^{\xi} \lambda_i}{\lambda_1 + \lambda_2}$$

The maximum value of ρ is achieved, when the numerator is maximized and, at the same time, the denominator is minimized. The maximum value of $\sum_{j=1}^{\nu} \mu_j$, called w_{max}, is the maximum "distance" from a sensing node to the BS (it may coincide with the eccentricity of vertex "BS" in G). The minimum value of $\sum_{i=3}^{\xi} \lambda_i$ can be zero, when the BS is the next hop after node "2". Hence, it follows that:

$$\rho > \frac{w_{max}}{\lambda_1 + \lambda_2}. \tag{11}$$

w_{max} must be recomputed during the next execution of the outer loop, since the weights of the graph have been increased using (10). It is obvious that the cost of computing w_{max} is quite high, specially in the case where the nodes communicate in a distributed way.