Location Uncertainty and Target Coverage in Wireless Sensor Networks Deployment

Mohamed H. Shazly, Ehab S. Elmallah, Janelle Harms
Department of Computing Science
University of Alberta
Edmonton, T6G 2E8, Canada
E-Mail: {shazly,elmallah,janelleh}@ualberta.ca

***Abstract*—In this paper we consider a wireless sensor network (WSN) deployed to monitor a set of targets with known positions. Each target has an associated desired level of coverage by its neighbouring sensor nodes. The network deployment process introduces node placement uncertainty described by known probability distributions. Consequently, deficiency in achieving the desired coverage levels occurs with certain probabilities. To estimate such probabilities, we formalize a target coverage deficiency (TCD) problem. We show that the TCD problem is #P-hard even when restricted to grid WSNs. We then consider networks where node transmission ranges guarantee that the network after deployment has the same connectivity as the planned network. For such networks, we devise a dynamic programming algorithm that can solve a discrete version of the problem exactly and can produce lower bounds on the solution of any arbitrary given instance of the problem. We present simulation results that investigate the accuracy of the algorithm, and illustrate its usefulness in evaluating performance of any given node deployment scheme.**

I. Introduction

In this paper, we are interested in wireless sensor networks (WSNs) deployed to monitor events that occur at certain known target points in a given geographical area. In such networks, different target points may require different sensing coverage levels depending on the physical characteristics and importance of the events occurring at these points. This problem context encourages deploying the network so as to realize a carefully optimized network layout that delivers adequate target coverage and network connectivity. Practical network deployments, however, may suffer from node placement errors and uncertainties due to a number of factors such as obstacles, or node movements due to wild life, wind, rain, or water currents. Our interest is on analyzing the effects of such node placement uncertainties, and developing methods to quantify the likelihood that the network after deployment will satisfy desired target coverage levels.

For a literature review, we first note that extensive literature exists on deployments of WSNs to achieve different coverage objectives. Deployment methods discussed in the literature include both *deterministic* methods (e.g., planar square, rectangular, triangular, and hexagonal grids), and *probabilistic* methods (e.g., uniform random distributions, and Poisson distributions with certain rate). Coverage objectives include area coverage (full and fractional), barrier coverage, and coverage of targets with known positions. Many of the existing results are devised under the Boolean disc sensing model while fewer results exist when utilizing directional sensing or probabilistic sensing models (see, e.g., the taxonomies and surveys in [1]–[4], and the work of [5]–[7]). Our general interest in this paper is on the effect of deployment errors on quality of coverage. Examples of deployment errors include placement errors in deterministic deployments and errors causing deviations from uniform random distributions in probabilistic deployments. Our emphasis here is on quantifying the effect of placement errors on the quality of coverage of a set of points with known positions.

Of the available related results, we mention the following examples. The work in [8] (and the references therein) focuses on developing centralized and distributed algorithms for dividing the nodes in a WSN into disjoint (or overlapping) *cover sets*. Each cover set can monitor a given set of targets with known positions, and thus, can be used by a scheduling algorithm to extend the network's lifetime. In [9], [10], analyzing exposure at certain points in a geographical area (obtained by dividing the area using a square grid) is the key for developing algorithms for detecting intruders crossing the area. The exposure analysis done in [10] is further used to develop a sequential deployment approach that aims to minimize the cost of deployment while achieving a desired detection probability.

In comparison to the volume of research done in the above directions, less work exists on analyzing the impact of placement uncertainty on network performance. Of the available results, we mention the following. In [11], the authors analyze the impact on sensing coverage of misaligned rows and columns in a grid-based WSN deployment. In [12], the authors analyze the effect of placement uncertainty on the average connectivity of 3D grid-based deployments. Research on minimizing localization errors for sensor nodes and targets is yet another important related field. Extensive research exists on localization methods (see, e.g., [13], [14]).

In contrast with the above work, our focus here is on assessing the likelihood that a WSN can provide the required levels of target coverage given the possible placement errors in the after-deployed network. To this end, we formalize a problem, called the *target coverage deficiency* (TCD) problem. Roughly speaking, the problem takes as input the desired coverage level of each target, probability models of node placement uncertainties, and a maximum target coverage deficiency threshold. The problem asks for computing the probability that the network after deployment will achieve a maximum deficiency bounded by the given threshold. We note that in target localization problems, nodes can benefit from the use

978-0-7695-5041-1/13 $26.00 © 2013 IEEE
DOI 10.1109/DCOSS.2013.50

of global positioning systems to get accurate results. In our problem, however, using such a system to know a node's position after deployment does not improve the node's ability to detect a target that is supposed to be covered by that node in the planned network.

Our main contributions here are: (1) we formalize a probabilistic target deficiency problem that deals with targets requiring heterogeneous coverage levels, (2) we show that the problem is #P-hard (a class of intractable counting problems) even when restricted to grid networks, and (3) we present an algorithm that computes a lower bound on the solution of any arbitrary instance of the problem for deployments where the after-deployed networks remain connected as the planned networks. The algorithm works by computing an exact solution of a modified set of input values obtained by converting the continuous problem into a discrete problem. To the best of our knowledge our problem formulation, solution approach, and devised algorithm are novel in the literature.

II. Problem Formulation

In this section we introduce notation to formalize the TCD problem, and draw some remarks about its complexity.

WSN Deployment and Connectivity. We denote the set of nodes forming a given WSN G by $V \bigcup \{s\}$ where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of sensor nodes, and s is a distinguished sink node that performs various control and data gathering tasks. Each node $v \in V \bigcup \{s\}$ in the network has a transmission range $R_{tr}(v)$. Two nodes can communicate with each other if they lie within the transmission range of each other. In the network design phase, each node v is assigned a nominal (x, y)-position in the deployment field, denoted $(x(v), y(v))$.

Due to errors introduced by the deployment process, however, the actual position of a node v after deployment lies in a random location inside a known deployment region, denoted $DR(v)$, that can be of any shape. The radius of this region, denoted $R_{deploy}(v)$, is the largest distance between v and any point in $DR(v)$. We assume knowledge of a probability density function $f(v)$ that can be used to compute the probability that v lies in any specified area within $DR(v)$. The exact form of $f(v)$ depends on the technology used in sensor placement as well as the terrain and obstacles of the deployment field; its exact form is not critical to our treatment, only that it is in computable form.

A network deployment scenario results in a *configuration* where nodes assume certain positions in their respective deployment areas. We use $\mathbf{D} = (V \bigcup \{s\}, DR, f)$ to denote the key parameters mentioned above concerning the deployment process.

Targets and Desired Coverage Levels. We use $T = \{t_1, t_2, \ldots, t_r\}$ to denote the set of target points monitored by the given WSN. Each target t has a known position in the WSN's deployment field. In addition, each target t has an associated sensing region (that can be of any shape), denoted $SR(t)$, determined by the target's physical characteristics (e.g., the amount of energy emitted from, or reflected by, the target). A sensor node v can sense target t only if it lies in $SR(t)$. The radius of the region, denoted $R_{sense}(t)$, is the largest distance between t and any point in $SR(t)$. For a successful network operation, we assume that we desire to cover each target t by at least a specified number of nodes, denoted $DC(t)$.

Target Coverage Deficiency. Given a network configuration and a target t, we use $AC(t)$ to denote the achieved coverage of target t by nodes that can sense the target and send this information to the sink node. Due to node position uncertainty in the deployment process, the value $AC(t)$ may differ from the desired minimum value $DC(t)$. We define the coverage *deficiency* of target t as $\delta(t) = \max(0, DC(t) - AC(t))$, and the coverage deficiency of the set T as $\delta(T) = \max(\delta(t) : t \in T)$.

Example. Fig. 1 illustrates a WSN on $|V| = 6$ nodes and a sink node deployed to monitor $|T| = 4$ targets. Setting $R_{tr} = 4$ units for each node ensures network connectivity. For simplicity of the illustration, we use circular deployment and sensing regions. Circular regions, however, are not constraints in the devised algorithm. In the figure, the deployment region $DR(v)$ of each node v has radius $R_{deploy} = 1.0$ units, and the sensing region $SR(t)$ of each target t has radius $R_{sense} = 2.0$ units. The minimum desired coverage of each target t is $DC(t) = 2$. As can be seen, the network has configurations that can achieve $\delta(T) = 0$, given variations in the after-deployment positions of the nodes. ■
We now define the TCD problem.

Definition (the Target Coverage Deficiency (TCD) problem): Given

- $\mathbf{D} = (V \bigcup \{s\}, DR, f)$: A deployment of a WSN G with a set $V \bigcup \{s\}$ of nodes, where each node v has a transmission radius $R_{tr}(v)$, a nominal deployment position $(x(v), y(v))$, and a probability density function $f(v)$ specifying the likelihood that v will lie in a particular subarea within $DR(v)$
- T: A set of target points where each target t has an associated sensing region $SR(t)$, and desired coverage level $DC(t)$
- $\delta_{thr} \geq 0$: A threshold deficiency value

The TCD problem is to compute the probability $TCDef(\mathbf{D}, T, [0, \delta_{thr}]) = \text{Prob}\,[\delta(T) \leq \delta_{thr}]$. ■

For a given deficiency value δ_0, we also use the notation $TCDef(\mathbf{D}, T, \delta_0) = \text{Prob}\,[\delta(T) = \delta_0]$. That is, $TCDef(\mathbf{D}, T, [0, \delta_{thr}]) = \sum_{\delta_0=0}^{\delta_{thr}} TCDef(\mathbf{D}, T, \delta_0])$.

Complexity Remarks. The TCD problem is computationally demanding. Indeed, we show in the Appendix that the problem is #P-hard (a class of intractable counting problems) even when the planned network is restricted to a grid network with only one target and exactly one node that covers the target. In essence, the proof quantifies the complexity of the problem when placement errors affect reachability to the sink node. Next, we consider networks where placement errors do not affect reachability to the sink node; a property that is satisfied when the placement errors are sufficiently smaller than node transmission ranges and/or when the network has sufficient

node density. Our goal is to focus on networks of arbitrary topology that are required to cover many targets. The resulting class of TCD problems is also expected to be intractable, and our devised algorithm provides lower bound on the solution.

III. Main Ideas

In this section we outline the main ideas used to develop our algorithm for computing lower bounds on the $TCDef$ measure.

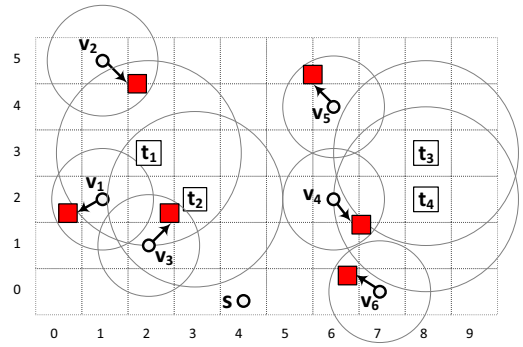


Fig. 1: A WSN on 6 nodes (nominal positions shown at the tail of arrows) and four targets. Here, node (target) circles are deployment (sensing) regions.

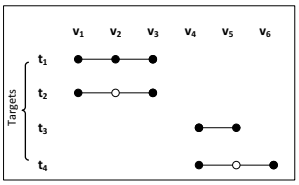


Fig. 2: Processing intervals of the four target nodes

Connectivity Preservation. Given the apparent computational intractability of solving the problem where both node connectivity and target coverage are taken into account when analyzing each possible network configuration, we choose to focus on the effect of location uncertainty on target coverage in the algorithm. To this end, we consider only deployments that preserve the planned network connectivity. That is, if nodes x and y are intended to be connected in the design phase, and $d_{x,y}$ is the distance between their nominal positions then the worst case separation distance after deployment is $d'_{x,y} = d_{x,y} + R_{deploy}(x) + R_{deploy}(y)$. We consider deployments where $d'_{x,y} \leq \min(R_{tr}(x), R_{tr}(y))$. That is, nodes x and y should use sufficient transmission power to maintain connectivity so that each intended link (x, y) exists after deployment, as required.

B-Grid Approximation. We approximate the deployment field of the entire WSN by the smallest encapsulating rectangle, and view this rectangle as being segmented into a matrix $W \times L$ of rectangular blocks: $\{B_{x,y} : x \in [0, L-1] \text{ and } y \in [0, W-1]\}$. We call such a matrix of blocks a *B-grid*. Since each block is a rectangle, the probability that sensor v_i lies in any given block $B_{x,y}$, denoted $p(v_i \rightarrow B_{x,y})$, can be easily computed using the probability density function $f(v)$.

Our devised algorithm obtains a lower bound on the $TCDef$ measure by making the following restrictions.

a) Only blocks that lie entirely within the deployment region $DR(v)$ of a sensor v are considered as possible locations for v after deployment. We denote this set of blocks by $blocks\_DR(v)$.

a) Only blocks that lie entirely within the sensing region $SR(t)$ of a target t are considered as possible locations of sensors covering the target. We denote this set of blocks by $blocks\_SR(t)$.

As can be seen, in (a) we ignore possible covering effects of a node v when it is placed outside $blocks\_DR(v)$, and in (b) we ignore possible coverage of target t with sensors that lie outside $blocks\_SR(t)$. Thus, restrictions (a) and (b) lead to computing a lower bound on the solution of the original input problem.

We also note that the gap introduced by such approximations can be reduced by increasing the granularity of the B-grid (i.e., decreasing the size of its blocks). Increasing the granularity of the B-grid, however, increases the running time of the algorithm.

Example. In Fig. 1, assume we use the coarse-grained B-grid shown in the diagram with dashed lines. Then $blocks\_DR(v_1) = \{B_{1,2}\}$ and $blocks\_SR(t_1) = \{B_{1,3}, B_{2,2}, B_{2,3}, B_{2,4}, B_{3,3}\}$. ■

The TCD Problem on B-grids. The above argument on B-grid approximation and connectivity preserving deployments leads to an interest in examining the TCD problem on B-grids (denoted B−TCD for short). The problem is defined below.

Definition (the B-TCD problem). Given

- $\mathbf{B} = (V \bigcup \{s\}, blocks\_DR, p)$: A deployment of a WSN G where each node v has an associated set $blocks\_DR(v)$ of blocks in a B-grid. Node v lies in any block $B \in blocks\_DR(v)$ with probability $p(v \rightarrow B)$.
- T: A set of target points where each target t has an associated set $blocks\_SR(t)$ of blocks. Target t can be sensed by a node lying anywhere in any block $B \in blocks\_SR(t)$.
- $\delta_{thr} \geq 0$: A threshold deficiency value

The B−TCD problem is to compute the probability $TCDef(\mathbf{B}, T, [0, \delta_{thr}]) = \text{Prob}[\delta(T) \leq \delta_{thr}]$. ■

Here, we also use $TCDef(\mathbf{B}, T, \delta_0) = \text{Prob}[\delta(T) = \delta_0]$, for any given integer $\delta_0 \geq 0$. From the above discussion, we conclude that $TCDef(\mathbf{B}, T, \delta_0)$ is a lower bound on $TCDef(\mathbf{D}, T, \delta_0)$. Similarly, for a given range $[0, \delta_{thr}]$, the probability $TCDef(\mathbf{B}, T, [0, \delta_{thr}])$ is a lower bound on the solution $TCDef(\mathbf{D}, T, [0, \delta_{thr}])$ of the TCD problem.

For the B−TCD problem, we note that a network deployment scenario results in a *B-configuration* described by a mapping where each node v is assigned a block $B \in blocks\_DR(v)$.

IV. Main Algorithm

In this section we present an algorithm for solving the B−TCD problem exactly. The algorithm (function Main in Fig. 3) takes as input an instance of the B−TCD problem where a B-grid of some known dimensions $W \times L$ is used. If we let $DC_{max} = \max(DC(t) : t \in T)$, the algorithm computes for every possible deficiency value $\delta_0 \in [0, DC_{max}]$, the exact value $TCDef(\mathbf{B}, T, \delta_0)$ achieved by a random deployment. The algorithm makes use of the following ingredients.

A. Processing Order of Sensor Nodes

We assume that the nodes in each block $B_{x,y}$ are listed in some order, denoted $[B_{x,y}]$ (any order is acceptable). The algorithm processes the nodes in the B-grid column-wise starting with the sequence of nodes in the blocks of column 0: $([B_{0,y}] : y = 0, 1, \ldots, W-1)$ then column 1: $([B_{1,y}] : y = 0, 1, \ldots, W-1)$, and so on. Let $(v_1, v_2, \ldots, v_n)$ denote the obtained total order of the nodes. The main loop of the algorithm (Step 2 in function Main) then processes the nodes in this given order.

B. States of the Dynamic Program

The dynamic program achieves its efficiency by consolidating key information of many possible B-configurations in one entry in a state table, denoted R. To describe the contents of state table R, we introduce the following definitions.

Definition (first and last processing nodes of a target). Given a processing sequence $(v_1, v_2, \cdots, v_n)$ of the nodes, one can associate with each target $t \in T$ an interval $[first(t), last(t)]$ where $first(t)$ is the first node in the sequence that may sense t (i.e., a node that may lie in a block $B \in blocks\_SR(t)$). Likewise, $last(t)$ is the last node in the sequence that may sense t. ■

Definition (active targets during processing of a node). Given a processing sequence $(v_1, v_2, \cdots, v_n)$ of the nodes, the set of active targets during the processing of node v_i, denoted $T_{active}(v_i)$, is the subset of targets whose intervals pass through node v_i in the above interval representation. ■

Example. In Fig. 1, let $(v_1, v_2, \ldots, v_6)$ be the processing sequence of the nodes. Assume we use a fine-grained B-grid having the solid rectangles (at the tip points of the arrows) as some of its blocks. Fig. 2 then illustrates the intervals associated with each target. For example, for target t_2, we have $[first(t_2) = v_1, last(t_2) = v_3]$. This latter interval follows since in such fine-grained B-grid, node v_1 (or v_3) may lie in a rectangle within the sensing region of t_2. Note that node v_2 lies on this interval, however, target t_2 is not covered by v_2 in any deployment where $R_{deploy}(v_2) = 1$. ■

Definition (dynamic program states). Given a processing sequence $(v_1, v_2, \ldots, v_n)$ of the nodes, and a B-configuration G_i on prefix $(v_1, v_2, \ldots, v_i)$, $i \in [1, n]$, a state of G_i is a tuple (δ_{hist}, δ) where:

- δ_{hist} is the maximum deficiency achieved in G_i over all targets t where $last(t) < v_i$. That is, $\delta_{hist} = \max(\delta(t) : last(t) < v_i)$.
- δ is an associative array whose domain is the set of active targets $T_{active}(v_i)$, and if target t is active then $\delta(t)$ is the deficiency of t in the configuration G_i. ■

Thus, each state (δ_{hist}, δ) in R stores information about a group of B-configurations. In the ith iteration of the algorithm, some targets have been processed and became inactive, while some other targets are active (i.e., under processing in this iteration and some subsequent iterations). All B-configurations summarized by a state (δ_{hist}, δ) have **(a)** the same maximum deficiency of the inactive targets stored in δ_{hist}, and **(b)** the same detailed information about deficiencies of the active targets stored in array δ.

Example. In Fig. 1, let $(v_1, v_2, \ldots, v_6)$ be the processing sequence. Again, assume the use a fine-grained B-grid having the solid rectangles (at the tip points of the arrows) as some of its blocks. Suppose that the algorithm is currently processing node v_5. For targets t_1 and t_2, we have $last(t_1) = last(t_2) = v_3 < v_5$. Hence, the δ_{hist} component of each state in table R summarizes deficiency information about these two targets. On the other hand, $T_{active}(v_5) = \{t_3, t_4\}$. Thus, the δ component of each state in table R encodes detailed information about the deficiencies of these two targets. For example, assuming that the minimum desired coverage of each target t is $DC(t) = 2$, the B-configuration indicated by the arrows gives rise to a state (δ_{hist}, δ) where $\delta_{hist} = \max(\delta(t_1) = 0, \delta(t_2) = 1) = 1$, $\delta(t_3) = 2$, and $\delta(t_4) = 1$. ■

Function Main$(\mathbf{B}, T, p, \delta_{thr})$**:**
Input: An instance of the B−TCD problem defined on a $W \times L$ B-grid
Output: The function computes $TCDef(\mathbf{B}, T, \delta_0)$ for each $\delta_0 \in [0, DC_{max}]$, and returns $TCDef(\mathbf{B}, T, [0, \delta_{thr}])$
Notation: R is a state table holding mappings from states to probabilities
1. **Initialization:** set $R(\delta_{hist} = -1, \delta = \emptyset) = 1.0$
(so, initially, R has has only one state)
2. **for** $(i = 1, 2, \ldots, n)$
{
3. call Update(R, v_i)
4. **if** (a target t exists such that $last(t) = v_i$)) call Post_process(R, v_i)
}
5. **for** $(\delta_0 = 0, 1, 2, \ldots, DC_{max})$
{
6. $TCDef(\delta_0) = \sum_{\delta_0 = \max(\delta_{hist}, (\delta(t): t \in domain(\delta)))} R(\delta_{hist}, \delta)$
}
7. **return** $\sum_{\delta_0 \in [0, \delta_{thr}]} R(\delta_{hist}, \delta_0)$

Fig. 3: Pseudo-code for function Main

Function Main. The main function initializes table R with the state $(\delta_{hist} = -1, \delta = \emptyset)$ of probability 1.0. The -1 is used as a special value indicating that no history deficiency value

Function Update(R, v_i)**:**
Input: State table R and a node v_i to be processed
Output: Updated content of table R
Notation: Q is a state table similar to R

1. clear table Q
2. **foreach** (state (δ_{hist}, δ) in R)
   {
3. **foreach** (block B in $blocks\_DR(v_i)$)
   {
4. set $\delta' = \delta$
5. **foreach** (target t covered by block B)
   {
6. **if** (t not in δ') set $\delta'(t) = DC(t)$
7. $\delta'(t) = \max(0, \delta'(t) - 1)$
   }
8. $Q(\delta_{hist}, \delta')$ $+= R(\delta_{hist}, \delta) \times p(v_i \rightarrow B)$
   }
   }
9. set $R = Q$
10. **return**

Fig. 4: Pseudo-code for function Update

Function Post_process(R, v_i)**:**
Input: State table R and a node v_i that has been processed
Output: Modify each state in R by moving information of targets that have just became inactive from δ to δ_{hist}
Notation: Q is a state table similar to R

1. clear table Q
2. **foreach** (state (δ_{hist}, δ) in R)
   {
3. set new state $(\delta'_{hist}, \delta')$= (δ_{hist}, δ)
4. **foreach** (target t such that $last(t) = v_i$)
   {
5. $\delta'_{hist} = \max(\delta_{hist}, \delta(t))$
6. delete t from δ'
   }
7. $Q(\delta'_{hist}, \delta')$ $+= R(\delta_{hist}, \delta)$
   }
8. set $R = Q$
9. **return**

Fig. 5: Pseudo-code for function Post_process

is stored in δ_{hist}. The setting of $\delta = \emptyset$ clears the domain of array δ indicating that no target is currently active. Step 2 processes the nodes in the specified total order $(v_1, v_2, \ldots, v_n)$. Processing each node is mainly done by function Update. Step 4 calls function Post_process if for some target t, we have $last(t) = v_i$. If t exists then it will be inactive in subsequent iterations and its associated deficiency information can be moved from δ to δ_{hist}.

After processing all nodes, the loop in Step 5 computes for each possible deficiency value $\delta_0 \in [0, DC_{max}]$, the probability $TCDef(\delta_0) = TCDef(\mathbf{B}, T, \delta_0)$. This probability is computed by aggregating entries in table R that satisfy the condition $\delta_0 = \max(\delta_{hist}, (\delta(t) : t \in domain(\delta)))$. We recall that the domain of the associative array δ (in any state (δ_{hist}, δ) in array R) is the set of active targets at the current point of the computations.

Function Update. The function incorporates the effect of positioning a given node v_i in each possible block $B \in blocks\_DR(v_i)$ on the contents of table R. In more detail, Step 1 clears table Q that will hold the computed results. The loop in Step 2 iterates over each state (δ_{hist}, δ) in table R. Step 3 then considers the effect of positioning v_i in each possible block $B \in blocks\_DR(v_i)$. Positioning v_i in a block B has the effect of decreasing the positive deficiency of a covered target t by one. In Step 8, the probability of obtaining this state is computed and accumulated in entry $Q(\delta_{hist}, \delta')$. Step 9 returns in R the newly computed values of Q.

We remark that Step 9 can be done in $O(1)$ time by exchanging pointers of arrays R and Q.

Function Post_process. After processing a node v_i, the function is invoked if there exists a target t such that $last(t) = v_i$. Target t will be inactive until the algorithm terminates, and hence its deficiency information can be moved from the δ component of each state to the δ_{hist} component. This is the main work done by the function.

In more detail, Step 1 clears table Q that will subsequently receive the computed values. Step 2 iterates over all states (δ_{hist}, δ) in R. For each state, Steps 3-6 process each target t where $v_i = last(t)$, by removing t from the associative array δ, and incorporating the deleted information $\delta(t)$ in the history value δ_{hist}. The resulting new state is denoted $(\delta'_{hist}, \delta')$. Step 7 then updates the $Q(\delta'_{hist}, \delta')$ probability. Finally, Step 8 returns in table R the updated contents.

C. Running Time

In this section, we derive the worst case running time of the algorithm as a function of the following parameters.

- $n = |V|$: The number of sensor nodes in the network
- $L_{max} = \max(|blocks\_DR(v)| : v \in V \bigcup \{s\})$: The maximum number of blocks a node can lie in due to uncertainties in deployment
- $DC_{max} = \max(DC(t) : t \in T)$: The maximum deficiency of any target in any B-configuration of the network
- $|T|$: the number of targets

Theorem.

a) In any iteration of the main loop in function Main (Steps 2-4), the length of array R is at most $R_{max} = DC_{max} \cdot \max_{v \in V}(\prod_{t \in T_{active}(v)} DC(t))$.
b) The worst case running time of the algorithm is $O(n \cdot R_{max} \cdot L_{max} \cdot |T|)$.

Proof. To see (a), consider an iteration of the loop in Steps 2-4 of function Main that processes some node, say v. During this iteration, any state (δ_{hist}, δ) in array R has two components: a non-negative integer δ_{hist}, and a sequence of non-negative integers in δ. We note that $\delta_{hist} \leq DC_{max}$. In addition, the number of different possible sequences stored in different instances of array δ is at most $\prod_{t \in T_{active}(v)} DC(t)$. Thus the length of array R during this iteration is at most $DC_{max} \cdot \prod_{t \in T_{active}(v)} DC(t)$, as required.

To see (b), note that the loop in Steps 2-4 of function Main iterates n times. The running time of each iteration is

dominated by the execution of function Update. The outer loop in function Update (Step 2-8) iterates at most R_{max} times. Each iteration of the inner loop in Step 3 considers the different blocks a node can lie in (at most L_{max} blocks). Processing a node when it lies in a block requires looking at the coverage of at most $|T|$ targets. Thus, in the worst case, the algorithm requires $O(n \cdot R_{max} \cdot L_{max} \cdot |T|)$, as claimed. ■

Thus, for problem instances where the number of active terminals processed at any stage is bounded by a constant, the maximum length R_{max} of the array R is polynominal in n (since the maximum deficiency of any target $\leq n$). For such instances, the running time of the algorithm is polynomial in the number of nodes and targets of the problem.

V. Simulation Results

We implemented the dynamic program in the C++ language, and conducted a number of experiments to evaluate the performance and use of our devised algorithm. In this section, we present a few selected results to illustrate some of the important findings. For experiments in parts (a) and (b) below, we use the following setup.

WSN deployment field: We use a WSN deployment field of dimensions $W_{field} \times L_{field} = 50 \times 100$ units, and assign (x, y)-coordinates to points in the field assuming the origin point is at the left-bottom corner.

Network deployment: We use a network of $n = 100$ nodes located randomly at integer coordinates in the WSN field. The deployment region $DR(v)$ of each node v is a square of width $W_{deploy} = 10$ units having v at the center.

Targets: The network monitors 4 targets, positioned at (x, y)-coordinates $x \in \{\frac{L_{field}}{3}, \frac{2L_{field}}{3}\}$, and $y \in \{\frac{W_{field}}{3}, \frac{2W_{field}}{3}\}$ (rounded to integers). The sensing region $SR(t)$ of each target t is a square of width $W_{sense} = 20$ units having t at the center. Our experiments analyze the following aspects.

a) Effect of Increasing B-grid Granularity. In Fig. 6a, we vary the desired coverage of each target in the range $[1, 4]$. The curves on the figure correspond to using three B-grids of different block sizes: $W_{block} = 1, 2$, and 5. We remark that using a B-grid with square 1×1 blocks results in covering each deployment region and sensing region exactly. Thus, the algorithm computes exact $TCDef$ measure.

As can be seen, using coarser B-grids (e.g., $W_{block} = 2$ and 5) results in less accurate lower bounds. This follows since using a coarser B-grid results in ignoring parts of a node's deployment region and/or a target's sensing region.

b) Effect of Increasing Node Deployment Regions. The cost of deploying a WSN is expected to increase when using more precise node positioning methods. Using less precise methods introduces unplanned node displacements. When targets are relatively close to each other, however, a displaced sensor node may serve in covering an unplanned neighbouring target. In such scenarios, node displacements may play a positive coverage role. The experiments done here demonstrate the use of our algorithm as a tool to assess the risks of using a less accurate positioning mechanism. We use the network and targets setup discussed above.

Fig. 6b illustrates three scenarios where $W_{deploy} = 0, 2$, and 10, and the desired coverage varies in the range $DC \in [1, 6]$. Here, the planned nominal positions of nodes (taking effect when $W_{deploy} = 0$) achieve the desired coverage for $DC = 1, 2$ (but not for $DC = 3, 4$). The obtained results show that the distribution of targets in the field and the availability of many nodes make the TCDef measure when $W_{deploy} = 10$ consistently better than when $W_{deploy} = 2$. Thus, in some deployment scenarios the designer may use inaccurate positioning mechanisms (and consequently, reduce the deployment cost) without sacrificing the obtained performance. Our algorithm provides a means for conducting the required cost-coverage analysis that enables detecting such scenarios.

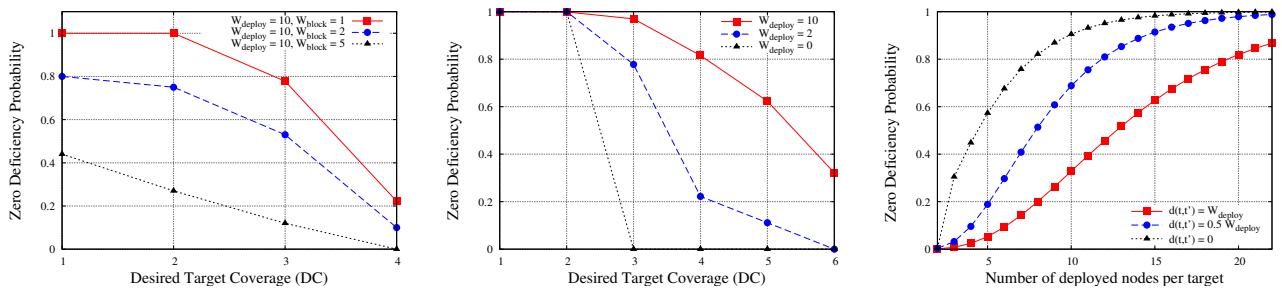
c) Application: Evaluating a Deployment Approach. In this section we tackle the following design problem: given a deficiency threshold δ_{thr} and a probability threshold P_{thr}, find a deployment scheme that satisfies the condition $\text{Prob}\,[\delta(T) \leq \delta_{thr}] \geq P_{thr}$.

One approach to tackle the problem may proceed as follows. **(a)** For each target t, we deploy a sufficiently large set $N(t)$ of nodes whose nominal positions coincide with t's position. **(b)** For each node $v \in N(t)$, we use information on $DR(v)$ and $SR(t)$ to compute the probability $p_{success}(v, t)$ that v covers t. **(c)** Using the Binomial distribution we then compute $\text{Prob}\,[\delta(t) \leq \delta_{thr}]$ by considering the success probabilities of nodes in $N(t)$ only. **(d)** We then utilize the lower bound given by $\text{Prob}\,[\delta(T) \leq \delta_{thr}] \geq \prod_{t \in T} \text{Prob}\,[\delta(t) \leq \delta_{thr}]$, as the basis of determining a suitable size of each set $N(t)$, $t \in T$.

We note that this approach considers coverage of each target t by nodes in $N(t)$ only. Possible *cross-coverage* of t by displaced nodes from another set $N(t')$, $t \neq t'$, is not considered. This limitation of the approach results in a possible overestimation of the minimum number of nodes required to solve the problem. Our algorithm, on the other hand, is designed to consider many potential cross-coverage effects.

Fig. 6c quantifies this overestimation aspect of the approach using a simple scenario where $T = \{t, t'\}$ has only two targets, and $DC(t) = 2$ for each target. Each target is the center of its square sensing region of width W_{sense}, and each node is the center of its square deployment region of width W_{deploy}. Here, we use $W_{sense} < W_{deploy}$. The targets are separated by a distance $d(t, t')$. When $d(t, t') \geq W_{deploy}$, cross-coverage events are not possible. On the other hand, when $d(t, t') \leq \frac{W_{deploy}}{2}$ many such events are possible.

Fig. 6c illustrates this behaviour. Here the x-axis varies $|N(t)|$ in the range $[2, 22]$, and the y-axis is the probability of achieving $\delta(t) = 0$. The curve where $d(t, t') = W_{deploy}$ obtained by the algorithm is found to coincide with the curve obtained by using the Binomial distribution in the approach described above. As can be seen, when $d(t, t') \leq \frac{W_{deploy}}{2}$, the approach overestimates the minimum $|N(t)|$ required to achieve $\text{Prob}\,[\delta(t) = 0] \geq P_{thr}$ for any possible P_{thr} value. This in turn illustrates the cost savings enabled by the use of our algorithm.



(a) Effect of B-grid granularity (b) Effect of location uncertainty (c) Evaluating a deployment approach

Fig. 6: Simulation Results

APPENDIX

We show that

Theorem. The TCD problem is #P-hard even when restricted to grid networks with one target node and the deployment region of each sensor node is divided into at most two distinguishable regions.

The proof reduces in polynomial time the following problem to the TCD problem.

Definition (the 2-terminal network reliability problem on grid networks). Given

- $G = (V, E)$: A connected grid network where each node $x \in V$ has integer Cartesian coordinates, and each edge $(x, y) \in E$ (either horizontal or vertical) has unit length
- s and t: Two distinguished non-adjacent nodes in G. Both nodes are assumed to be perfectly reliable
- p: Each node in $V - \{s, t\}$ operates with probability p, and fails with probability $1 - p$, independent of other nodes

The problem is to compute the probability that after an event of random node failure the set of operating nodes contain a path from s to t. We denote this probability by $Rel(G, s, t, p)$. ■

Fig. 7 illustrates an example grid network with nodes s and t. The above reliability problem has been shown to be #P-hard in [15]. Since grid graphs form a proper subset of unit-disk graphs, the result applies to wireless networks where the quality of the received signal at a point of a certain distance from a transmitter depends on the distance but not the direction of transmission. We now prove the theorem.

Proof. Given an instance (G, s, t, p) of the reliability problem, we construct in polynomial time an instance $(\mathbf{D}, T, \delta_{thr})$ of the TCD problem such that $Rel(G, s, t, p) = TCDef(\mathbf{D}, T, \delta_{thr})$. Thus, showing that the TCD problem is #P-hard, as required. In more detail, the proof maps each possible subgraph G' of G formed by the operating nodes after an event of random failure in the reliability problem to a unique configuration, denoted C', obtained by deploying the sensor nodes in the constructed TCD problem. This one-to-one mapping guarantees that

(a) the probability of obtaining G' equals the probability of obtaining C', and

(b) G' is an operating subgraph (i.e., contains a path from s to t) if and only if configuration C' satisfies the connectivity and deficiency conditions required to satisfy δ_{thr}.

The above two conditions ensure that $Rel(G, s, t, p) = TCDef(\mathbf{D}, T, \delta_{thr})$, as required. The reduction constructs the instance $(\mathbf{D}, T, \delta_{thr})$ as follows (Fig. 8 illustrates the key aspects of the reduction).

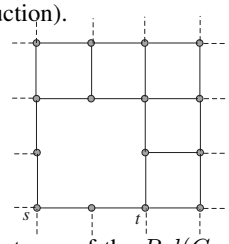


Fig. 7: An instance of the $Rel(G, s, t, p)$ problem

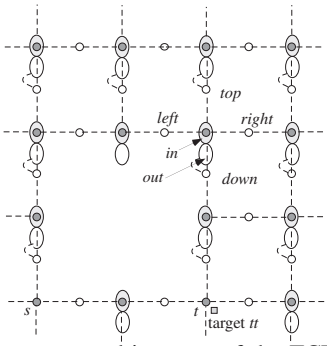


Fig. 8: A constructed instance of the TCD problem

- Node s in the reliability problem is chosen as the sink of the WSN. The constructed TCD problem has one target, denoted tt, that is covered only by terminal t of the reliability problem. The desired coverage of this target is 1, and we set $\delta_{thr} = 0$ to require that a path exists from the covering node t to the sink s.
- The nominal positions of the nodes in the constructed TCD problem are obtained as follows. We replace each horizontal (or vertical) edge (x, y) in G by two adjacent horizontal (respectively, vertical) edges $(x, v_{x,y})$ and $(v_{x,y}, y)$ where node $v_{x,y}$ is a new added

node of degree two (the hollow nodes in Fig. 8). We call a new node horizontal (vertical) *linkage* node if it is introduced by replacing a horizontal (respectively, vertical) edge. The Cartesian coordinates of x and y are adjusted to allow this transformation (the distance between x and y is 2 after the transformation). The new coordinates of a node after transforming all edges of G define its nominal position in the constructed TCD instance.

- The deployment region of each of the two terminals s and t, and each of the horizontal and vertical linkage nodes is the node's nominal position (that is, they are not subject to placement errors).
- The deployment region of each node $x \in V - \{s, t\}$ is composed of two adjacent regions, denoted x^{in} and x^{out}, respectively. Node x falls in x^{in} (x^{out}) with probability p (respectively, $1 - p$).
- All nodes in V and all horizontal linkage nodes have transmission range denoted $R_{horizontal}$. Likewise, all vertical linkage nodes have transmission range denoted $R_{vertical}$. The setting of $R_{horizontal}$, $R_{vertical}$ and the shape of regions of type x^{in} and x^{out} satisfy the following conditions:
  - No pair of linkage nodes or pair of nodes in V communicate directly with each other.
  - Node x can communicate with its left, top, and right linkage nodes only if x lies in region x^{in}.
  - Node x can communicate with its down linkage node if it lies anywhere in $x^{in} \bigcup x^{out}$.

The dashed lines in Fig. 8 illustrate the communicating pairs in the constructed instance. One way to realize the above conditions is to let $\epsilon = 0.05$, and set **(1)** $R_{horizontal} = 1 + \epsilon$ $(= 1.05)$, **(2)** x^{in} to the intersection of two circles of radius $R_{horizontal}$ centered at x_{right} and x_{left}, respectively (thus, the vertical radius of x^{in}, denoted $x^{in}_{vRadius}$, equals $\sqrt{(1+\epsilon)^2 - 1} \approx 0.32$), **(3)** $R_{vertical} = 1 + x^{in}_{vRadius} (\approx 1.32)$, and **(4)** x^{out} to have the same shape as x^{in} but the vertical radius of $x^{out} = \frac{1}{4} x^{in}_{vRadius}$.

Thus, if node x is the top neighbour of node y in G then the transformation makes the smallest distance between two points in y^{in} and x^{out} to be $2 - 2.5x^{in}_{vRadius}$ (≈ 1.2) which is larger than $R_{horizontal}$. Consequently, nodes x and y can not communicate directly with each other. In addition $\max(R_{horizontal} = 1.05, R_{vertical} \approx 1.32) < \sqrt{2}$, so no pair of linkage nodes can communicate directly with each other, as required.

Given a subgraph $G' \subseteq G$ of operating nodes obtained after an event of random failure, we map G' to a configuration C' of the TCD problem as follows: if node x is operating in G' then node x lies in x^{in} in C'. Else (if x is failed) then node x lies in region x^{out} in C'. One may verify that the above transformation satisfies property **(a)** above.

We now show that it satisfies condition **(b)**. Suppose G' has an operating (s, t)-path $(s, a, b, c, \cdots, t)$ then $(s, v_{s,a}, a, v_{a,b}, b, v_{b,c}, c, \cdots, t)$ is a path that makes the maximum deficiency of configuration C' zero, as required. On the other hand, if G' has no operating (s, t)-path then no path between s and t exists in configuration C' since no simple path can pass through a node x located in the x^{out} region. This completes the proof. ■

VI. Concluding Remarks

Practical deployments of WSNs may introduce node placement uncertainty in the after-deployed networks, and consequently, deviations from the planned network behaviour. Our work here introduces an approach for assessing the likelihood that the deployed network will continue to perform as required. The approach is independent of the shapes and probability distributions describing the uncertainty in the placement process. The presented simulation results show the usefulness of the approach in evaluating network deployment schemes.

References

[1] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad-hoc sensor network," *Computer Communications*, vol. 29, pp. 413–420, 2006.

[2] A. Ghosh and S. K. Das, "Coverage and connectivity issues in wireless sensor networks: A survey," *Pervasive and Mobile Computing*, vol. 4, pp. 303–334, 2008.

[3] B. Wang, "Sensor activity scheduling," in *Coverage Control in Sensor Networks*, ser. Computer Communications and Networks, A. J. Sammes, Ed. Springer London, 2010, pp. 121–153.

[4] S. S. Iyengar, A. Tandon, Q. Wu, E. Cho, N. S. V. RAO, and V. Vaishnavi, "Deployment of sensors: an overview," in *Distributed Sensor Networks*, S. S. Iyengar and R. R. Brooks, Eds. CRC Press, Boca Raton, Florida, 2005, pp. 483–503.

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

[6] S. Kumar, T. H. Lai, and J. Balogh, "On k-coverage in a mostly sleeping sensor network," *Wireless Networks*, vol. 14, no. 3, pp. 277–294, Jun. 2008.

[7] P. Brass, "Bounds on coverage and target detection capabilities for models of networks of mobile sensors," *ACM Transactions on Sensor Networks*, vol. 3, no. 2, Jun. 2007.

[8] D. Zorbas, D. Glynos, P. Kotzanikolaou, and C. Douligeris, "Solving coverage problems in wireless sensor networks using cover sets," *Ad Hoc Networks*, vol. 8, no. 4, pp. 400 – 415, 2010.

[9] S. Megerian, F. Koushanfar, G. Qu, G. Veltri, and M. Potkonjak, "Exposure in wireless sensor networks: theory and practical solutions," *Wireless Network*, vol. 8, pp. 443–454, September 2002.

[10] T. Clouqueur, V. Phipatanasuphorn, P. m. Ramanathan, and K. K. Saluja, "Sensor deployment strategy for detection of targets traversing a regi on," *Mob. Netw. Appl.*, vol. 8, pp. 453–461, August 2003.

[11] K. Xu, G. Takahara, and H. S. Hassanein, "On the robustness of grid-based deployment in wireless sensor networks," in *International Conference on Wireless Communications and Mobile Computing*, 2006, pp. 1183–1188.

[12] F. M. Al-Turjman, H. S. Hassanein, and M. A. Ibnkahla, "Quantifying connectivity of grid-based Wireless Sensor Networks under practical errors," in *IEEE Conference on Local Computer Networks*, 2010, pp. 220–223.

[13] J. Wang, R. K. Ghosh, and S. K. Das, "A survey on sensor localization," *Journal of Control Theory and Applications*, vol. 8, pp. 2–11, 2010.

[14] A. Boukerche, H. Oliveira, E. Nakamura, and A. Loureiro, "Localization systems for wireless sensor networks," *IEEE Wireless Communications*, vol. 14, no. 6, pp. 6–12, december 2007.

[15] H. AboElFotoh and C. Colbourn, "Computing 2-terminal reliability for radio-broadcast networks," *IEEE Transactions on Reliability*, vol. 38, no. 5, pp. 538–555, December 1989.