An Iterative Method for Strong Barrier Coverage under Practical Constraints

Xiaoyun Zhang, Mathew L. Wymore, and Daji Qiao
Iowa State University, Ames, IA 50011
{zxydut, mlwymore, daji}@iastate.edu

***Abstract*—Barrier coverage is a fundamental application for wireless sensor networks. In this paper, we consider a practical probabilistic sensing model and propose an iterative scheme, called BaCo, to provide strong barrier coverage under this model, with the objective of minimizing the number of active sensors. Moreover, we build the barrier under practical constraints of minimum detection probability and maximum false alarm probability. We use simulations to show that BaCo converges quickly and achieves better results than previous work while also bounding the system false alarm probability.**

I. Introduction

Intruder detection and border surveillance are intuitive applications of sensor networks. In these applications, sensors are deployed in a long belt region such that no intruder can cross the belt without being detected, a scenario referred to as *barrier coverage* in sensor networks.

The barrier coverage problem has attracted great interest from researchers in the last decade [1]–[8]. Kumar defined *weak* and *strong* barrier coverage in [1]. In weak barrier coverage, an intruder is assumed to only follow the shortest straight paths to cross the belt. In strong barrier coverage, an intruder can take any path. Since strong barrier coverage produces a more complete barrier, many works [2], [3], [5]–[8], including this paper, focus on strong barrier coverage.

The majority of the work on barrier coverage has adopted the simple disk sensing model [1]–[3], [6]–[8], in which a target is considered detected simply if it enters a detection radius. This model makes the barrier coverage problem easier, but it does not reflect the sensing behavior of many real-world sensors. In contrast, in the probabilistic sensing model proposed in [9]–[11], a target is detected by a sensor with a probability between 0 and 1, depending on factors such as distance and noise. This approximates real-world sensing behavior much better. In this paper, we adopt the probabilistic model.

Another benefit of the probabilistic model is that it allows data fusion among sensors [9], [12]–[14]. When data fusion is employed, the detection probability for a target is calculated by either fusing the raw readings (*value fusion*) or decisions (*decision fusion*) of sensors. Data fusion can increase the detection probability of a target, or, from another perspective, it can expand the coverage region of sensors [12], [13]. We employ decision fusion, because it is relatively light-weight compared to value fusion. In addition to fusing the decisions of different sensors, we also fuse the decisions at a sequence of sampling points along the intruder's path.

We also define and consider a system false alarm probability for our practical setup. This consideration of false alarm probability distinguishes our work from other works such as [5], where only the probability of detecting a target is considered. Dealing with the detection probability without considering the false alarm probability makes little sense under the probabilistic sensing model, as we can simply lower the alarm threshold of sensors to get a higher detection probability, which may result in an unacceptable false alarm probability.

To maximize cost-effectiveness and energy efficiency, we seek the minimum number of sensors for building a barrier. Given this goal, to address the strong barrier coverage problem under the practical constraints of minimum detection probability and maximum false alarm probability, we propose a novel iterative algorithm, called BaCo, to identify a minimal set of active sensors from a given deployment to build a barrier. BaCo adjusts the alarm threshold of sensors iteratively to find a compromise between detection probability and system false alarm probability.

The paper is organized as follows: Section II presents the probabilistic model and the problem statement. Section III describes the proposed scheme. Section IV presents results from our evaluation of BaCo. Finally, Section V discusses the related work, and Section VI concludes the paper.

II. Model and Problem Statement

A. System Model

We consider a network of N sensors randomly and uniformly deployed to monitor a long rectangular region with two parallel sides: an *entrance side* and a *destination side*. The size of the region is ℓ (length) by h (width). Let $\mathcal{S}$ denote the set of N sensors. We assume that sensors in $\mathcal{S}$ know their locations in the region and that they have an identical communication range R_c. We also assume the sensors have a finite sampling rate f and are synchronized in their sensing activities. An intruder, or *target*, may take any path traversing the region from the *entrance side* to the *destination side*. A target is assumed to move continually at its maximum speed $v_{\max}$ in order to minimize the probability of being detected.

B. Sensing Model

We use a probabilistic sensing model, in which sensor readings are affected by randomly varying noise and sensor nodes use a decision threshold to determine if an intruder is present or not. The model consists of a source model, a detection model, and a false alarm model.

1) Source Model: We assume either the target or its motion produces a physical signal, such as sound, electromagnetic waves, or vibrations. We assume the strength of the signal decays according to the power law, meaning that if the target is at point t, the signal strength at the location of sensor s_i is [12], [14]:

$$\omega_i(t) = \frac{\Omega}{1 + \big(d(s_i, t)\big)^{\alpha}}, \tag{1}$$

where Ω is the signal amplitude at the target, α is a known decay exponent, and $d(\cdot,\cdot)$ denotes the distance between two points.

2) Detection Model: We assume that background noise affects sensor readings. When a target is present at point t, a sensor s_i observes a signal $\mathbf{x}_i$ that depends on (1) and the background noise n, as follows:

$$\mathbf{x}_i = \omega_i(t) + n. \tag{2}$$

When no target is present, $\mathbf{x}_i = n$. Let $F_N(n)$ denote the cumulative distribution function of noise, and assume that it is identical and independent for all sensors. We also assume that F_N is known by the base station.

To detect a target, sensors use a decision threshold T. When a sensed reading exceeds T, the sensor generates an alarm to report the presence of a target. Therefore, given T, the probability that sensor s_i detects a target at point t is:

$$P_d(s_i,t) = 1 - F_N\big(T - \omega_i(t)\big). \tag{3}$$

We apply the "OR" rule to fuse the decisions made by all active sensors. Under the "OR" rule, a target is said to have been detected if at least one active sensor reports its presence. Thus, given S_A, the set of active sensors, the overall probability of detecting a target at a point t is:

$$P_{D,t} = 1 - \prod_{s_i \in S_A} \big(1 - P_d\left(s_i,t\right)\big). \tag{4}$$

For the purpose of detection, a target's intruding path φ is composed of a set Q of discrete, evenly-spaced points q_j that correspond to the points on φ where the target is when the sensors take samples. The target only needs to be detected at one q_j, so we apply the "OR" rule over all $q_j \in Q$ as well. Thus, given Q, the probability of detecting a target traveling along φ is:

$$P_{D,\varphi} = 1 - \prod_{q_j \in Q} (1 - P_{D,q_j}), \tag{5}$$

where P_{D,q_j} is calculated according to (4).

Given an intruding path φ, the set Q depends on the target's maximum speed $v_{\max}$, the sensors' sampling rate f, and the sampling phase relative to the arrival of the target at the entrance side. Since we want to place a lower bound on the probability of detection, we are interested in the Q that yields $P^l_{D,\varphi}$, the minimum detection probability for a target taking the path φ. For a given f and $v_{\max}$, we define $P^l_{D,\varphi}$ as follows:

$$P^l_{D,\varphi} = \min_Q P_{D,\varphi}, \tag{6}$$

where the choice of points $q_j \in Q$ is constrained as described above.

We extend this minimum detection probability concept to all possible intruding paths and define P_D as the system's overall minimum detection probability, as follows:

$$P_D = \min_\varphi P^l_{D,\varphi} = \min_\varphi \min_Q \big(1 - \prod_{q_j \in Q} \prod_{s_i \in S_A} \big(1 - P_d\left(s_i,q_j\right)\big)\big). \tag{7}$$

3) False Alarm Model: Due to excessive noise, a sensor may generate an alarm and report the presence of a target when no target is present. This type of alarm is called a *false alarm*. The probability of false alarms should be bounded in order to avoid burdening the end user. For each sample taken, the probability of a particular sensor generating a false alarm is:

$$P_f = 1 - F_N\left(T\right). \tag{8}$$

Since we use the "OR" rule for target detection, a single sensor reporting a false alarm for any given sample constitutes a system false alarm. Therefore, we define the system false alarm probability P_F as the probability that any sensor produces a false alarm for a particular sample, as follows:

$$P_F = 1 - (1 - P_f)^{|S_A|}, \tag{9}$$

where $|S_A|$ is the total number of active sensors. This definition of P_F is consistent with the system false alarm probability defined in [4] and [12] and the network false alarm rate in [14].

C. Problem Statement

To summarize, we define strong $(P_D^{\min}, P_F^{\max})$-barrier coverage under the probabilistic models presented above as follows: strong $(P_D^{\min}, P_F^{\max})$-barrier coverage is achieved if and only if

1) the system's minimum probability of detecting a target taking any intruding path is at least $P_D^{\min}$, and
2) the system false alarm probability is at most $P_F^{\max}$.

In this paper, we study how to achieve strong $(P_D^{\min}, P_F^{\max})$-barrier coverage with the minimum number of sensors from a given set $\mathcal{S}$ of static sensors in an $\ell \times h$ region, given R_c, f, and $v_{\max}$. We seek the minimum number of sensors for cost-effectiveness and energy efficiency. Formally, our problem is to minimize $|S_A|$, subject to $P_D \geq P_D^{\min}$, $P_F \leq P_F^{\max}$, and $S_A \subseteq \mathcal{S}$.

D. Transformed Problem

In order to simplify the calculation of detection probability along a path, we transform the problem by adopting the concept of *detection gain* introduced in [5]. The detection gain $\mathcal{G}(p)$ associated with a probability p is defined as follows:

$$\mathcal{G}(p) = -\log(1-p). \tag{10}$$

$\mathcal{G}(p)$ is a monotonically increasing function of p, with $\mathcal{G}(0) = 0$ and $\mathcal{G}(1) = \infty$.

We apply the gain concept by first substituting (4) into (5) and rearranging to obtain:

$$1 - P_{D,\varphi} = \prod_{q_j \in Q} \prod_{s_i \in S_A} \big(1 - P_d\left(s_i,q_j\right)\big). \tag{11}$$

By applying the log function to both sides of (11), we obtain an expression for $\mathcal{G}_\varphi$, the total detection gain for a target that takes intruding path φ, as follows:

$$\mathcal{G}_\varphi = \sum_{q_j \in Q} \sum_{s_i \in S_A} \mathcal{G}\left(s_i,q_j\right), \tag{12}$$

where $\mathcal{G}\left(s_i,q_j\right)$ is the detection gain of sensor s_i on a target located at q_j. We then define $\mathcal{G}_D$ as the minimum detection gain for all φ, analogous to our definition of P_D, as follows:

$$\mathcal{G}_D = \min_\varphi \min_Q \big(\sum_{q_j \in Q} \sum_{s_i \in S_A} \mathcal{G}\left(s_i,q_j\right)\big). \tag{13}$$

We also define $\mathcal{G}_D^{\min} = \mathcal{G}\left(P_D^{\min}\right)$. Then our equivalent transformed problem is to minimize $|S_A|$, subject to $\mathcal{G}_D \geq \mathcal{G}_D^{\min}$, $P_F \leq P_F^{\max}$, and $S_A \subseteq \mathcal{S}$. The following section presents a practical method for obtaining a best-effort feasible solution to this problem.

III. Proposed Scheme

In this section, we present BaCo's iterative design that allows it to achieve strong $(P_D^{\min}, P_F^{\max})$-barrier coverage while minimizing the number of active sensors. The main idea of BaCo is to first assume a number of active sensors N_A, which is used to set the decision threshold T. Then, given that T, we check whether strong $(P_D^{\min}, P_F^{\max})$-barrier coverage can be achieved with N_A sensors. If not, we update our assumption for N_A and iterate.

BaCo is divided into four modules, shown in Fig. 1. The setup module takes N_A as input and provides T as output. The mapping and solution modules then identify a minimized set of active sensors S_A that satisfies the $P_D^{\min}$, or equivalently, $\mathcal{G}_D^{\min}$, constraint. The iteration controller either terminates the algorithm or starts the next iteration, depending on whether the $P_F^{\max}$ constraint is met by S_A.

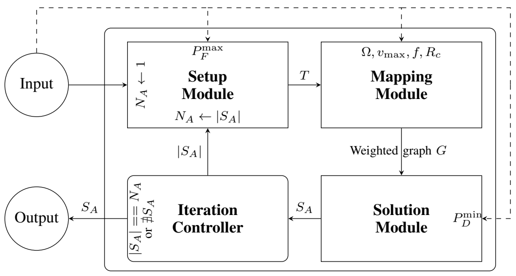


Fig. 1. Overview of BaCo. The dashed lines indicate parameter entry. Inputs are labeled with the action taken upon receiving the input. Outputs are labeled with any applicable decision criteria.

A. Setup Module

The setup module calculates the decision threshold T using two inputs, N_A and $P_F^{\max}$. In the first iteration, we set $N_A = 1$, starting with a small N_A because we want to minimize $|S_A|$, the size of the set of active sensors. For all other iterations, N_A is set to the $|S_A|$ found in the previous iteration. This influences T, as follows.

According to (9), to satisfy $P_F \leq P_F^{\max}$, we need

$$P_f \leq 1 - (1 - P_F^{\max})^{1/N_A} . \tag{14}$$

Using (8), we then have

$$T = F_N^{-1}(1 - P_f) \geq F_N^{-1}\left((1 - P_F^{\max})^{1/N_A}\right). \tag{15}$$

According to (3), to maximize the probability of detection, we minimize T; therefore, we use

$$T = F_N^{-1}\left((1 - P_F^{\max})^{1/N_A}\right) \tag{16}$$

as the output of the setup module.

B. Mapping Module

The mapping module maps the sensor network to an undirected weighted graph G, which consolidates the detection gain and the network connectivity information. As shown in Fig. 1, the mapping module takes T as input, as well as several of the system-related parameters introduced in Section II. These inputs determine the edges and edge weights of G. Fig. 2 provides an example of the mapping procedure, which is composed of the steps described below.

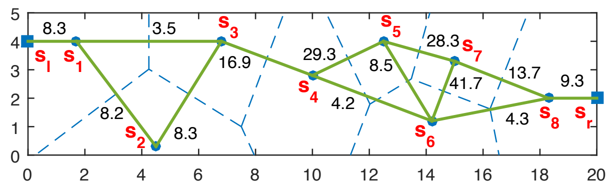


Fig. 2. Example of the mapping procedure. Eight sensors are deployed in a 20×5 m region. The inputs are $T = 1.64$ mW, $\Omega = 10$ mW, $v_{\max} = 1$ m/s, $f = 2$ Hz, and $R_c = 6$ m. Sensors s_l and s_r are virtual sensors which represent the left and right boundary of the monitored region, respectively. The dashed lines are the Voronoi diagram of the sensors. The solid lines between sensors are the edges of G, labeled with their weights.

1) Vertex Identification: The vertices in G include (a) all physical sensors in $\mathcal{S}$, and (b) two virtual sensors s_l and s_r, which represent the left and right boundary of the monitored region, respectively.

2) Edge Identification: The edges of G are all the edges of the Delaunay triangulation of $\mathcal{S}$ whose lengths are shorter than the communication range R_c. The Delaunay triangulation is used because, according to the conclusion in [5], from all the possible intruding paths, the path with the minimum detection gain is composed of Voronoi edges. Each edge in G thus corresponds to a section of a possible worst-case intruding path.

If a section of the left or right boundary is contained within the Voronoi cell of a physical sensor $s_i \in \mathcal{S}$, and s_i is within R_c of the boundary, then an edge between s_i and a virtual sensor is added to G.

3) Weight Assignment: The weight of the edge $s_i s_j$ in G is the minimum accumulative detection gain of s_i and s_j for a target traveling along the Voronoi edge between s_i and s_j, $\mathrm{Vor}(s_i, s_j)$. According to [5], when an intruder travels along the perpendicular bisector of the line segment $s_i s_j$, sensors s_i and s_j will have the minimum accumulative detection gain if the sampling points

- are symmetrically distributed on the two sides of the line segment $s_i s_j$, and
- the distance between two adjacent sampling points is $v_{\max}/f$.

These requirements are illustrated in Fig. 3, where the crosses show the worst-case sampling points for a target traveling between s_3 and s_4 in the example in Fig. 2. The points are $v_{\max}/f$ apart, and they are symmetrically distributed on either side of the line segment $s_3 s_4$.

For an intruder traveling between a physical sensor s_i and a virtual sensor s_l or s_r, the worst-case sampling points are found along the section of boundary that is within s_i's Voronoi cell, and they are symmetrically distributed on either side of the horizontal line between s_i and the boundary.

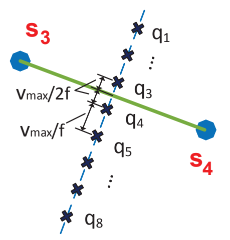


Fig. 3. The worst-case sampling points for a target traveling between s_3 and s_4. The crosses represent the sampling points and the dashed line is the Voronoi edge between the two sensors.

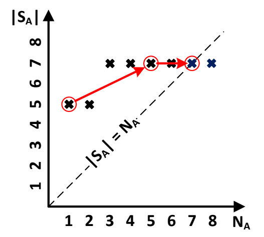


Fig. 4. (Referenced in Section III-D) Illustration of iteration progress for the example shown in Fig. 5. The crosses show the output $|S_A|$ value corresponding to each input N_A. The arrows show the progression of the three iterations from Fig. 5. Only the circled crosses are checked by BaCo.

Once the worst-case sampling points are identified, we can obtain the minimum accumulative detection gain of s_i and s_j on an intruder traveling along $\text{Vor}(s_i, s_j)$. We use this value as w_{ij}, the weight of the edge $s_i s_j$ in G. From (12), we have

$$w_{ij} = \sum_{k=1}^{m} \big(\mathcal{G}(s_i, q_k) + \mathcal{G}(s_j, q_k)\big), \tag{17}$$

where q_k is a sampling point, m is the number of sampling points along $\text{Vor}(s_i, s_j)$, and $\mathcal{G}(s_i, q_k)$ and $\mathcal{G}(s_j, q_k)$ are the detection gains of s_i and s_j on q_k. Note that only the detection gains of s_i and s_j are considered, while in reality, other sensors may also provide detection gain for an intruder traveling along $\text{Vor}(s_i, s_j)$. Therefore, w_{ij} is a lower bound of the actual detection gain from all $s_i \in \mathcal{S}$.

C. Solution Module

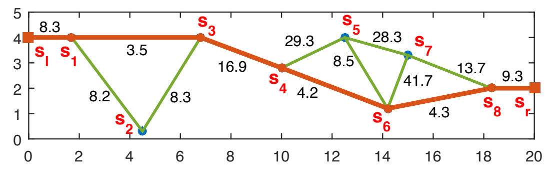
Given the weighted graph G, the solution module finds a minimum set of sensors whose minimum detection gain for any intruding path is larger than $\mathcal{G}_D^{\min}$. With the edge weight defined in the mapping module as the capacity of each edge, the minimum detection gain $\mathcal{G}_D$ of the system for any intruding path, assuming that all sensors are active, is equal to the maximum flow from s_l to s_r in G. Therefore, our goal is to find a minimum subset of sensors in $\mathcal{S}$ whose maximum flow is larger than $\mathcal{G}_D^{\min}$. However, selecting the minimum number of sensors in a graph which can deliver a certain amount of flow is NP-hard [5]. Therefore, in BaCo, we use a two-phase heuristic solution.

1) Phase 1: Prune all edges in G whose weights are no more than $\mathcal{G}_D^{\min}$ and call the resulting graph $\tilde{G}$. Run Dijkstra's algorithm on $\tilde{G}$ to find the shortest path, in terms of number of hops, from s_l to s_r. If a path is found, then S_A is composed of the physical sensors on that path, and the solution module is done. Otherwise, continue to Phase 2.

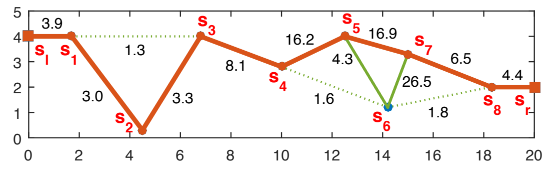
2) Phase 2: If Dijkstra's algorithm cannot find a path, then s_l is disconnected from s_r in $\tilde{G}$, meaning that no single path can deliver $\mathcal{G}_D^{\min}$ flow from s_l to s_r. In this case, we look for a flow network that can deliver $\mathcal{G}_D^{\min}$ flow by running the maximum-flow based algorithm proposed in [5] on G. Briefly, in this algorithm, the edge weight in G becomes the capacity of each edge, and the algorithm heuristically searches for an S_A which can deliver at least $\mathcal{G}_D^{\min}$ flow. The algorithm attempts to minimize the number of nodes in the flow network, but the solution found is likely sub-optimal.

We try Dijkstra's algorithm prior to the max-flow based algorithm because a solution with a single path tends to use less sensors than a solution with multiple branches, due to the sub-optimal nature of Phase 2's algorithm. This intuition is verified by simulation. However, we include Phase 2 as a backup, because when Phase 1 fails due to the pruning operation disconnecting the graph, the max-flow based algorithm of Phase 2 may still produce a solution. If Phase 2 does not produce a solution, then for the purposes of BaCo, S_A does not exist.

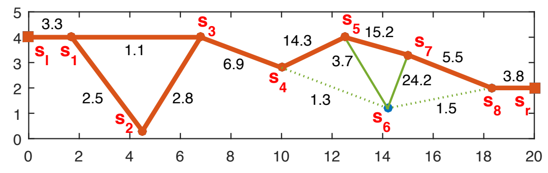
Fig. 5 illustrates the heuristic two-phase algorithm with three example graphs (see the next section for an explanation of these graphs as iterations). In these examples, $\mathcal{G}_D^{\min} = 3$, which corresponds to $P_D^{\min} = 0.95$. In Fig. 5(a), $\tilde{G}$ is identical to G, as all the edge weights are larger than $\mathcal{G}_D^{\min}$. Dijkstra's algorithm finds a shortest path in $\tilde{G}$ that yields $S_A = \{s_1, s_3, s_4, s_6, s_8\}$, so the solution module does not run Phase 2. In Fig. 5(b), three edges are pruned in Phase 1, but Dijkstra's algorithm still works, so Phase 2 is again not used. In the graph in Fig. 5(c), edges $s_1 s_3$, $s_1 s_2$, $s_2 s_3$, $s_4 s_6$ and $s_6 s_8$ are all pruned in Phase 1, disconnecting s_l from s_r. Therefore, this graph requires Phase 2, which produces the solution shown in the figure, $S_A = \{s_1, s_2, s_3, s_4, s_5, s_7, s_8\}$.



(a) Iteration 1: $N_A = 1$, $S_A = \{s_1, s_3, s_4, s_6, s_8\}$, $|S_A| = 5$.



(b) Iteration 2: $N_A = 5$, $S_A = \{s_1, s_2, s_3, s_4, s_5, s_7, s_8\}$, $|S_A| = 7$.



(c) Iteration 3: $N_A = 7$, $S_A = \{s_1, s_2, s_3, s_4, s_5, s_7, s_8\}$, $|S_A| = 7$.

Fig. 5. An illustration of iterations in BaCo. The minimum detection gain $\mathcal{G}_D^{\min} = 3$, which corresponds to $P_D^{\min} = 0.95$, and $P_F^{\max} = 0.05$. The thick edges compose the path or flow network which can deliver $\mathcal{G}_D^{\min}$ flow. The dotted edges are pruned in $\tilde{G}$.

D. Iteration Controller

To understand the iteration controller, we first give an overview of BaCo's iterations. In the first iteration, $N_A = 1$. At the end of any iteration, if $|S_A| == N_A$, or if S_A does not exist, we terminate. Otherwise, we set $N_A = |S_A|$ and iterate. Given these rules, we have the following property.

Theorem 1. Let S_A be the output of the solution module, given N_A as the input of the setup module. In each iteration of BaCo, if S_A exists, then $|S_A| \geq N_A$.

Proof: We will prove this with induction. Let $N_A^{(k)}$ and $S_A^{(k)}$ denote the input and output of iteration k, respectively. Let $G^{(k)}$ denote the graph created in iteration k. In BaCo, $N_A^{(1)} = 1$, so we have the following base case.

Base case: $|S_A|^{(1)} \geq N_A^{(1)} = 1$. This is obvious, because if a solution exists, it must have at least one sensor.

Inductive step: If $|S_A|^{(k-1)} \geq N_A^{(k-1)}$, then $|S_A|^{(k)} \geq N_A^{(k)}$ for $k > 1$. This is true because, in BaCo, if we do not terminate in iteration $k-1$, then we set $N_A^{(k)} = |S_A|^{(k-1)}$. Therefore,

$$N_A^{(k)} = |S_A|^{(k-1)} \geq N_A^{(k-1)}. \tag{18}$$

From (16), we can then conclude that $T^{(k)} \geq T^{(k-1)}$. Next, from (3) and (10), we know that a higher T for a sensor s_i leads to a lower detection probability and gain for s_i, given an intruder at any point t:

$$\begin{aligned} T^{(k)} \geq T^{(k-1)} &\Rightarrow P_d^{(k)}(s_i,t) \leq P_d^{(k-1)}(s_i,t) \\ &\Rightarrow \mathcal{G}^{(k)}(s_i,t) \leq \mathcal{G}^{(k-1)}(s_i,t),\ \forall i, \forall t. \end{aligned} \tag{19}$$

From (17), we see that this leads to the weight of any particular edge in the graph $G^{(k)}$ being lower than the weight of the corresponding edge in $G^{(k-1)}$:

$$\left.\begin{aligned} \mathcal{G}^{(k)}(s_i,t) \leq \mathcal{G}^{(k-1)}(s_i,t) \\ \mathcal{G}^{(k)}(s_j,t) \leq \mathcal{G}^{(k-1)}(s_j,t) \end{aligned}\right\} \Rightarrow w_{ij}^{(k)} \leq w_{ij}^{(k-1)}. \tag{20}$$

Consequently, the set of sensors $S_A^{(k)}$ which can deliver $\mathcal{G}_D^{\min}$ amount of flow in $G^{(k)}$ can also deliver at least $\mathcal{G}_D^{\min}$ amount of flow in $G^{(k-1)}$. However, there may exist a better solution in $G^{(k-1)}$, because its edges can deliver more flow. Therefore, $|S_A|^{(k)} \geq |S_A|^{(k-1)}$, and since $N_A^{(k)} = |S_A|^{(k-1)}$ according to our iteration rule, we have

$$|S_A|^{(k)} \geq |S_A|^{(k-1)} = N_A^{(k)}. \qquad \blacksquare$$

Given this property, we discuss the iteration controller in more detail. The iteration controller takes S_A as input from the solution module and decides if another iteration is required, according to the relationship between $|S_A|$ and N_A, as follows.

1) $|S_A| = N_A$: Terminate and output S_A. The assumption for N_A has been validated, meaning that S_A is a feasible solution because it meets the requirements for both $\mathcal{G}_D$ (from the solution module) and P_F (from the setup module). Note $|S_A| = N_A$ means $P_F = P_F^{\max}$. Furthermore, S_A is the best feasible solution that can be found by BaCo, because N_A is the smallest valid assumption. Thus, the iteration controller terminates the algorithm and outputs S_A. Note that $|S_A| < N_A$ also means that S_A is a feasible solution. However, this case will not occur in BaCo, as demonstrated by Theorem 1.

2) S_A does not exist: Terminate. The solution module could not find an S_A that satisfies $\mathcal{G}_D^{\min}$. Further iterations would also not produce a solution, because S'_A would not exist for any $N'_A > N_A$, as all the edge weights in G' would be less than the edge weights in G, using reasoning similar to that of the proof of Theorem 1. This means that BaCo cannot find a solution for strong $(P_D^{\min}, P_F^{\max})$-barrier coverage with the given sensor deployment.

3) $|S_A| > N_A$: Set $N_A = |S_A|$ and iterate. The assumed N_A has not been validated and the solution S_A violates the $P_F^{\max}$ constraint. The iteration controller outputs $|S_A|$ to the setup module, which sets $N_A = |S_A|$ and starts the next iteration. Thus, BaCo does not exhaustively try all values of N_A. The proof of correctness for skipping the values between N_A and $|S_A|$ is detailed as follows.

Theorem 2. For any $N'_A \in [N_A, |S_A|)$, the solution module cannot find a set of sensors S'_A which satisfies $P'_F \leq P_F^{\max}$ and $\mathcal{G}'_D \geq \mathcal{G}_D^{\min}$, where $\mathcal{G}'_D$ and P'_F are the minimum detection gain and the system false alarm probability of S'_A.

Proof: Suppose there exists an N'_A with $N_A \leq N'_A < |S_A|$ that creates the graph G' and produces a feasible solution S'_A, meaning that $N'_A \geq |S'_A|$.

Using reasoning similar to that of the proof of Theorem 1, we know that

$$N'_A \geq N_A \Rightarrow w'_{ij} \leq w_{ij}. \tag{21}$$

Therefore, any solution S'_A that is feasible on G' is also feasible on G, but since G's edges can deliver more flow, there may exist a better solution in G. Thus, $|S'_A| \geq |S_A|$. Combining this with the assumption that S'_A is a feasible solution for the input $N'_A < |S_A|$, we have

$$|S_A| > N'_A \geq |S'_A| \geq |S_A|, \tag{22}$$

which is a contradiction. $\blacksquare$

An example of BaCo's iterations is shown in Fig. 5. Fig. 4 illustrates the iteration progress for this example, showing $|S_A|$ versus N_A. The algorithm terminates the first time it finds a solution on the line $|S_A| = N_A$, and since we start with $N_A = 1$ and $|S_A|$ increases with each iteration (Theorem 1), BaCo thus attempts to minimize $|S_A|$. The circled crosses and arrows in Fig. 4 show the iterations in Fig. 5, starting with $N_A = 1$ in the first iteration. This iteration outputs $|S_A| = 5$, which becomes the new N_A for the second iteration. The second iteration yields $|S_A| = 7$ and with an input of $N_A = 7$, the third iteration yields $|S_A| = 7$, and the algorithm terminates.

IV. Evaluation

In this section, we evaluate the importance of considering system false alarm probability, the performance of BaCo in terms of the number of active sensors, and BaCo's convergence speed. In our simulations, 200 sensors are randomly deployed in a 100 × 10 m belt region, similar to the simulation setups in other barrier coverage papers [1]–[8]. The default simulation parameters are shown in Table I.

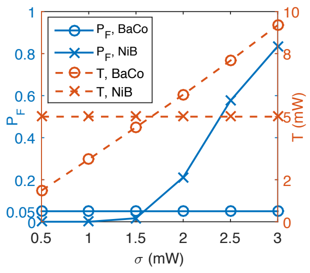
TABLE I
Default Simulation Parameters

Parameter	Meaning	Default Value
$P_D^{\min}$	Detection probability constraint	0.95
$P_F^{\max}$	System false alarm probability constraint	0.05
R_c	Communication range	20 m
$v_{\max}$	Target's maximum moving speed	1 m/s
Ω	Source signal strength	30 mW
α	Source signal decay exponent	2
f	Sensor sampling rate	5 Hz
F_N	CDF of noise distribution	CDF of Gaussian
μ	Noise mean	0 mW
σ	Noise standard deviation	1 mW

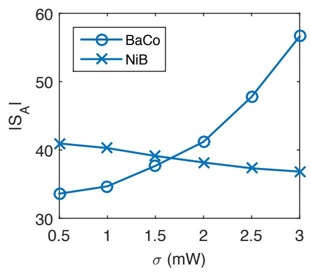
A. System False Alarm Probability

We first demonstrate the importance of considering system false alarm probability P_F when designing a scheme. We compare BaCo to a non-iterative version of itself, which we call *NiB*, that is based on MWBA [5]. NiB uses BaCo's source and detection model, but like MWBA, it does not consider system false alarm probability P_F. Therefore, it uses a fixed decision threshold T and only runs BaCo's solution algorithm once, selecting a set of active sensors S_A that satisfies the detection probability constraint $P_D \geq P_D^{\min}$. The P_F for NiB is then calculated according to (8) and (9).

Fig. 6(a) shows P_F and the decision threshold T versus σ, the standard deviation of the background noise, for BaCo and NiB. By design, BaCo's P_F is constant at $P_F^{\max} = 0.05$. To achieve this, BaCo automatically increases T as σ increases, effectively dealing with the larger fluctuations in noise. In contrast, NiB's P_F grows quickly with σ, reaching over 0.2 when $\sigma = 2$ mW. The P_F of NiB is lower than that of BaCo when σ is small, but as a tradeoff, NiB's number of active sensors $|S_A|$ is larger at small σ, as can be seen in Fig. 6(b). BaCo balances this tradeoff using its iterative algorithm, achieving smaller $|S_A|$ when the P_F constraint allows, and sacrificing $|S_A|$ for P_F when needed.



(a) P_F and T vs. σ



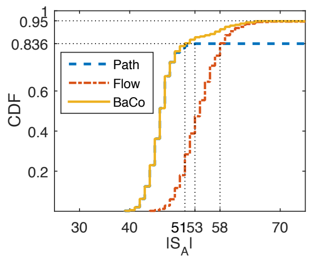
(b) $|S_A|$ vs. σ

Fig. 6. Comparison of schemes with and without the P_F constraint.

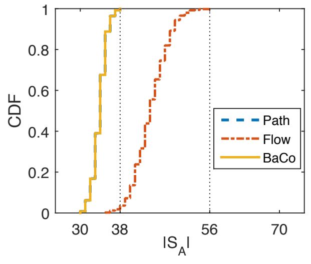
B. Number of Active Sensors

We now evaluate BaCo's performance in terms of the number of active sensors $|S_A|$ versus the source signal strength Ω and the sampling frequency f. Since BaCo's performance cannot be fairly compared to schemes without the P_F constraint, we compare BaCo to two reduced versions of itself: *Path*, in which the solution module utilizes only the shortest-path based algorithm (Phase 1 of the solution module), and *Flow*, in which the solution module utilizes only the flow-based algorithm of [5] (Phase 2 of the solution module). The full BaCo scheme utilizes both, as described in Section III-C. The results are collected from 500 runs of each scheme, with random deployments for each run. If a scheme cannot achieve barrier coverage for a run, an $|S_A|$ of ∞ is recorded.

1) The Effect of Signal Strength Ω: Fig. 7(a) shows the CDF of $|S_A|$ when $\Omega = 12$ mW, a weak source signal, and Fig. 7(b) shows the CDF of $|S_A|$ when $\Omega = 50$ mW, a strong source signal. The curve of BaCo overlaps that of the Path scheme for $|S_A| < 51$ in Fig. 7(a) and completely in Fig. 7(b). Note the left shift in the CDFs between Fig. 7(a) and Fig. 7(b), indicating that $|S_A|$ is smaller when the source signal is stronger. This is because, with a stronger signal, each sensor has a higher probability of detecting the target, so fewer sensors are needed.



(a) $\Omega = 12$ mW



(b) $\Omega = 50$ mW

Fig. 7. CDFs of $|S_A|$ for two Ω values.

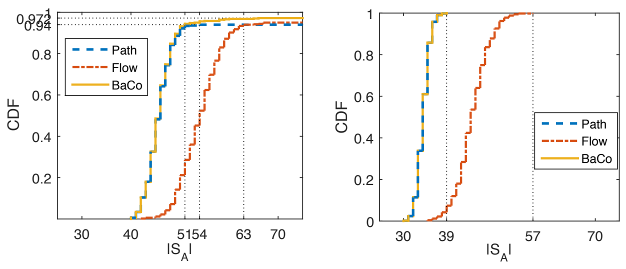
When Ω is low (Fig. 7(a)), the Path scheme can only obtain a feasible solution 83.6% of the time, while the other two schemes can achieve coverage 95% of the time. This is because the Path scheme limits the search space to a single path. However, when the Path scheme does produce a feasible solution, it tends to use less sensors than the Flow scheme. At the highest percentile for which the Path scheme produces a feasible solution, BaCo uses 51 sensors, the Path scheme uses 53 sensors, and the Flow scheme uses 58 sensors. Thus, BaCo achieves the performance of Path and the coverage percentile of Flow by combining the two.

When Ω is high (Fig. 7(b)), all schemes are able to achieve coverage in 100% of the runs. The Flow scheme activates more sensors than both BaCo and the Path scheme, due to the sub-optimal nature of the flow-based algorithm. BaCo and the Path scheme use at most 38 sensors, while the Flow scheme uses at most 56 sensors. Therefore, BaCo performs well with both large and small Ω values.

2) The Effect of Sampling Rate f: Fig. 8(a) shows the CDF of $|S_A|$ when $f = 0.5$ Hz and Fig. 8(b) shows the CDF of $|S_A|$ when $f = 10$ Hz. Again, BaCo largely overlaps the Path scheme. Comparing the two figures, we see that the number of active sensors of all three schemes is smaller for the higher sampling frequency. This is because a higher sampling frequency gives each sensor more chances to detect the target, so fewer sensors are required to achieve the same P_D.

When f is low (Fig. 8(a)), BaCo and the Flow scheme achieve a higher coverage percentile than the Path scheme. The BaCo scheme achieves coverage in 97.2% of the runs, with the Path scheme at 94% and the Flow scheme between the two. The lower coverage percentile of Flow than BaCo implies that the Path scheme (Phase 1 of BaCo) sometimes finds a valid solution when the Flow scheme does not. This can happen because the Flow scheme generally finds a solution with a larger $|S_A|$ in each iteration. Then in the next iteration the Flow scheme may not be able to find a feasible solution using the larger $|S_A|$ as N_A, because the threshold T will be higher and the gains will be lower. We again see that BaCo performs the best in terms of $|S_A|$ and that, in the cases where the Path scheme achieves coverage, the Path scheme uses less sensors than the Flow scheme. Therefore, BaCo's two-phase algorithm again proves advantageous in terms of both coverage percentile and the number of activated sensors.

When f is higher (Fig. 8(b)), all schemes have a coverage percentile of 100, and BaCo and the Path scheme activate fewer sensors than the Flow scheme. Thus, BaCo also performs well for varied sampling rates.



(a) $f = 0.5$ Hz (b) $f = 10$ Hz

Fig. 8. CDFs of $|S_A|$ for two sampling rates.

C. Convergence Speed

We also verified that BaCo's iterative algorithm converges quickly, regardless of N, the number of sensors. We found that an average of around three iterations were required with $N = 200$ in our test scenario. This low number of iterations is explained as follows. Since each iteration outputs an S_A that would be a feasible solution if false alarm probability was not being considered, $|S_A|$ from the first iteration is relatively large. BaCo then skips from $N_A = 1$ to this relatively large value for the second iteration. At relatively large values of N_A, small changes in N_A have little impact on the threshold T, as can be seen in (16). Therefore, the detection gains do not change much. An example of the gain change can be seen in Fig. 5, where the decrease of the gain from Fig. 5(b) to Fig. 5(c) is smaller than that from Fig. 5(a) to Fig. 5(b). So as the iterations continue, with increasing probability, the solution in the previous iteration is still available in the next iteration, and hence the algorithm settles quickly.

V. Related Work

The probabilistic sensing model was first applied to the area coverage problem. Ahmed [10] proposed an algorithm to evaluate area coverage under a probabilistic model. In [11], the authors designed distributed protocols to achieve full coverage under a probabilistic model. Clouqueur [9] compared value fusion and decision fusion under the probabilistic model. Following this work, Xing [12] analysed the impact of value fusion on area coverage from a statistical perspective. Wang [13] investigated the coverage region of sensors under the value fusion model and proposed a greedy algorithm to select the minimum number of sensors to achieve area coverage. These works established the fundamentals of probabilistic sensing models and provided insights for the application of these models to the barrier coverage problem.

A majority of the work on barrier coverage [1]–[3], [7], [8] adopted the disk sensing model. Under this model, Kumar and Xing investigated the critical conditions of weak and strong barrier coverage in [1] and [2]. They also proposed schemes to identify barriers formed by randomly deployed sensors. In [3], Kumar designed a scheme to schedule barriers such that the lifetime of the sensor network is maximized. Mostafaei in [7] proposed a distributed learning automata-based method to find the minimum number of sensors to construct barriers. Wang in [8] considered the barrier coverage problem in hybrid sensor networks where both static and mobile sensors are utilized. They investigated the impact of localization errors on the minimum number of mobile sensors required. These works provide the fundamental theories of barrier coverage. However, the disk model adopted is overly simple and cannot approximate the real world well.

Recently, the probabilistic sensing model is applied to barrier coverage and intruder detection problems [4], [5], [14]. Tan [14] explored the detection delay of intruders under the value fusion model. Yang proposed a scheme in [4] to achieve weak barrier coverage under the value fusion model with a system false alarm probability constraint. In this scheme, a sensor's coverage region is first modeled back to a disk by cutting off the region with small detection probability, then the coverage problem is solved with a disk model. No fusion along the intruding path is considered, so this solution cannot be applied to our problem. Chen proposed a scheme in [5] to achieve strong barrier coverage under the decision fusion model, where fusion along the intruding path is considered. However, the scheme only considers detection probability and ignores system false alarm probability. In contrast, BaCo considers both fusion along the intruding path and system false alarm probability, making BaCo a more practical scheme for real-world applications.

VI. Conclusion

In this paper, we propose an iterative scheme, called BaCo, to provide strong barrier coverage under the probabilistic sensing model, with the objective of minimizing the number of active sensors. BaCo considers both target detection probability and system false alarm probability, hence providing a more practical barrier coverage. Simulations show BaCo converges quickly and achieves better results than previous work, and it can automatically adjust the decision threshold to keep the system false alarm probability under a constant threshold.

References

[1] S. Kumar, T. H. Lai, and A. Arora, "Barrier coverage with wireless sensors," in *Proc. of ACM MobiCom*, Sept. 2005.

[2] B. Liu, O. Dousse, J. Wang, and A. Saipulla, "Strong barrier coverage of wireless sensor networks," *in Proc. of ACM MobiHoc*, May 2008.

[3] S. Kumar, T. H. Lai, M. E. Posner, and P. Sinha, "Optimal sleep-wakeup algorithms for barriers of wireless sensors," *in Proc. of IEEE BROADNETS*, Sept. 2007.

[4] G. Yang and D. Qiao, "Barrier information coverage with wireless sensors," in *Proc. of IEEE INFOCOM*, Apr. 2009.

[5] J. Chen, J. Li, and T. H. Lai, "Energy-efficient intrusion detection with a barrier of probabilistic sensors: Global and local," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, 2013.

[6] A. Saipulla, B. Liu, G. Xing, X. Fu, and J. Wang, "Barrier coverage with sensors of limited mobility," in *Proc. of ACM MobiHoc*, Sept. 2010.

[7] H. Mostafaei, "Stochastic barrier coverage in wireless sensor networks based on distributed learning automata," *Computer Communications*, vol. 55, 2015.

[8] Z. Wang, H. Chen, Q. Cao, H. Qi, and Z. Wang, "Fault tolerant barrier coverage for wireless sensor networks," in *Proc. of IEEE INFOCOM*, Apr. 2014.

[9] T. Clouqueur, K. K. Saluja, and P. Ramanathan, "Fault tolerance in collaborative sensor networks for target detection," *IEEE Transactions on Computers*, vol. 53, no. 3, 2004.

[10] N. Ahmed, S. S. Kanhere, and S. Jha, "Probabilistic coverage in wireless sensor networks," in *Proc. of IEEE LCN*, Nov. 2005.

[11] M. Hefeeda and H. Ahmadi, "A probabilistic coverage protocol for wireless sensor networks," in *Proc. of IEEE ICNP*, Oct. 2007.

[12] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C.-W. Yi, "Data fusion improves the coverage of wireless sensor networks," in *Proc. of ACM Mobicom*, Sept. 2009.

[13] W. Wang, V. Srinivasan, K.-C. Chua, and B. Wang, "Energy-efficient coverage for target detection in wireless sensor networks," in *Proc. of IEEE IPSN*, Apr. 2007.

[14] R. Tan, G. Xing, J. Wang, and B. Liu, "Performance analysis of real-time detection in fusion-based sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 9, 2011.