Barrier Counting in Mixed Wireless Sensor Networks

Shambhavi Srinivasa   Carey Williamson   Zongpeng Li
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada T2N 1N4
{*srinivs, carey, zongpeng*}*@cpsc.ucalgary.ca*

***Abstract*—Barrier coverage problems in sensor networks involve detecting intruders that attempt to cross a region of interest. In this paper, we formulate the k-connect barrier count problem for Mixed Sensor Networks (MSNs). The k-connect barrier count problem is to find the maximum number of barriers in an arbitrary MSN where at most k distinct mobile sensors can be used to construct any given *virtual edge* used in a barrier. We present the solution for the k-connect barrier count problem for $k \in \{0, 1, 2\}$ via Integer Linear Programming. Using simulation results, we show that as k increases, the density of sensors required to achieve barrier coverage decreases. The results quantitatively demonstrate the benefits of mobile sensors.**

***Keywords*-Wireless sensor networks; barrier coverage**

I. Introduction

A Wireless Sensor Network (WSN) typically consists of stationary sensors with no movement capability. With advancements in robotics, however, sensors with movement capability are now possible. These sensors are mounted on a mobile platform, and are known as mobile sensors.

A Mixed Sensor Network (MSN) is a special type of WSN that consists of stationary sensors as well as mobile sensors. While mobile sensors are expensive, and have limited movement capability, they provide distinct advantages for MSNs over WSNs. In particular, the ability to move selected sensor nodes as needed can improve sensor coverage, and provides tactical advantages for tracking and surveillance in hostile or environmentally-sensitive environments (e.g., military scenarios, wildlife corridors).

An important problem in sensor network deployments is *barrier coverage*, for detecting intruders that attempt to cross an area of interest. The solution requires a chain (barrier) of sensors across the deployed region such that the sensing areas of adjacent sensors mutually overlap. A strong barrier can protect critical infrastructure against all possible intruder crossing paths. Thus, barrier coverage has received significant research attention in recent years.

There are two general approaches to achieving strong barrier coverage. One approach is to scatter sufficiently many stationary sensors so that a strong barrier is present with high probability. Another approach is to deploy a mix of stationary and mobile sensors, using the latter to heal the coverage gaps, if any, that occur in barrier formation for the stationary sensors.

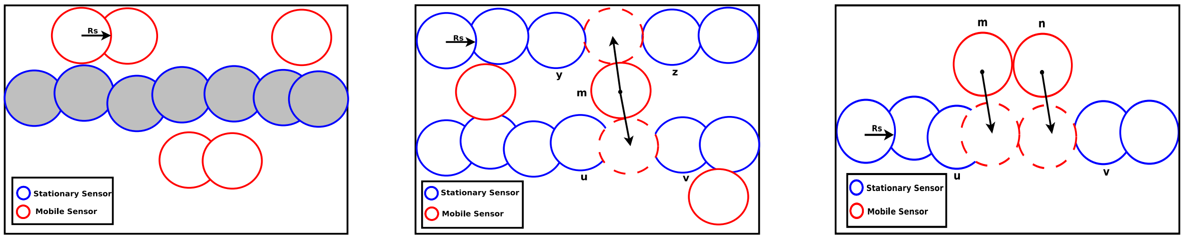
The benefits of mobile sensors have been explored primarily in the context of *area coverage* in fully-mobile sensor networks (i.e., WSNs that consist entirely of mobile sensors). However, Saipulla *et al.* [9] mathematically analyze the fundamental limits of barrier coverage with mobile sensors. The authors show that sensor mobility can effectively lower the critical density of sensors required (i.e., *percolation threshold*) for barrier formation in a network. The authors also devise an algorithm that determines the presence or absence of a barrier under constrained sensor mobility in an arbitrary MSN.

In this paper, we further investigate the benefits of mobile sensors. We formulate a novel problem called the k-connect barrier count problem, which considers not just the presence or absence of a barrier, but determines the maximum number of barriers that can be formed in an arbitary MSN. That is, our approach seeks to provide fault-tolerance in barrier coverage. We define this problem in terms of k, which is the maximum number of mobile sensors that can be used to construct any given *virtual edge* used in a barrier (see Figure 1). We formulate the k-connect barrier count problem as a maximum flow problem, which we solve using Integer Linear Programming. We also use simulation and visualization to test and verify the accuracy of our solution on randomly-generated MSN topologies.

There are two primary contributions in this paper:

- We formally define the k-connect barrier count problem, and solve it using Integer Linear Programming for $k \in \{0, 1, 2\}$.
- We use simulation to study the effects of sensing radius, movement radius, and the number of mobile sensors on the percolation threshold. In general, increasing k decreases the network size at which barrier coverage is achieved, thus demonstrating the benefits of mobile sensors.

The remainder of the paper is organized as follows. Section II reviews recent literature on barrier coverage in WSNs and MSNs. Section III presents the k-connect Barrier Count Problem. Section IV presents an overview of our solution methodology for the k-connect Barrier Count Problem for $k \in \{0, 1, 2\}$. Section V presents simulation results. Finally, Section VI concludes the paper.



(a) Barrier using $k = 0$ mobile sensors (b) Barrier using $k = 1$ mobile sensors (c) Barrier using $k = 2$ mobile sensors

Figure 1. Examples of the k-connect barrier count problem in Mixed Sensor Networks (MSNs)

II. Related Work

There are three broad categories of coverage problems in WSNs: point coverage, area coverage, and barrier coverage. In this paper, we focus solely on barrier coverage.

Barrier coverage refers to the ability of a WSN to protect an area of interest from intruder penetration, typically in military and homeland security applications. While barrier coverage is a relatively new topic in the field of WSNs, it has garnered significant research interest over the past few years. The bulk of this work is for stationary sensors, but recent works have considered mobile sensors as well.

Gage [3] introduced the notion of barrier coverage in the context of *ensemble motion behaviors* of robots. The objective of barrier coverage is to achieve a static arrangement of sensors that minimizes the probability of an intruder penetrating the barrier without being detected.

The barrier coverage problem has also been viewed in terms of exposure, expressed as either *worst-case coverage* or *best-case coverage*. In worst-case coverage, the objective is to find a path through the sensing field for which the probability of detecting an intruder is the lowest. This path is called the *Minimal Exposure Path* [7] or the *Maximal Breach Path* [5], [8]. Finding such a worst-case path is important because additional sensors could be deployed along that path to increase the quality of coverage. In the best-case formulation, the objective is to find a path through the sensing field for which the probability of detecting an intruder is highest. Such a path is called the *Maximal Exposure Path* [11] or the *Maximal Support Path* [8]. Finding such a path can be useful for applications that are security-sensitive and highly resource-constrained. An example is a solar-powered autonomous robot traversing a light-detecting WSN. By using the best-coverage path, the robot obtains the maximum amount of sunlight within a limited time.

Kumar *et al.* [4] introduce the notion of *k-barrier coverage* (not to be confused with our k-connect barrier count problem). A given area is *k-covered* if every crossing path across the width of the area intersects the coverage areas of at least k sensors. They establish whether or not it is locally possible to determine if a given area of interest is k-covered or not. They establish the optimal deployment pattern for k-barrier coverage when the sensors are deployed deterministically, and also consider the case when the sensors are deployed randomly. Next, they introduce the notions of weak and strong barrier coverage, deriving the critical conditions for the existence of weak k-barrier coverage, and the minimum number of sensors required to provide it.

Chen *et al.* [2] propose the notion of *local barrier coverage*. They provide a localized algorithm for sensors to determine whether or not an intruder can cross the area of interest undetected. Although local barrier coverage does not always guarantee global barrier coverage, the authors show through simulations that for thin belt regions, local barrier coverage always translates into global barrier coverage.

Deriving the critical density is a fundamental problem in coverage. Below the critical density, the network (w.h.p.) does not provide barrier coverage, while above this critical threshold, barrier coverage almost surely exists. Balister *et al.* [1] derive reliable density estimates for barrier coverage in thin strips. Recently, Liu *et al.* [6] present the critical conditions for strong barrier coverage in rectangular strips.

Barrier coverage with mobility is just beginning to appear in the literature. Existing papers deal with the barrier coverage problem with mobility-constrained sensors.

The work most relevant to our own is that by Saipulla *et al.* [9], in which the authors present a mathematical analysis of the fundamental limits of barrier coverage with sensor mobility. When m mobile sensors are deployed in a rectangular area of dimension $l \times w$, if all sensors have a sensing radius of r, they show that a maximum of $\frac{2mr}{l}$ barriers can be formed. They also present a sensor mobility scheme that achieves the maximum barrier coverage, while minimizing the maximum sensor movement distance. The authors devise an algorithm that detects the presence or absence of a barrier under (discrete) constrained sensor mobility. The authors also show that sensor mobility can effectively reduce the percolation threshold of a network, with the extent of the reduction depending on the number of mobile sensors as well as their movement radius.

Our work considers a novel (and more difficult) variant of the barrier coverage problem, in which we determine the maximum number of barriers that can be concurrently established across a given region, using the mobile sensors.

III. Problem Formulation

A. Network Model and Assumptions

We consider a *Mixed Sensor Network* deployed in a *rectangular deployment area*. The deployment follows the *random deployment model* and sensors adopt the commonly used *Binary Sensing Model*. In the following, we define the terms that we use in the description of the network model.

1) Wireless Sensor Network: A wireless sensor network $WSN(N, R_{\mathtt{S}}, R_{\mathtt{C}})$ is a network of N sensors. Every sensor has a unique ID and a unique location (x, y). A sensor can sense any object within a maximum distance $R_{\mathtt{S}}$ from its current position, where $R_{\mathtt{S}}$ is the *sensing radius*. It can also communicate with any sensor lying within a maximum distance $R_{\mathtt{C}}$ from its current position, where $R_{\mathtt{C}}$ is the *communication radius*. It is assumed that $R_{\mathtt{C}} \geq 2 \cdot R_{\mathtt{S}}$.

2) Mixed Sensor Network: A mixed sensor network $MSN(N, R_{\mathtt{S}}, R_{\mathtt{C}}, M, R_{\mathtt{M}})$ is a wireless sensor network $WSN(N + M, R_{\mathtt{S}}, R_{\mathtt{C}})$ consisting of N stationary sensors and M mobile sensors. A stationary sensor is a sensor with no movement capability, while a mobile sensor can move a maximum distance $R_{\mathtt{M}}$ from its initial position. $R_{\mathtt{M}}$ is called the *movement radius*. Let the set of N stationary sensors be denoted by $V_{\mathcal{S}}$ and the set of M mobile sensors be denoted by $V_{\mathcal{M}}$. Let s denote the source sensor and t denote the destination sensor. s and t are virtual sensor nodes representing the left boundary and the right boundary of the region, respectively.

3) Deployment Area: A deployment area $D(MSN, l, w)$ is a rectangular area of length l and width w, upon which the sensors of MSN are uniformly and independently distributed. The deployment area is devoid of obstacles. That is, sensors within distance $R_{\mathtt{C}}$ of each other can communicate with each other without any transmission errors.

4) Random Deployment Model: The random deployment model is a physical deployment model that can be modeled by a *two-dimensional Poisson point process*. Let the density of the underlying Poisson process be λ. In this model, the number of sensors, $N(R)$, located in a region R follows a Poisson distribution with parameter $\lambda \cdot A(R)$, where $A(R)$ is the area of the deployment region R. Specifically,

$$P(N(R) = k) = \frac{e^{-\lambda \cdot A(R)}(\lambda \cdot A(R))^k}{k!}$$

5) Binary Sensing Model: We use the standard Binary Sensing Model from the WSN literature. In this model, a sensor at position P_1 can detect an event at position P_2 if the Euclidean distance between the two positions satisfies $d(P_1, P_2) \leq R_{\mathtt{S}}$. Specifically, the *coverage area* for a sensor is a circular area of radius $R_{\mathtt{S}}$ around the sensor. Corresponding definitions apply for the *communication area* (with radius $R_{\mathtt{C}}$) and the *movement area* (with radius $R_{\mathtt{M}}$).

B. Preliminaries

We provide formal definitions of the key terms and concepts used in our problem formulation, and its solution. Figure 1 shows an example scenario illustrating several of these definitions. In this diagram, a set of sensor nodes (e.g., u, v) are scattered in a rectangular region. For notational convenience, we assume a virtual sensor node s (source) to the left of the region, and a virtual sensor node t (target destination) to the right of the region.

Definition 1 (**Inter-sensor Distance** $\delta(u, v)$)**:** The distance between two (stationary, mobile, or virtual) sensors u, v located at points $p_u(x_u, y_u)$ and $p_v(x_v, y_v)$ is:

$$\delta(u,v) = \begin{cases} d(p_u, p_v) & \text{if } u, v \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \\ x_v & \text{if } u = s \text{ and } v \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \\ l - x_u & \text{if } u \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \text{ and } v = t \\ l & \text{if } u = s \text{ and } v = t \end{cases}$$

Definition 2 (**Communication Connectivity**)**:** A type of connectivity that exists between two sensors, u and v, if their communication areas overlap, i.e., $\delta(u, v) \leq R_{\mathtt{C}}$.

Definition 3 (**Sensing Connectivity**)**:** A type of connectivity that exists between two sensors, u and v, if:

$$\delta(u,v) \leq \begin{cases} 2 \cdot R_{\mathtt{S}} & \text{if } u, v \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \\ R_{\mathtt{S}} & \text{if } u = s \text{ and } v \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \\ R_{\mathtt{S}} & \text{if } u \in V_{\mathcal{S}} \cup V_{\mathcal{M}} \text{ and } v = t \end{cases}$$

Definition 4 (**Coverage Gap**)**:** An area that is not covered by the coverage area of a sensor.

Definition 5 (**Barrier**)**:** A chain of sensors with shared sensing connectivity such that their aggregate sensing area covers the entire length of the area of interest without any coverage gaps.

Definition 6 (**Crossing Path**)**:** A path in the area of interest that crosses the entire width of the area (i.e., from top to bottom) without touching the coverage area of any sensor.

Definition 7 (**Strong Barrier**)**:** A barrier for which no crossing path exists. That is, the barrier is guaranteed to detect intruders moving along any attempted crossing path.

Definition 8 (**Connectivity Graph** $G_{\mathtt{C}} = (V_{\mathtt{C}}, E_{\mathtt{C}})$)**:** A graph whose vertex set consists of all stationary sensors $V_{\mathcal{S}}$, all mobile sensors $V_{\mathcal{M}}$, the source sensor s, and the destination sensor t, and whose edge set consists of all pairs of sensors that can communicate directly with each other.

Definition 9 (**Sensing Graph** $G_{\mathtt{S}} = (V_{\mathtt{S}}, E_{\mathtt{S}})$)**:** A graph whose vertex set consists of all stationary sensors $V_{\mathcal{S}}$, the source sensor s, and the destination sensor t, and whose edge set consists of the following:

- All pairs of stationary sensors that share sensing connectivity with each other.

- The source s and a stationary sensor if the coverage area of the stationary sensor reaches the left boundary.
- A stationary sensor and the destination t if the coverage area of the stationary sensor reaches the right boundary.

Definition 10 (**Component**)**:** A set of stationary sensors such that for any two sensors u, v in the set, there exists a path in the sensing graph from u to v. The component set $\mathcal{C}$ is the set of all components.

Definition 11 (**Gateway Sensor**)**:** A stationary sensor that has communication connectivity with a stationary sensor from another component. H is the set of all gateway sensors.

Definition 12 (**Virtual Edge**)**:** An edge that can be created with the assistance of at most k unique mobile sensors between any of the following:

- Two stationary sensors
- The source sensor s and a stationary sensor
- A stationary sensor and the destination sensor t
- The source sensor s and the destination sensor t

Definition 13 (**LP Graph** $G^k_{lp} = (V^k_{lp}, E^k_{lp})$)**:** A graph whose vertex set consists of all stationary sensors $V_\mathcal{S}$, the source sensor s, and the destination sensor t, and whose edge set consists of all the edges in the sensing graph G_s, plus the virtual edge set (i.e., the set of all virtual edges).

Definition 14 (**Flow Network**)**:** A directed graph in which each edge has a capacity and each edge receives a flow governed by the following constraints:

- **Capacity Constraint**: The amount of flow on an edge cannot exceed the capacity of the edge.
- **Flow Conservation Constraint**: The amount of incoming flow into a vertex equals the amount of its outgoing flow, except for the source (which has more outgoing flow) and the sink (which has more incoming flow).
- **Mobility Constraint**: A mobile sensor can be used in at most one of its candidate virtual edges in the MSN.

C. Problem Definition

The k**-connect barrier count problem** is to find the maximum number (say η) of strong barriers in an MSN, where at most k unique mobile sensors can be used to construct any given virtual edge used in a barrier. An intruder crossing the area of interest is detected by at least η sensors.

We focus on small values of k (i.e., $k \in \{0, 1, 2\}$) for three reasons. First, mobile sensors are assumed to be an expensive and scarce resource in an MSN, and should be used judiciously when constructing virtual edges. For example, using five mobile sensors to build five virtual edges is usually more effective than building one virtual edge that requires all five mobile sensors. Second, the state-space complexity of the problem grows as the value of k increases, since the number of virtual edges grows rapidly. Third, there is a diminishing returns effect as k increases, which does not merit the extra complexity associated with larger k.

IV. Proposed k-connect Barrier Count Solution

A. Overview

The solution for the k-connect Barrier Count Problem for $k \in \{0, 1, 2\}$ can be summarized in the following steps:

- First, the LP Graph is built.
- We formulate a network flow problem by constructing a flow network from the LP Graph. This network flow problem is a variation of the Maximum Flow (max flow) problem. The objective of the max flow problem is to find the maximum feasible flow through a single-source, single-destination flow network.
- The network flow problem is solved using an Integer Linear Program (ILP) formulation.
- The solution to the ILP gives us the maximum number of concurrent edge-disjoint and vertex-disjoint flows. These disjoint flows correspond to disjoint paths in the LP Graph. One disjoint path, in turn, corresponds to one barrier. Hence, we obtain the maximum number of concurrent barriers that can be formed.

In the 0-connect Barrier Count Problem, the LP Graph is equivalent to the sensing graph G_S, since no mobile sensors are allowed to be used in barrier formation.

The construction of the LP Graph for the 1-connect Barrier Count Problem, denoted by $G^1_{lp} = (V^1_{lp}, E^1_{lp})$, is as follows. For every edge $\{u, v\} \in E_\mathrm{S}$ (i.e., for every edge in the sensing graph G_S), we add an edge to E^1_{lp} denoted by $(\{u, v\}, (r))$. Here, r is a character constant indicating that the edge is a real edge (i.e., between two stationary sensors, or between s and a stationary sensor, or between a stationary sensor and t) rather than a virtual edge (i.e., using a mobile sensor). The virtual edge set is the set of all virtual edges that can be created with the assistance of a single mobile sensor. Each virtual edge is denoted by $(\{u, v\}, (m))$, where u and v are either gateway sensors in different components that can be connected via the mobile sensor m, or stationary sensors belonging to the same component that can be connected via m. While there may be many candidate virtual edges, at most M of them can be instantiated in barrier formation, with each such edge requiring one (i.e., $k = 1$) of the available mobile nodes.

In the next section, we present the detailed solution for the 2-connect Barrier Count Problem.

B. Solution Details

In the **2-connect barrier count problem**, at most two distinct mobile sensors can be used together to connect two stationary sensors. In this problem, the virtual edge set consists of all virtual edges that can be created with either one or two mobile sensors. We build this virtual edge set using information from two data structures constructed for the $k = 1$ case: the component set $\mathcal{C}$, and the LP graph G^1_{lp}.

The algorithm for the construction of the LP Graph $G^2_{lp} = (V^2_{lp}, E^2_{lp})$ is presented in Algorithm 1. The inputs

Algorithm 1 LP Graph Construction

```
1: $V_{lp}^2 \leftarrow V_{\mathcal{S}} \cup \{s,t\}$    ▷ Assume $u.x \leq v.x$
2: $E_m \leftarrow \{\ (\{u,v\},(m,n)) \mid u,v \in V_{\mathcal{S}} \cup \{s,t\} \wedge m,n \in V_{\mathcal{M}} \wedge u,v \notin \mathcal{C}_i \wedge TwoConnect(u,v,m,n,R_{\mathtt{M}},R_{\mathtt{S}})\ \}$
3: $E_m^* \leftarrow \{\ (\{u,v\},(m,n)) \mid u,v \in V_{\mathcal{S}} \wedge m,n \in V_{\mathcal{M}} \wedge u,v \in \mathcal{C}_i \wedge \{u,v\} \notin E_{\mathtt{S}} \wedge TwoConnect(u,v,m,n,R_{\mathtt{M}},R_{\mathtt{S}})\ \}$
4: $E_{lp}' \leftarrow \{(\{u,v\},(a,l)) \mid (\{u,v\},(a)) \in E_{lp}^1\}$    ▷ Let $l$ be a unique character identifier
5: $E_{lp}^2 \leftarrow E_m \cup E_m^* \cup E_{lp}'$
6: $G_{lp}^2 \leftarrow (V_{lp}^2, E_{lp}^2)$
```

to this algorithm are the component set $\mathcal{C}$, and G_{lp}^1 from the 1-connect Barrier Count Problem. The explanation of the algorithm is as follows. The vertex set V_{lp}^2 in line 1 is initialized to include all stationary sensors $V_{\mathcal{S}}$, the source sensor s, and the destination sensor t. In lines 2 and 3, we construct the edge sets E_m and E_m^*. Combined, they constitute the virtual edge set. In line 2, we consider every tuple (u,v,m,n), where u and v are in $V_{\mathcal{S}} \cup \{s,t\}$, u is located to the left of v, and they do not belong to the same connected component. m and n are two arbitrary mobile sensors. In line 3, we consider the possibility of pairs of sensors within the *same* connected component that can be connected via a mobile sensor. We consider all tuples (u,v,m,n), where u and v belong to the same component, and there exists no edge between u and v in the sensing graph $G_{\mathtt{S}}$. Again, m and n are two arbitrary mobile sensors.

We also define a function `TwoConnect()`, presented in Algorithm 2, to check if u and v can be connected using m and n. If this function returns `True`, then an undirected edge denoted by $(\{u,v\},(m,n))$ is added to the edge set E_m. Here, the notation m,n is used as a unique identifier to indicate that the virtual edge between the two stationary sensors u and v is created by the mobile sensors m and n. To ensure compatibility with the previous edge set, we add a character constant l to the tuples representing the edges belonging to E_{lp}^1 in line 4, and create a new edge set E_{lp}'. In line 5, the edge set E_{lp}^2 becomes the union of E_m, E_m^*, and E_{lp}'. Finally, in line 6, the graph G_{lp}^2 is constructed from V_{lp}^2 and E_{lp}^2.

In the function `TwoConnect()`, we first check if it is possible for u and v to be connected to each other using two mobile sensors. We also define a boolean variable `flag`. The following cases are possible:

- If u and v are both stationary sensors, they can be connected via two mobile sensors only if the distance, $\delta(u,v)$, between them satisfies $4R_{\mathtt{S}} < \delta(u,v) \leq 6R_{\mathtt{S}}$ (see line 18). If $\delta(u,v) \leq 4R_{\mathtt{S}}$, then u and v are eligible to be connected using a single mobile sensor.
- If u is the source sensor s, and v is an arbitrary stationary sensor, then we are trying to connect v to the left boundary. In this case (line 9), u and v can be connected using a mobile sensor only if the distance, $\delta(u,v)$, between them satisfies $3R_{\mathtt{S}} < \delta(u,v) \leq 5R_{\mathtt{S}}$. If so, the (x,y) coordinates of u are set to $(0, v.y)$.

Algorithm 2 TwoConnect(u,v,m,n,$R_{\mathtt{M}}$,$R_{\mathtt{S}}$)

```
1: function TwoConnect(u,v,m,n,$R_{\mathtt{M}}$,$R_{\mathtt{S}}$)
2:     Bool flag ← False
3:     if $(u = s \wedge v = t)$ then
4:         if $(\delta(u,v) > 2 \cdot R_{\mathtt{S}} \wedge \delta(u,v) \leq 4 \cdot R_{\mathtt{S}})$ then
5:             $u.x = 0$; $u.y = m.y$
6:             $v.x = l$; $v.y = n.y$
7:             flag ← True
8:         end if
9:     else if $(u = s)$ then
10:        if $(\delta(u,v) > 3 \cdot R_{\mathtt{S}} \wedge \delta(u,v) \leq 5 \cdot R_{\mathtt{S}})$ then
11:            $u.x = 0$; $u.y = v.y$; flag ← True
12:        end if
13:    else if $(v = t)$ then
14:        if $(\delta(u,v) > 3 \cdot R_{\mathtt{S}} \wedge \delta(u,v) \leq 5 \cdot R_{\mathtt{S}})$ then
15:            $v.x = l$; $v.y = u.y$; flag ← True
16:        end if
17:    else
18:        if $(\delta(u,v) > 4 \cdot R_{\mathtt{S}} \wedge \delta(u,v) \leq 6 \cdot R_{\mathtt{S}})$ then
19:            flag ← True
20:        end if
21:    end if
22:    if flag then
23:        $C_1.x \leftarrow u.x + \frac{1}{3}(v.x - u.x)$
24:        $C_1.y \leftarrow u.y + \frac{1}{3}(v.y - u.y)$
25:        $C_2.x \leftarrow u.x + \frac{2}{3}(v.x - u.x)$
26:        $C_2.y \leftarrow u.y + \frac{2}{3}(v.y - u.y)$
27:        Return Feasible(u,v,m,n,$R_{\mathtt{M}}$,$R_{\mathtt{S}}$,$C_1$,$C_2$)
28:    else
29:        Return False
30:    end if
31: end function
```

- By a symmetry argument, the previous case also applies when u is an arbitrary stationary sensor and v is the destination sensor t (see line 13). That is, we are trying to connect u to the right boundary, and the (x,y) coordinates of v are set to $(l, u.y)$.
- When u is the source sensor and v is the destination sensor, they can be connected only if the distance between them satisfies $2R_{\mathtt{S}} < \delta(u,v) \leq 4R_{\mathtt{S}}$ (line 3). In this case, we are trying to connect the left boundary to the right boundary using two mobile sensors. As a

Algorithm 3 Feasible(u,v,m,n,$R_{\mathtt{M}}$,$R_{\mathtt{S}}$,C_1, C_2)

```
 1: function FEASIBLE(u,v,m,n,$R_{\mathtt{M}}$,$R_{\mathtt{S}}$,$C_1$, $C_2$)
 2:     Float $\mathtt{check_u} \leftarrow 0.0$, $\mathtt{check_v} \leftarrow 0.0$
 3:     if $(u = s \wedge v = t)$ then
 4:         $\mathtt{check_u} \leftarrow R_{\mathtt{S}}$; $\mathtt{check_v} \leftarrow R_{\mathtt{S}}$
 5:     else if $(u = s)$ then
 6:         $\mathtt{check_u} \leftarrow R_{\mathtt{S}}$; $\mathtt{check_v} \leftarrow 2 \cdot R_{\mathtt{S}}$
 7:     else if $(v = t)$ then
 8:         $\mathtt{check_u} \leftarrow 2 \cdot R_{\mathtt{S}}$; $\mathtt{check_v} \leftarrow R_{\mathtt{S}}$
 9:     else
10:         $\mathtt{check_u} \leftarrow 2 \cdot R_{\mathtt{S}}$; $\mathtt{check_v} \leftarrow 2 \cdot R_{\mathtt{S}}$
11:     end if
12:     if $(\delta(u,m), \delta(v,n) \leq 2 \cdot R_{\mathtt{S}} + R_{\mathtt{M}})$ then
13:         if $(d(m, C_1) \leq R_{\mathtt{M}})$ then
14:             $C_3.x \leftarrow C_1.x$; $C_3.y \leftarrow C_1.y$
15:         else
16:             $C_3.x \leftarrow m.x + \frac{R_{\mathtt{M}}}{d(m,C_1)} \cdot (m.x - C_1.x)$
17:             $C_3.y \leftarrow m.y + \frac{R_{\mathtt{M}}}{d(m,C_1)} \cdot (m.y - C_1.y)$
18:         end if
19:         if $(d(n, C_2) \leq R_{\mathtt{M}})$ then
20:             $C_4.x \leftarrow C_2.x$; $C_4.y \leftarrow C_2.y$
21:         else
22:             $C_4.x \leftarrow n.x + \frac{R_{\mathtt{M}}}{d(n,C_2)} \cdot (n.x - C_2.x)$
23:             $C_4.y \leftarrow n.y + \frac{R_{\mathtt{M}}}{d(n,C_2)} \cdot (n.y - C_2.y)$
24:         end if
25:         if $(d(u, C_3) \leq \mathtt{check_u} \wedge d(C_4, v) \leq \mathtt{check_v} \wedge d(C_3, C_4) \leq 2 \cdot R_{\mathtt{S}})$ then
26:             Return True
27:         else
28:             Return False
29:         end if
30:     end if
31:     Return False
32: end function
```

result, the (x, y) coordinates of u are set to $(0, m.y)$ and the (x, y) coordinates of v are set to $(l, n.y)$.

If none of these conditions are satisfied, then `flag` is set to `False`. If `flag` is set to `False`, we know that the sensors u and v cannot be connected using two mobile sensors, and the function `TwoConnect()` returns `False`.

If any of the foregoing conditions are satisfied, then `flag` is set to `True`. If `flag` is set to `True`, we calculate where to relocate the mobile sensors m and n in order to connect u and v. These locations are called candidate locations, and are denoted by C_1 and C_2, respectively. Each candidate location has a location (x, y). The locations of C_1 and C_2 are determined such that they split the line joining u and v into three equal segments. See Figure 2. We then call the function `Feasible()`, presented in Algorithm 3.

The function `Feasible()` takes the stationary sensors u and v, the mobile sensors m and n, the movement radius $R_{\mathtt{M}}$, the sensing radius $R_{\mathtt{S}}$, and the candidate locations C_1 and C_2 as its parameters. It determines whether it is geometrically possible to connect the two stationary sensors with the two mobile sensors. If so, then the function returns `True`. Otherwise, it returns `False`.

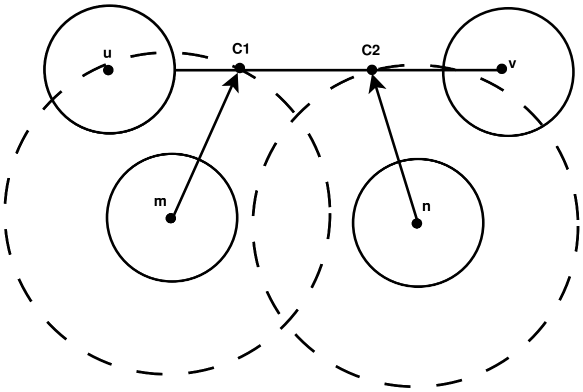


Figure 2. $d(m, C_1), d(n, C_2) \leq R_{\mathtt{M}}$

In the function `Feasible()`, we check the proximity of m to u, and the proximity of n to v, to ensure that connectivity is possible using these mobile sensors (see line 12). The maximum reach of the coverage area of a mobile sensor is $R_{\mathtt{S}} + R_{\mathtt{M}}$. For the boundary of a mobile sensor's coverage area to touch the boundary of a stationary sensor's coverage area, the maximum distance between them can be $2 \cdot R_{\mathtt{S}} + R_{\mathtt{M}}$. For m to be able to re-locate itself to the right of u such that its coverage area overlaps with that of u, we need $\delta(u, m) \leq 2 \cdot R_{\mathtt{S}} + R_{\mathtt{M}}$. A similar condition applies for n and v. If these conditions are not met, then it is not feasible to connect u and v using m and n, so the function returns `False`.

In line 13, we next check if the distance, $d(m, C_1)$, between m and C_1 is less than or equal to $R_{\mathtt{M}}$. If this condition is satisfied, then m can move to the candidate location on the line joining u and v, as shown in Figure 2. Otherwise, in lines 16-17, a new candidate location C_3 is determined. This candidate location is determined such that the mobile sensor moves as close as possible to C_1. That is, it moves to the boundary of its movement radius in the direction of C_1, as shown in Figure 3. In line 19, we make the corresponding check between n and C_2, and calculate a new candidate location C_4 if necessary.

In line 25, we check for the following conditions:

- We check if the Euclidean distance between C_3 and C_4 satisfies $d(C_3, C_4) \leq 2 \cdot R_{\mathtt{S}}$.
- We check if $d(u, C_3) \leq \mathtt{check_u}$, where $\mathtt{check_u}$ is either $R_{\mathtt{S}}$ if $u = s$, or $2 \cdot R_{\mathtt{S}}$ otherwise.
- We check if $d(C_4, v) \leq \mathtt{check_v}$ where $\mathtt{check_v}$ is $R_{\mathtt{S}}$ if $v = t$, and $2 \cdot R_{\mathtt{S}}$ otherwise.

If these conditions are satisfied, then the mobile sensors have successfully connected the two stationary sensors, and the

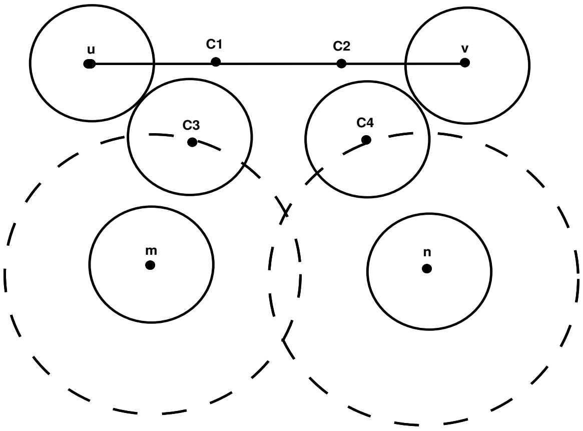


Figure 3. $d(m, C_1), d(n, C_2) > R_{\mathrm{M}}$

function `Feasible()` returns `True`. If these conditions are not satisfied, then the mobile sensors have failed at connecting u and v. Hence, it is not feasible to connect u and v using m and n, and the function `Feasible()` returns `False`.

C. Integer Linear Program Formulation

We now construct a network flow problem, which when solved will give us the maximum number of disjoint flows for the 2-connect barrier count problem. The translation to the ILP formulation is relatively straightforward. Consider a flow network in the form of a directed graph $G^2_{fn} = (V^2_{fn}, E^2_{fn})$. The vertex set of this graph is set to V^2_{lp}. For every undirected edge $(\{u,v\},(a,b)) \in E^2_{lp}$, we add two directed edges (u,v,a,b) and (v,u,a,b) to E^2_{fn}. The capacity of every edge in E^2_{fn} is constrained to 1, to guarantee that the flows found by the ILP are edge-disjoint. Given a source $s \in V^2_{fn}$ and a sink $t \in V^2_{fn}$, we aim to find flow assignments, denoted by x_{uv}, through every edge in the flow network such that the unit capacity constraint (for edge-disjoint flows) and flow conservation constraints at all vertices are satisfied, and the total flow through the network from s to t is maximized. In addition, we add the following extra constraints:

- **Vertex Capacity Constraint**: In addition to being edge-disjoint, we require all flows between s and t to be *vertex-disjoint*. A set of flows in G^2_{fn} are vertex-disjoint if each vertex in G^2_{fn} appears in at most one of the flows. In order to do so, we require that for any vertex v belonging to the set of all stationary sensors $V_{\mathcal{S}}$, the total incoming flow is at most 1.
- **Mobility Constraint**: The mobility constraint is introduced to guarantee that a mobile sensor can assist in connecting at most one pair of stationary sensors, since it can ultimately move to only one location.

We now formulate the ILP for the flow network as follows:

$$\text{Max} \sum_{(s,j,a,b)\in E^2_{fn}} x_{sjab} - \sum_{(j,s,a,b)\in E^2_{fn}} x_{jsab} \tag{1}$$

$$\text{s. t.} \sum_{(s,j,a,b)\in E^2_{fn}} x_{sjab} - \sum_{(j,s,a,b)\in E^2_{fn}} x_{jsab} \geq 0 \tag{2}$$

$$\sum_{(i,j,a,b)\in E^2_{fn}} x_{ijab} - \sum_{(j,i,a,b)\in E^2_{fn}} x_{jiab} = 0 \tag{3}$$

$$\sum_{(t,j,a,b)\in E^2_{fn}} x_{tjab} - \sum_{(j,t,a,b)\in E^2_{fn}} x_{jtab} \leq 0 \tag{4}$$

$$\sum_{(i,j,a,b)\in E^2_{fn}} x_{ijab} \leq 1, \quad \forall a \in V_{\mathcal{M}} \tag{5}$$

$$\sum_{(i,j,a,b)\in E^2_{fn}} x_{ijab} \leq 1, \quad \forall b \in V_{\mathcal{M}} \tag{6}$$

$$\sum_{(i,j,a,b)\in E^2_{fn}} x_{ijab} \leq 1, \quad \forall j \in V_{\mathcal{S}} \tag{7}$$

$$x_{ijab} \in \{0,1\}, \quad \forall (i,j,a,b) \in E^2_{fn} \tag{8}$$

In the ILP formulation, we maximize the net flow from the source (line 1). Lines 2, 3, and 4 denote the flow conservation constraint: in (2), the constraint is restricted to the source sensor s, which only has outgoing flow; in (4), the constraint is restricted to the destination sensor t, which only has incoming flow; and (3) is the general case for all other stationary sensors i in $V_{\mathcal{S}}$. Lines 5 and 6 express the mobility constraint for the cases where a and b are mobile sensors. Line 7 denotes the vertex capacity constraint. Line 8 states the unit capacity constraint. In our ILP, the only valid flow values that can be assigned are 0 and 1. Furthermore, summing the end-to-end flow volume is logically equivalent to counting the number of barriers formed.

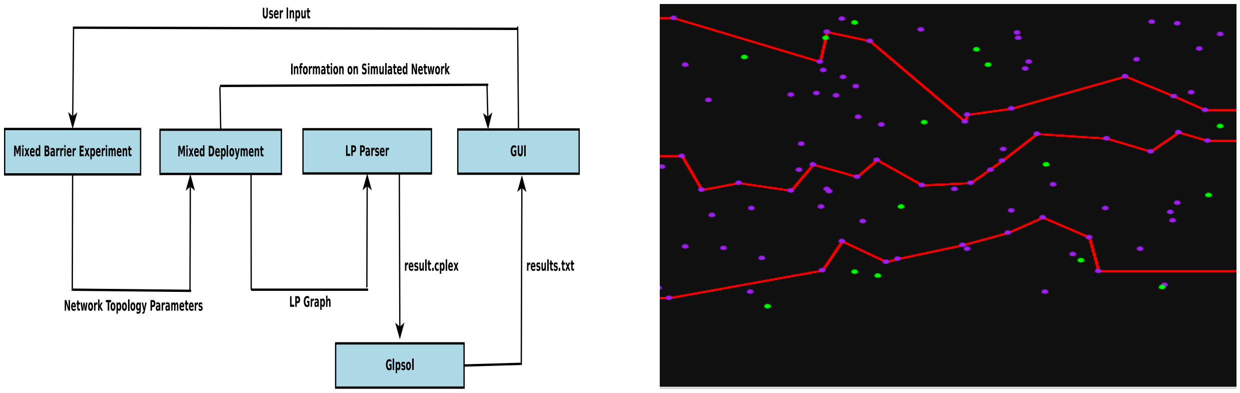
The solution to this ILP gives the maximum number of edge-disjoint and vertex-disjoint paths in the graph G^2_{fn}, which represents the maximum number of strong barriers.

V. Simulation Results

A. Simulator Overview

Figure 4(a) shows a structural overview of our MSN simulator [10]. The simulator has a GUI to enter MSN configuration parameters, and separate modules to do random sensor deployment, graph algorithms, and LP formulation. The LP itself is solved by invoking `glpsol` externally, and then parsing and displaying the results. Figure 4(b) shows an example of the three barriers found on a random MSN topology using our simulator.

In our simulations, sensors are deployed according to a Poisson point process in a 50 x 500 rectangular deployment area. We carefully constructed a set of 10 test scenarios to verify the correctness of our barrier formation algorithms, including scenarios with and without possible barriers [10]. These tests have confirmed the correctness of our solution.



(a) Simulator overview

(b) Barrier coverage example with three barriers

Figure 4. Simulation example for barrier coverage in mixed sensor networks

We used our simulator to generate and test a large set of randomly generated topologies, to assess the benefits of mobile sensors. In particular, we consider the prevalence of MSN topologies for which no barrier is possible, compared to those for which (one or more) barriers are possible. We express our results as *barrier coverage probability*.

B. Simulation Methodology

Our simulation experiments are structured using a one-factor-at-a-time experimental design. We study the effects of sensing radius, movement radius, and the number of mobile sensors on the barrier coverage probability for $k \in \{0, 1, 2\}$. According to percolation theory, when sensors are deployed according to a Poisson point process with density λ, there exists a critical density, λ_c, above which an arbitrarily large sensor cluster is formed. Above this critical density, the probability of finding a barrier approaches 1. Hence, barrier coverage probability is related to the percolation threshold.

The factors and levels used in our simulation experiments are summarized in Table I. The default parameter values are shown in bold font. In each graph of simulation results, each data point represents the average from 50 replications, each with a different seed for the random number generator.

Table I
FACTORS AND LEVELS IN SIMULATION EXPERIMENTS

Factor	Description	Levels
N	Network Size (Sensors)	0, 10, 20, . . . 200
p	Mobile Sensor Percentage	0%, 10%, **30%**, 50%
k	Virtual edge constraint	0, 1, 2
$R_{\mathtt{S}}$	Sensing Radius	**20**, 75
$R_{\mathtt{M}}$	Movement Radius	25, **50**, 75

C. Effect of Sensing Radius

In our first set of results, we study the effect of the sensing radius, $R_{\mathtt{S}}$, on barrier formation as a function of network size. For $R_{\mathtt{S}} \in \{20, 75\}$, we observe the barrier coverage probability for different values of k. The communication radius, $R_{\mathtt{C}}$, for each sensor is 50. Mobile sensors constitute 30% of the network nodes, and their movement radius, $R_{\mathtt{M}}$, is 50. The network size ranges from 0 to 200 in steps of 10.

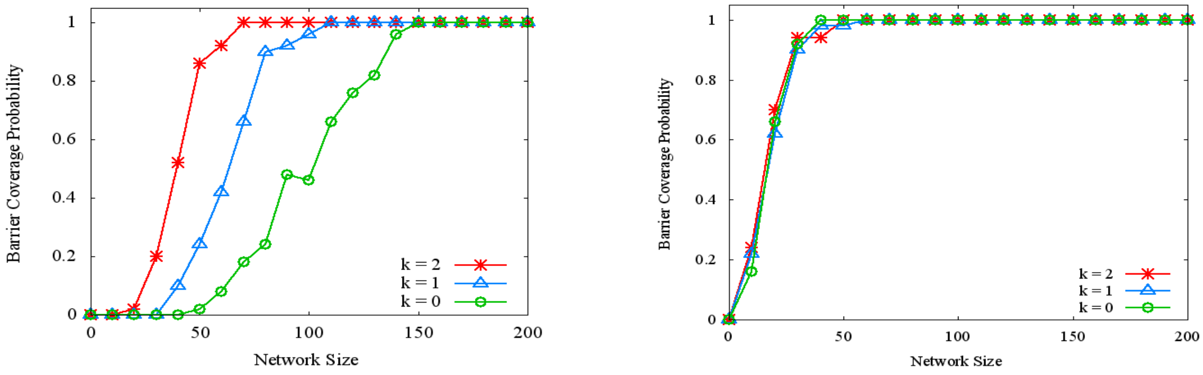
In Figure 5(a), each curve represents the value of barrier coverage probability obtained for a different k value, as the network size is varied. For each curve, there exists a *phase transition* behavior where the barrier coverage probability rises sharply and becomes 1. For $k = 2$, this occurs at a smaller network size compared to $k = 1$. Mobile sensors assist in joining disconnected clusters of sensors. Since two mobile sensors can assist in such connections when $k = 2$, the phase transition occurs at a much smaller network size compared to $k = 1$. For $k = 0$, this transition occurs at a much larger network size. In this particular simulation scenario, the critical density is roughly halved (compared to $k = 0$) when $k = 2$, even though only 30% of the sensor nodes are mobile.

In Figure 5(b), when $R_{\mathtt{S}}$ is set to 75, we observe that all three curves align very closely. This result makes sense intuitively because the sensing radius is much larger with respect to the length of the deployment area. In most MSN instances, there invariably exists one large component that has sensing connectivity to the source as well as the destination, so the results for $k \in \{0, 1, 2\}$ are similar. In other words, the benefits of mobile sensors are more apparent when $R_{\mathtt{S}}$ is small relative to the length of the deployment area.

D. Effect of Movement Radius

In our second set of results, we study the effect of the movement radius, $R_{\mathtt{M}}$. For $R_{\mathtt{M}} \in \{25, 75\}$, we observe the barrier coverage probability for $k \in \{1, 2\}$. (We do not plot the barrier coverage probability for $k = 0$ because $R_{\mathtt{M}}$ is irrelevant in this case.) The other parameters remain at their default settings.

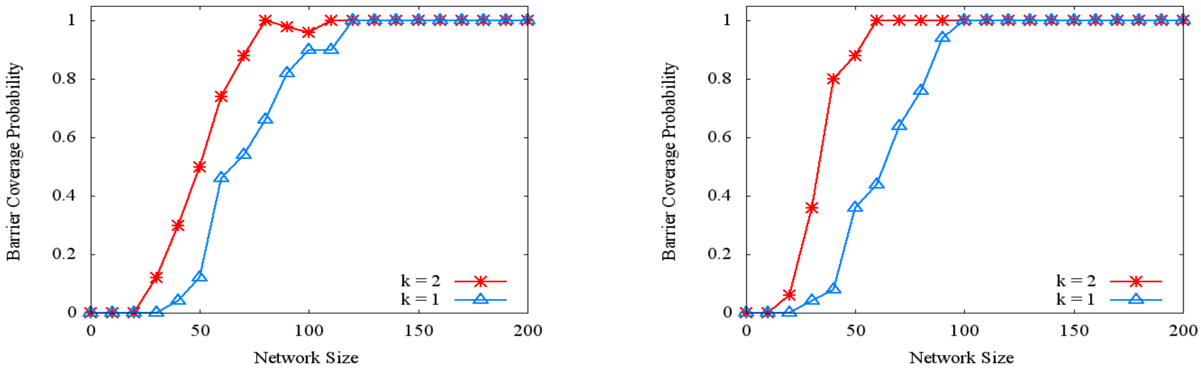
Figure 6(a) shows the barrier coverage probability attained for each value of k as the network size is increased. In each



(a) Barrier coverage probability vs network size at $R_S = 20$

(b) Barrier coverage probability vs network size at $R_S = 75$

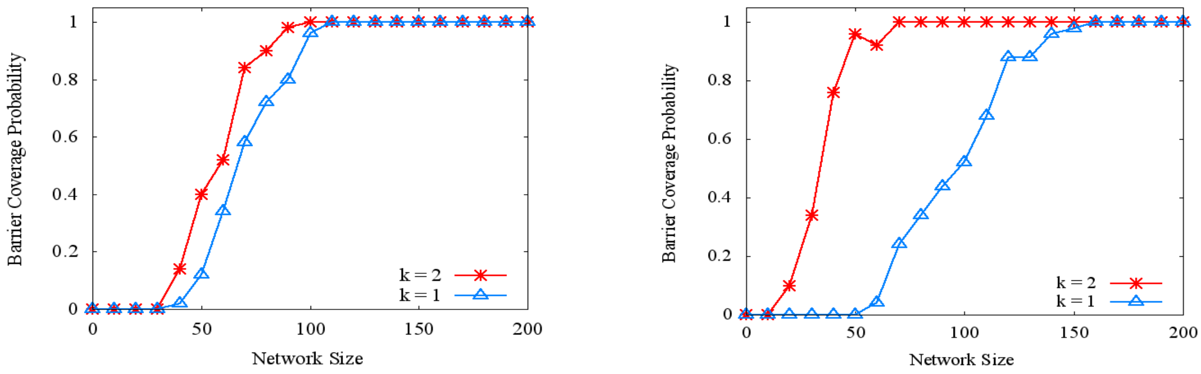
Figure 5. Effect of sensing radius



(a) Barrier coverage probability vs network size at $R_M = 25$

(b) Barrier coverage probability vs network size at $R_M = 75$

Figure 6. Effect of movement radius



(a) Barrier coverage probability vs network size (10% mobile)

(b) Barrier coverage probability vs network size (50% mobile)

Figure 7. Effect of percentage of mobile sensors

curve, there still exists a *phase transition* behavior, though it seems more gradual than in the previous scenario. That is, the sensing radius $R_{\mathtt{S}}$ seems to have greater impact than the movement radius $R_{\mathtt{M}}$, though both do have an effect. As expected, the phase transition for $k = 2$ occurs at a smaller network size compared to $k = 1$. This behavior is consistent when $R_{\mathtt{M}}$ is increased to 75 as shown in Figure 6(b). As $R_{\mathtt{M}}$ increases, we see a decrease in percolation threshold for $k = 1$ as well as $k = 2$, since the mobile sensors can move farther distances to bridge coverage gaps. There is slightly greater separation between the two curves when $R_{\mathtt{M}}$ is larger, implying greater advantages for using $k > 1$ mobile sensors when their movement is less constrained.

E. Effect of Number of Mobile Sensors

In our final set of results, we study the effect of the percentage of mobile sensors. We vary this parameter between 10% and 50%, and observe the barrier coverage probability for $k \in \{1, 2\}$. The other parameters retain their defaults.

In Figure 7(a), each curve shows the barrier coverage probability obtained for each k value when the network size is increased. Here, the percentage of mobile sensors is set to 10%. Again, we observe a phase transition at some point, when the barrier coverage probability becomes 1. This transition occurs at a lower density for $k = 2$ compared to $k = 1$. This behavior is consistent when the percentage of mobile sensors is 50%, as shown in Figure 7(b). However, we also observe greater separation between the two curves when the percentage of sensors increases. There are two underlying reasons for this phenomenon. First, for a given fixed network size ($N+M$), there are fewer stationary sensor nodes that can be used to build the initial sensing graph. As a result, the connected components are (on average) sparser and further apart, making them less likely to be connectable using a single mobile sensor. Second, there are more mobile sensors available to fill coverage gaps. In this scenario, with 50% mobile sensors, allowing up to $k = 2$ mobile sensors per virtual edge reduces the percolation threshold dramatically compared to $k = 1$.

VI. Conclusions

This paper introduces the k-connect barrier count problem, which involves finding the maximum number of distinct barriers in an arbitrary MSN. A specific goal behind this problem is to provide fault-tolerant barrier coverage using mobile sensors. A key constraint, though, is that at most k unique mobile sensors can be used to construct any given virtual edge in a barrier.

The solutions to the k-connect barrier count problem for $k \in \{0, 1, 2\}$ are provided through Integer Linear Program formulations. Through simulation experiments, we show that as k increases, the network size at which barrier coverage is achieved decreases. That is, mobile sensors are effective in healing coverage gaps, and their benefits are quantified. However, the benefits of mobile sensors depend on many factors, including the sensing radius, movement radius, and the number of mobile sensors available.

For future work, the next logical step is solving the k-connect barrier count problem for general k. Another interesting problem to consider is building the maximum number of barriers while minimizing power consumption.

Acknowledgements

The authors thank the anonymous IEEE MASCOTS 2012 reviewers for their constructive feedback on this paper, which helped to improve the clarity of the final version. Financial support for this work was provided by Canada's Natural Sciences and Engineering Research Council (NSERC).

References

[1] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar, "Reliable Density Estimates for Coverage and Connectivity in Thin Strips of Finite Length", *Proceedings of ACM Mobicom*, Montreal, QC, pp. 75–86, September 2007.

[2] A. Chen, S. Kumar, and T. Lai, "Designing Localized Algorithms for Barrier Coverage", *Proceedings of ACM Mobicom*, Montreal, QC, pp. 63–74, September 2007.

[3] D. Gage, "Command Control for Many-robot Systems", *Unmanned Systems*, Vol. 10, pp. 28–34, June 1992.

[4] S. Kumar, T. Lai, and A. Arora, "Barrier Coverage with Wireless Sensors" *Proceedings of ACM Mobicom*, Cologne, Germany, pp. 284–298, August 2005.

[5] X. Li, P. Wan, and O. Frieder, "Coverage in Wireless Adhoc Sensor Networks", *IEEE Trans. on Computers*, Vol. 52, pp. 753–763, June 2003.

[6] B. Liu, O. Dousse, J. Wang, and A. Saipulla, "Strong Barrier Coverage of Wireless Sensor Networks", *Proceedings of ACM MobiHoc*, Hong Kong, China, pp. 411–420, May 2008.

[7] S. Megerian, F. Koushanfar, G. Qu, G. Veltri, and M. Potkonjak, "Exposure in Wireless Sensor Networks: Theory and Practical Solutions", *Wireless Networks*, Vol. 8, pp. 443–454, 2002.

[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage Problems in Wireless Ad-hoc Sensor Networks", *Proceedings of IEEE INFOCOM*, Anchorage, AK, pp. 1380–1387, April 2001.

[9] A. Saipulla, B. Liu, G. Xing, X. Fu, and J. Wang, "Barrier Coverage with Sensors of Limited Mobility", *Proceedings of ACM MobiHoc*, Chicago, IL, pp. 201–210, September 2010.

[10] S. Srinivasa, *Barrier Coverage in Mixed Sensor Networks*, M.Sc. Thesis, University of Calgary, December 2011.

[11] G. Veltri, Q. Huang, G. Qu, and M. Potkonjak, "Minimal and Maximal Exposure Path Algorithms for Wireless Embedded Sensor Networks", *Proceedings of ACM SenSys*, Los Angeles, CA, pp. 40–50, November 2003.