A study of sensor movement and selection strategies for strong barrier coverage

by

Xiaoyun Zhang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering (Computing and Networking Systems)

Program of Study Committee:
Daji Qiao, Major Professor
Neil Zhenqiang Gong
Manimaran Govindarasu
Yong Guan
Sang W. Kim

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2017

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me on my research and the writing of this dissertation.

First and foremost, I would like to thank Dr. Daji Qiao for his guidance, patience and support throughout this research and the writing of this dissertation. Without these, this dissertation would not come forth. His insights and words of encouragement have often inspired me and renewed my hopes for completing my PhD program.

I would also like to thank my colleague Mat Wymore for his help on my paper. I would additionally like to thank my committee members for their suggestions on this research.

Last, I would like to express my gratitude to my family and friends who have always supported me.

ABSTRACT

Intruder detection and border surveillance are some of the many applications of sensor networks. In these applications, sensors are deployed along the perimeter of a protected area such that no intruder can cross the perimeter without being detected. The arrangement of sensors for this purpose is referred to as the barrier coverage problem in sensor networks. A primary question centering such a problem is: "How to achieve barrier coverage?" On the other hand, sensor nodes are usually battery-powered and have limited energy. It is critical to design energy-efficient barrier construction schemes while satisfying the coverage requirement.

First, we studied how to achieve strong barrier coverage with mobile sensors. We leverage the mobility of sensors and relocate them to designated destinations to form a strong horizontal barrier after the random deployment. Algorithms were proposed to calculate the optimal relocating destinations such that the maximum moving distance of sensors is minimized. Depending on the number of sensors on the final barrier, two problems were investigated: (1) constructing a barrier with the minimum number of sensors on the final barrier, and (2) constructing a barrier with any number of sensors on the final barrier. For both problems, we optimized the barrier location instead of fixing it a priori as other works. We proposed algorithms which first identify a set of discrete candidates for the barrier location, then check the candidates iteratively. Both problems could be solved in polynomial time.

Second, we investigated how to achieve strong barrier coverage by selectively activating randomly deployed static sensors. We aimed to select the minimum number of sensors to be active to achieve barrier coverage under a practical probabilistic model. The system false alarm probability and detection probability were jointly considered, and a $(P_D^{\min}, P_F^{\max})$-barrier coverage was defined where $P_D^{\min}$ is the minimum system detection probability and $P_F^{\max}$ is the maximum system false alarm probability. Our analysis showed that with the constraint on the system false alarm probability, the number of active sensors affects the detection capability

of sensors, which would bring new challenges to the min-num sensor selection problem. We proposed an iterative framework to solve the sensor selection problem under the probabilistic model. Depending on whether the decision fusion was applied, different detection capability evaluation methods were used in the iterative framework.

Finally, we studied how to achieve strong barrier coverage in a hybrid network with a mix of mobile and static sensors. A two-step deployment strategy was adopted where static sensors are first randomly deployed, and then mobile sensors are deployed to merge the coverage gap left by the static sensors. We aimed to find the proper coverage gaps to deploy mobile sensors such that $(P_D^{\min}, P_F^{\max})$-barrier coverage is achieved, and the total cost of the barrier is minimized. Under the probabilistic model, we solved the problem by iteratively trying multiple assumptions of the number of active sensors, and obtained the min-cost deployment strategy with the help of graph algorithms.

CHAPTER 1. INTRODUCTION

1.1 Coverage in Sensor Networks

Coverage is one of the fundamental and underlying functionalities of sensor networks. Almost all the applications of sensor networks require some level of coverage to support other functionalities. Various coverage problems have been investigated. These coverage problems differ from each other regarding the coverage types, coverage models, coverage qualities, sensor types and deployment strategies. In the following, we will discuss the various coverage problems in detail, with an emphasis on the barrier coverage problems.

1.1.1 Coverage Type

According to the goals of coverage, coverage in sensor networks can be classified into three categories: *area coverage*, *point coverage*, and *barrier coverage*.

1.1.1.1 Area Coverage

Area coverage aims to cover every point in the area of interest, as shown in Fig. 1.1.

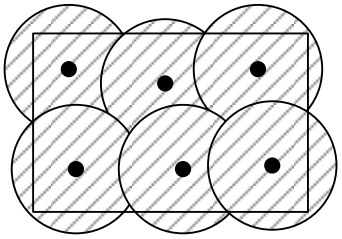


Figure 1.1 Area coverage.

The research works for area coverage have been focusing on (1) the optimal deterministic deployment pattern, (2) the relationship between sensor density and coverage ratio, and (3) energy-efficient sensor selection and scheduling algorithms. Kershner in [Kershner (1939)] proved that the optimal deterministic deployment pattern for area coverage which uses the

minimum number of sensors is the triangular lattice. Liu in [Liu and Towsley (2004)] revealed the relationship between the sensor density and the area coverage ratio, which can be used to estimate the number of sensors required to achieve a certain level of coverage. When sensors are randomly deployed, redundant sensors exist. Sensor selection and scheduling algorithms are then needed to achieve area coverage energy efficiently. In [Xing et al. (2005)] and [Zhang and Hou (2005)], the authors proposed distributed sensor scheduling algorithms with redundancy check and sleeping strategies to achieve area coverage.

1.1.1.2 Point Coverage

Point coverage aims to cover target points in the area of interest instead of the entire region, as shown in Fig. 1.2. These target points may be the locations of critical facilities or resources.

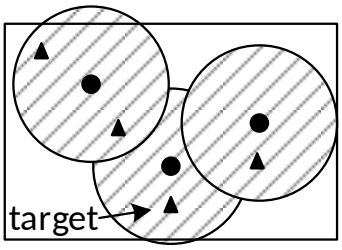
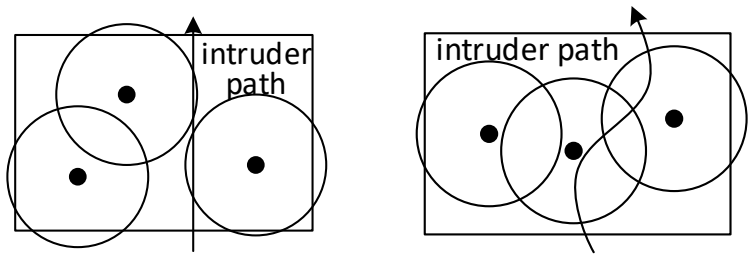


Figure 1.2 Point coverage.

Research works for point coverage have been focusing on (1) optimal placement of sensors, and (2) energy/cost-efficient sensor selection and scheduling algorithms. In [Chakrabarty et al. (2002); Wang and Zhong (2006); Xu and Sahni (2007)], the authors studied the sensor placement problem with the objective of minimizing the cost of sensors. If there are multiple sensor sets with each set itself can cover all the targets, scheduling algorithms which alternatively activate these sets are needed to maximize the coverage time. The authors in [Cardei et al. (2005a)] and [Wu et al. (2012)] investigated this problem under the disk and probabilistic model, respectively.

1.1.1.3 Barrier Coverage

Barrier coverage aims to cover the intruder paths which traverse the perimeter of the area of interest. If the perimeter of the area of interest is well protected by sensors, any intruder crossing the perimeter will be detected by the sensors placed along the perimeter. This kind of applications is referred as barrier coverage, and the sensors along the perimeter are called barrier. Fig. 1.3(b) shows a small part of the sensor barrier along the perimeter of the protected area. As we can see, any intruder who attempts to cross the perimeter will be detected by at least one sensor.



(a) Weak barrier coverage. (b) Strong barrier coverage

Figure 1.3 Barrier coverage.

There are two kinds of barrier coverage, weak barrier coverage and strong barrier coverage. In weak barrier coverage, the intruders are unaware of the sensor locations and hence will take the shortest path to traverse the perimeter of the area of the interest, as shown in Fig. 1.3(a). In strong barrier coverage, the intruders are aware of the sensor locations and may take any path to transverse the perimeter of the area of the interest, as shown in Fig. 1.3(b). It is easy to conclude that weak barrier coverage does not guarantee strong barrier coverage, while strong barrier coverage always guarantees weak barrier coverage.

Research works for barrier coverage have been focusing on (1) the critical sensor density to achieve barrier coverage (2) sensor selection and scheduling algorithms, and (3) constructing barriers with mobile sensors. Kumar and Liu investigated the critical sensor density problem for weak barrier coverage and strong barrier coverage in [Kumar et al. (2005)] and [Liu et al. (2008)], respectively. When sensors are randomly deployed, sensor selection and scheduling algorithms are needed to selectively activate redundant sensors to conserve energy. Kumar in

[Kumar et al. (2005)] showed that finding k barriers is equivalent to finding k node-disjoint paths in a graph. Liu in [Liu et al. (2008)] then proposed a more practical barrier construction algorithm based on divide-and-conquer. In [Kumar et al. (2007)], Kumar presented a barrier scheduling algorithm which maximizes the barrier lifetime. A comprehensive literature review for the sensor movement problems will be given in Chapter 2.

1.1.2 Coverage Models

Coverage model describes the coverage capability of a sensor. There are mainly two coverage models: the disk model and the probabilistic model.

1.1.2.1 Disk Model

In the disk model, the coverage region of a sensor is modeled as a disk centered at that sensor with coverage radius R, as shown in Fig. 1.4. By reason of its simplicity, the disk model is widely adopted by researchers. In the disk model, a target will be detected by a sensor if and only if the distance from the target to the sensor is no more than R.

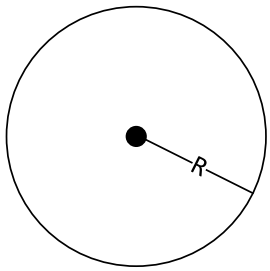


Figure 1.4 Disk model.

1.1.2.2 Probabilistic Model

In the probabilistic model, the coverage capability of a sensor is characterized by the probability that a target is detected by the sensor. For any point in the surrounding area of a sensor, if a target is locating at that point, then it can be detected by the sensor with a certain probability. The further the target from the sensor, the lower the probability that the target is detected by the sensor. As shown in Fig. 1.5, with the sensor locating at the center, a target

locating at the inner circle has a detection probability of 0.95, while a target locating at the outer circle has a detection probability of 0.9. Compared with the disk model, the probabilistic model is more realistic and approximates the real detection capability of a sensor better.

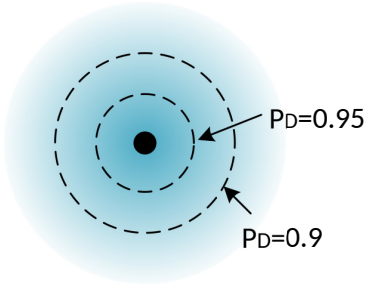


Figure 1.5 Probabilistic model.

Various probabilistic models have been proposed. In [Zou and Chakrabarty (2004), Zou and Chakrabarty (2005)], the detection probability is directly defined as a function of the distance between the target and the sensor. In [Clouqueur et al. (2004), Wang et al. (2007), Xing et al. (2009)], the detection probability is derived from the distribution of the sensor reading, whose average is determined by a signal attenuation model with the distance from the target to the sensor as a parameter. Detailed probabilistic models will be discussed in Chapter 3.

1.1.3 Quality of Coverage

For barrier coverage, coverage quality defines how well the perimeter of the area of interest is protected from intruders. Sorting by the coverage quality from low to high, there is partial coverage, full coverage and redundant k-coverage.

1.1.3.1 Partial Coverage

Partial coverage means only some intruder paths are covered. Breaches exist on the sensor barrier, and hence an intruder can penetrate the barrier without being captured. Nevertheless, only the paths which are rarely chosen by intruders are left uncovered so that partial coverage can prevent most of the intruders. For example, in [Chen et al. (2007)], Chen designed a scheme to provide partial barrier coverage which just covers paths expanding less than length l

horizontally. Partial coverage trades off the coverage quality against the number of the sensors required to achieve coverage.

1.1.3.2 Full Coverage

Full coverage means all intruder paths are covered. No breach exists on the sensor barrier and any intruder penetrating the barrier will be captured. Most of the applications aim to provide full barrier coverage. Under the disk model, this requires the coverage regions of adjacent sensors on the barrier overlap with each other.

1.1.3.3 Redundant k-coverage

Redundant k-coverage means all intruder paths are covered by k sensors. No breach exists on the sensor barriers and any intruder penetrating the barriers will be captured by at least k sensors. k-coverage is more robust than full coverage and resists to node failure. It trades off the coverage cost against coverage quality. With random deployed sensors, k-coverage can be achieved by finding k node-disjoint paths in a graph [Kumar et al. (2005)]. When more than k barriers exist, scheduling algorithms are needed to provide k-barrier coverage in an efficient way. In [Kumar et al. (2007)], the authors proposed such a scheduling algorithm which maximizes the coverage time.

1.1.4 Sensor Type

1.1.4.1 Homogeneous vs. Heterogeneous

Sensors may differ in many aspects, like coverage radius, power consumption, cost, and mobility. Coverage can be achieved with either homogeneous sensors or heterogeneous sensors. While it is much easier to achieve coverage with homogeneous sensors, heterogeneous sensors provide diverse sensing capabilities and are more flexible [Cardei et al. (2005b), Lazos and Poovendran (2006), Ammari and Giudici (2009)]. Sometimes we may need different types of sensors to sense various physical signals. We may want to cover some critical points with expensive but fine-grained sensors and cover the rest of a barrier with cheaper sensors. We can also use mobile sensors to enhance the coverage provided by static sensors.

1.1.4.2 Mobile vs. Static

Coverage can be achieved with mobile sensors, static sensors, or a combination of them. Static sensors have a lower cost while they cannot adjust their locations once deployed. Mobile sensors have a higher cost while they can relocate themselves to build a barrier or move constantly to patrol the perimeter of the area of interest. Usually, sensors are deployed randomly, if only static sensors are used, a large amount of sensors are required to fully cover an area or all intruder paths [Kumar et al. (2005), Liu et al. (2008)]. If mobile sensors are used, they can relocate themselves to achieve area or barrier coverage. Research works have been focusing on how to relocate mobile sensors energy-efficiently, for example, minimizing the moving distance of sensors. A third and better option might be utilizing both static and mobile sensors to achieve coverage. Mobile sensors can relocate themselves to merge the coverage holes left by static sensors.

1.1.4.3 Omnidirectional vs. Directional

Most sensor network applications use omnidirectional sensors like seismic sensor and microphone. An omnidirectional sensor has the same detection capability for all directions. Some sensor network applications may use directional sensors. A typical kind of directional sensors is the camera. The coverage area of a directional sensor can be described by a three-tuple composed of the coverage radius, the expanding angle, and the sensing direction [Wang et al. (2014b), Han et al. (2008)]. There are many differences between designing coverage schemes with directional sensors and designing coverage schemes with omnidirectional sensors. First, the method of determining whether two directional sensors overlap is more complicated than that of omnidirectional sensors. Second, when considering sensor movement, directional sensors can rotate while staying at the same location.

1.1.5 Deployment Strategies

1.1.5.1 Deterministic Deployment

Deterministic deployment can place sensor nodes at the exact desired locations, but it is labor-intensive. For area coverage, the optimal deterministic deployment pattern of using the minimum number of sensors is equilateral triangle tessellation [Kershner (1939)]. For barrier coverage, to minimize the number of sensors, we can place the sensors along the barrier, or the convex hull of the area of interest, without any overlap between sensor coverage regions.

1.1.5.2 Random Deployment

Random deployment is favored by most network designers. Sensor nodes can be dropped by an airplane which does not need much labors. Another reason is that in some harsh environments like battlefield and desert, it is even impossible to deploy sensor nodes manually and deterministically. However, random deployment usually requires a significant amount of sensors to be deployed to achieve full barrier coverage due to the randomness of sensor landing points [Kumar et al. (2005), Liu et al. (2008)]. In order to utilize the deployed sensors efficiently, strategies are designed to select and schedule subsets of sensors to be active to achieve barrier coverage [Kumar et al. (2007)].

1.1.5.3 Relocation and Patrolling

If sensors have the moving capability, then sensors can move to the desired locations to achieve barrier coverage. There are mainly two movement strategies: one-time relocation [Wang et al. (2004), Saipulla et al. (2010), Li and Shen (2015), Zhang et al. (2015)] and consistent patrolling [Liu et al. (2005), Yang et al. (2007), He et al. (2012)]. In the one-time relocation strategy, sensors are first randomly deployed; then they relocate to the desired locations to form a barrier. Once the sensors reach the destinations, they will stay there until the energy runs out. In the consistent patrolling strategy, sensors are scheduled to move constantly with objectives such as maximizing the intruder detection probability.

1.2 Research Main Contributions and Organization of the Dissertation

This dissertation focuses on barrier coverage. We will investigate three problems arisen when achieving barrier coverage: sensor movement problem, static sensor selection problem and sensor placement problem in a hybrid network.

Chapter 2 will study two sensor movement problems under the disk model. We aim to achieve strong barrier coverage with homogeneous mobile sensors. Sensors are omnidirectional. Mobile sensors are first randomly deployed in the area of interest, and then they relocate themselves to form a barrier. Our objective is to minimize the maximum moving distance of sensors and hence to prolong the lifetime of the barrier. Section 2.2 will study a sensor movement problem where only the minimum number of sensors are allowed on the barrier. Section 2.3 will relax the above constraint and allow any number sensors on the barrier to get a smaller moving distance.

Chapter 3 will investigate two sensor selection problems under a probabilistic model. We aim to select the minimal set of randomly deployed static sensors to achieve $(P_D^{\min}, P_F^{\max})$-barrier coverage. This means an intruder will be detected with a probability of at least $P_D^{\min}$ and the system false alarm probability will be no more than $P_F^{\max}$. Sensors are omnidirectional. Section 3.2 will study a sensor selection problem without data fusion. Section 3.3 will study a sensor selection problem with decision fusion applied among neighboring sensors and sampling points along intruder paths.

Chapter 4 will study how to achieve barrier coverage with a mix of mobile and static sensors, under a probabilistic model. Static sensors are first dropped, and then mobile sensors are deployed and relocate themselves to merge the coverage gaps left by the static sensors. The coverage gaps to be merged are carefully selected to minimize the overall cost of active sensors and achieve $(P_D^{\min}, P_F^{\max})$-barrier coverage.

Chapter 5 will conclude this dissertation and provide future research directions.

CHAPTER 2. ON MINIMIZING THE MAXIMUM SENSOR MOVEMENT FOR STRONG BARRIER COVERAGE UNDER DISK MODEL

2.1 Literature Review

2.1.1 Overview

Many research works focus on achieving barrier coverage with mobile sensors. Fig. 2.1 summarizes all the sensor movement problems in the area of barrier coverage under the disk model.

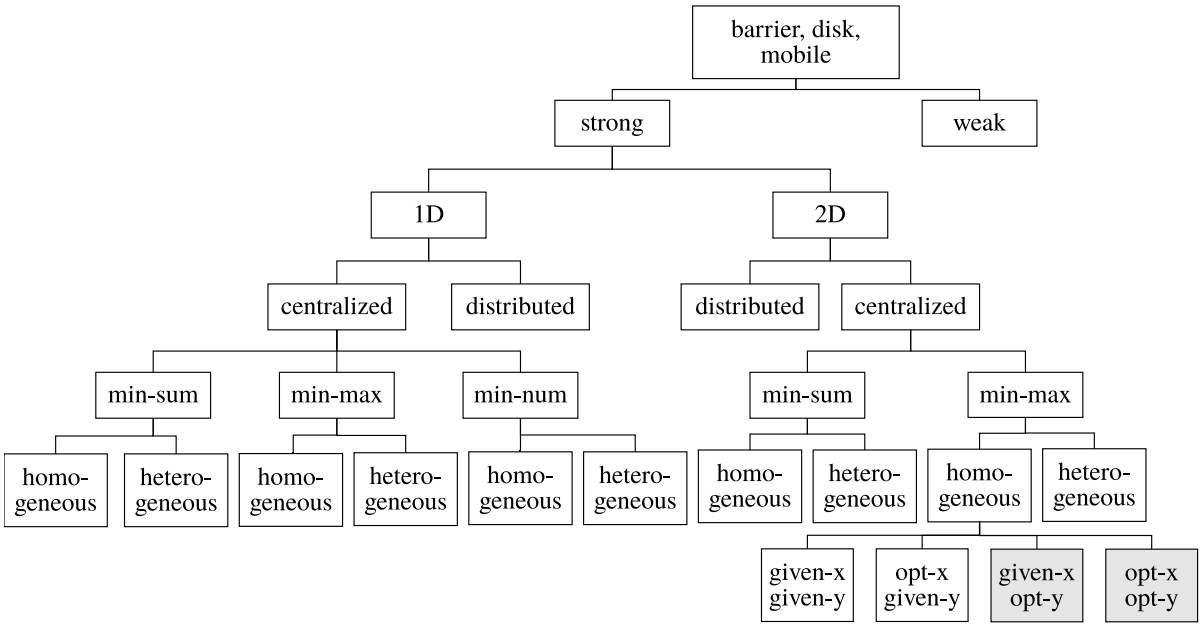


Figure 2.1 Sensor movement problems under the disk model. The "given-x, opt-y" problem was studied in Section 2.2. The "opt-x, opt-y" problem was studied in Section 2.3.

Sensors can move to form either strong or weak barriers. According to the dimensions of sensor deployment regions, the sensor movement problems can be classified into two categories:

1-dimensional (1D) sensor movement problems and 2-dimensional (2D) sensor movement problems. In 1D sensor movement problems, the sensors deployment and movement regions are restricted to a 1-dimensional line. In 2D sensor movement problems, the sensors deployment and movement regions are a 2-dimensional region. Centralized and distributed schemes have been proposed for the sensor movement problems. While the distributed schemes mainly focus on constructing barriers in a distributed manner, the centralized schemes have been focusing on constructing barriers energy-efficiently. To reduce the energy consumption on movements and therefore prolong the lifetime of the barrier, researchers have designed movement strategies with the objective of minimizing the maximum moving distance (min-max) or minimizing the total moving distance (min-sum) of sensors. In each sub-problem, sensors can either be homogeneous or heterogeneous.

Our work focused on building a strong horizontal barrier with homogeneous mobile sensors in a 2D rectangular region, with the objective of minimizing the maximum moving distance of sensors. In this branch, depending on whether the x-coordinates or y-coordinates of sensor final positions are given or optimized, there are four variants of such a problem as shown in Fig. 2.1. We studied and solved the "given-x, opt-y" and "opt-x, opt-y" problems in Section 2.2 and Section 2.3, respectively. Centralized algorithms were proposed to construct a strong horizontal barrier in the "given-x, opt-y" and "opt-x, opt-y" problems. We assume there is a central unit in the sensor network to which sensors send their initial positions. The proposed algorithms are run on the central unit. After obtaining the optimized final positions of sensors, the central unit disseminates them to sensors. In a multi-level hierarchical network, the workload of initial position collection, algorithm execution and final position dissemination can be divided and performed on lower-level central units, the higher-level central units are then responsible for merging the barriers formed by the lower-level central units.

In the following, we will give a comprehensive literature review for all the problems listed in Fig. 2.1.

2.1.2 Strong 1D Barrier Coverage

In the strong 1-D barrier coverage problems, sensors are initially deployed on a line, and they move along the line to form a barrier. Let L be the length of the line and N be the number of sensors deployed.

2.1.2.1 Centralized Solutions for the Min-max Problems

The maximum moving distance of sensors is minimized to maximize the lifetime of the barrier. Assuming all sensors have the same initial energy, the sensor with the maximum moving distance will die first and consequently breach the barrier, therefore, minimizing the maximum moving distance of sensors is equivalent to maximize the lifetime of the barrier.

Homogeneous Sensors Let R denote the identical coverage radius of sensors, and $N_{\min}$ denote the minimum number of sensors required to cover the line. $N_{\min}$ should be $\lceil L/2R \rceil$. Depending on the relationship between N and $N_{\min}$, we have the following variants of the min-max problem.

- $N = N_{\min}$ Bhattacharya in [Bhattacharya et al. (2009)] proposed an $O(N)$ algorithm to relocate $N_{\min}$ sensors to $(2i-1)R$ where $1 \leq i \leq N_{\min}$. The key of the algorithm is that the sensor final positions will preserve the order of the sensor initial positions on the barrier. Czyzowicz gave the same solution for this problem in [Czyzowicz et al. (2009)].
- $N < N_{\min}$ Czyzowicz in [Czyzowicz et al. (2009)] investigated the case when there are not enough sensors. Two objectives were considered: constructing the longest continuous barrier and constructing the longest non-continuous barrier. Both can be solved in $O(N)$ time.
- $N > N_{\min}$ This case has been extensively studied. Czyzowicz gave an $O(N^2)$ optimal, $O(N)$ 2-approximation, and $O\left(N\frac{\log(C/g)}{\log(1+\epsilon)}\right)$ $(1+\epsilon)$-approximation solution in [Czyzowicz et al. (2009)], where C and g are parameters related to the sensor initial positions and the line length L. Chen in [Chen et al. (2013a)] improved the time complexity of the optimal solution from $O(N^2)$ to $O(N \log N)$.

Heterogeneous Sensors Two kinds of heterogeneous sensor movement problems have been explored. In one set of problems, sensors have different coverage radii. In the other set of problems, the moving distances of sensors are multiplied with different weights, because sensors may have distinctive power consumption for movements.

Czyzowicz in [Czyzowicz et al. (2009)] showed that a variation of the min-max problem with heterogeneous coverage radii is NP-complete, in which one sensor is assigned a predetermined position. It was open according to Czyzowicz's paper that whether the original min-max problem with heterogeneous coverage radii is NP-complete.

Chen in [Chen et al. (2013a)] claimed that the min-max problem with heterogeneous coverage radii is not NP-complete and provided an $O(N^2 \log N \log\log N)$ solution for it.

Wang in [Wang and Zhang (2015)] solved a weighted min-max movement problem where the moving distances of sensors are multiplied with different weights. An $O(N^2 \log N \log\log N)$ solution was given.

2.1.2.2 Centralized Solutions for the Min-sum Problems

The sum of the moving distances of sensors is minimized to reduce the overall energy consumption for movement and hence to prolong the lifetime of the barrier.

Homogeneous Sensors Let R denote the identical coverage radius of sensors, and $N_{\min}$ denote the minimum number of sensor required to cover the line. Depending on the relationship between N and $N_{\min}$, we have the following variants of the min-sum problem.

- $N = N_{\min}$ Same as the min-max problem, the order of sensor final positions is the same as the order of sensor initial positions in the min-sum problem. Based on this fact, Czyzowicz in [Czyzowicz et al. (2010)] gave an $O(N)$ solution.
- $N < N_{\min}$ Similar to the min-max problem, Czyzowicz in [Czyzowicz et al. (2010)] investigated the case when not enough sensors are provided. Two objectives were considered: constructing the longest continuous barrier and constructing the longest non-continuous barrier. Both can be solved in $O(N)$ time.

- $N > N_{\min}$ An optimal $O(N^2)$ solution was proposed by Czyzowicz in [Czyzowicz et al. (2010)]. Andrew in [Andrews and Wang (2017)] improved the time complexity of the optimal solution from $O(N^2)$ to $O(N \log N)$.

Heterogeneous Sensors Similar to the min-max problem, two kinds of heterogeneous sensor movement problems have been explored. One kind of heterogeneous min-sum problem dealt with sensors with different coverage radii. The other one weighed the moving distances of sensors differently.

Czyzowicz in [Czyzowicz et al. (2010)] proved that the heterogeneous min-sum problem with different coverage radii is NP-complete, with the restriction that sensors initially locate on the line to be covered.

Benkoczi in [Benkoczi et al. (2015)] proved the heterogeneous min-sum problem with different coverage radii remains NP-complete when the initial sensor coverage ranges are disjoint from the line. They then proposed a $(1+\epsilon)$-approximation algorithm with a time complexity of $O(\frac{N^7}{\epsilon^3})$.

Benkoczi in [Benkoczi et al. (2016)] proposed a greedy-based 2-approximation algorithm to solve the weighted min-sum problem with arbitrary coverage radii, which runs in $O(N)$ time.

2.1.2.3 Centralized Solutions for the Min-num Problems

Mehrandish in [Mehrandish et al. (2011)] investigated both the homogeneous and heterogeneous min-num problem, where the number of moving sensors is minimized.

Homogeneous Sensors For the homogeneous case, two problems were considered: achieving the maximum coverage on an infinite line and achieving the maximum coverage on a finite line. For the infinite line coverage, the covered line segment can either be continuous or non-continuous. An $O(N)$ algorithm was proposed to achieve non-continuous maximum coverage, and an $O(N^2)$ algorithm was proposed to achieve continuous maximum coverage. For the finite line coverage, when $N < N_{\min}$, an $O(N^3)$ algorithm was proposed to achieve non-continuous maximum coverage, and an $O(N^2)$ algorithm was proposed to achieve continuous maximum

coverage. When $N = N_{\min}$, an $O(N)$ algorithm was proposed to achieve the maximum coverage. When $N > N_{\min}$, an $O(N^3)$ algorithm was proposed to achieve the maximum coverage.

Heterogeneous Sensors Similar to the heterogeneous case, achieving the maximum coverage on an infinite and finite line were studied. For the infinite line coverage, an $O(N \log N)$ algorithm was proposed to achieve non-continuous maximum coverage. In contrast, it is NP-complete to achieve the continuous maximum coverage. For the finite line coverage, all cases were proved to be NP-complete with a reduction to a partition problem.

2.1.2.4 Distributed Solutions

Unlike the centralized solutions, most of the distributed solutions do not have a specific objective other than forming a barrier. They aim to form a barrier in a distributed manner with communications between neighboring sensor nodes.

Eftekhari in [Eftekhari et al. (2013)] presented two simple distributed algorithms which achieve barrier coverage with $N \geq N_{\min}$ homogeneous sensors. The algorithms are synchronized. Sensors have limited visibility and can only move constant distance in a time slot. One algorithm is stateless where sensors do not know the moving direction in the previous time slot and converges in $O(N^2)$ time. The other algorithm is stateful where sensors memorize the moving directions in the previous time slot and converges in $O(N)$ time.

2.1.3 Strong 2D Barrier Coverage

In the strong 2D barrier coverage problems, sensors are initially deployed in a 2-dimensional long belt region and they move in the 2-dimensional region to form a barrier. In the following, we use L to denote the length of the belt region and W to denote the width of it. The coverage radius of sensors is R if all sensors are homogeneous.

2.1.3.1 Centralized Solutions for the Min-max Problems

Homogeneous Sensors Assume the x axis is along the length of the belt, the y axis is along its width. We classify the 2D min-max barrier coverage problems to four categories

depending on whether the x-coordinates and y-coordinates of final sensor positions are given. "Given-x" and "Given-y" mean the x or y coordinates of final sensor positions are known as a priori, respectively. "Opt-x" and "Opt-y" mean the x or y coordinates of final sensor positions are unknown and optimized by the algorithm, respectively.

- Given-x, given-y In [Saipulla et al. (2010)], an algorithm was proposed to relocate the minimum number of sensors to form a horizontal barrier with the min-max objective function. Since only the minimum number of sensors are used, the x-coordinates of final sensor positions can be pre-calculated as $\{R, 3R, 5R, ...\}$, as shown in Fig. 2.2. The barrier location, i.e., the identical y-coordinate of final sensor positions, is assumed to be fixed and given. Note although the set of x-coordinates of final sensor positions are known, the match between the sensor initial and the final positions is unknown.

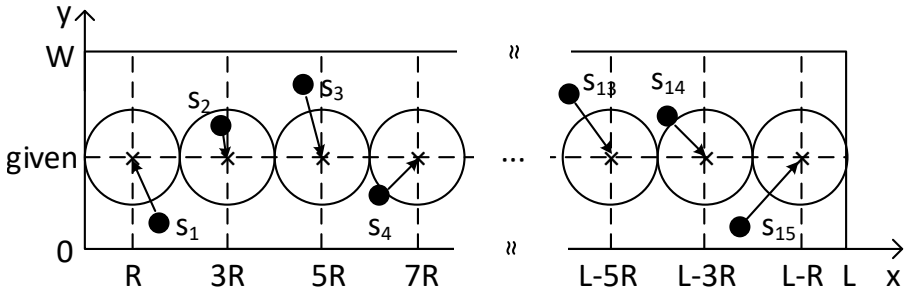


Figure 2.2 Given x, given y.

The proposed algorithm combined the bipartite matching algorithm and the binary search framework. Bipartite matching is used to get a match between the sensor initial and final positions given a limited maximum moving distance; binary search is employed to adjust the maximum moving distance.

- Opt-x, given-y The authors of [Li and Shen (2015)] also proposed an algorithm to relocate sensors to form a horizontal barrier with the min-max objective function. Compared with [Saipulla et al. (2010)], any number of sensors can be on the final barrier in [Li and Shen (2015)], as shown in Fig. 2.3. In consequence, the x-coordinates of final sensor positions are unknown, and they are optimized by the algorithm so that the maximum moving distance is minimized. On the other hand, same as [Saipulla et al. (2010)], the barrier location is assumed to be fixed and given.

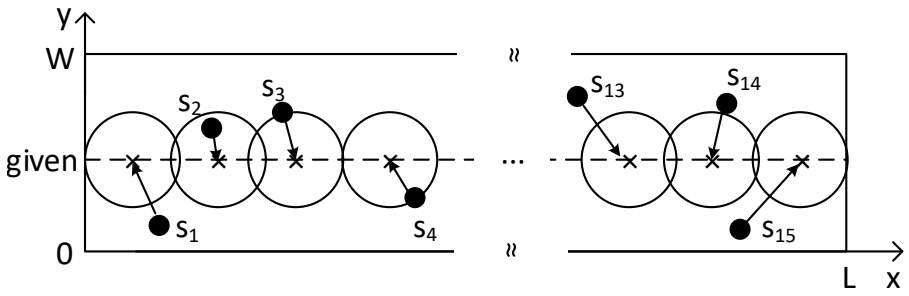


Figure 2.3 Optimized x, given y.

The essential of the solution in [Li and Shen (2015)] is a greedy algorithm which solves a feasibility problem, that is, whether there is a sensor movement strategy to achieve barrier coverage with a limited maximum moving distance. The greedy algorithm covers the horizontal barrier from the left to the right. In each step of the greedy algorithm, a sensor is selected and assigned a final position so that the total coverage length to the left is maximized.

- Given-x, opt-y In our work [Zhang et al. (2015)], we proposed an algorithm to relocate the minimum number of sensors (hence known x-coordinates of final sensor positions) to form a horizontal barrier with the min-max objective function without, however, a pre-decided barrier location, as shown in Fig. 2.4. Compared with the "given x, given y" problem, we optimized the barrier location, and hence the maximum moving distance will be smaller.

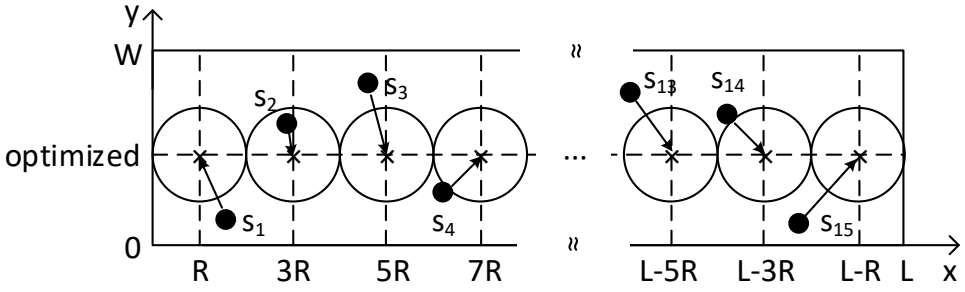


Figure 2.4 Given x, optimized y.

To solve the problem, we first identified a discrete set of candidates for the final barrier location, and then proposed an iterative algorithm to search over the discrete set to find the optimal one. The detailed algorithm will be introduced in Section 2.2.

- Opt-x, opt-y In this problem, the goal is to relocate sensors to form a horizontal barrier with the min-max objective function, without either pre-decided x-coordinates of the final sensor positions or pre-decided barrier location, as shown in Fig. 2.5. Any number of sensors can be on the final barrier. This problem was studied in Section 2.3.

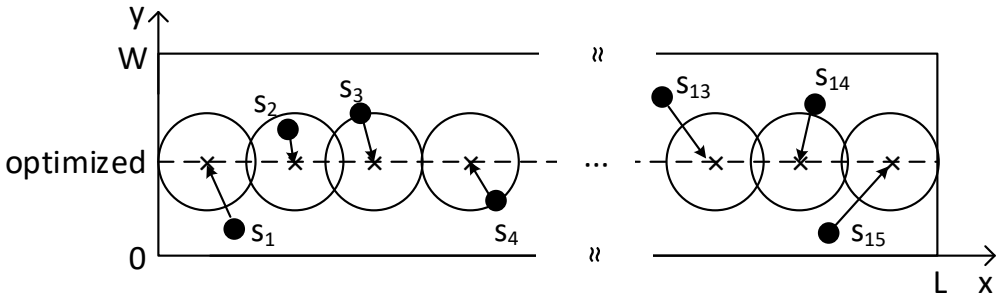


Figure 2.5 Optimized x, optimized y.

Heterogeneous Sensors In [Dobrev et al. (2015)], Dobrev proved that with the objective of minimizing the maximum moving distance, to cover one or more given barriers with heterogeneous sensors which have different coverage radii is NP-complete.

2.1.3.2 Centralized Solutions for the Min-sum Problems

Homogeneous Sensors Shen in [Shen et al. (2008)] studied a 2D min-sum sensor movement problem without knowing either the x or y coordinates of final sensor positions. Instead, they assumed the final barrier is a straight line, and the final sensor positions distribute evenly along the barrier line, and the sensors preserve their order along the x-axis. With all these assumptions, the min-sum problem could be modeled as a convex optimization problem.

Ban in [Ban et al. (2010)] proved the 2D min-sum sensor movement problem is NP-hard. Alternatively, they constructed a horizontal barrier with given x-coordinates of final sensor positions. The barrier location is optimized heuristically to reduce the total moving distances

of sensors. The match between the sensor initial and final positions on the barrier is obtained by solving a bipartite weighted problem with Hungarian method.

Heterogeneous Sensors In [Dobrev et al. (2015)], Dobrev proved that with the objective of minimizing the sum of moving distances, to cover one or more given barriers with heterogeneous sensors which have different coverage radii is NP-complete.

2.1.3.3 Distributed Solutions

A set of classical distributed algorithms to achieve barrier coverage in 2D region are "virtual force" based. The concept of virtual force was proposed by Shen in [Shen et al. (2008)]. They assume the sensors on the 2D plane have attractive forces along the y axis and repulsive forces along the x axis to each other. The sensors move based on the virtual forces from their neighbors. Finally, sensors will distribute evenly on a line parallel to the x-axis.

Cheng in [Cheng and Savkin (2009)] proposed a distributed algorithm based on consensus theory which relocates sensors from the two ends of a line to the line. By communicating with the immediate adjacent sensors only, sensors eventually distribute evenly along the line.

2.1.4 Weak Barrier Coverage

Weak barrier coverage covers the shortest intruder paths traversing the area of interest. For a long belt region, weak barrier coverage covers just the paths which are perpendicular to the length of the belt. As long as the projections of sensor coverage regions cover the length of the belt, the belt is weak barrier covered. Since the y-coordinates of sensors have no effect on the position of the projections of sensor coverage regions, achieving 1D/2D weak barrier coverage is equivalent to achieving 1D strong barrier coverage. If sensors are initially deployed in a 2D region, sensors need to move along the x-axis only to achieve weak barrier coverage.

2.2 On Minimizing the Maximum Sensor Movement for Horizontal Barrier Construction with the Minimum Number of Sensors

2.2.1 Introduction

In this work, we will study a strong barrier construction problem in a 2D rectangular region. Homogeneous mobile sensors are employed to construct the barrier. After the random deployment of mobile sensors, the minimum number of sensors relocate to form a strong barrier along a horizontal line that spans the length of the deployment region. We aim to minimize the maximum moving distance of sensors.

Since only the minimum number of sensors are utilized, the x-coordinates of the sensors on the final barrier can be pre-calculated as $\{R, 3R, 5R, ...\}$ where R is the coverage radius of sensors. The y-coordinates of the sensors on the final barrier are identical yet unknown. We define the identical y-coordinates as the *barrier location*. The unknown barrier location distinguishes this problem from the strong barrier construction problem studied in [Saipulla et al. (2010)] where the barrier location was given.

We will present a centralized algorithm that decides the optimal barrier location and identifies a minimal subset of sensors to move to their corresponding destinations so that the maximum moving distance of sensors is minimized. Our algorithm first discretizes the search space for the barrier location, then quickly iterates over the entire search space by identifying the subset of barrier location candidates that need to be checked.

The proposed algorithm was evaluated regarding the sensor maximum moving distance and time complexity. Simulation results showed that our algorithm outperforms the algorithm proposed [Saipulla et al. (2010)] regarding the maximum moving distance since we optimized the barrier location.

2.2.2 Model and Problem Statement

2.2.2.1 System Model

Sensor Network We study a network of N mobile sensors deployed in a long rectangular region of size $L \times W$, where $L \gg W$, as shown in Fig. 2.6. Sensors are named s_1 to s_N from

left to right of the deployment region, and the initial position of sensor s_i is denoted by (x_i, y_i). The set of all the sensors is denoted by S. We adopt the widely used disk coverage model and denote the coverage radius as R. An intruder can be detected by a sensor if and only if it is within R of the sensor. In addition, we assume sensors can acquire their locations from GPS or other localization schemes.

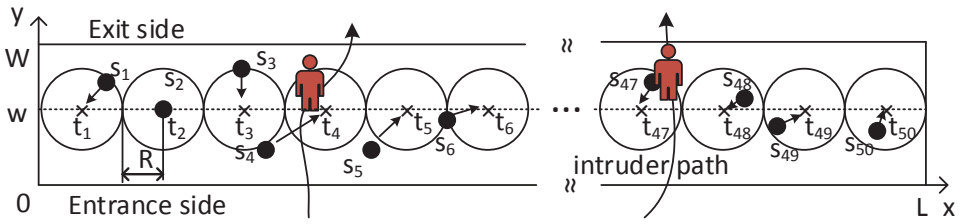


Figure 2.6 An example sensor network of 50 mobile sensors.

Intruder and Barrier We assume that intruders may take any path to cross the deployment region from bottom to top, as shown in Fig. 2.6. In order to detect such intruders, *strong barrier coverage* is required. This is in contrast to *weak barrier coverage*, which assumes intruders take only paths perpendicular to the x-axis of the deployment region.

The minimum number of sensors needed to form a strong barrier is $N_{\min} = \lceil \frac{L}{2R} \rceil$. For simplicity, and without loss of generality, we assume that L is a multiple of $2R$; hence, $N_{\min} = \frac{L}{2R}$. To form a strong barrier with $N_{\min}$ sensors, which is the focus of our study, these sensors must be aligned along a horizontal line parallel to the x-axis of the deployment region. In other words, the destination positions of these sensors (denoted by t_j, $1 \leq j \leq N_{\min}$) must have the coordinates of (α_j, w_j), where $\alpha_j = (2j-1)R$, $1 \leq j \leq N_{\min}$, and $w_1 = w_2 = \cdots = w_{N_{\min}}$. We use T to denote the set of destination positions, and use w to denote their common y-coordinate, called the *barrier location*.

System We assume that the sensor network remains connected during sensor movement, and that there is a central processing unit which collects information from sensors, executes the proposed algorithm, and disseminates the movement strategy to sensors.

2.2.2.2 Problem Statement

Our ultimate goal is to maximize the lifetime of the barrier. We assume all sensors have the same amount of energy initially, which is E_{total}. We also assume the sensor which has the least remaining energy after movement runs out of energy first, and consequently breaches the barrier. To maximize the lifetime of the barrier, we need to maximize the minimum remaining energy, i.e., $\min\limits_{s_i \in S, M(i) \neq 0} \left(E_{\text{total}} - P_m \sqrt{\left(x_i - \alpha_{M(i)}\right)^2 + (y_i - w)^2} \right)$, where P_m is the energy consumed for moving one unit and M is a mapping function from S to T. In general, a mapping function $M : S \to T$ is defined as follows: $M(i) = j \neq 0$ means that sensor s_i moves to destination t_j, while $M(i) = 0$ means that sensor s_i remains stationary. Sensors not used in the initial barrier may participate in forming future barriers after the operational lifetime of the current barrier has expired. The above problem is equivalent to minimizing the maximum moving distance of sensors, i.e., $\max\limits_{s_i \in S, M(i) \neq 0} \sqrt{\left(x_i - \alpha_{M(i)}\right)^2 + (y_i - w)^2}$.

To minimize the maximum moving distance of mobile sensors that form a strong barrier, with selected $N_{\min}$ from N ($N \geq N_{\min}$) sensors with known initial positions, our algorithm must decide (1) where the barrier shall be formed, i.e., the optimal barrier location, denoted by w^{opt}; and (2) how to form the barrier at w^{opt}, i.e., the optimal sensor movement strategy M^{opt} from S to T. Formally, the problem we try to address is described as follows:

Given:

- rectangular deployment region: $L \times W$
- sensing range: R
- total number of deployed sensors: N
- initial sensor positions: (x_i, y_i) for each $s_i \in S$
- x-coordinate of the destinations: α_j for each $t_j \in T$

Output:

- optimal barrier location w^{opt} and sensor movement strategy M^{opt}: $\{w^{\text{opt}}, M^{\text{opt}}\} = \arg\min\limits_{\{w,M\}} D(M, w)$ where

$$D(M, w) = \max_{s_i \in S, M(i) \neq 0} \sqrt{\left(x_i - \alpha_{M(i)}\right)^2 + (y_i - w)^2}.$$

Constraint: $0 \leq w^{\text{opt}} \leq W$

We use $D(M, w)$ to denote the maximum moving distance of the sensors when they follow strategy M to form a barrier at location w. We restrict the barrier location between 0 and W.

2.2.3 Proposed Scheme

2.2.3.1 Overview of the Proposed Scheme

To determine the optimal barrier location w^{opt}, we first reduce its search space from a continuous range $[0, W]$ to a discrete set with less than $N^2 N_{\min}^2$ points, which is then called *candidate barrier locations*. Afterwards, we propose an efficient iterative algorithm to search over this discrete set. Even though in the worst case we may need $O(N^2 N_{\min}^2)$ iterations to complete the search, in practice, the number of iterations is approximately $O(N N_{\min})$.

2.2.3.2 Identification of Candidate Barrier Locations

The candidate barrier locations are derived from the minimum and intersection points of a group of functions defined below.

Definition 2.1 (Function of Moving Distance)**.** *Suppose a sensor s_i moves to a destination t_j, the moving distance of s_i can be represented as a function of the barrier location w, i.e.,*

$$f_{i,j}(w) = \sqrt{[x_i - (2j-1)R]^2 + (y_i - w)^2} \tag{2.1}$$

where $0 \leq w \leq W$.

We use F to denote the set of all the functions of moving distance associated with a sensor network of N sensors, i.e.,

$$F = \{f_{i,j}(w) | 1 \leq i \leq N, 1 \leq j \leq N_{\min}, 0 \leq w \leq W\}. \tag{2.2}$$

For simplicity, we also define $f_{i,0}(w) = 0, \forall i, \forall w$.

Definition 2.2. *[Candidate Barrier Locations] The set of candidate barrier locations is $\Phi = \Phi^{mins} \cup \Phi^{ints}$, where*

- $\Phi^{mins} = \bigcup_{\forall f_{i,j} \in F} \{w | \arg\min_{w} f_{i,j}(w)\}$ *includes all the* w *values that yield a minimum value for at least one of the* $f_{i,j}$ *functions in* F*;*
- $\Phi^{ints} = \bigcup_{\forall f_{i,j} \in F, f_{m,n} \in F} \{w | f_{i,j}(w) = f_{m,n}(w)\}$ *includes all the* w *values where two functions in* F *intersect with each other.*

Let $|\Phi|$ denote the total number of candidates in Φ. We sort these candidates in an ascending order and name them as: $\{K_i |\ i = 1, \cdots, |\Phi|\}$. Let $|\Phi^{\text{mins}}|$ and $|\Phi^{\text{ints}}|$ denote the number of candidates in Φ^{mins} and Φ^{ints}, respectively. Then we have:

$$|\Phi| = |\Phi^{\text{mins}}| + |\Phi^{\text{ints}}| \leq NN_{\min} + \binom{NN_{\min}}{2} = O(N^2 N_{\min}^2) = O(N^4). \tag{2.3}$$

Fig. 2.7 plots all the functions in F corresponding to a sensor network shown in Fig. 2.8. In this example, $|\Phi^{\text{ints}}| = 10$, $|\Phi^{\text{mins}}| = 3$, and $|\Phi| = 13$.

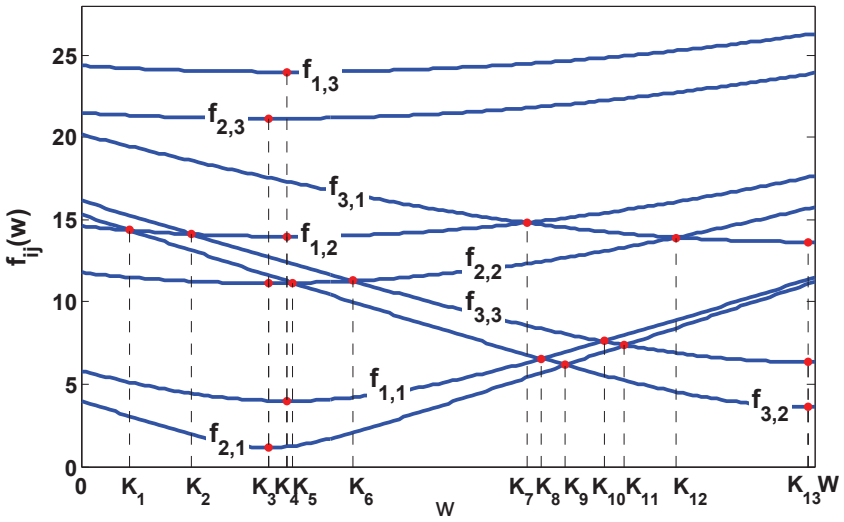


Figure 2.7 There is a total of 9 functions in F and 13 candidate barrier locations. $\Phi^{\text{ints}} = \{K_1, K_2, K_5, K_6, K_7, K_8, K_9, K_{10}, K_{11}, K_{12}\}$, $\Phi^{\text{mins}} = \{K_3, K_4, K_{13}\}$.

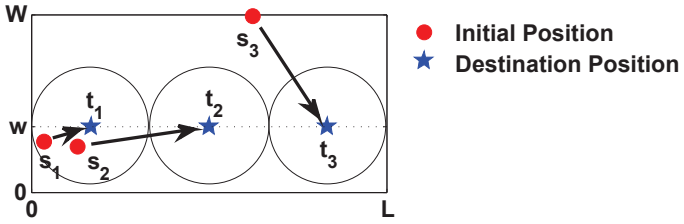


Figure 2.8 An example sensor network of 3 mobile sensors.

The optimal barrier location w^{opt} is guaranteed to be at one of the candidate barrier locations, as Theorem 2.1 below proves that any movement strategy M for a particular barrier location w (between two adjacent candidate locations) can always yield a smaller maximum moving distance by setting the barrier location instead to one of these two candidates.

Theorem 2.1. *$\forall M$, $\forall w \in (K_j, K_{j+1})$, where $1 \leq j \leq |\Phi|-1$, $D(M,w) > \min\{D(M,K_j), D(M,K_{j+1})\}$.*

Proof: Let s_{i^*} denote the sensor that has the maximum moving distance in M at w. This means

$$\forall i,\ f_{i,M(i)}(w) \leq f_{i^*,M(i^*)}(w). \tag{2.4}$$

Since K_j and K_{j+1} are adjacent candidate locations, no other functions in F intersect between them. Therefore, (2.4) implies that

$$\begin{cases} f_{i,M(i)}(K_j) \leq f_{i^*,M(i^*)}(K_j),\ \forall i, \\ f_{i,M(i)}(K_{j+1}) \leq f_{i^*,M(i^*)}(K_{j+1}),\ \forall i. \end{cases} \tag{2.5}$$

In other words,

$$\begin{cases} f_{i^*,M(i^*)}(K_j) = \max\limits_{1\leq i\leq N} f_{i,M(i)}(K_j), \\ f_{i^*,M(i^*)}(K_{j+1}) = \max\limits_{1\leq i\leq N} f_{i,M(i)}(K_{j+1}). \end{cases} \tag{2.6}$$

Moreover, according to the definition of candidate barrier locations, all the functions in F are monotone between two adjacent candidates such as K_j and K_{j+1}. Therefore, we have:

$$\begin{aligned} D(M,w) &= \max_{1\leq i\leq N} f_{i,M(i)}(w) = f_{i^*,M(i^*)}(w) && \text{(Definition)} \\ &> \min\left\{f_{i^*,M(i^*)}(K_j), f_{i^*,M(i^*)}(K_{j+1})\right\} && \text{(Monotone)} \\ &= \min\left\{\max_{1\leq i\leq N} f_{i,M(i)}(K_j), \max_{1\leq i\leq N} f_{i,M(i)}(K_{j+1})\right\} \\ &= \min\{D(M,K_j), D(M,K_{j+1})\}. \end{aligned} \tag{2.7}$$

■

2.2.3.3 Iterative Algorithm

With the candidate barrier locations being identified, we further reduce the search complexity by proposing an iterative algorithm in which only a small portion of candidates are checked. Fig. 2.9 shows the flowchart of our iterative algorithm, which is explained in detail next.

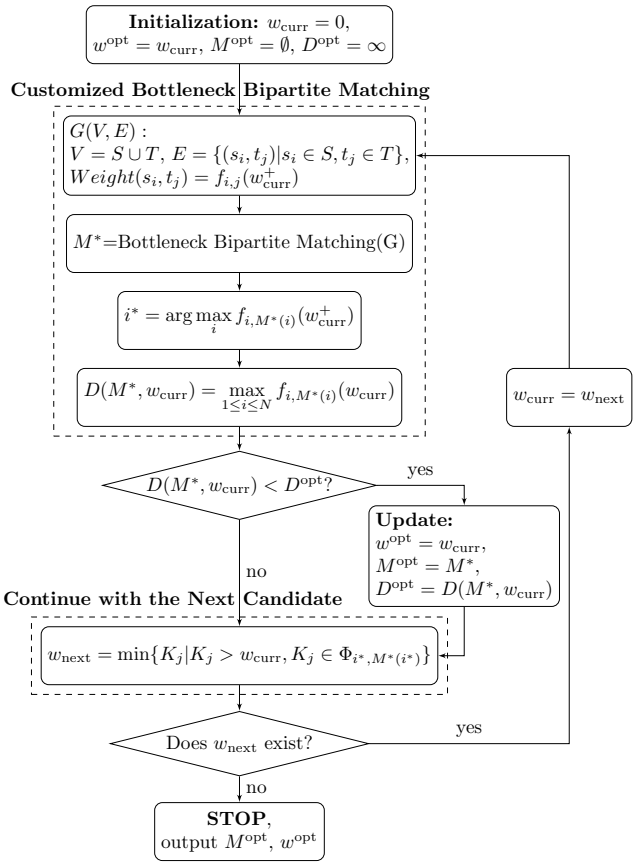


Figure 2.9 Flowchart of the iterative algorithm (note that $w^+_{\text{curr}} = w_{\text{curr}} + \delta w$).

The algorithm starts with $w_{\text{curr}} = 0$. After initialization, with the optimal minimax moving distance D^{opt} set to infinity, the Bottleneck Bipartite Matching (BBM) algorithm [Punnen and Nair (1994)] is applied to determine the best movement strategy for the current barrier location as follows. The input to BBM is a weighted bipartite graph $G(V, E)$:

- $V = S \cup T$ where S is the set of sensors, T is the set of destinations along w_{curr};
- $E = \{(s_i, t_j) |\ s_i \in S, t_j \in T\}$;
- Weight$(s_i, t_j) = f_{i,j}(w^+_{\text{curr}})$ where $w^+_{\text{curr}} = w_{\text{curr}} + \delta w$ and δw is a positive offset which is sufficiently small so that w^+_{curr} does not reach the next candidate in Φ. Recall that $f_{i,j}$ is the function of moving distance for sensor s_i to reach destination t_j.

BBM returns a max-cardinality matching whose maximum edge weight is minimized. In other words, it produces a movement strategy which minimizes the maximum moving distance of sensors from their initial positions to the destination positions along w_{curr}.

In this customized BBM algorithm, the offset δw is used as a tie breaker in case there are multiple movement strategies that all yield the same minimax moving distance at w_{curr}. For example, in Fig. 2.10, at $w_{\text{curr}} = K_2$, there exist two such movement strategies: $M_1 = \{1\to2, 2\to1, 3\to 3\}$ and $M_2 = \{1\to1, 2\to2, 3\to3\}$. By adding a small offset, the tie is broken and M_2 is chosen which yields a smaller maximum moving distance than M_1 at K_2^+. This is important as it serves as the basis for the next round of iteration. In detail, the start of the next iteration is determined based on the minimax matching at w_{curr}^+, which is introduced and proved in the following.

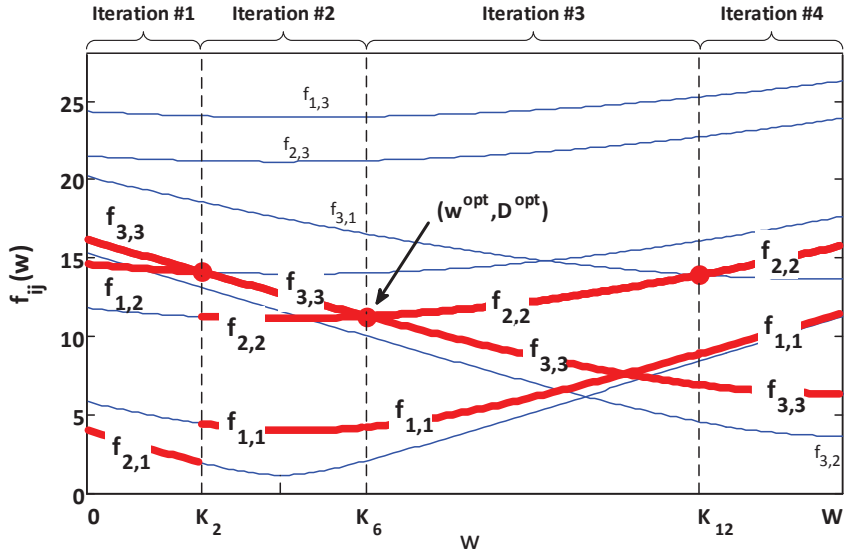


Figure 2.10 The iterative algorithm starts with $w_{\text{curr}} = 0$. Iteration #1: The minimax matching at w_{curr}^+ is $\{1\to2, 2\to1, 3\to3\}$, in which s_3 has the maximum moving distance at w_{curr}^+. The next candidate to check is the first candidate location along $f_{3,3}$ after $w_{\text{curr}} = 0$, which is K_2. Iteration #2: The minimax matching at K_2^+ is $\{1\to1, 2\to2, 3\to3\}$, in which s_3 has the maximum moving distance at K_2^+. The next candidate to check is the first candidate along $f_{3,3}$ after K_2, which is K_6. The same process repeats for Iteration #3 and Iteration #4 till the algorithm terminates.

As shown in the flowchart, we record the output of the customized BBM algorithm as follows:

- M^*: the minimax matching at w_{curr}^+;
- i^*: the index of the sensor that has the maximum moving distance in M^* at w_{curr}^+;
- $D(M^*, w_{\text{curr}})$: the minimax moving distance at w_{curr}.

The next candidate location to check is

$$w_{\text{next}} = \min\{K_j | K_j > w_{\text{curr}}, K_j \in \Phi_{i^*,M^*(i^*)}\}, \tag{2.8}$$

where $\Phi_{i^*,M^*(i^*)}$ is the set of candidates along $f_{i^*,M^*(i^*)}$ which is formally defined below.

Definition 2.3 (Candidate Barrier Locations along $f_{i,j}$)**.** *The set of candidate barrier locations along the $f_{i,j}$ function is $\Phi_{i,j} = \Phi_{i,j}^{mins} \cup \Phi_{i,j}^{ints}$, where*

- $\Phi_{i,j}^{mins} = \{w | \arg\min_w f_{i,j}(w)\}$ *includes the w value where $f_{i,j}$ achieves its minimum;*
- $\Phi_{i,j}^{ints} = \bigcup_{\forall f_{m,n} \in F} \{w | f_{i,j}(w) = f_{m,n}(w)\}$ *includes all the w values where $f_{i,j}$ intersects with another function in F.*

Note that w_{next} is the first candidate location along the $f_{i^*,M^*(i^*)}$ function after w_{curr}. There might exist other candidate locations between w_{curr} and w_{next} but belong to a different f function, which are skipped in our iterative algorithm to reduce the search complexity. For example, in Fig. 2.10, the next candidate to check after K_2 is K_6. It is determined with the following steps: (1) the minimax matching at K_2^+ is $M^* = \{1{\to}1, 2{\to}2, 3{\to}3\}$; (2) the sensor that has the maximum moving distance in M^* at K_2^+ is s_3; (3) the first candidate along $f_{3,3}$ after K_2 is K_6. (Note here if we choose a minimax matching at K_2 instead of K_2^+, which may be $M = \{1{\to}2, 2{\to}1, 3{\to}3\}$, then by following the above three steps, we may search along a wrong function $f_{1,2}$ to find the next candidate.) Comparing Fig. 2.10 with Fig. 2.7, we can see that candidate locations K_3, K_4, and K_5 are skipped. The reason why these candidate locations can be skipped is that, as we will show in Theorem 2.2, at any w between w_{curr} and w_{next}, the maximum moving distance of any movement strategy is always larger than or equal to that of M^* at w_{curr} or w_{next}.

Theorem 2.2. *$\forall M'$, $\forall w \in [w_{curr}, w_{next}]$, where w_{curr} is the current candidate location in the iterative algorithm, and w_{next} is the next candidate location to check as defined in (2.8), we always have $D(M', w) \geq \min\{D(M^*, w_{curr}), D(M^*, w_{next})\}$, where M^* is the minimax matching at w_{curr}^+.*

Proof: Recall that i^* is the index of the sensor that has the maximum moving distance in M^* at w^+_{curr}, i.e.,

$$i^* = \arg\max_i f_{i,M^*(i)}(w^+_{\text{curr}}). \tag{2.9}$$

Therefore, the following inequality holds for all i:

$$f_{i^*,M^*(i^*)}(w^+_{\text{curr}}) \geq f_{i,M^*(i)}(w^+_{\text{curr}}). \tag{2.10}$$

As w_{next} is the first candidate location along the $f_{i^*,M^*(i^*)}$ function after w_{curr}, no other functions in F intersect with $f_{i^*,M^*(i^*)}$ between w_{curr} and w_{next}. Consequently, we have:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}], \forall i, f_{i^*,M^*(i^*)}(w) \geq f_{i,M^*(i)}(w). \tag{2.11}$$

Hence, we have:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}], f_{i^*,M^*(i^*)}(w) = D(M^*, w). \tag{2.12}$$

On the other hand, let i' denote the index of the sensor that has the maximum moving distance in M' at w^+_{curr}, i.e.,

$$i' = \arg\max_i f_{i,M'(i)}(w^+_{\text{curr}}). \tag{2.13}$$

As M^* is the minimax matching at w^+_{curr}, we have:

$$f_{i',M'(i')}(w^+_{\text{curr}}) \geq f_{i^*,M^*(i^*)}(w^+_{\text{curr}}). \tag{2.14}$$

Similarly, due to the fact that no other functions in F intersect with $f_{i^*,M^*(i^*)}$ between w_{curr} and w_{next}, we have:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}], f_{i',M'(i')}(w) \geq f_{i^*,M^*(i^*)}(w). \tag{2.15}$$

As the definition of $D(M', w)$ implies:

$$D(M', w) = \max_{1 \leq i \leq N} f_{i,M'(i)}(w) \geq f_{i',M'(i')}(w), \tag{2.16}$$

we have:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}], D(M', w) \geq f_{i^*,M^*(i^*)}(w). \tag{2.17}$$

Furthermore, as the $f_{i^*,M^*(i^*)}$ function is monotone between w_{curr} and w_{next}, (2.17) implies:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}],\ D(M', w) \geq \min\{f_{i^*,M^*(i^*)}(w_{\text{curr}}), f_{i^*,M^*(i^*)}(w_{\text{next}})\}. \tag{2.18}$$

Combining (2.12) with (2.18), we have:

$$\forall w \in [w_{\text{curr}}, w_{\text{next}}],\ D(M', w) \geq \min\{D(M^*, w_{\text{curr}}), D(M^*, w_{\text{next}})\}. \tag{2.19}$$

■

Finally, the algorithm terminates when w_{next} cannot be found, meaning that we have completed the search over the entire range $[0, W]$. Fig. 2.10 illustrates the iterative algorithm with an example and the iteration process is explained in the caption of the figure.

2.2.3.4 Complexity Analysis

In theory, there is a total of $O(N^2 N_{\min}^2)$ candidates in Φ, which means that, in the worst case, we may need $O(N^2 N_{\min}^2)$ number of iterations to complete the search. However, in practice, the number of iterations is more comparable to $O(NN_{\min})$. This is because our algorithm essentially checks the candidate barrier locations iteratively along a single continuous function that is composed of multiple sections from different f functions. Thus, the total number of checked locations is on the same order as the number of candidate locations along a single f function, which is $O(NN_{\min})$. In Section 2.2.4, we validate this complexity analysis by simulation.

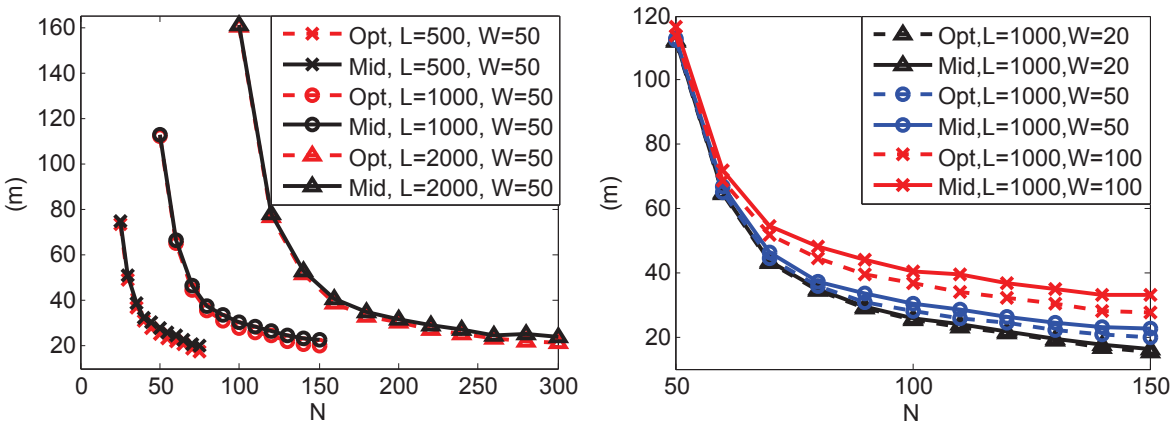
2.2.4 Evaluation Results

We evaluate our algorithm by simulating two types of sensor deployment: random uniform deployment, and line-based deployment. Random uniform deployment is adopted by most of the studies of wireless sensor networks [Kumar et al. (2005); Liu et al. (2008); Yang and Qiao (2009); Saipulla et al. (2010); Chen et al. (2013b); Wang et al. (2014a); Mostafaei (2015); Li and Shen (2015)]. While line-based deployment was investigated in [Saipulla et al. (2009)] to model the sensors dropped by an airplane. In both scenarios, we compare the optimal minimax moving distance found by our algorithm to that of a naive algorithm that fixes the barrier location to $\frac{W}{2}$, which is the average sensor initial y-coordinates in both deployment strategies. We refer to our algorithm as "Opt" and the naive algorithm as "Mid". The time complexity of our algorithm is also evaluated in terms of the number of checked candidate locations.

2.2.4.1 Uniform Deployment

We first evaluate our algorithm with a random uniform deployment of N sensors in an $L \times W$ region.

Minimax Moving Distance Fig. 2.11 shows the minimax moving distances of the two algorithms varying with N in the uniform case. We test different sizes of deployment region. For each size, the effect of redundant sensors on the minimax moving distance is also tested.



(a) Fixed W and varying L.

(b) Fixed L and varying W.

Figure 2.11 Minimax moving distance when N sensors are uniformly deployed in an $L \times W$ region. The coverage radius of each sensor is $R = 10$ m.

We first note that, regardless of the size of the deployment region, the minimax moving distance of both algorithms decreases as the number of sensors increases. This is an intuitive result, as more sensors deployed in the area means a higher chance that one or more sensors will already be close to each final destination. We also note that, regardless of the size of the deployment region, Opt outperforms Mid more when N is larger. This is because the horizontal moving distance tends to dominate the overall 2D moving distance as $L \gg W$, but as the number of sensors increases, the horizontal moving distance decreases. Therefore, the benefit of optimizing vertical moving distance becomes relatively larger. Finally, we note from Fig. 2.11(b) that, given the same L and N, the difference between Opt and Mid is larger when W is larger. This is again due to the increased proportion of vertical distance in the total moving distance.

Fig. 2.11 shows the average minimax moving distances of Opt and Mid. To further illustrate the improvement of Opt over Mid, Fig. 2.12 presents the cumulative distribution function (CDF) of the absolute and relative improvement of Opt over Mid in terms of minimax moving distance, which is a result of 1000 trials for each of three different setups shown in the figure.

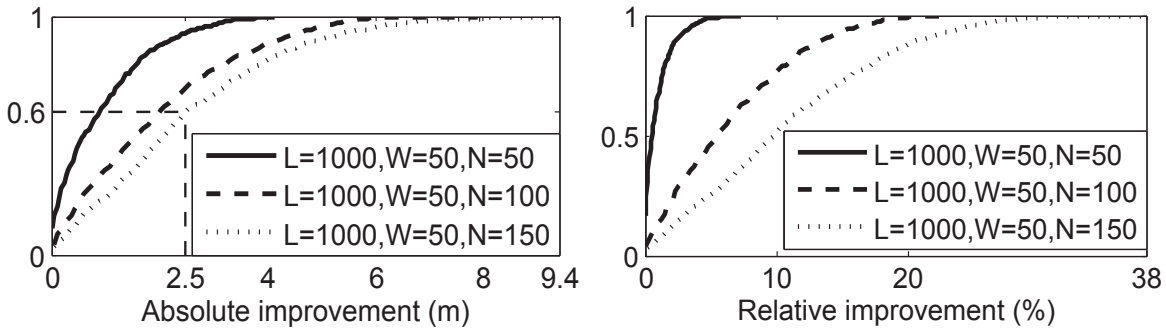


Figure 2.12 CDF of the absolute and relative improvement of Opt over Mid in terms of minimax moving distance.

In the scenario where $N = 150$, Opt has a minimax moving distance which is on average 2.5 m or 11.2% less than that of Mid, but in around 40% of the trials, the improvement is greater. In the most extreme case, Opt reduces the minimax moving distance by 9.4 m or 38%. Similar observations can be made for the other two setups.

Number of Checked Candidate Locations Table 2.1 shows the number of total and checked candidate locations of selected experiments from the uniform scenario. "Total" is the number of candidates in Φ, which is up to $N^2 N_{\min}^2$, while "check" is the number of candidates checked in our iterative algorithm. When N, L, or W increases, the number of total and checked candidates increases accordingly. However, in any scenario, only a small portion of candidates are checked, with a number even less than $NN_{\min}$, which indicates that our algorithm is efficient and scalable. For example, when $L = 2000$ m, $W = 100$ m, $N_{\min} = 100$, given $N = 300$ sensors, $NN_{\min} = 30000$, but only 1783 candidates are checked.

Table 2.1 Total and checked candidates for the uniform deployment case. $R = 10$ m and $N_{\min} = L/2R$.

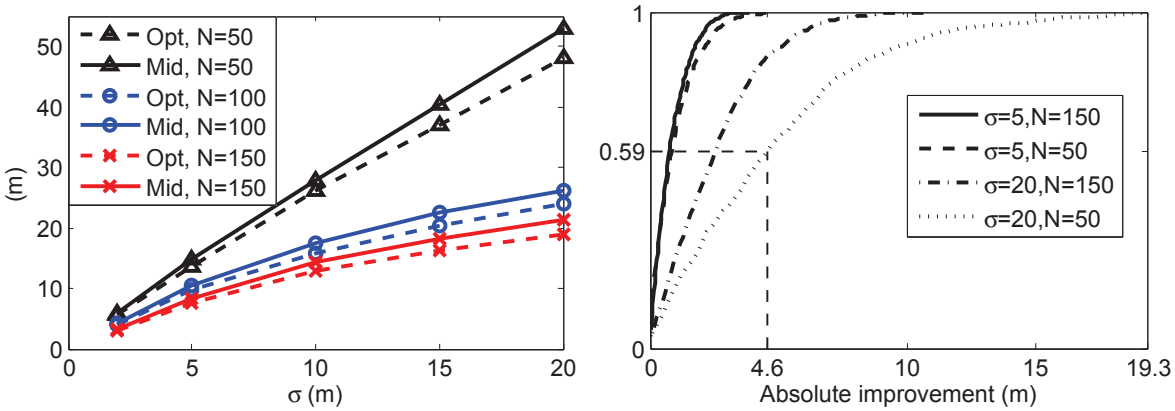
L=1000,W=50,$N_{\min}$=50			L=2000,W=50,$N_{\min}$=100			L=2000,W=100,$N_{\min}$=100		
N	total	check	N	total	check	N	total	check
50	31656	38	100	156114	54	100	512099	208
100	128167	306	200	629182	591	200	2062666	1286
150	288983	469	300	1422913	959	300	4663225	1783

2.2.4.2 Line-based Deployment

The second deployment scenario is the line-based sensor deployment strategy proposed in [Saipulla et al. (2009)], where N sensors are deployed in an $L \times W$ region along the line of $y = \frac{W}{2}$. Each sensor s_i is deployed at its final position $[(2j-1)R, \frac{W}{2}]$ with a random x-axis error δ_i^x and y-axis error δ_i^y, where $\delta_i^x, \delta_i^y \sim N(0, \sigma)$. In practice, this error could be the result of wind or other environmental conditions during an air drop. When redundant sensors are deployed, sensors are assumed to be dropped in groups, e.g., for $N = 2N_{\min}$, two sensors are dropped at each position.

Minimax moving distance Figure 2.13(a) shows the minimax moving distances of Opt and Mid with varying σ. As σ increases, the minimax moving distances of both Opt and Mid increase, because the sensors tend to be initially deployed farther from their final positions. Also, as σ increases, Opt outperforms Mid more. This is because when σ is larger, the sensors will be scattered in a wider region. It is more necessary to optimize the vertical moving distance in this case. Interestingly, when σ is fixed, the difference between Opt and Mid is larger when N is smaller, which is the opposite of the observations in the uniform deployment case. This is due to the following reasons. Firstly, when N increases and multiple sensors are dropped at each point, we tend to have a combination of sensors that yields an optimal barrier location close to $\frac{W}{2}$; therefore, the difference between the vertical moving distance of Opt and Mid is smaller. Secondly, the vertical and horizontal moving distances are comparable in the line-

based deployment case; hence, the reduction of the vertical moving distance can be reflected in the overall 2D moving distance.



(a) Mimimax moving distance vs. σ given three different N values.

(b) CDF of the absolute improvement of Opt over Mid in terms of minimax moving distance.

Figure 2.13 The minimax moving distance for the line-based deployment strategy, with $L = 1000$ m, $W = 50$ m, $R = 10$ m, and $N_{\min} = 50$.

Similar to before, we plot in Figure 2.13(b) the CDF of the improvement of Opt over Mid in four different setups. We can see that when $\sigma = 20$ m and $N = 50$, on average, the minimax moving distance of Opt is 4.6 m or 8.7% less than Mid. In the most extreme case, the minimax moving distance of Opt may be up to 19.3 m or 36% less than Mid. Similar observations can be made for the other setups.

Number of Checked Candidate Locations Table 2.2 shows the number of total and checked candidate locations of selected experiments from the line-based deployment scenario. Similar to before, when N or σ increases, the number of total and checked candidates increases, but only a small proportion of the total candidates are checked in any scenario.

Table 2.2 Total and checked candidates for the line-based deployment case. $L = 1000$ m, $W = 50$ m, $R = 10$ m, and $N_{\min} = 50$.

N	$\sigma = 5$		$\sigma = 20$	
	total	check	total	check
50	6997	83	39561	197
100	30833	104	161154	338
150	72210	159	363082	495

2.3 On Minimizing the Maximum Sensor Movement for Horizontal Barrier Construction

2.3.1 Introduction

In this work, we will study a more practical strong barrier construction problem in a 2D rectangular region, with homogeneous mobile sensors. Our goal is to identify an optimal sensor relocation strategy to build a horizontal barrier so that the maximum sensor moving distance is minimized. There is no requirement on the number of sensors on the final barrier, or any prior knowledge of either the barrier location or the final x-coordinates of the sensors on the barrier. The only assumption in this work is that the formed barrier shall be horizontal – parallel to the sides of the region. In contrast, the solutions in [Saipulla et al. (2010)] and Section 2.2 were based on the assumption that the final horizontal barrier shall be constructed with the minimum number of sensors (hence their final x-coordinates can be calculated *a priori*). While the solutions in [Saipulla et al. (2010); Li and Shen (2015)] were based on the assumption that the final barrier location is fixed and known *a priori*.

To achieve our goal, we proposed an iterative algorithm to apply a binary search over all possible maximum moving distances. During each iteration, a particular maximum moving distance is considered, and we first identified a discrete set of candidate barrier locations that may be possible with this maximum moving distance. Then, a fast polynomial-time algorithm is used to determine the feasibility of forming a barrier at each of these barrier locations while satisfying the maximum moving distance. Base on the feasibility checking results, the search space for the maximum moving distance and the barrier location is reduced and thus may expedite the search process at the next iteration.

Finally, we evaluated the proposed algorithm regarding the maximum moving distance and the time complexity, with various deployment regions and different number of deployed sensors. Evaluation results show that the algorithm in this section can get a smaller maximum moving distance than the algorithms in [Li and Shen (2015)] and Section 2.2. Moreover, our algorithm is efficient and scalable as a benefit of the binary search algorithm.

2.3.2 Model and Problem Statement

2.3.2.1 System Model

Sensor Network We study a network of N mobile sensors deployed in a long rectangular region of size $L \times W$, where $L \gg W$, as shown in Fig. 2.14.

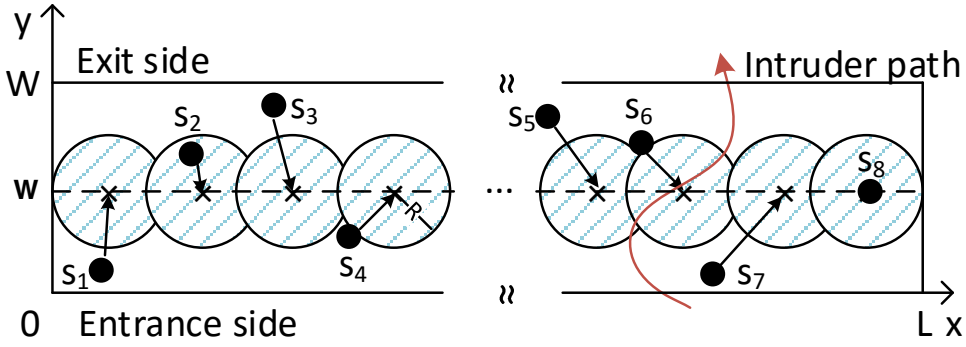


Figure 2.14 System model.

Sensors are named s_1 to s_N from the left to right of the deployment region, and the initial position of sensor s_i is denoted by (x_i, y_i). The set of all the sensors is denoted by S. We adopt the widely-used disk coverage model and denote the sensor coverage radius as R. An intruder can be detected by a sensor if and only if it is within R of the sensor. In addition, we assume sensors can acquire their positions from GPS or another localization scheme.

Intruder and Barrier We assume that intruders may take any path to cross the deployment region, as shown in Fig. 2.14. In order to detect such intruders, strong barrier coverage is required. For simplicity, in the following when we say barrier coverage we mean strong barrier coverage. The focus of our study is to identify a subset of sensors S_b out of S to form a horizontal barrier parallel to the sides of the deployment region. In other words, the final positions of sensors on the barrier, denoted by (x'_i, y'_i), must satisfy: $\forall s_i, s_j \in S_b, y'_i = y'_j = w$. We use w to denote the final barrier location.

System We assume that the sensor network remains connected during sensor movement, and that there is a central processing unit which collects information from sensors, executes the proposed algorithm, and disseminates the movement strategy to sensors.

2.3.2.2 Problem Statement

Our ultimate goal is to maximize the lifetime of the barrier. We assume all sensors have the same amount of energy initially, which is E_{total}. We also assume the sensor which has the least remaining energy after movement runs out of energy first, and consequently breaches the barrier. To maximize the lifetime of the barrier, we need to maximize the minimum remaining energy, i.e., $\min_{s_i \in S_b}(E_{\text{total}} - P_m\sqrt{(x_i - x_i')^2 + (y_i - y_i')^2})$, where P_m is the energy consumed by moving one unit and S_b is the set of sensors forming the barrier. This is equivalent to minimizing the maximum moving distance of sensors in S_b, i.e., $\max_{s_i \in S_b}\sqrt{(x_i - x_i')^2 + (y_i - y_i')^2}$. Sensors not used to form the barrier will remain at their initial positions and may participate in forming future barriers after the operational lifetime of the current barrier has elapsed. Formally, the problem addressed in this work is described as follows:

Given:

- Rectangular deployment region: $L \times W$
- Sensor coverage radius: R
- Total number of deployed sensors: N
- Initial sensor positions: $(x_i, y_i), \forall s_i \in S$

Constraint:

- $\exists\ S_b \subseteq S$, where the final positions of sensors in S_b form a strong barrier.
- $\forall s_i \in S_b, y_i' = w$, where $0 \leq w \leq W$ is the final barrier location.

Output:

- S_b and the final positions of sensors in S_b such that $\max_{s_i \in S_b}\sqrt{(x_i - x_i')^2 + (y_i - y_i')^2}$ is minimized.

2.3.3 Proposed Scheme

2.3.3.1 Overview

The key challenge in solving our barrier coverage problem is to determine the final barrier location. If it is known, then we can apply the algorithm in [Li and Shen (2015)] to find the optimal x-coordinates of final sensor positions. Fig. 2.15 gives an overview of the proposed

scheme. The scheme takes the initial sensor positions, the dimensions of the deployment region, and the sensor coverage radius as the input, and outputs the optimal maximum moving distance, optimal barrier location, and the final sensor positions. The scheme reduces the solution space iteratively till the final solution is identified.

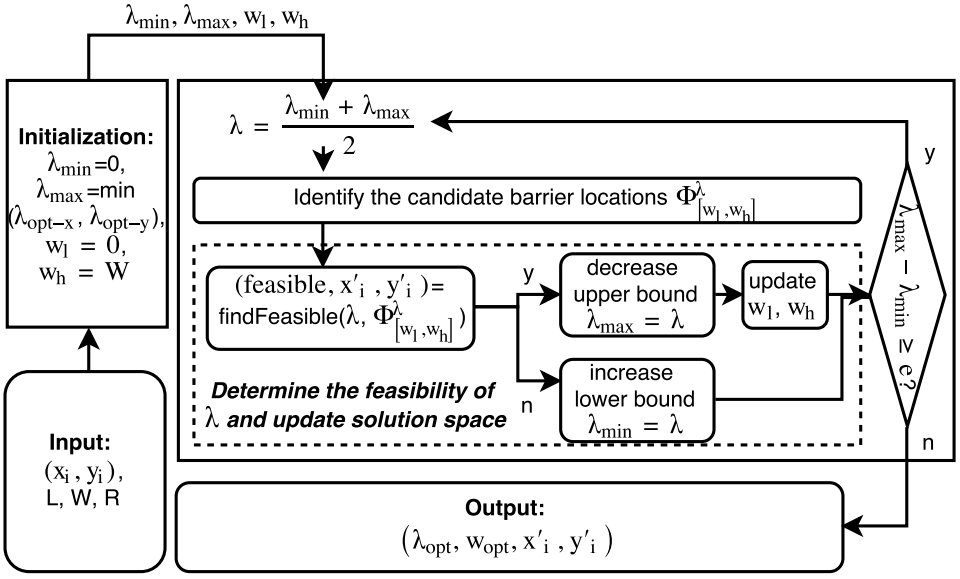


Figure 2.15 An overview of the proposed scheme.

2.3.3.2 Initialization

The initialization module initializes the solution space. It is represented by a four-tuple $\{\lambda_{\min}, \lambda_{\max}, w_l, w_h\}$, where $\lambda_{\min}$ is a lower bound of the optimal maximum moving distance, $\lambda_{\max}$ is an upper bound of the optimal maximum moving distance, w_l is the lowest possible barrier location, and w_h is the highest possible barrier location. Initially, $w_l = 0$, $w_h = W$, $\lambda_{\min} = 0$, and $\lambda_{\max}$ is set to:

$$\lambda_{\max} = \min(\lambda_{opt-x}, \lambda_{opt-y}), \tag{2.20}$$

where λ_{opt-x} and λ_{opt-y} are the minimized maximum moving distance yielded by the barrier coverage schemes in [Li and Shen (2015)] and [Zhang et al. (2015)], respectively. In [Li and Shen (2015)], the barrier coverage problem was solved based on an assumption that the barrier location is known while optimizing the final sensor x-coordinates along the barrier. λ_{opt-x} can

be obtained by setting the barrier location to $\frac{W}{2}$. In [Zhang et al. (2015)], we solved the barrier coverage problem by assuming minimal number of sensors on the barrier (hence, the final sensor x-coordinates are known) while optimizing the barrier location. Both schemes would produce a feasible solution to our problem, thus providing an upper bound of the optimal maximum moving distance.

The lowest possible barrier location w_l is set to 0, and the highest possible barrier location w_h is set to W.

2.3.3.3 Main Scheme

Our scheme is an iterative scheme. It takes the initial $\lambda_{\min}$, $\lambda_{\max}$, w_l, and w_h as the input, and then updates them in each iteration, till the difference between $\lambda_{\max}$ and $\lambda_{\min}$ is less than a small positive value e.

The binary search technique is adopted to reduce the solution space iteratively. In each iteration, the *findFeasible* algorithm checks whether it is possible to form a horizontal barrier with a maximum sensor moving distance no larger than $\lambda = \frac{\lambda_{\min}+\lambda_{\max}}{2}$, and then updates $\lambda_{\min}$, $\lambda_{\max}$, w_l, and w_h according to the result. Before we explain the details of our scheme, we first examine the relation between a barrier location w and the min-max sensor moving distance.

Min-max Moving Distance Function We use function $f^*(w)$ to represent the min-max moving distance of sensors that form a barrier at location w. It has the following two properties.

Lemma 2.1. *Function $f^*(w)$ is a piecewise function composing of segments from functions $f_1^{i,k}(w)$, $f_2^{i,k}(w)$, $f_3^{i,j,k}(w)$, and $f_4^i(w)$, where*

1. $f_1^{i,k}(w) = \sqrt{\{x_i - [L - (2k-1)R]\}^2 + (y_i - w)^2}$,
2. $f_2^{i,k}(w) = \sqrt{[x_i - (2k-1)R]^2 + (y_i - w)^2}$,
3. $f_3^{i,j,k}(w) = \sqrt{aw^2 + bw + c}$ *where* $a = c_1^2/c_0^2 + 1$, $b = -2y_i - c_1 - c_1^2 c_2/c_0^2$, $c = y_i^2 + (c_0^2 + c_1 c_2)^2/4c_0^2$, $c_0 = x_i - x_j + 2kR$, $c_1 = y_j - y_i$, $c_2 = y_j + y_i$,
4. $f_4^i(w) = |y_i - w|$,

where $1 \le i, j \le N$, $i \ne j$, *and* $k = 1, \cdots, \lceil \frac{L}{2R} \rceil$.

Proof: To form a barrier at location w, the min-max moving distance must occur to sensors that belong to one (but not necessarily all) of the following cases. which are illustrated in Fig. 2.16 and will be explained in detail below. It has been proved in [Li and Shen (2015)] that the maximum sensor moving distance can always be reduced further by rearranging the sensors so that one of these four possible cases occurs.

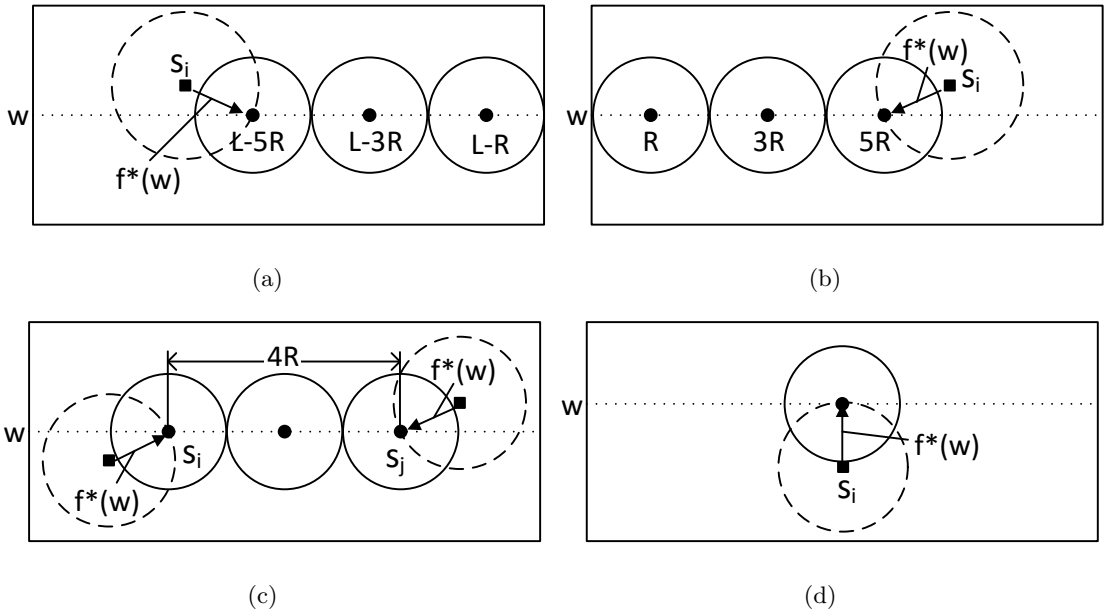


Figure 2.16 Four possible cases when the min-max moving distance may occur, in order to form a barrier at location w.

- Case a). The min-max moving distance may occur to a sensor s_i, which (1) relocates to the final barrier, and (2) all sensors to its right along the final barrier are in attaching positions, meaning that the coverage regions of these sensors are right next to each other, without any overlap or gap between them. In this case, the moving distance of s_i can be represented with one of the following functions at some $k \in [1, \lceil \frac{L}{2R} \rceil]$:

$$f_1^{i,k}(w) = \sqrt{\{x_i - [L - (2k-1)R]\}^2 + (y_i - w)^2}. \tag{2.21}$$

- Case b). The min-max moving distance may occur to a sensor s_i, which relocates to the final barrier and all sensors to its left along the final barrier are in attaching positions. Similar to Case a), the moving distance of s_i can be represented with one of the following

functions at some $k \in [1, \lceil \frac{L}{2R} \rceil]$:

$$f_2^{i,k}(w) = \sqrt{[x_i - (2k-1)R]^2 + (y_i - w)^2}. \tag{2.22}$$

- Case c). The min-max moving distance may occur to two sensors s_i and s_j, where (1) both sensors relocate to the final barrier, (2) the moving distances of both sensors are the same, and (3) all sensors between them along the final barrier are in attaching positions. In this case, if we use $f_3^{i,j,k}(w)$ to denote the equal moving distance of s_i and s_j, the following condition holds for some $k \in [1, \lceil \frac{L}{2R} \rceil]$:

$$x_i + \sqrt{f_3^{i,j,k}(w)^2 - (y_i - w)^2} + 2kR = \ x_j - \sqrt{f_3^{i,j,k}(w)^2 - (y_j - w)^2}. \tag{2.23}$$

  Rewriting (2.23), we have

$$f_3^{i,j,k}(w) = \sqrt{aw^2 + bw + c}, \tag{2.24}$$

  where $a = c_1^2/c_0^2 + 1$, $b = -2y_i - c_1 - c_1^2 c_2/c_0^2$, $c = y_i^2 + (c_0^2 + c_1 c_2)^2/4c_0^2$, and $c_0 = x_i - x_j + 2kR$, $c_1 = y_j - y_i$, $c_2 = y_j + y_i$.

- Case d). The min-max moving distance may occur to a sensor s_i that moves perpendicularly to the final barrier. In this case, the moving distance of sensor s_i is:

$$f_4^i(w) = |y_i - w|. \tag{2.25}$$

Regardless of the barrier location w, the min-max sensor moving distance must occur in one of these four cases. In other words, $f^*(w)$ must be composed of segments from $f_1^{i,k}(w)$, $f_2^{i,k}(w)$, $f_3^{i,j,k}(w)$, and $f_4^i(w)$. ■

Lemma 2.2. *$f^*(w)$ is a continuous function.*

Proof: Let $x'_{i,w}$ and $x'_{i,w+\Delta w}$ denote the x-coordinate of sensor s_i's optimal final position when the barrier is formed at location w and $w + \Delta w$, respectively. Then, we have:

$$\begin{aligned}
\lim_{\Delta w \to 0} f^*(w + \Delta w) &= \lim_{\Delta w \to 0} \max_{1 \le i \le N} \{(x_i - x'_{i,w+\Delta w})^2 + (y_i - w - \Delta w)^2\} \\
&= \max_{1 \le i \le N} \{(x_i - x'_{i,w+\Delta w})^2 + (y_i - w)^2\} \\
&\ge \max_{1 \le i \le N} \{(x_i - x'_{i,w})^2 + (y_i - w)^2\} \\
&= f^*(w).
\end{aligned} \tag{2.26}$$

The "$\geq$" in (2.26) holds because $x'_{i,w}$ is the x-coordinate of s_i's optimal final position when the barrier is formed at location w; therefore, any other values including $x'_{i,w+\Delta w}$ will result a larger or equal maximum moving distance.

Similarly, we have:

$$\begin{aligned} f^*(w) &= \max_{1\leq i\leq N}\{(x_i - x'_{i,w})^2 + (y_i - w)^2\} \\ &= \lim_{\Delta w\to 0}\max_{1\leq i\leq N}\{(x_i - x'_{i,w})^2 + (y_i - w - \Delta w)^2\} \\ &\geq \lim_{\Delta w\to 0}\max_{1\leq i\leq N}\{(x_i - x'_{i,w+\Delta w})^2 + (y_i - w - \Delta w)^2\} \\ &= \lim_{\Delta w\to 0} f^*(w+\Delta w) \end{aligned} \tag{2.27}$$

Therefore, $f^*(w) = \lim_{\Delta w\to 0} f^*(w+\Delta w)$, meaning that $f^*(w)$ is a continuous function.

■

Let $\mathcal{F}$ denote the set of all $f_1^{i,k}(w)$, $f_2^{i,k}(w)$, $f_3^{i,j,k}(w)$, and $f_4^i(w)$ functions. Fig. 2.17 plots all the functions in $\mathcal{F}$ for the example scenario in Fig. 2.18(a), where $\lambda_{\min} = 0$, $\lambda_{\max} = 2.25$, $w_l = 0$, and $w_h = 10$. $f^*(w)$ is highlighted as bold; it is a continuous function and composed of segments from functions f_4^1 and $f_3^{3,4,1}$.

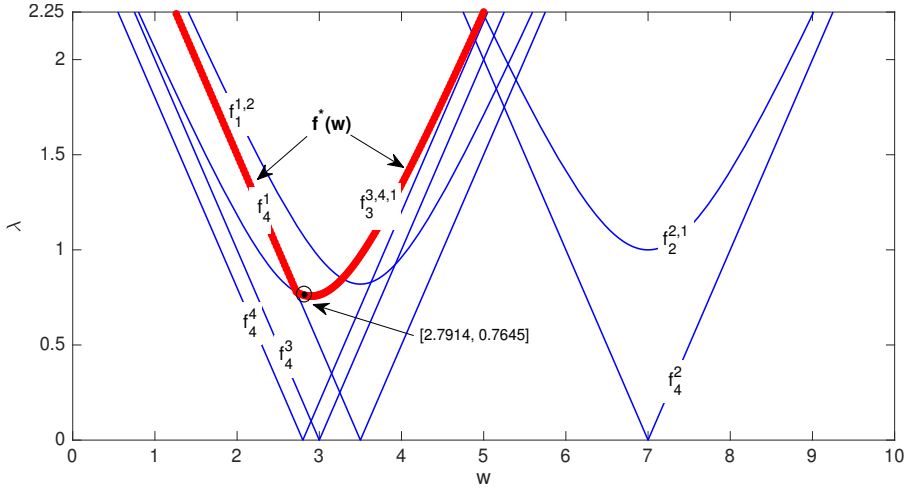
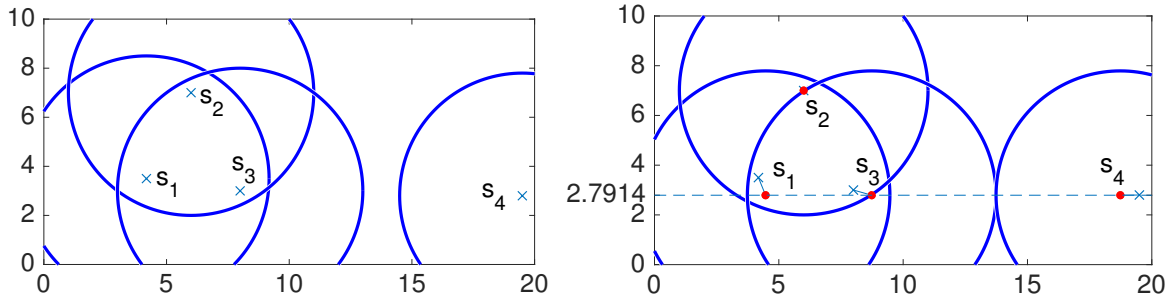


Figure 2.17 Functions in $\mathcal{F}$ corresponding to the example scenario in Fig. 2.18(a). Each curve is labeled with the name of the function. $f^*(w)$ is highlighted as bold. The final solution found by the proposed scheme is marked with $\odot$ and will be discussed in detail in Section 2.3.3.3, Termination Criteria.



(a) Initial deployment (b) Optimal movement strategy

Figure 2.18 An example scenario, where four sensors are deployed in a rectangular region with $L = 20$ and $W = 10$. The sensor coverage radius is 5. The optimal movement strategy is shown in (b) and will be discussed in detail in Section 2.3.3.3, Termination Criteria.

Identification of Candidate Barrier Locations Based on the properties of the min-max moving distance function, we design a polynomial-time algorithm to check the feasibility of λ. λ is deemed feasible if there exists a sensor movement strategy to form a horizontal barrier between w_l and w_h, with the maximum sensor moving distance no larger than λ.

To determine the feasibility of λ, we first discretize the continuous solution space of the possible barrier locations. In other words, we first identify a discrete set of *candidate barrier locations* between w_l and w_h, denoted by $\Phi^{\lambda}_{[w_l,w_h]}$. Candidate barrier locations are the barrier locations where a horizontal line at λ intersects with any of the functions in $\mathcal{F}$. In other words:

$$\begin{aligned}\Phi^{\lambda}_{[w_l,w_h]} = \bigcup_{1\le i,j\le N, i\neq j, 1\le k\le \lceil \frac{L}{2R} \rceil} \Big\{ & \{w | f_1^{i,k}(w) = \lambda, w_l \le w \le w_h\} \\ & \cup \{w | f_2^{i,k}(w) = \lambda, w_l \le w \le w_h\} \\ & \cup \{w | f_3^{i,j,k}(w) = \lambda, w_l \le w \le w_h\} \\ & \cup \{w | f_4^{i}(w) = \lambda, w_l \le w \le w_h\} \cup w_l \cup w_h \Big\}. \end{aligned} \tag{2.28}$$

Then, we check whether λ is feasible at each candidate in $\Phi^{\lambda}_{[w_l,w_h]}$. We will prove later, if λ is feasible at a barrier location between $[w_l, w_h]$, then λ must be feasible at one of the candidate barrier locations in $\Phi^{\lambda}_{[w_l,w_h]}$.

Fig. 2.19 gives an example of how to identify the candidate barrier locations.

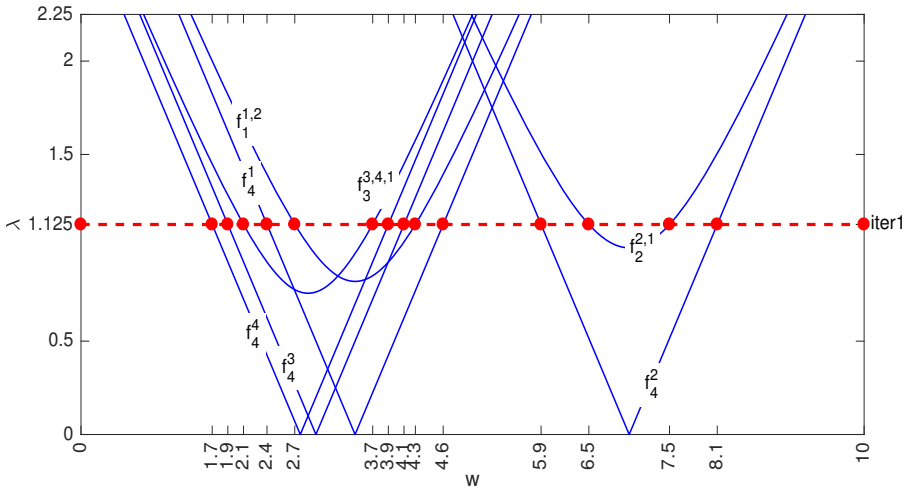


Figure 2.19 Example of the identification of candidate barrier locations. In the first iteration, λ is set to 1.125. 16 barrier location candidates are identified along the line $\lambda = 1.125$ and are marked with solid dots.

Continue with the example shown in Fig. 2.17, in the first iteration, $\lambda_{\min} = 0$, $\lambda_{\max} = 2.25$, and hence λ is set to 1.125. The candidate barrier locations are identified by intersecting the functions in $\mathcal{F}$ with the horizontal line $\lambda = 1.125$. 16 candidate barrier locations are identified, which are marked with solid dots in the figure.

Next, we prove that we can determine the feasibility of λ by checking only the w values in the discrete set $\Phi^{\lambda}_{[w_l,w_h]}$, instead of all the w values in the continuous range $[w_l, w_h]$.

Theorem 2.3. *If there exists a barrier location $w \in [w_l, w_h]$ where λ is feasible, then there must exist a barrier location $w' \in \Phi^{\lambda}_{[w_l,w_h]}$ at which λ also is feasible.*

Proof: If λ is feasible at w_l, then the statement is obviously true because $w_l \in \Phi^{\lambda}_{[w_l,w_h]}$. Therefore, in the following we only consider the situation when λ is not feasible at w_l, i.e., $f^*(w_l) > \lambda$.

Suppose λ is feasible at some $w \in (w_l, w_h]$, meaning that $f^*(w) \leq \lambda$. Because $f^*(w_l) > \lambda$ and we know that $f^*(w)$ is a continuous function (from Lemma 2), there must exist a $w' \in (w_l, w]$ such that $f^*(w') = \lambda$. Considering the fact that $f^*(w)$ is composed of segments of the functions in $\mathcal{F}$ (from Lemma 1), one of the following four equations must hold for some $k \in [1, \lceil \frac{L}{2R} \rceil]$: $f_1^{i,k}(w') = \lambda$, $f_2^{i,k}(w') = \lambda$, $f_3^{i,j,k}(w') = \lambda$, or $f_4^{i,k}(w') = \lambda$. Therefore, $w' \in \Phi^{\lambda}_{[w_l,w_h]}$. ■

Determine the Feasibility of λ and Update the Solution Space After identifying the candidate barrier locations $\Phi^{\lambda}_{[w_l,w_h]}$, we determine the feasibility of λ by only checking whether λ is feasible at any $w \in \Phi^{\lambda}_{[w_l,w_h]}$, and then update the solution space accordingly. Details are shown in Algorithm 2.1.

Algorithm 2.1: *findFeasible()*

```
   Input: λ, Φ^λ_[w_l,w_h]
   Output: feasible, λ_min, λ_max, w_l, w_h
 1 feasible = false;
 2 Sort w ∈ Φ^λ_[w_l,w_h] in ascending order;
 3 length = |Φ^λ_[w_l,w_h]|;
   /* Determine the feasibility of λ, and update w_l                  */
 4 for i from 1 to length do
 5     if isFeasible(λ, w[i]) then
 6         feasible = true;
 7         λ_max = λ;
 8         w_l = w[i];
 9         break;
10     end
11 end
   /* Update w_h                                                      */
12 for i from length to 1 do
13     if isFeasible(λ, w[i]) then
14         w_h = w[i];
15         break;
16     end
17 end
18 if not feasible then λ_min = λ;
```

The feasibility of λ at a particular candidate barrier location w is determined by applying a feasibility checking algorithm proposed in [Li and Shen (2015)], which is called *isFeasible(λ, w)*. It is a simply greedy algorithm that selects sensors to achieve the maximal coverage from left to right along the barrier in each step. In addition to the feasibility result, the *isFeasible* algorithm also outputs the final sensor positions if λ is feasible, which are not included in Algorithm 2.1 for ease of presentation.

Lines 1 - 3 of Algorithm 2.1 initialize the algorithm. From line 4 to line 11, we check the candidate barrier locations, starting from w_l in an ascending order. Once a candidate w has

been found so that λ is feasible at barrier location w, we mark λ as feasible, and update the upper bound $\lambda_{\max}$ to λ and w_l to w. The reason for such an update is as follows. Because λ is not feasible at any candidates in $\Phi^{\lambda}_{[w_l,w)}$, according to Theorem 2.3, λ also is not feasible for any barrier location between w_l and w. Since we have already identified a feasible solution at w where the maximum sensor moving distance is no larger than λ, it is therefore unnecessary to check $[w_l, w)$ in future iterations. Similarly, from line 12 to line 17, we check the candidate barrier locations from another direction, starting from w_h in a descending order. Once a candidate w has been identified so that λ is feasible, we update w_h to w.

On the other hand, If λ is not feasible for any of the candidate barrier locations in $\Phi^{\lambda}_{[w_l,w_h]}$, $\lambda_{\min}$ is increased to λ, and w_l and w_h remain the same. For example, in the first iteration shown in Fig. 2.20, given the 16 candidate barrier locations identified, we first check them from $w_l = 0$. $w = 2.4$ is the first candidate barrier location found feasible with $\lambda = 1.125$. Therefore, we update $\lambda_{\max}$ to 1.125 and w_l to 2.4. We then check the candidate barrier locations from $w_h = 10$. The first candidate barrier location found feasible with $\lambda = 1.125$ is $w = 3.7$. Therefore, we update w_h to 3.7.

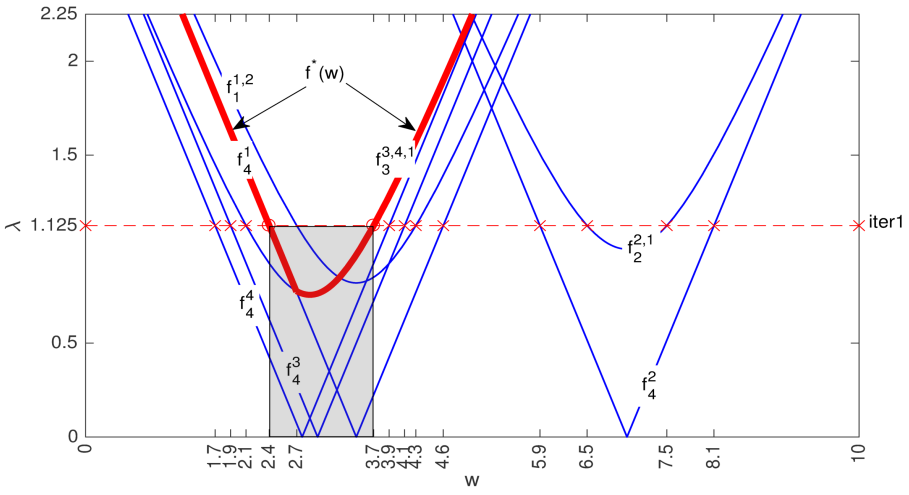


Figure 2.20 Iteration 1: Initially, $\lambda_{\min} = 0$, $\lambda_{\max} = 2.25$, $w_l = 0$, $w_h = 10$. After the feasibility check, they are updated to $\lambda_{\min} = 0$, $\lambda_{\max} = 1.125$, $w_l = 2.4$, $w_h = 3.7$. Candidate barrier locations are marked with $\times$ (infeasible) or $\circ$ (feasible).

In the second iteration shown in Fig. 2.21, initially, $w_l = 2.4$ and $w_h = 3.7$, and λ is set to 0.5625. Five candidate barrier locations are identified along the horizontal line at $\lambda = 0.5625$. As all of them are found infeasible with $\lambda = 0.5625$, we increase $\lambda_{\min}$ to 0.5625, w_l and w_h remain at 2.4 and 3.7, respectively.

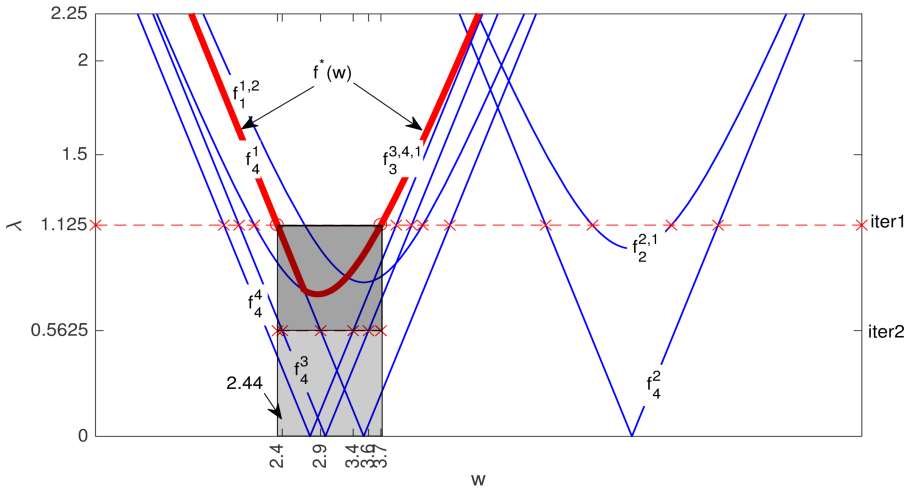


Figure 2.21 Iteration 2: Initially, $\lambda_{\min} = 0$, $\lambda_{\max} = 1.125$, $w_l = 2.4$, $w_h = 3.7$, and five candidate barrier locations are identified. After the feasibility check, the solution space is updated to $\lambda_{\min} = 0.5625$, $\lambda_{\max} = 1.125$, $w_l = 2.4$, $w_h = 3.7$. Candidate barrier locations are marked with $\times$ (infeasible) or $\circ$ (feasible).

Termination Criteria The proposed scheme reduces the solution space iteratively, till the difference between $\lambda_{\max}$ and $\lambda_{\min}$ is less than e. Continue with the example shown in Fig. 2.21, we set e to 0.01. The scheme terminates when $\lambda_{\max} = 0.7645$ and $\lambda_{\min} = 0.7557$. The optimal barrier location w_{opt} is 2.7914 and the maximum moving distance is $\lambda_{opt} = 0.7645$, which are marked in Fig. 2.17. The corresponding optimal sensor movement strategy is shown in Fig. 2.18(b), where sensors s_3 and s_4 move the maximum distance, and they attach to each other on the final barrier. This conforms with the fact that, when $w = 2.7914$, a segment of $f_3^{3,4,1}$ is part of $f^*(w)$, as shown in Fig. 2.17. In this example, the min-max sensor moving distance occurs under Case c) of Lemma 2.1.

2.3.3.4 Complexity Analysis

The total number of iterations is $O(\log \frac{\lambda_{\max}-\lambda_{\min}}{e})$ where $\lambda_{\max}$ and $\lambda_{\min}$ are the initial upper and lower bound of the maximum moving distance, respectively. In each iteration, the candidate identification process has a time complexity of $O(|\mathcal{F}|) = O(N^2\lceil\frac{L}{2R}\rceil)$. In reality, $|\mathcal{F}|$ is much smaller than $N^2\lceil\frac{L}{2R}\rceil$, since we only consider the functions that intersect with the solution space identified by the initial $\lambda_{\min}$, $\lambda_{\max}$, w_l, and w_h values. The *findFeasible* algorithm has a time complexity of $O(|\mathcal{F}|N\log N)$ where $O(N\log N)$ is the time complexity of the *isFeasible* algorithm. Overall, the time complexity is $O(\log \frac{\lambda_{\max}-\lambda_{\min}}{e} N^3\lceil\frac{L}{2R}\rceil \log N)$.

2.3.4 Evaluation Results

We use Matlab simulations to evaluate the proposed scheme in terms of the maximum moving distance and the time complexity. All mobile sensors are deployed in an $L \times W$ rectangular region uniformly at random, which is consistent with most of the studies of wireless sensor networks [Kumar et al. (2005); Liu et al. (2008); Yang and Qiao (2009); Saipulla et al. (2010); Chen et al. (2013b); Wang et al. (2014a); Mostafaei (2015); Li and Shen (2015)]. The default sensing radius is $R = 10$, and the default termination criteria e is 0.01. All values which represent distance have a unit of meter. All results are averaged over 50 experiments.

2.3.4.1 Maximum Moving Distance

Fig. 2.22 and Fig. 2.23 plot the maximum moving distance of the proposed scheme, labeled as λ_{opt}, and compare with λ_{opt-x}, λ_{opt-y}, λ_{curve}^{lower}. λ_{opt-x} is the optimized maximum moving distance obtained by the scheme in [Li and Shen (2015)], which attempts to optimize the x-coordinates of the sensors that form the final barrier at location $\frac{W}{2}$. λ_{opt-y} is the optimized maximum moving distance obtained by the scheme in [Zhang et al. (2015)], which attempts to optimize the final barrier location under the assumption that the x-coordinates of the final sensor positions are known. λ_{curve}^{lower} is a lower bound of the maximum moving distance of sensors when forming a curve strong barrier, in contrast to the horizontal strong barrier in this work. We use the minimized maximum moving distance of sensors in achieving weak barrier coverage

as λ_{curve}^{lower}, which can be obtained with the algorithms proposed in [Czyzowicz et al. (2009)]. Since sensors move only horizontally when achieving weak barrier coverage, λ_{curve}^{lower} is a value unrelated to the width of the deployed region W. We compare λ_{opt} with λ_{curve}^{lower} to evaluate the sub-optimality of the min-max moving distance when enforcing a horizontal barrier.

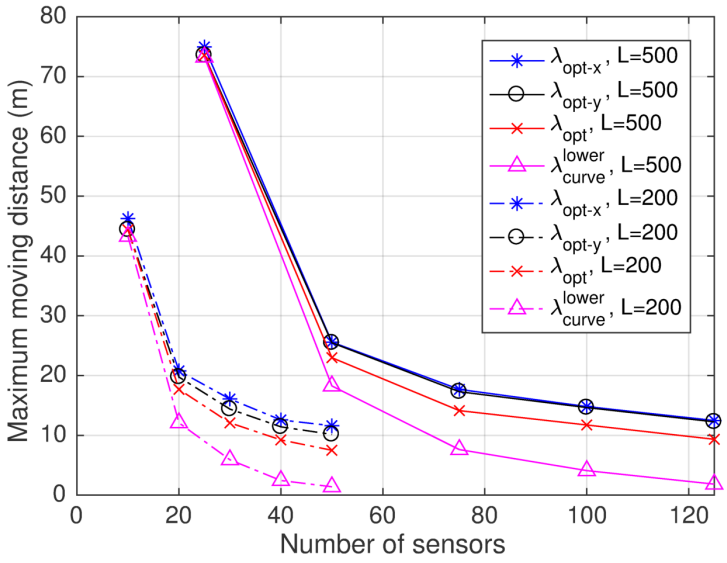


Figure 2.22 Simulation results with $W = 50$ and varying L.

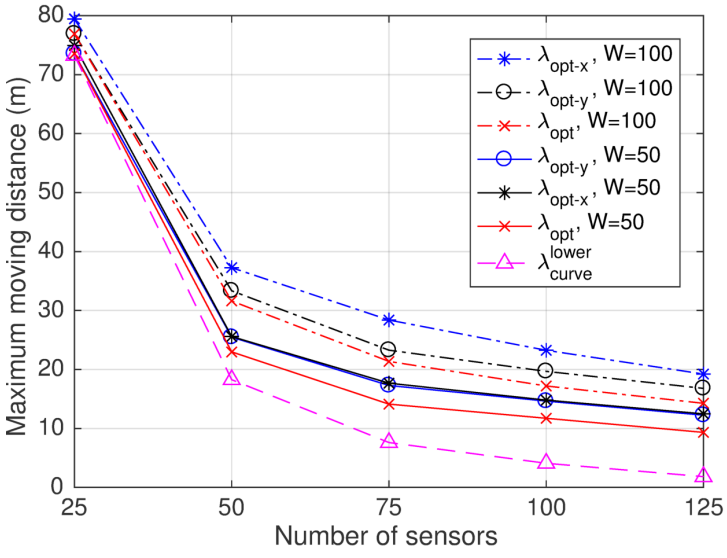


Figure 2.23 Simulation results with $L = 500$ and varying W.

In Fig. 2.22, the width of the deployment region is fixed but the length varies, while in Fig. 2.23, the length is fixed but the width varies. As we can see from the figures, our scheme always yields a smaller maximum moving distance than λ_{opt-x} and λ_{opt-y} under all scenarios, with the total number of deployed sensors varying from $N_{\min}$ to $5N_{\min}$. This verifies the correctness of our scheme.

Another observation is that, when the width to length ratio of the deployment region gets larger, λ_{opt} and λ_{opt-y} outperform λ_{opt-x} more, because the vertical moving distance of sensors contributes more in the overall sensor moving distance and thus we benefit more from optimizing the barrier location. We also observe that, when N gets larger, λ_{opt} outperforms λ_{opt-y} more. This is because λ_{opt} is the result of utilizing all sensors to form a barrier, while λ_{opt-y} is the result of utilizing only the minimum number of sensors.

Additionally, for all scenarios, when only the minimum number of sensors are provided, λ_{opt} is very close to λ_{curve}^{lower}. This is because in this case we can only form a horizontal barrier and λ_{opt} is the optimal maximum moving distance when forming a horizontal barrier. When more than the minimum number of sensors are available, it is possible to construct a curve barrier. In an extreme scenario with large number of sensors provided, a strong barrier may have already formed without any movement. Forcing sensors to form a horizontal barrier would involve additional movements and hence result a larger maximum moving distance.

2.3.4.2 Time Complexity

We evaluate the time complexity of our scheme by investigating the number of functions, iterations, and feasibility-checks. By default, the termination criteria e is set to 0.01 m.

Number of Functions Table 2.3 lists the number of functions in $\mathcal{F}$ under various deployment scenarios. The number of functions affects the number of candidate barrier locations and consequently the maximum number of feasibility-checks in each iteration. In the worst case, the number of functions, i.e., the size of $\mathcal{F}$, is $N^2\lceil\frac{L}{2R}\rceil$. In reality, as listed in the table, the number of functions is far less than $N^2\lceil\frac{L}{2R}\rceil$, since we only consider the functions that intersect with the initial solution space identified by $\lambda_{\min}$, $\lambda_{\max}$, w_l and w_h.

Table 2.3 Number of functions

	L=200, W=50		L=500, W=50		L=500, W=100	
	N=20	N=40	N=50	N=100	N=50	N=100
$N^2\lceil\frac{L}{2R}\rceil$	4000	16000	62500	250000	62500	250000
# of functions	275	481	2508	4475	2818	4741

Number of Iterations Table 2.4 lists the number of iterations under various deployment scenarios. As we can see, the number of iterations in all scenarios is small, as a result of the adopted binary search strategy. Interestingly, the more sensors are deployed, the less iterations are needed. This is because, when more sensors are deployed, there are more opportunities for schemes in [Li and Shen (2015); Zhang et al. (2015)] to produce better λ_{opt-x} and λ_{opt-y}, which, in turn, results in a smaller upper bound of the maximum moving distance ($\lambda_{\max}$). Thus, less number of iterations are required.

Table 2.4 Number of iterations

	L=200, W=50		L=500, W=50		L=500, W=100	
	N=20	N=40	N=50	N=100	N=50	N=100
# of iterations	11.3	10.6	11.7	10.9	12.1	11.3

Number of Feasibility-checks Table 2.5 shows the total number of feasibility-checks in all iterations. We expect that the total number of feasibility-checks is less than the product of the corresponding number of iterations and the number of functions. This is validated in Table 2.5. The main reason is that in an iteration where λ is feasible, the feasibility-checks are performed only on a fraction of the candidate barrier locations.

Table 2.5 Number of feasibility-checks

	L=200, W=50		L=500, W=50		L=500,W=100	
	N=20	N=40	N=50	N=100	N=50	N=100
# of checks	2360	2549	18921	20995	26318	24131

2.3.4.3 Effect of the Termination Criteria e

Table 2.6 shows the number of iterations, feasibility-checks and the code running time under different values of e. As we can see, when e decreases, the maximum moving distance decreases. As a tradeoff, the number of iterations and feasibility-checks, as well as the running time increase. In particular, when e decreases from 0.01 to 0.001, the maximum moving distance decreases but the improvement is almost negligible, while the complexity increases considerably. This justifies the choice of $e = 0.01$ in the implementation of our algorithm.

Table 2.6 Effect of e ($L = 500, W = 50, N = 50$)

e	0.001	0.01	0.1	1
λ_{opt}	22.9847	22.9879	23.0191	23.2777
# of iterations	15.1	11.7	8.3	5.1
# of feasibility-checks	20532	18921	16983	13341
Total running time (s)	7.0933	6.4111	5.5685	3.7649

2.4 Conclusions

In this chapter, we explored two sensor movement problems arisen when constructing strong barrier coverage in a 2D rectangular region. First, we studied a sensor movement problem which only allows the minimum number of sensors on the final barrier. Then we studied an extended sensor movement problem which allows any number of sensors on the final barrier. Different from the previous works, we assume the barrier location is unknown and proposed efficient strategies to find the optimal one. Both problems can be solved in polynomial time and the effectiveness of our solutions is supported by simulation results.

CHAPTER 3. ON MINIMIZING THE NUMBER OF ACTIVE SENSORS FOR STRONG BARRIER COVERAGE UNDER PROBABILISTIC MODEL

3.1 Literature Review

3.1.1 Overview

A probabilistic model describes the sensing capability of a sensor better than the disk model. In a probabilistic model, the sensing capability of a sensor is represented by continuous probabilities instead of the binary values 0 and 1. With a probabilistic model, new challenges will arise regarding the definition of coverage, the interaction between the false alarm probability and detection probability, and sensor collaboration strategies. Fig. 3.1 summarizes the barrier coverage problems under a probabilistic model, with an emphasis on the sensor selection and deployment problems when achieving strong barrier coverage.

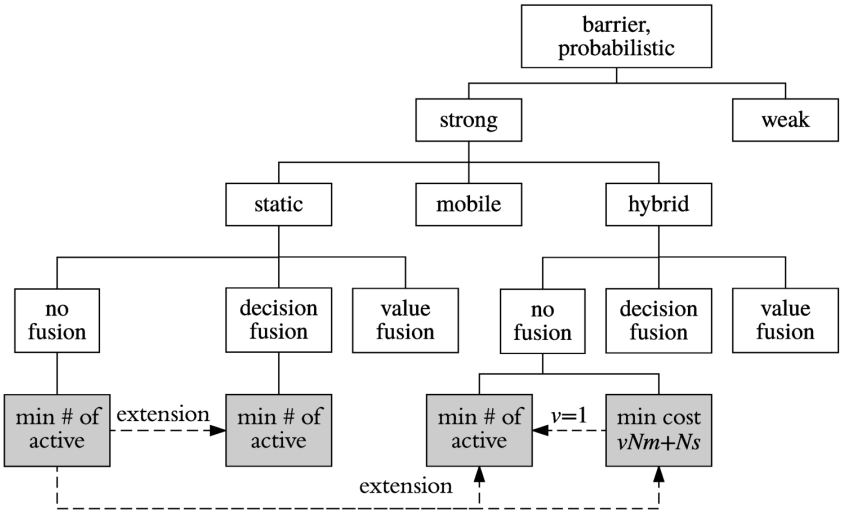


Figure 3.1 Research problems for barrier coverage under the probabilistic model. The min-num static sensor selection problem without data fusion was studied in Section 3.2. The same problem with decision fusion was studied in Section 3.3. The min-cost sensor deployment problem in a hybrid network without data fusion was studied in Chapter 4.

Strong or weak barriers can be formed under a probabilistic model. Strong barrier coverage guarantees a minimum level of detection probability for the intruders who may take any crossing paths. While weak barrier coverage guarantees a minimum level of detection probability for the intruders who take orthogonal crossing paths. Strong and weak barriers can be formed with static or mobile sensors or a combination of them. Data fusion may be applied under a probabilistic model. There are two data fusion methods: decision fusion and value fusion. We studied a fundamental sensor selection problem in this chapter, selecting the minimum number of static sensors to achieve strong barrier coverage under a probabilistic model. We started from the min-num sensor selection problem without data fusion and then studied an extended min-num sensor selection problem with decision fusion. In Chapter 4, we would study another extended problem, constructing strong barrier under the probabilistic model in a hybrid network, with mobile and static sensors.

For all problems, centralized algorithms were proposed to identify a subset of sensors to form a strong barrier under the probabilistic model. We assume there is a central unit in the sensor network to which sensors send their initial positions. The proposed algorithms are run on the central unit. After obtaining the activating information of sensors, the central unit disseminates it to sensors, and then sensors can choose to sleep or be active accordingly. In a multi-level hierarchical network, the workload of initial position collection, algorithm execution and activation information dissemination can be divided and performed on lower-level central units, the higher-level central units are then responsible for merging the barriers formed by the lower-level central units.

3.1.1.1 Probabilistic Model

In a probabilistic model, the coverage capability of a sensor is characterized by probability, rather than the binary values 0 and 1 in the disk model. Typically, a probabilistic model defines two probabilities, the detection probability and the false alarm probability. The detection probability is the probability that a sensor alarms when there is a target. It attenuates as the distance between the sensor and the target increases. The false alarm probability is the

probability that a sensor alarms when there is no target, and it holds the same value for the entire field around the sensor.

Various probabilistic models have been proposed. These models can be classified to two categories. In one category, the detection probability is defined explicitly as a function of the distance from the target to the sensor [Zou and Chakrabarty (2004), Zou and Chakrabarty (2005)]. A classical definition of such a detection probability function is given in [Zou and Chakrabarty (2004)], as shown below,

$$P_d(t) = \begin{cases} 1 & d_{s,t} \leq d_1 \\ e^{-\lambda(d_{s,t}-d_1)\beta} & d_1 < d_{s,t} < d_2 \\ 0 & d_2 \leq d_{s,t} \end{cases} \tag{3.1}$$

where $d_{s,t}$ is the distance from the sensor s to the target t, d_1 and d_2 are parameters which control the start and end points of the probability approximation, λ and β are parameters which control the shape of the probability function. However, this probabilistic model does not define a false alarm probability.

In the other category, the detection probability is calculated based on the distribution of the sensor reading and the decision threhold of sensors. The sensor reading follows a signal attenuation model where the distance from the target to the sensor is a parameter. In [Xing et al. (2009)], Xing gave such a probabilistic model. First, a model for the sensor reading $\mathbf{x}_i$ is defined, as shown below,

$$\mathbf{x}_i = \begin{cases} \frac{\Omega}{1+d_{s,t}^{\alpha}} + n, & \text{if target exists,} \\ n, & \text{otherwise,} \end{cases} \tag{3.2}$$

where Ω is the signal amplitude at the target, $d_{s,t}$ is the distance from the sensor s to the target t, α is the path loss exponent, n is the noise. Each sensor sets a decision threshold T. If the sensor's reading exceeds T, then the sensor alarms the existence of a target. Thus, the detection probability of the sensor for the target is the probability that $\mathbf{x}_i = \frac{\Omega}{1+d_{s,t}^{\alpha}} + n \geq T$, and the false alarm probability is the probability that $\mathbf{x}_i = n \geq T$. Similiar probabilistic models can be found in [Ahmed et al. (2005), Clouqueur et al. (2004), Wang et al. (2007)].

In the probabilistic model, the coverage region of a sensor can be defined as the set of points whose detection probability is larger than a pre-defined threshold $P_d^{\min}$, and the false alarm probability is less than a pre-defined threshold $P_f^{\max}$.

3.1.1.2 Data Fusion

Another advantage of the probabilistic model over the disk model is that data fusion may be applied among sensors. Data fusion can enhance the detection capability of sensors and may expand the coverage regions of sensors. If there is no fusion among sensors, then the coverage region of multiple sensors as a whole is the geometric union of the coverage regions of all sensors. In the following, we name the set of sensors applying data fusion as a *virtual sensor*. There are two data fusion methods: decision fusion and value fusion.

Decision Fusion In the decision fusion model, each sensor makes its own decision; then the decisions are fused at a sink sensor. The decisions can be fused according to the "or" rule or by averaging all decisions and then comparing with a threshold [Clouqueur et al. (2004)]. If the "or" rule is applied, the detection probability of the virtual sensor composed of sensors in $\mathcal{S}$ for a point is,

$$P_d(t) = 1 - \prod_{s_i \in S} \left(1 - P_d(s_i, t)\right). \tag{3.3}$$

With decision fusion, a point which is not covered by any physical sensor in a virtual sensor may be covered by the virtual sensor.

Value Fusion In the value fusion model, each sensor sends its sensing reading to the fusion center where the sensing readings of multiple sensors are fused. The fusion center can use the sum rule or the l^2 rule [Xing et al. (2009), Wang and Zhong (2006)] to fuse the sensing readings. If the sum rule is applied, the detection probability of a virtual sensor composed of sensors in $\mathcal{S}$ for a point is,

$$P_d(t) = P(\sum_{s_i \in S} \mathbf{x}_i \geq T_f), \tag{3.4}$$

where T_f is the decision threshold of the virtual sensor. The value of T_f depends the distribution of the sensing noise, and the number of sensors in the virtual sensor.

3.1.2 Strong Barrier Coverage under Probabilistic Model

3.1.2.1 Static Sensor Selection Problem

No Fusion When no fusion is applied in the probabilistic model, the coverage region of multiple sensors as a whole is the geometric union of the coverage region of each sensor, which is the same as that in the disk model. Under the disk model, selecting the minimal set of sensors to achieve barrier coverage is equivalent to finding the node-disjoint paths with the minimal weight in a graph [Kumar et al. (2005)]. Though the way to calculate the coverage region of multiple sensors is the same as that of the disk model, in the probabilistic model the sensor coverage radius is affected by the number of active sensors, in contrast to the fixed coverage radius in the disk model. This is because in the probabilistic model the sensors have to adjust their decision threshold based on the number of active sensors to keep the system false alarm probability below a threshold. This brings new challenges to the sensor selection problem. We will discuss the minimal static sensor selection problem when no fusion is considered in Section 3.2.

Decision Fusion Chen proposed a scheme in [Chen et al. (2013b)] to achieve strong barrier coverage under the decision fusion model, where the decisions of neighboring sensors and the decisions at multiple sampling points along the intruding path are fused. However, their scheme only considers the detection probability and ignores the system false alarm probability. In contrast, we in Section 3.3 proposed a more practical scheme to achieve strong barrier coverage, which considers both the detection probability and the system false alarm probability with decision fusion applied.

Value Fusion Though there is a lack of research works in achieving barrier coverage under the value fusion model, there are some research works in achieving area coverage under the value fusion model, which may provide insights into the barrier coverage problem. Xing [Xing et al. (2009)] analyzed the impact of value fusion on area coverage from a statistical perspective. They found with value fusion significant fewer sensors are required to achieve full area coverage. Wang [Wang et al. (2007)] 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 full area coverage. The analysis of the coverage regions of sensors under the value fusion model can be applied to barrier coverage.

3.1.2.2 Hybrid Sensor Deployment Problem

No Fusion Hybrid networks have been adopted in many sensor network applications. For barrier coverage, Wang in [Wang et al. (2014b)] considered how to achieve k-barrier coverage with the minimum number of mobile directional sensors which fill the gaps left by static sensors, under the directional non-probabilistic model. In Chapter 4, we will study how to achieve 1-barrier coverage with a combination of static and mobile sensors under a probabilistic model. We consider constraints on both the system detection probability and false alarm probability. Additionally, we consider a more general sensor cost instead of minimizing the number of mobile sensors. In [Kim et al. (2017)], Kim tried to maximize the lifetime of the barriers by scheduling mobile sensors to move around among partial barriers composed of static sensors. Xu [Xu et al. (2014)] investigated the problem of allocating mobile sensors to fortify weak points in a barrier, where intruders may have a higher probability of visiting. The objective was to provide a minimum level of coverage, instead of minimizing the cost.

Decision Fusion & Value Fusion There is no research work in this area so far, though it is an interesting and promising topic. Data fusion will bring new challenges to the hybrid sensor deployment problem. The biggest challenge might be how to form the best sensor fusion pairs or clusters to facilitate the deployment and movement of mobile sensors. Different sensor pairing or clustering schemes will result in different coverage gaps and thus different arrangements for mobile sensors. Dedicated strategies need to be designed to achieve barrier coverage in the most energy/cost-efficient way.

3.1.3 Weak Barrier Coverage under Probabilistic Model

Yang proposed a scheme in [Yang and Qiao (2009)] to achieve weak barrier coverage under the value fusion model with a system false alarm probability constraint. Yang first derived the

projection of two value-fusing sensors and then designed a greedy algorithm to achieve weak barrier coverage.

3.2 Min-num Strong Barrier Coverage under Probabilistic Model without Data Fusion

3.2.1 Introduction

In this work, we will investigate a min-num sensor selection problem arisen when achieving strong barrier coverage with randomly deployed static sensors, under a the probabilistic model. We will jointly consider the detection probability and system false alarm probability and define $(P_D^{\min}, P_F^{\max})$-barrier coverage. Dealing with the detection probability without considering the false alarm probability makes little sense under a probabilistic sensing model, as we can simply lower the decision threshold of sensors to get a higher detection probability, which may result an unacceptable false alarm probability.

To minimize the cost and maximize the energy efficiency, we seek the minimum number of sensors for building a barrier. However, this is not as easy as that under the disk model where the minimal set of sensors can be identified with one run of the shortest path algorithm on a graph. With the constraint on the system false alarm probability, the number of active sensors affects the decision threshold of sensors, which then influences the detection capability of sensors. The detection capability of sensors, in turn, determines the number of active sensors in the system. One run of the shortest path algorithm may not find a set of sensors satisfying both the constraints on the system false alarm probability and detection probability.

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, to identify a minimal set of active sensors from a given deployment to build a barrier. The proposed scheme assumes the number of active sensors and adjusts the decision threshold of sensors iteratively to find a compromise between detection probability and system false alarm probability.

3.2.2 Model and Problem Statement

3.2.2.1 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 W (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. An intruder, or *target*, may take any path traversing the region from the *entrance side* to the *destination side*.

3.2.2.2 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.

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 [Xing et al. (2009); Tan et al. (2011)]:

$$\omega_i(t) = \frac{\Omega}{1 + \big(d\left(s_i, t\right)\big)^{\alpha}}, \tag{3.5}$$

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.

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 (3.5) and the background noise n, as follows:

$$\mathbf{x}_i = \omega_i(t) + n. \tag{3.6}$$

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:

$$P_d(s_i, t) = 1 - F_N\big(T - \omega_i(t)\big). \tag{3.7}$$

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:

$$P_f = 1 - F_N\left(T\right). \tag{3.8}$$

Considering the fact that a single sensor reporting a false alarm constitutes a system false alarm, we define the system false alarm probability P_F as the probability that any sensor produces a false alarm, as follows:

$$P_F = 1 - \left(1 - P_f\right)^{|S_A|}, \tag{3.9}$$

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 [Yang and Qiao (2009)] and [Xing et al. (2009)] and the network false alarm rate in [Tan et al. (2011)].

3.2.2.3 Problem Statement

We then 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 detection probability of 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 work, 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 $L \times W$ region. We seek the minimum number of sensors for cost-effectiveness and energy efficiency.

3.2.3 Proposed Scheme

3.2.3.1 Overview

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

The scheme is divided into four modules, shown in Fig. 3.2. 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 $P_D^{\min}$ 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.

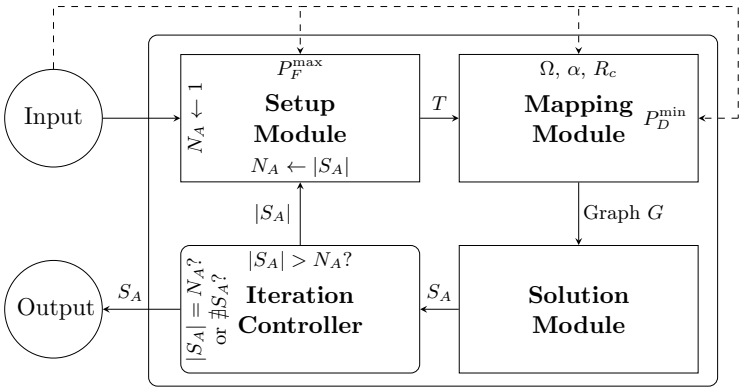


Figure 3.2 Overview of the proposed scheme. The dashed lines indicate parameter entry. Inputs are labeled with the action taken upon receiving the input. Outputs are labeled with any applicable decision criteria.

3.2.3.2 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 (3.9), to satisfy $P_F \leq P_F^{\max}$, we need

$$P_f \leq 1 - (1 - P_F^{\max})^{1/N_A}\,. \tag{3.10}$$

Using (3.8), we then have

$$T = F_N^{-1}(1 - P_f) \geq F_N^{-1}\big((1 - P_F^{\max})^{1/N_A}\big). \tag{3.11}$$

According to (3.7), to maximize the probability of detection, we minimize T; therefore, we use

$$T = F_N^{-1}\big((1 - P_F^{\max})^{1/N_A}\big) \tag{3.12}$$

as the output of the setup module.

3.2.3.3 Mapping Module

The mapping module maps the sensor network to an undirected graph G, which consolidates the detection probability and the network connectivity information. As shown in Fig. 3.2, the mapping module takes T as input, as well as several system-related parameters introduced in Section 3.2.2. These inputs determine the edges of G. Fig. 3.3 provides an example of the mapping procedure, which is composed of the steps described below.

The mapping procedure takes the following steps:

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.

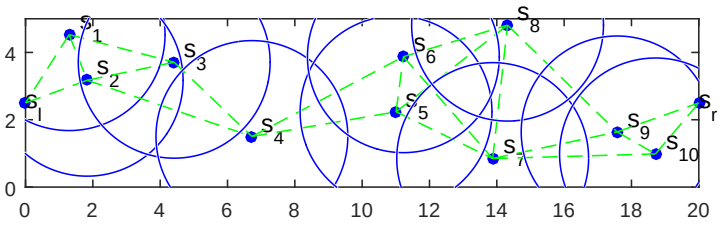


Figure 3.3 Example of the mapping procedure. Ten sensors are deployed in a 20 × 5 m region. The inputs are $T = 1.64$ mW, $\Omega = 30$ mW, $\alpha = 2$, and $R_c = 6$ m. Sensors s_l and s_r are virtual sensors which represent the left and right boundary of the monitored region, respectively. The circles are the coverage regions of sensors. The dash lines between sensors are the edges of G.

Edge Identification To identify the edges in G, a sensor's coverage region under the probabilistic model must be determined in advance. In probabilistic model, a target at point t is covered if it can be detected by a sensor with a probability no less than $P_D^{\min}$, and the system false alarm probability is no more than $P_F^{\max}$. Given the decision threshold T which satisfies $P_F \leq P_F^{\max}$, the coverage region of a sensor is the region around a sensor where the detection probability is no less than $P_D^{\min}$. That is,

$$P_d(s_i, t) = 1 - F_N\big(T - \omega_i(t)\big) \geq P_D^{\min} \Longrightarrow \omega_i(t) \geq T - F_N^{-1}(1 - P_D^{\min}). \tag{3.13}$$

According to (3.5), we have the coverage region of sensor s_i,

$$\mathcal{A} = \left\{ t | d(s_i, t) \leq \left(\frac{\Omega}{T - F_N^{-1}(1 - P_D^{\min})} - 1 \right)^{\frac{1}{\alpha}} \equiv R \right\}, \tag{3.14}$$

which is a disk centered at s_i with radius R as shown in Fig. 3.3.

For any two physical sensors, if their coverage regions overlap and they are in the communication range of each other, then an edge exists between them. For the edge between a physical sensor and s_l or s_r, we require the coverage region of the physical sensor intersect the left or right boundary of the monitored region.

3.2.3.4 Solution Module

Given the graph G, the solution module finds a minimum set of sensors whose detection probability for any intruding path is no less than $P_D^{\min}$. Such a set of sensors can be found by

applying the shortest path algorithm on G, with source node as s_l, and destination node as s_r. The sensors on the path from s_l to s_r form the barrier. For any intruder traversing the barrier, it will be detected with a probability higher than or equal to $P_D^{\min}$.

3.2.3.5 Iteration Controller

To understand the iteration controller, we first give an overview of iterations in our scheme. 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 3.1. Let S_A be the output of the solution module, given N_A as the input of the setup module. In each iteration of the proposed scheme, 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 the proposed scheme, $N_A^{(1)} = 1$, 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 the proposed scheme, if we do not terminate in iteration $k-1$, then we set $N_A^{(k)} = |S_A|^{(k-1)}$. Therefore,

$$N_A^{(k)} = |S_A|^{(k-1)} \geq N_A^{(k-1)}. \tag{3.15}$$

From (3.12), we can then conclude that $T^{(k)} \geq T^{(k-1)}$. Next, from (3.14), we know that a higher T for a sensor s_i leads to a smaller coverage radius for s_i, that is,

$$T^{(k)} \geq T^{(k-1)} \Longrightarrow R^{(k)} \leq R^{(k-1)}. \tag{3.16}$$

Consequently, the set of sensors $S_A^{(k)}$ which can form a barrier in G^k can also form a barrier in $G^{(k-1)}$. However, there may exist a better solution in $G^{(k-1)}$. 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.

$|S_A| = N_A$ 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 the detection probability (from the solution module) and system false alarm probability (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 our scheme, 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, as demonstrated by Theorem 3.1.

S_A does not exist Terminate. The solution module could not find an S_A that meet the requirement on detection probability. Further iterations would also not produce a solution, because S'_A would not exist for any $N'_A > N_A$, as all the coverage radius would be smaller for a larger N_A, using reasoning similar to that of the proof of Theorem 3.1. This means that the proposed scheme cannot find a solution for strong $(P_D^{\min}, P_F^{\max})$-barrier coverage with the given sensor deployment.

$|S_A| > N_A$ 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, our scheme 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 3.2. For any $N'_A \in [N_A, |S_A|)$, the solution module cannot find a path from s_l to s_r in G' which satisfies $P'_F \leq P_F^{\max}$, where G' is the output of the mapping module when N'_A sensors are assumed to be active, and P'_F is 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 3.1, we know that

$$N'_A \geq N_A \Rightarrow R' \leq R. \tag{3.17}$$

Therefore, any solution S'_A that is feasible on G' is also feasible on G, but 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

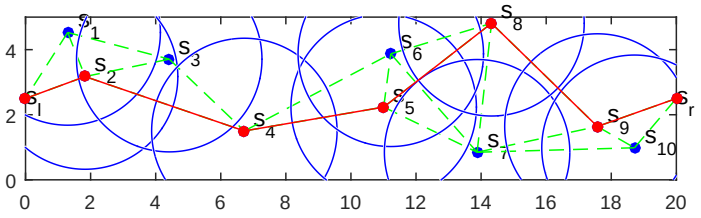
$$|S_A| > N'_A \geq |S'_A| \geq |S_A|, \tag{3.18}$$

which is a contradiction. ■

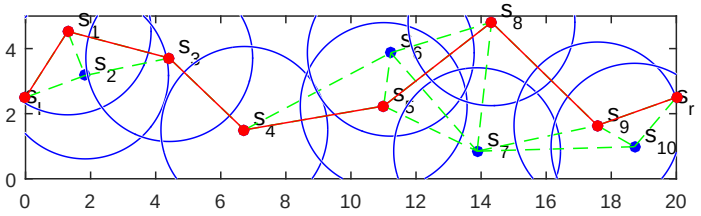
3.2.4 Evaluation Results

3.2.4.1 Trace Study

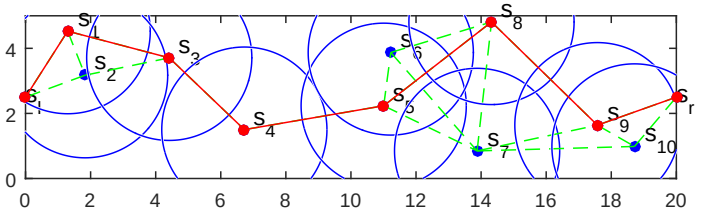
An example of the iterations in the proposed scheme is shown in Fig. 3.4.



(a) Iteration 1: $N_A = 1$, $S_A = \{s_2, s_4, s_5, s_8, s_9\}$, $|S_A| = 5$.



(b) Iteration 2: $N_A = 5$, $S_A = \{s_1, s_3, s_4, s_5, s_8, s_9\}$, $|S_A| = 6$.



(c) Iteration 3: $N_A = 6$, $S_A = \{s_1, s_3, s_4, s_5, s_8, s_9\}$, $|S_A| = 6$.

Figure 3.4 An illustration of iterations in the proposed scheme. $P_D^{\min} = 0.95$ and $P_F^{\max} = 0.05$. The red edges compose the shortest path.

In the first iteration, $N_A = 1$ and the shortest path from s_l to s_r contains five physical sensors $\{s_2, s_4, s_5, s_8, s_9\}$. In the second iteration, $N_A = 5$. We can see due to the shrinking of sensor coverage region, the edges in G become less. Now the shortest path from s_l to s_r contains six physical sensors $\{s_2, s_4, s_5, s_8, s_9\}$. In the third iteration, $N_A = 6$. The edges in G keep reducing. But the shortest path in the last iteration is still a shortest path in current iteration, and hence $|S_A| = N_A$, the scheme terminates.

3.2.4.2 Number of Active Sensors

In this section, we show the variation of the number of active sensors when varying Ω, $P_D^{\min}$ and $P_F^{\max}$, to verify the correctness of the proposed scheme. 200 sensors are randomly deployed in 200 $m \times 10$ m region, the communication range of sensors is 20 m, the decay factor α is 2. We assume the noise follows Gaussian distribution with a mean of 0 and standard variance of 1. By default, $P_D^{\min} = 0.9$ and $P_F^{max} = 0.05$.

The Effect of Signal Amplitude Ω As we can see in Fig. 3.5, when the amplitude of the target emitted signal increases, the number of active sensors decreases. This is because a larger Ω will increase the coverage radius of sensors, less sensors are needed to reach the coverage level.

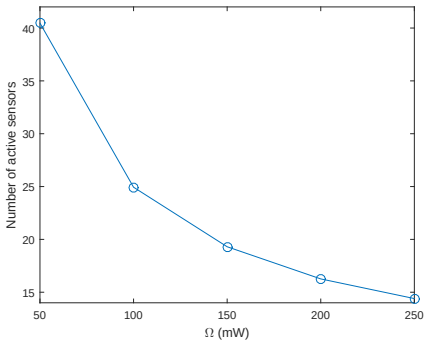
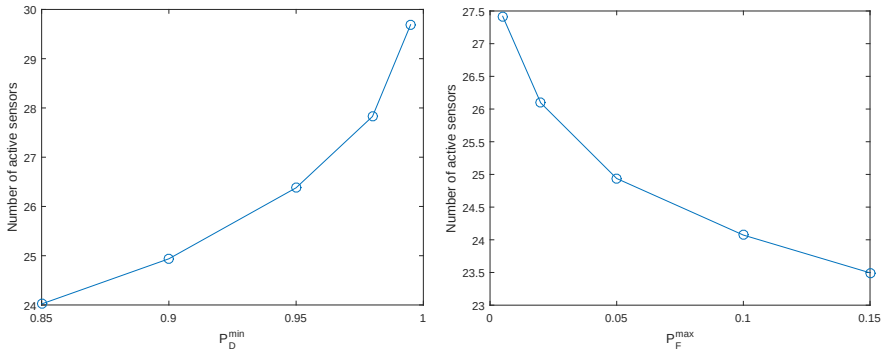


Figure 3.5 Number of active sensors vs. amplitude of the target emitted signal Ω.

The Effect of P_D^{min} and P_F^{max} As predicted, in Fig. 3.6(a), when P_D^{min} increases, the number of active sensors increases, since the coverage radius of sensors decreases. In Fig.

3.6(b), when P_F^{max} increases, the number of active sensors decreases. This is because a higher P_F^{max} allows a sensor to choose a lower decision threshold, which will result an increase of sensor coverage radius.



(a) Number of active sensors v.s. P_D^{min} (b) Number of active sensors v.s. P_F^{max}

Figure 3.6 The effect of P_D^{min} and P_F^{max} on the number of active sensors.

3.3 Min-num Strong Barrier Coverage under Probabilistic Model with Data Fusion

3.3.1 Introduction

In this section, we will investigate a similar min-num sensor selection problem as in Section 3.2. The minimum number of randomly deployed static sensors are selected to achieve $(P_D^{\min}, P_F^{\max})$-barrier coverage under a probabilistic model. The difference between this work and the work in Section 3.2 is that data fusion is employed in this work. Data fusion can enhance the detection capability of sensors and hence expanding the coverage regions of sensors. We adopted decision fusion over value fusion in this work because it is relatively light-weight compared to value fusion. Decision fusion was applied to neighboring sensors and the sequence of sampling points along an intruder's path. A system false alarm probability was considered as in Section 3.2, which distinguishes our work from other works such as [Chen et al. (2013b)] with decision fusion applied, where only the probability of detecting a target was considered.

Under a probabilistic model, when the system false alarm probability and the detection probability are jointly considered, the number of active sensors and the detection capability of

sensors affect each other. Therefore, we still need the iterative framework proposed in Section 3.2 to test the assumptions on the number of active sensors and adjust the decision threshold. However, the detection capability evaluation method in Section 3.2 will underestimate the detection capability of sensors with decision fusion. In this section, to address the min-num sensor selection problem under the decision fusion model, we applied the detection capability evaluation method proposed in [Chen et al. (2013b)] within the iterative framework proposed in Section 3.2. Moreover, we improved the method of finding the minimal set of sensors proposed in [Chen et al. (2013b)]. We name the proposed scheme as BaCo.

3.3.2 Model and Problem Statement

3.3.2.1 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.

3.3.2.2 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.

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 [Xing et al. (2009); Tan et al. (2011)]:

$$\omega_i(t) = \frac{\Omega}{1 + \big(d\,(s_i, t)\,\big)^{\alpha}}, \tag{3.19}$$

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.

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 (3.19) and the background noise n, as follows:

$$\mathbf{x}_i = \omega_i(t) + n. \tag{3.20}$$

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:

$$P_d(s_i, t) = 1 - F_N\big(T - \omega_i(t)\big). \tag{3.21}$$

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:

$$P_{D,t} = 1 - \prod_{s_i \in S_A} \big(1 - P_d\,(s_i, t)\,\big). \tag{3.22}$$

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:

$$P_{D,\varphi} = 1 - \prod_{q_j \in Q} (1 - P_{D,q_j}), \tag{3.23}$$

where P_{D,q_j} is calculated according to (3.22).

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:

$$P^l_{D,\varphi} = \min_{Q} P_{D,\varphi}, \tag{3.24}$$

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:

$$P_D = \min_{\varphi} P^l_{D,\varphi} = \min_{\varphi} \min_{Q} \left(1 - \prod_{q_j \in Q} \prod_{s_i \in S_A} \left(1 - P_d\left(s_i, q_j\right)\right)\right). \tag{3.25}$$

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:

$$P_f = 1 - F_N\left(T\right). \tag{3.26}$$

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:

$$P_F = 1 - \left(1 - P_f\right)^{|S_A|}, \tag{3.27}$$

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 [Yang and Qiao (2009)] and [Xing et al. (2009)] and the network false alarm rate in [Tan et al. (2011)].

3.3.2.3 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 work, 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 $S_A \subseteq \mathcal{S}$.

3.3.2.4 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 [Chen et al. (2013b)]. The detection gain $\mathcal{G}(p)$ associated with a probability p is defined as follows:

$$\mathcal{G}(p) = -\log(1-p). \tag{3.28}$$

$\mathcal{G}(p)$ 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 (3.22) into (3.23) and rearranging to obtain:

$$1 - P_{D,\varphi} = \prod_{q_j \in Q} \prod_{s_i \in S_A} \big(1 - P_d\left(s_i, q_j\right)\big). \tag{3.29}$$

By applying the log function to both sides of (3.29), we obtain an expression for $\mathcal{G}_\varphi$, the total detection gain for a target that takes intruding path φ, as follows:

$$\mathcal{G}_\varphi = \sum_{q_j \in Q} \sum_{s_i \in S_A} \mathcal{G}\left(s_i, q_j\right), \tag{3.30}$$

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:

$$\mathcal{G}_D = \min_{\varphi} \min_{Q} \big(\sum_{q_j \in Q} \sum_{s_i \in S_A} \mathcal{G}\left(s_i, q_j\right) \big). \tag{3.31}$$

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.

3.3.3 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. 3.7. 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.

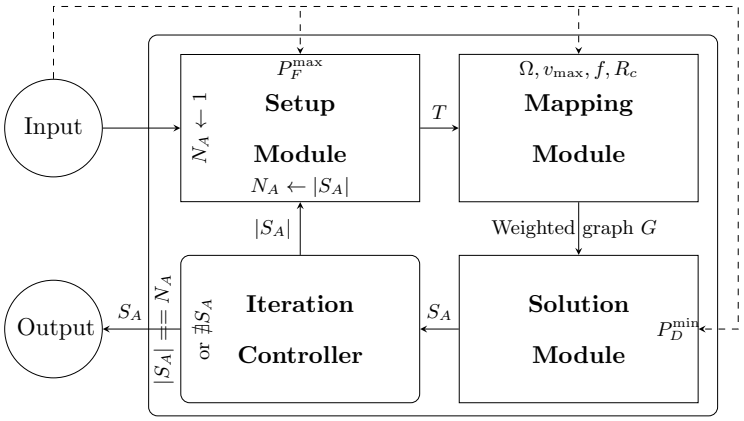


Figure 3.7 Overview of BaCo. The dashed lines indicate parameter entry. Inputs are labeled with the action taken upon receiving the input. Outputs are labeled with any applicable decision criteria.

3.3.3.1 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 (3.27), to satisfy $P_F \leq P_F^{\max}$, we need

$$P_f \leq 1 - (1 - P_F^{\max})^{1/N_A} . \tag{3.32}$$

Using (3.26), we then have

$$T = F_N^{-1}(1 - P_f) \geq F_N^{-1}\big((1 - P_F^{\max})^{1/N_A} \big). \tag{3.33}$$

According to (3.21), to maximize the probability of detection, we minimize T; therefore, we use

$$T = F_N^{-1}\big((1 - P_F^{\max})^{1/N_A} \big) \tag{3.34}$$

as the output of the setup module.

3.3.3.2 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. 3.7, the mapping module takes T as input, as well as several of the system-related parameters introduced in Section 3.3.2. These inputs determine the edges and edge weights of G. Fig. 3.8 provides an example of the mapping procedure, which is composed of the steps described below.

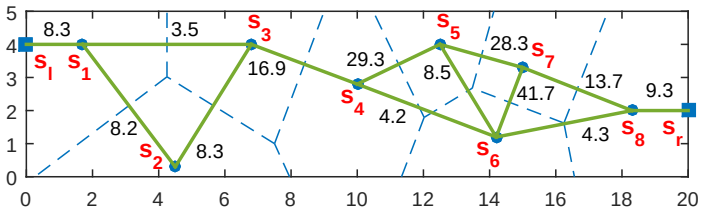


Figure 3.8 Example of the mapping procedure. Eight sensors are deployed in a 20 × 5 m region. The inputs are $T = 1.64$ mW, $\Omega = 10$ mW, $v_{\max} = 1$ m/s, $f = 2$ Hz, and $R_c = 6$ m. Sensors s_l and s_r are virtual sensors which represent the left and right boundary of the monitored region, respectively. The dashed lines are the Voronoi diagram of the sensors. The solid lines between sensors are the edges of G, labeled with their weights.

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.

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 [Chen et al. (2013b)], 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.

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 [Chen et al. (2013b)], 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. 3.9, where the crosses show the worst-case sampling points for a target traveling between s_3 and s_4 in the example in Fig. 3.8. 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.

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 $\mathrm{Vor}(s_i, s_j)$. We use this value as

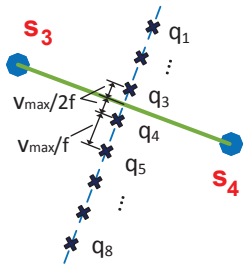


Figure 3.9 The worst-case sampling points for a target traveling between s_3 and s_4. The crosses represent the sampling points and the dashed line is the Voronoi edge between the two sensors.

w_{ij}, the weight of the edge $s_i s_j$ in G. From (3.30), we have

$$w_{ij} = \sum_{k=1}^{m} \big(\mathcal{G}(s_i, q_k) + \mathcal{G}(s_j, q_k)\big), \tag{3.35}$$

where q_k is a sampling point, m is the number of sampling points along $\mathrm{Vor}(s_i, s_j)$, 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 $\mathrm{Vor}(s_i, s_j)$. Therefore, w_{ij} is a lower bound of the actual detection gain from all $s_i \in \mathcal{S}$.

3.3.3.3 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 [Chen et al. (2013b)]. Therefore, in BaCo, we use a two-phase heuristic solution.

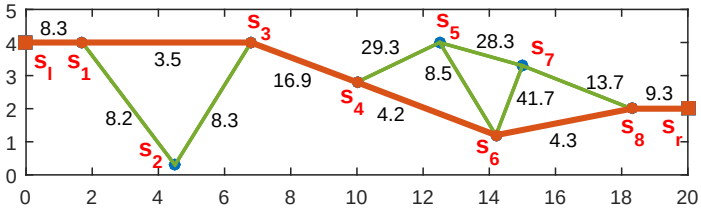
Phase 1 Prune all edges in G whose weights are no more than $\mathcal{G}_D^{\min}$ and call the resulting graph $\tilde{G}$. Run Dijkstra's algorithm on $\tilde{G}$ to find the shortest path, in terms of number of hops,

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.

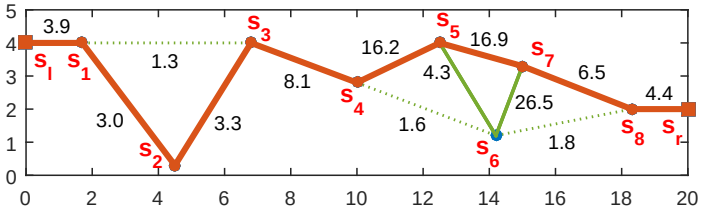
Phase 2 If Dijkstra's algorithm cannot find a path, then s_l is disconnected from s_r in $\tilde{G}$, meaning that no single path can deliver $\mathcal{G}_D^{\min}$ flow from s_l to s_r. In this case, we look for a flow network that can deliver $\mathcal{G}_D^{\min}$ flow by running the maximum-flow based algorithm proposed in [Chen et al. (2013b)] 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.

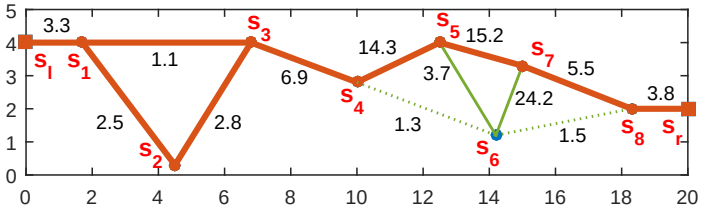
Fig. 3.10 illustrates the heuristic two-phase algorithm with three example graphs (see the next section for an explanation of these graphs as iterations). In these examples, $\mathcal{G}_D^{\min} = 3$, which corresponds to $P_D^{\min} = 0.95$. In Fig. 3.10(a), $\tilde{G}$ is identical to G, as all the 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. 3.10(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. 3.10(c), edges s_1s_3, s_1s_2, s_2s_3, s_4s_6 and s_6s_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: $N_A = 7$, $S_A = \{s_1, s_2, s_3, s_4, s_5, s_7, s_8\}$, $|S_A| = 7$.

Figure 3.10 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 $\tilde{G}$.

3.3.3.4 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 3.3. 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,

$$N_A^{(k)} = |S_A|^{(k-1)} \geq N_A^{(k-1)}. \tag{3.36}$$

From (3.34), we can then conclude that $T^{(k)} \geq T^{(k-1)}$. Next, from (3.21) and (3.28), 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:

$$\begin{aligned} T^{(k)} \geq T^{(k-1)} &\Rightarrow P_d^{(k)}(s_i,t) \leq P_d^{(k-1)}(s_i,t) \\ &\Rightarrow \mathcal{G}^{(k)}(s_i,t) \leq \mathcal{G}^{(k-1)}(s_i,t),\ \forall i, \forall t. \end{aligned} \tag{3.37}$$

From (3.35), 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 $G^{(k-1)}$:

$$\left.\begin{aligned} \mathcal{G}^{(k)}(s_i,t) \leq \mathcal{G}^{(k-1)}(s_i,t) \\ \mathcal{G}^{(k)}(s_j,t) \leq \mathcal{G}^{(k-1)}(s_j,t) \end{aligned}\right\} \Rightarrow w_{ij}^{(k)} \leq w_{ij}^{(k-1)}. \tag{3.38}$$

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.

$|S_A| = N_A$ 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 3.3.

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 3.3. This means that BaCo cannot find a solution for strong $(P_D^{\min}, P_F^{\max})$-barrier coverage with the given sensor deployment.

$|S_A| > N_A$ 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 3.4. 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 3.3, we know that

$$N'_A \geq N_A \Rightarrow w'_{ij} \leq w_{ij}. \tag{3.39}$$

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

$$|S_A| > N'_A \geq |S'_A| \geq |S_A|, \tag{3.40}$$

which is a contradiction. ■

An example of BaCo's iterations is shown in Fig. 3.10. Fig. 3.11 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 3.3), BaCo thus attempts to minimize $|S_A|$. The circled crosses and arrows in Fig. 3.11 show the iterations in Fig. 3.10, 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.

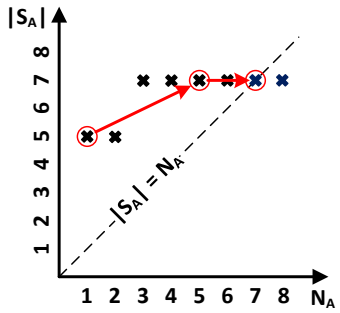


Figure 3.11 Illustration of iteration progress for the example shown in Fig. 3.10. The crosses show the output $|S_A|$ value corresponding to each input N_A. The arrows show the progression of the three iterations from Fig. 3.10. Only the circled crosses are checked by BaCo.

3.3.4 Evaluation Results

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 uniformly randomly deployed in a 100 $\times$ 10 m belt region, similar to the simulation setups in other barrier coverage papers [Kumar et al. (2005); Liu et al. (2008); Yang and Qiao (2009); Saipulla et al. (2010); Chen et al. (2013b); Wang et al.

(2014a); Mostafaei (2015); Li and Shen (2015)]. The default simulation parameters are shown in Table 3.1.

Table 3.1 Default simulation parameters

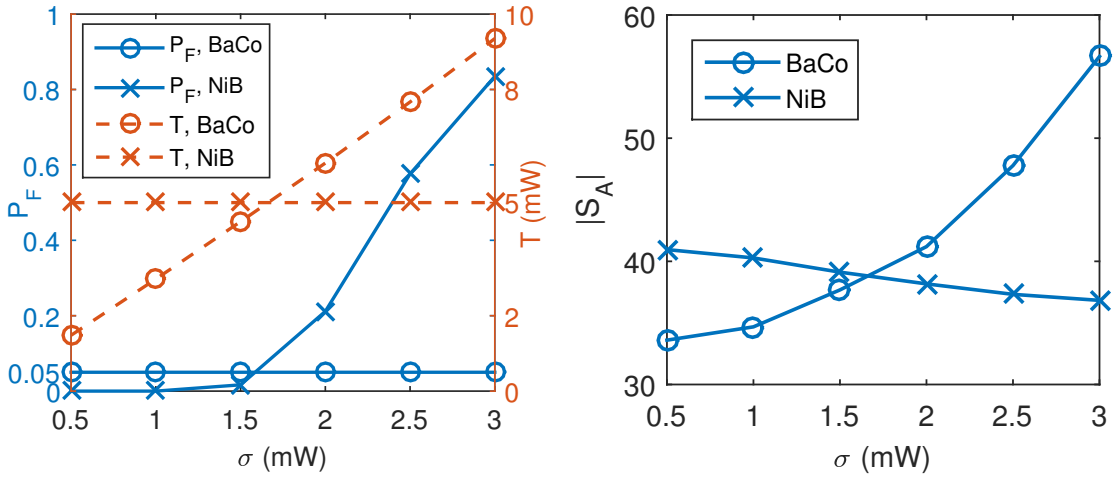
Parameter	Meaning	Default value
$P_D^{\min}$	Detection probability constraint	0.95
$P_F^{\max}$	System false alarm probability constraint	0.05
R_c	Communication range	20 m
$v_{\max}$	Target's maximum moving speed	1 m/s
Ω	Source signal strength	30 mW
α	Source signal decay exponent	2
f	Sensor sampling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

3.3.4.1 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 [Chen et al. (2013b)]. 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 (3.26) and (3.27).

Fig. 3.12(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. 3.12(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

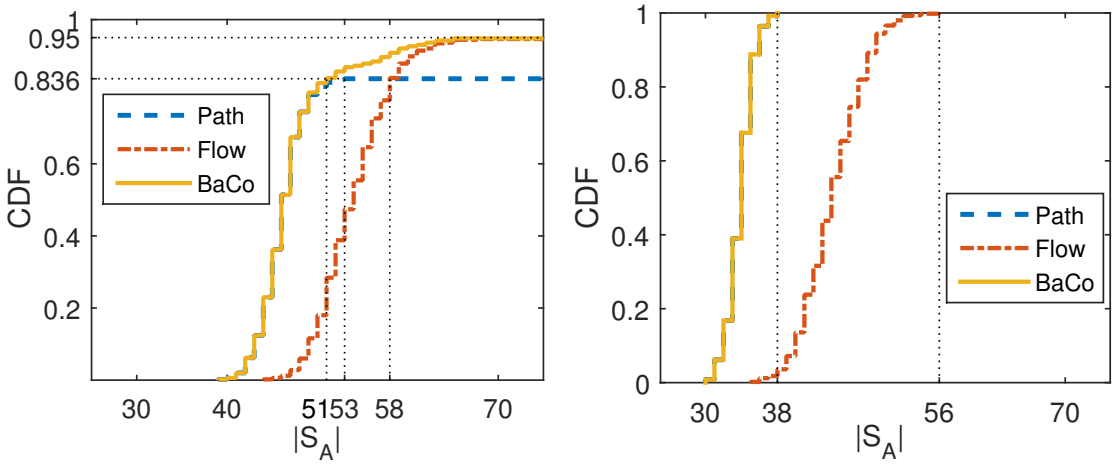
Figure 3.12 Comparison of schemes with and without the P_F constraint.

3.3.4.2 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 [Chen et al. (2013b)] (Phase 2 of the solution module). The full BaCo scheme utilizes both, as described in Section 3.3.3.3. 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.

The Effect of Signal Strength Ω Fig. 3.13(a) shows the CDF of $|S_A|$ when $\Omega = 12$ mW, a weak source signal, and Fig. 3.13(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. 3.13(a)

and completely in Fig. 3.13(b). Note the left shift in the CDFs between Fig. 3.13(a) and Fig. 3.13(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

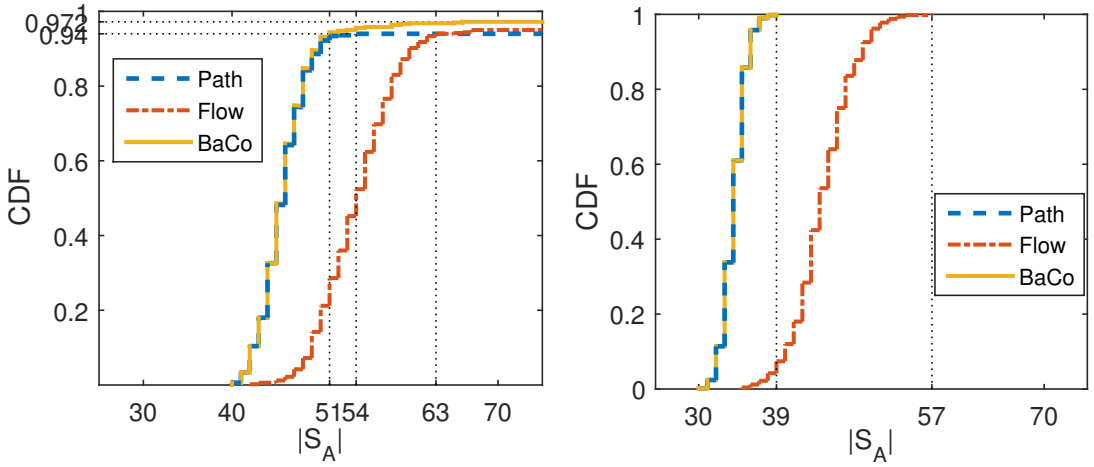
(b) $\Omega = 50$ mW

Figure 3.13 CDFs of $|S_A|$ for two Ω values.

When Ω is low (Fig. 3.13(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. 3.13(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.

The Effect of Sampling Rate f Fig. 3.14(a) shows the CDF of $|S_A|$ when $f = 0.5$ Hz and Fig. 3.14(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.



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

Figure 3.14 CDFs of $|S_A|$ for two sampling rates.

When f is low (Fig. 3.14(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. 3.14(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.

3.3.4.3 Convergence Speed

We also verified that BaCo's iterative algorithm converges quickly, regardless of 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 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 (3.34). Therefore, the detection gains do not change much. An example of the gain change can be seen in Fig. 3.10, where the decrease of the gain from Fig. 3.10(b) to Fig. 3.10(c) is smaller than that from Fig. 3.10(a) to Fig. 3.10(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.

3.4 Conclusions

In this chapter, we studied two min-num sensor selection problems under the probabilistic model. Static sensors are first randomly deployed and then selected to achieve $(P_D^{\min}, P_F^{\max})$-barrier coverage. Section 3.2 investigated the min-num sensor selection problem without data fusion. From our analysis, we learned that the sensor coverage radius is affected by the number of active sensors under the probabilistic model, if considering both the system false alarm probability and detection probability. To tackle this new challenge, we proposed an iterative framework to check the possible number of active sensors selectively. Section 3.3 studied the min-num sensor selection problem with decision fusion applied. The concept of detection gain was introduced to evaluate the detection capability of sensors. The detection gain is also affected by the number of active sensors in the system. A similar iterative algorithm was proposed to solve the min-num sensor selection problem with decision fusion.

CHAPTER 4. ON MINIMIZING THE COST OF ACTIVE SENSORS FOR STRONG BARRIER COVERAGE UNDER PROBABILISTIC MODEL

4.1 Introduction

To achieve barrier coverage, sensors can be manually deployed at desired positions, which is labor-intensive and may be dangerous in some scenarios, such as a battlefield. Alternatively, sensors can be dropped from an airplane or helicopter, resulting in a random deployment that may have coverage gaps. If the sensors are mobile, they can relocate themselves to the desired positions to form a barrier. However, mobile sensors typically cost more, and an all-mobile sensor barrier would be expensive. A potential compromise is to deploy a hybrid network, with both static and mobile sensors, to achieve barrier coverage.

In our scenario, static sensors are first deployed, potentially leaving coverage gaps. Then, mobile sensors are deployed to fill the gaps and form a barrier. Our goal is to select a subset of static sensors to use in the barrier and then determine the number of mobile sensors needed to fill any gaps. Our objective is to minimize the total cost of the active sensors, meaning those used to build the barrier. We leave the task of optimizing the actual sensor movement to other work, e.g. [Saipulla et al. (2010); Li and Shen (2015); Zhang et al. (2015)].

We consider *strong barrier coverage*, in which intruders may take any path to cross the monitored region. Unlike previous work such as [Wang et al. (2014b)] studying the hybrid sensor deployment problem under the disk model, we employ a probabilistic model that allows us to consider the practical constraints of system detection probability and false alarm probability in our solution. As the work presented in Chapter 3, we face the same challenge: the number of active sensors affects the decision threshold required to meet the false alarm probability constraint, which in turns affects the density of sensors required to meet the detection probability constraint.

Summarizing, this work provides the following contributions:

- We define the min-cost strong barrier problem under a probabilistic model, and transform the barrier construction problem with probabilistic constraints to a graph problem.
- We propose an efficient iterative algorithm to solve the problem, including speed-up strategies that skip some iterations and prune the graph in each iteration.

The chapter is organized as follows: Section 4.2 presents the probabilistic model and the problem statement. Section 4.3 describes the proposed scheme. Section 4.4 presents results from our evaluation of the proposed scheme. Finally, Section 4.5 concludes the chapter.

4.2 Model and Problem Statement

4.2.1 System Model

We consider a hybrid network of static and mobile sensors deployed to monitor a rectangular region with length L and width W, with the goal of detecting any intruders traversing the width of the region. We consider *strong barrier coverage*, meaning that an intruder, or *target*, may take any path to traverse the width of the region.

4.2.2 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. Our model consists of a source model, a detection model, and a false alarm model.

4.2.2.1 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 [Xing et al. (2009); Tan et al. (2011)]:

$$\omega_i(t) = \frac{\Omega}{1 + \big(d\,(s_i, t)\,\big)^{\alpha}}, \tag{4.1}$$

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

4.2.2.2 Detection Model

In our 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 (4.1) and the background noise n, as follows:

$$\mathbf{x}_i = \omega_i(t) + n. \tag{4.2}$$

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 set a decision threshold T. When a sensed reading $\mathbf{x}_i$ 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:

$$P_d(s_i,t) = 1 - F_N\big(T - \omega_i(t)\big). \tag{4.3}$$

Given a traversing path φ, we define $P_D(\varphi)$ as the maximum probability of detection, by any active sensor, for any point along the path. In other words, $P_D(\varphi)$ is the detection probability of the most well-monitored point in path φ. If we use S_A to denote the set of active sensors, then:

$$P_D(\varphi) = \max_{t\in\varphi} \max_{s_i\in S_A} P_d(s_i,t). \tag{4.4}$$

Strong barrier coverage assumes that the target may take any traversing path φ. Thus, the worst-case probability of detecting any given intruder is $P_D(\varphi)$ of the least-monitored φ. We call this worst-case probability the *system detection probability*, P_D, and define it as follows:

$$P_D = \min_{\varphi} P_D(\varphi) = \min_{\varphi} \max_{t\in\varphi} \max_{s_i\in S_A} P_d(s_i,t). \tag{4.5}$$

4.2.2.3 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. The probability of a particular sensor generating a false alarm is:

$$P_f = 1 - F_N(T). \tag{4.6}$$

We then define the *system false alarm probability* P_F as the probability that any sensor produces a false alarm, as follows:

$$P_F = 1 - (1 - P_f)^{|S_A|}, \tag{4.7}$$

where $|S_A|$ is the total number of active sensors. This definition of P_F is consistent with system false alarm probability in [Xing et al. (2009)] and [Yang and Qiao (2009)] and network false alarm rate in [Tan et al. (2011)].

4.2.3 Problem Statement

We define *strong ($P_D^{\min}$, $P_F^{\max}$)-barrier coverage* as a barrier coverage that requires $P_D \geq P_D^{\min}$ and $P_F \leq P_F^{\max}$ for any traversing path through the region. In this work, we consider a two-phase deployment strategy to achieve this coverage. First, N_s^{total} static sensors are randomly deployed in the monitored region, potentially leaving some coverage gaps. Then, mobile sensors are deployed to fill the coverage gaps between static sensors, ultimately forming a barrier. Mobile sensors usually are more expensive than static sensors, and we use ν to represent the mobile-to-static sensor cost ratio and we assume $\nu \geq 1$.

The goal is to minimize the overall cost of sensors used to achieve strong ($P_D^{\min}$, $P_F^{\max}$)-barrier coverage. Any static sensors not chosen to be active in the barrier remain in the monitored region. These inactive sensors may participate in construction of future barriers and are therefore not included in the cost of the current barrier. Let N_s be the number of static sensors selected to be active, and N_m be the number of mobile sensors needed to fill the coverage gaps between active static sensors. Formally, our problem is to minimize $(\nu N_m + N_s)$, subject to $P_D \geq P_D^{\min}$, $P_F \leq P_F^{\max}$, and $N_s \leq N_s^{\text{total}}$.

4.3 Proposed Scheme

4.3.1 Overview

To solve our min-cost barrier coverage problem, we propose a scheme that iterates over the assumed number of active sensors, N_A. The basic idea is to first assume a value for N_A, which is used to set the decision threshold T and calculate the sensing radius R_s. Then, we check whether the minimum cost for 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.

The scheme is composed of four modules, shown in Fig. 4.1. The initialization module determines the N_A value for the iteration and accelerates the scheme by skipping N_A values that will not produce a valid solution, meaning that $P_D^{\min}$ and $P_F^{\max}$ will not be satisfied. The initialization module also outputs the sensing radius corresponding to the value of N_A. The mapping module transforms the original problem into a graph problem by generating, updating, and pruning a weighted graph G, which is used in the next two modules.

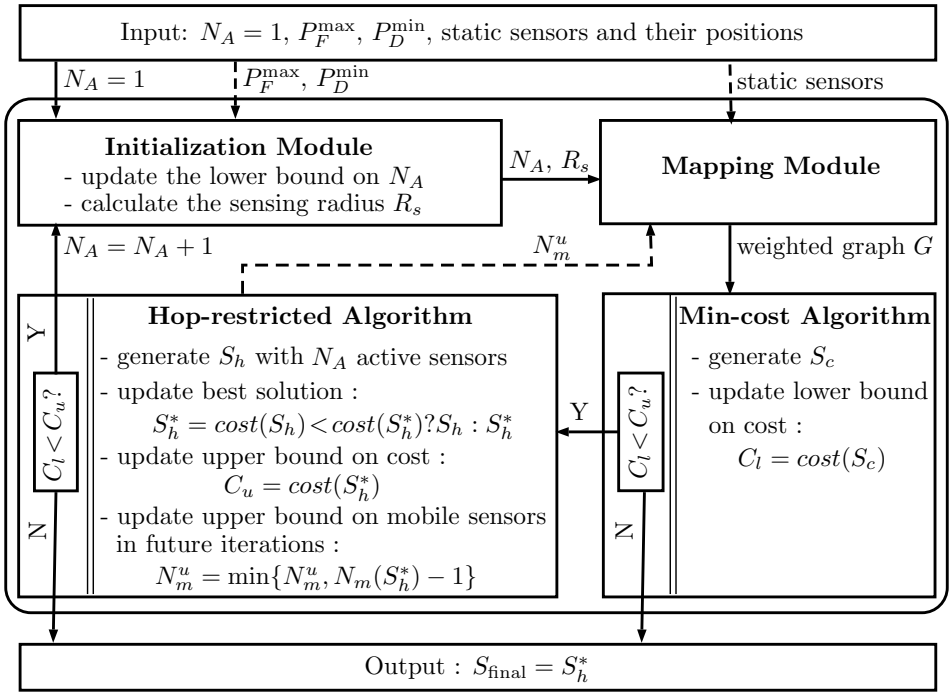


Figure 4.1 Overview of proposed scheme. The solid lines indicate iteration flow and the dashed lines indicate parameter flow.

The min-cost algorithm finds the cost lower bound C_l by identifying the min-cost set of sensors, denoted by S_c. The hop-restricted algorithm finds the cost upper bound C_u by identifying the best strategy with exactly N_A sensors, denoted by S_h. S_h^* represents the best feasible solution if the current N_A assumption is correct, and it is stored as a potential output. The number of mobile sensors in S_h^* is used to update N_m^u, the upper bound on the number of mobile sensors for future iterations. N_m^u is provided to the mapping module in order to prune G in each iteration. If the upper and lower bounds on cost meet (i.e., $C_l \geq C_u$), the scheme terminates and outputs a set of active sensors S_{final} as the final optimal deployment strategy. Otherwise, N_A is updated and iterations continue. Next, we introduce each module in detail.

4.3.2 Initialization Module

The initialization module, shown in Fig. 4.2, initializes each iteration. It determines the assumption of N_A to use for the iteration. The initialization module accelerates the scheme by skipping N_A values which will not produce a valid solution. The module takes a tentative N_A as input, and outputs the next value of N_A that has a valid solution.

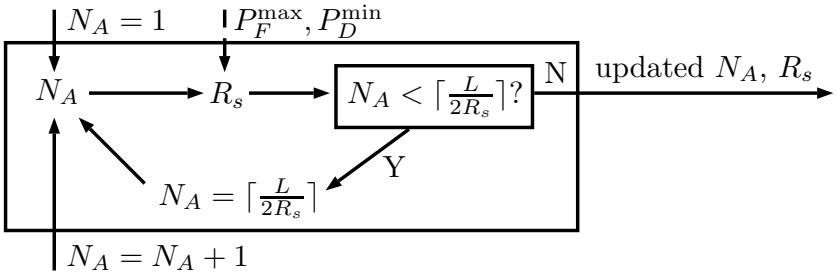


Figure 4.2 Initialization module.

As shown in Fig. 4.2, when entering this module, N_A is set to one for the first iteration, and $N_A + 1$ in subsequent iterations. Given N_A, $P_D^{\min}$, $P_F^{\max}$ and other sensing model-related parameters, this module performs the following steps.

1. Calculate the sensing threshold T. A sensed reading greater than T shall trigger an alarm. From (4.6) and (4.7), to satisfy $P_F \leq P_F^{\max}$, T is calculated as:

$$T \geq F_N^{-1}\left((1 - P_F^{\max})^{1/N_A}\right). \tag{4.8}$$

To maximize the coverage region, we choose the minimum value for T.

2. Calculate the sensing radius R_s. Intruders within this radius of a sensor shall be detected by that sensor with probability $P_D^{\min}$. Combining (4.1) and (4.3), we obtain:

$$R_s = \left(\frac{\Omega}{T - F_N^{-1}(1 - P_D^{\min})} - 1 \right)^{1/\alpha}. \tag{4.9}$$

3. Compute the minimum number of sensors required to achieve barrier coverage, $\lceil \frac{L}{2R_s} \rceil$.
4. If $\lceil \frac{L}{2R_s} \rceil$ is larger than N_A, then $N_A = \lceil \frac{L}{2R_s} \rceil$ and repeat the above steps; otherwise, the module outputs the current N_A.

The first three steps may need to be repeated because the value of N_A affects T, which affects R_s. When $\lceil \frac{L}{2R_s} \rceil > N_A$, the potential values of N_A between N_A and $\lceil \frac{L}{2R_s} \rceil$ are skipped because they will not lead to valid solutions, as proved in Theorem 4.1.

Theorem 4.1. *If $\lceil \frac{L}{2R_s} \rceil > N_A$, then for any $N'_A \in [N_A, \lceil \frac{L}{2R_s} \rceil)$, all deployment strategies with the assumption of N'_A active sensors would yield a P_F larger than $P_F^{\max}$, making them invalid solutions.*

Proof: Since $N'_A \geq N_A$, we have $R'_s \leq R_s$, because R_s is a decreasing function of N_A according to (4.9). This leads to $\lceil \frac{L}{2R'_s} \rceil \geq \lceil \frac{L}{2R_s} \rceil$. Let T' denote the sensor decision threshold and S'_A denote the set of active sensors in any deployment strategy corresponding to N'_A. Since $|S'_A| \geq \lceil \frac{L}{2R'_s} \rceil$, we have

$$\begin{aligned} P_F &= 1 - F_N\left(T'\right)^{|S'_A|} \geq 1 - F_N\left(T'\right)^{\lceil \frac{L}{2R'_s} \rceil} \\ &\geq 1 - F_N\left(T'\right)^{\lceil \frac{L}{2R_s} \rceil} > 1 - F_N\left(T'\right)^{N'_A} = P_F^{\max}. \end{aligned}$$ ■

As an example of how the initialization module works, suppose we want to build a barrier in an area of length $L = 14$ m. The signal amplitude emitted by the target, Ω, is 30 mW and the standard deviation of the environmental noise is $\sigma = 1$ mW. Given $P_F^{\max} = 0.05$, $P_D^{\min} = 0.95$, and $N_A = 1$, we obtain $R_s = 2.85$ m in Step 2 and $\lceil \frac{L}{2R_s} \rceil = 3$ in Step 3. Since $\lceil \frac{L}{2R_s} \rceil > N_A$, we return to Step 1 with $N_A = 3$. In the following steps, we obtain $R_s = 2.64$ m and $\lceil \frac{L}{2R_s} \rceil = 3$. Now $\lceil \frac{L}{2R_s} \rceil = N_A$, so we output $N_A = 3$ and $R_s = 2.64$ m.

4.3.3 Mapping Module

The mapping module maps the sensor network to an undirected weighted graph G to allow our scheme to use graph-based algorithms for optimization. G is initialized during the first iteration, and updated and pruned during the following iterations. Initially, G is constructed as follows.

- Vertices: There is a vertex for each static sensor. Two additional vertices, s_l and s_r, represent the left and right boundaries of the region, respectively.
- Edges: There is an edge between any two vertices.
- Weight: The weight of an edge is the cost of mobile sensors required to fill the gap between them, plus the cost of the static sensor at one end of the edge. In detail, the weights are listed below. R_s is the coverage radius corresponding to the N_A assumed for this iteration.
  - Between two physical, static sensors s_i and s_j:

$$w_{i,j} = \lceil \frac{\max\{\text{dist}(s_i, s_j) - 2R_s, 0\}}{2R_s} \rceil * \nu + 1. \tag{4.10}$$

  - Between a physical sensor s_i and a boundary:

$$w_{s_l,i} = \lceil \frac{\max\{x_i - R_s, 0\}}{2R_s} \rceil * \nu + 1. \tag{4.11}$$

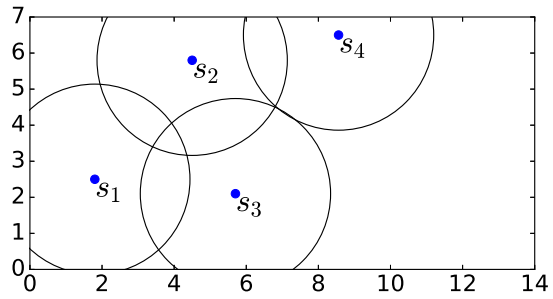
$$w_{i,s_r} = \lceil \frac{\max\{L - x_i - R_s, 0\}}{2R_s} \rceil * \nu. \tag{4.12}$$

  - Between the left and right boundaries:

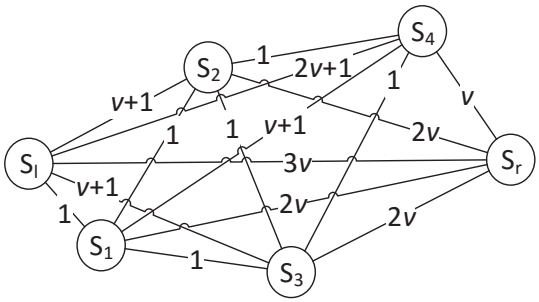
$$w_{s_l,s_r} = \lceil \frac{L}{2R_s} \rceil * \nu. \tag{4.13}$$

Continuing with the example from the previous section, let the input to the mapping module be $N_A = 3$ and $R_s = 2.64$ m, with the static sensors shown in Fig. 4.3(a). The graph G is then constructed as shown in Fig. 4.3(b).

We have two observations regarding G. First, any path from s_l to s_r in G represents a possible deployment strategy, and the sum of the weights of the edges in the path is the cost



(a) Sensor deployment. The circles are the coverage regions of the static sensors with $N_A = 3$.



(b) Weighted graph G. s_l and s_r represent the left and right boundaries. Edges are labeled with weights.

Figure 4.3 The graph initially constructed by the mapping module.

of the corresponding strategy. However, if this corresponding strategy requires more than N_A active sensors, then it is not necessarily valid with respect to $P_F^{\max}$ and $P_D^{\min}$.

For instance, in Fig. 4.3(b), $\{s_l, s_1, s_2, s_3, s_4, s_r\}$ is a possible deployment strategy that uses one mobile sensor between s_4 and s_r. But this strategy uses five sensors in total, which violates the assumption of $N_A = 3$, so this strategy may not be a valid solution. As another example, $\{s_l, s_1, s_r\}$ is a valid strategy with three sensors and a cost of $1 + 2\nu$; it uses two mobile sensors between s_1 and s_r.

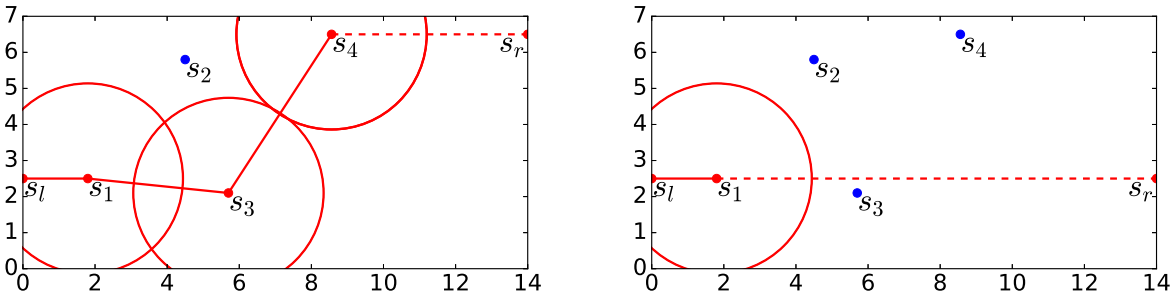
The second observation is that G initially is constructed as a fully connected graph. This is necessary because, in the optimal solution, mobile sensors may be deployed to fill the gap between *any* pair of static sensors, and not necessarily the smallest gap between two static sensor clusters. Although a barrier built by only filling the shortest gaps between static sensor clusters may use fewer mobile sensors, it also may use many more static sensors, possibly at a higher cost. To increase efficiency, our scheme prunes edges from the graph in future iterations. This pruning process will be discussed in Section 4.3.6.

4.3.4 Min-cost Algorithm to Update Cost Lower Bound

The goal of the min-cost algorithm is to find the solution that minimizes the active sensor cost. We achieve this by finding a minimum cost path from s_l to s_r on G using Dijkstra's

algorithm. The output of this algorithm is a set of active sensor nodes, S_c, that correspond to the vertices in the path, plus any mobile sensors that need to be added.

As an example, Fig. 4.4(a) shows the min-cost path S_c from the example in Fig. 4.3(b) when $\nu = 3$. This path is $\{s_l, s_1, s_3, s_4, s_r\}$. Three static sensors, s_1, s_3, and s_4, are used. One mobile sensor is required to fill the gap between s_4 and the right boundary s_r. The total cost of S_c is $cost(S_c) = 6$.



(a) Output of the min-cost algorithm when $N_A = 3$ and $\nu = 3$.

(b) Output of the hop-restricted algorithm when $N_A = 3$ and $\nu = 3$.

Figure 4.4 Example outputs of the graph algorithms. An edge between two sensors means that both sensors are active. A solid edge means their coverage regions overlap, requiring no mobile sensors. A dashed edge means mobile sensors are needed to fill the coverage gap between the sensors. In (a), one mobile sensor is required between s_4 and s_r, and in (b), two are required between s_1 and s_r.

S_c is the min-cost solution on G, but if $|S_c| > N_A$, the $P_F^{\max}$ requirement may not be satisfied by this solution. Therefore, S_c may be an invalid solution. However, even in this case, $cost(S_c)$ can still serve as a valid lower bound on the cost for future iterations. This is because, in future iterations, (1) the assumed number of active sensors increases, the sensor coverage radius decreases, and the edge weights on G increase; (2) the edge set of G is smaller because it is pruned in each iteration. Thus, any solution found in future versions of G would yield a higher cost than that of the current S_c.

Accordingly, the lower cost bound C_l is updated to $cost(S_c)$. The updated C_l is then compared to the cost upper bound, C_u. If $C_l < C_u$, a better solution may be found in future iterations, so the scheme continues to the hop-restricted algorithm. Otherwise, the scheme terminates and outputs the best valid solution yet found (from the hop-restricted algorithm

in a previous iteration). C_u is initialized to infinity, so in the example in Fig. 4.4(a), $C_l = cost(S_c) = 6 < C_u = \infty$ and we continue to the hop-restricted algorithm.

4.3.5 Hop-restricted Algorithm to Update Cost Upper Bound

The hop-restricted algorithm identifies the best path S_h from s_l to s_r that has exactly N_A physical sensors. This restriction ensures that S_h is a valid solution, distinguishing the hop-restricted algorithm from the min-cost algorithm in the previous module, which does not restrict hops and thus may not find a valid solution. The hop-restricted algorithm tracks the best valid solution found so far, and updates the cost upper bound C_u, terminating the scheme if the termination criterion is met. This algorithm also outputs the upper bound on the number of mobile sensors, N_m^u, which is used for graph pruning in the next iteration.

4.3.5.1 Hop-restricted Path S_h and Cost Upper Bound C_u

S_h is obtained by running a dynamic programming-based algorithm on G. For the algorithm, two sets of weights are needed:

- $w_{i,j}$: the cost of active sensors to fill the gap between s_i and s_j, as defined in Section 4.3.3, and
- $w_{i,j}^t$: the number of active sensors to fill the gap between s_i and s_j, obtained by setting the cost ratio ν in $w_{i,j}$ to one.

Let c_i^k denote the minimum cost of the path from s_l to s_i with k physical sensors. For $k \geq 1$,

$$c_i^k = \min_{j \in \Gamma_i} \{c_j^{k-w_{i,j}^t} + w_{i,j}\}, \quad \text{for } i = [1, 2, ..., s_r], \tag{4.14}$$

where Γ_i is s_i's neighborhood set. For $k \leq 0$, we define:

$$c_{s_l}^0 = 0, \tag{4.15}$$

$$c_i^0 = \infty, \quad \text{for } s_i \neq s_l, \tag{4.16}$$

$$c_i^k = \infty, \quad \text{for any } i \text{ if } k < 0. \tag{4.17}$$

The hop-restricted algorithm iterates over k and terminates when $c_{s_r}^{N_A}$ is obtained. S_h is then the path that reaches s_r with cost $c_{s_r}^{N_A}$. We define S_h^* as the best solution found so far. If

$cost(S_h)$ is less than $cost(S_h^*)$, we store S_h as S_h^* and update the cost upper bound C_u to $cost(S_h^*)$.

Fig. 4.4(b) shows the hop-restricted path S_h for the example in Fig. 4.3(b). With $N_A = 3$ and $\nu = 3$, S_h is $\{s_l, s_1, s_r\}$. One static sensor, s_1, is used. Two mobile sensors are needed to fill the gap between s_1 and the right boundary. The cost of S_h is 7, which is less than $cost(S_h^*) = \infty$ (as no S_h^* has previously been stored). S_h is then stored as S_h^*, and C_u is updated to 7.

Once C_u is updated, it is compared with C_l. If $C_l < C_u$, the scheme continues to the next iteration, with $N_A = N_A + 1$ as the input to the initialization module. Otherwise, the optimal solution has been found, and the scheme terminates and outputs S_h^*. In the above example, after executing the hop-restricted algorithm with $N_A = 3$, $C_l = 6 < C_u = 7$. Therefore, we continue with the next iteration and pass $N_A = 4$ to the initialization module.

4.3.5.2 Upper Bound on the Number of Mobile Sensors N_m^u

Another output of the hop-restricted algorithm is N_m^u, the upper bound of the number of mobile sensors for any solution in future iterations that has a lower cost. N_m^u is set as follows:

$$N_m^u = \min\{N_m^u, N_m^* - 1\}, \tag{4.18}$$

where N_m^* is the number of mobile sensors in S_h^*, shown as $N_m(S_h^*)$ in Fig. 4.1. For a solution S_h in any future iteration, we have

Theorem 4.2. *$cost(S_h) > cost(S_h^*)$ if $N_m \geq N_m^*$.*

Proof: Since N_A increases as the scheme iterates, we have $|S_h| > |S_h^*|$, where $|S_h|$ and $|S_h^*|$ are the number of active sensors on S_h and S_h^*, respectively. Then, we have

$$N_m + N_s > N_m^* + N_s^* \Longrightarrow N_s > N_m^* + N_s^* - N_m.$$

Further,

$$\begin{aligned} cost(S_h) = \nu N_m + N_s &> \nu N_m + N_m^* + N_s^* - N_m \\ &= (\nu - 1)N_m + N_m^* + N_s^* \\ &\geq (\nu - 1)N_m^* + N_m^* + N_s^* \\ &= \nu N_m^* + N_s^* = cost(S_h^*). \end{aligned}$$

■

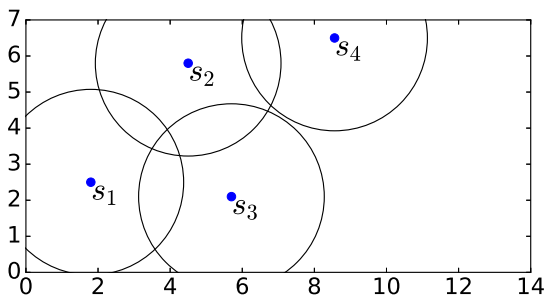
Theorem 4.2 shows that a solution S_h in future iterations with N_m^* or more sensors will have a higher cost than the current best solution S_h^*. Therefore, $N_m^* - 1$ is the upper bound of the number of mobile sensors in any solution in future iterations that has a lower cost.

4.3.6 Mapping Module Revisited

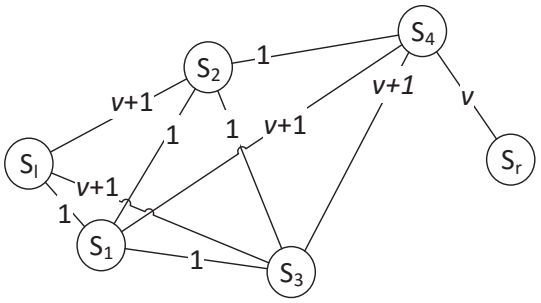
In the mapping module, after the first iteration, G only needs to be updated and pruned. G's vertex set remains the same for all iterations, while G's edge weights are updated and the edge set is pruned, using the following procedure:

1. Update: With the updated N_A and R_s in a new iteration, the number of mobile sensors needed to fill the gaps between static sensors is recalculated, and the edge weights of G are updated correspondingly.
2. Prune: After updating the weights of G, the edges of G that need more than N_m^u mobile sensors to fill the coverage gap are pruned.

Returning to the example, the solution shown in Fig. 4.4(b) was stored as S_h^* and N_m^u was updated to $N_m^u = N_m^* - 1 = 1$. Fig. 4.5 shows the sensor deployment and graph of this example in the following iteration, with $N_A = 4$. Comparing with Fig. 4.3, we can see that the weights of some edges have been updated. For example, a coverage gap has appeared between s_3 and s_4 due to the decreased R_s, and the weight of the edge has increased by ν to reflect that a mobile sensor is now required. Additionally, several edges have been pruned, such as the edge from s_3 to s_r.



(a) Sensor deployment. R_s is calculated with $N_A = 4$.



(b) Pruned weighted graph with $N_m^u = 1$.

Figure 4.5 The graph updated and pruned by the mapping module.

4.3.7 Terminating Condition

The scheme terminates when the upper and lower bounds for the cost meet or cross, meaning $C_l \geq C_u$. This can occur in either the min-cost algorithm or the hop-restricted algorithm. In our example, the min-cost algorithm is run on the newly pruned G in Fig. 4.5(b). The min-cost path in G is $\{s_l, s_1, s_2, s_4, s_r\}$, shown in Fig. 4.6.

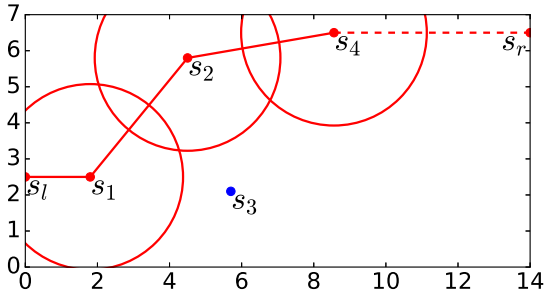


Figure 4.6 Output of the min-cost algorithm and hop-restricted algorithm when $N_A = 4$ and $\nu = 3$. The output consists of three static sensors (s_1, s_2, and s_4) and one mobile sensor to fill the gap between s_4 and the right boundary.

Its cost is 6, so C_l is updated to 6. Since C_l is still less than C_u, the hop-restricted algorithm is run on G, producing the same path. This path is saved as S_h^*. The cost upper bound C_u is then updated to 6 and now equals C_l, so the scheme terminates and outputs $S_{\text{final}} = S_h^* = \{s_l, s_1, s_2, s_4, s_r\}$. This solution consists of three static sensors (s_1, s_2, and s_4) and one mobile sensor to fill the gap between s_4 and the right boundary.

4.3.8 Complexity Analysis

Now let us do a complexity analysis. The total number of iterations will not be more than $|S_{\text{final}}|$, since we skip some N_A values in the initialization module. In an iteration with the assumed number of active sensors as N_A, the min-cost algorithm has a worst-case complexity of $O((N_s^{\text{total}})^2)$, and the hop-restricted algorithm has a worst-case complexity of $O(N_A|E|) = O(N_A(N_s^{\text{total}})^2)$ where $|E|$ is the number of edges on the weighted graph G. Sum up all the iterations, the worst-case complexity should be $O(|S_{\text{final}}|^2(N_s^{\text{total}})^2)$. In practice, the scheme

performs far more better than the worst case due to the skipping of N_A and the graph pruning strategy which is verified by simulation.

4.4 Evaluation Results

We evaluate the performance of the proposed scheme by varying the mobile-to-static sensor cost ratio (ν), the total number of deployed static sensors (N_s^{total}), and the $P_F^{\max}$ and $P_D^{\min}$ parameters. We also demonstrate the effectiveness of N_A skipping and the graph pruning strategy. In our simulations, sensors are uniformly randomly deployed in a 100 $\times$ 10 m rectangular region. The random uniform deployment of sensors is consistent with most of the studies of wireless sensor networks [Kumar et al. (2005); Liu et al. (2008); Yang and Qiao (2009); Saipulla et al. (2010); Chen et al. (2013b); Wang et al. (2014a); Mostafaei (2015); Li and Shen (2015)]. The default simulation parameters are shown in Table 4.1, which are chosen according to the real-world data mentioned in [Xing et al. (2009)]. All the simulation results are an average of 50 experiments.

Table 4.1 Default simulation parameters

Parameter	Meaning	Default value
$P_D^{\min}$	Minimum system detection probability	0.95
$P_F^{\max}$	Maximum system false alarm probability	0.05
Ω	Source signal strength	30 mW
α	Source signal decay exponent	2
F_N	CDF of noise distribution	Gaussian
μ	Noise mean	0 mW
σ	Noise standard deviation	1 mW
ν	Mobile-to-static sensor cost ratio	5
N_s^{total}	Total number of deployed static sensors	100

4.4.1 Effect of Cost Ratio

Fig. 4.7 demonstrates the effect of cost ratio on the number of active mobile and static sensors. When the cost ratio is one, meaning mobile sensors and static sensors have the same cost, the scheme only uses mobile sensors. This is because, in this case, a barrier between the left and right boundaries can be formed at minimum cost with a horizontal line of mobile sensors. At higher cost ratios, meaning more expensive mobile sensors, the scheme favors static sensors over mobile sensors. Additionally, as the cost ratio increases, the total number of active sensors also increases. This is because the scheme must seek out solutions that are more indirect and winding, requiring more static sensors, in order to reduce coverage gaps and the number of mobile sensors.

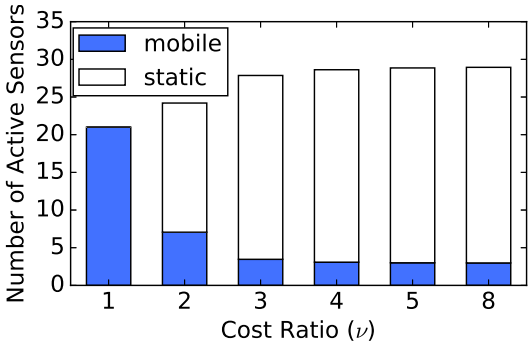
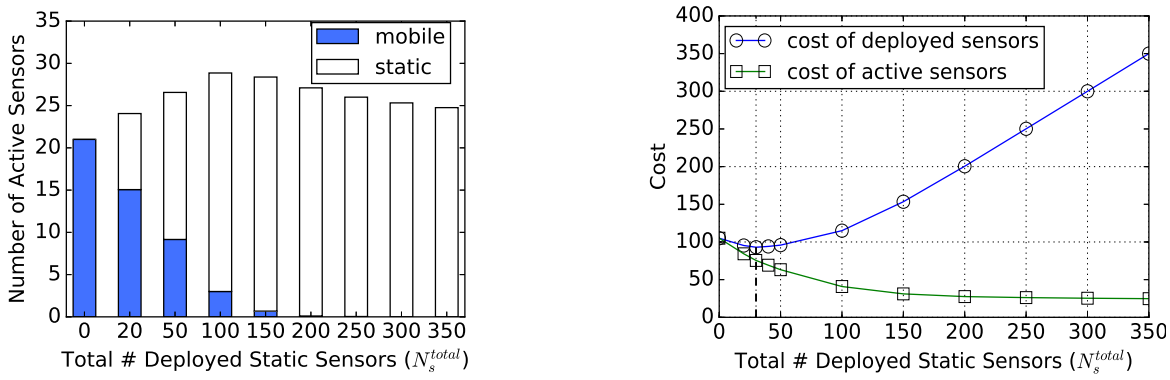


Figure 4.7 Effect of cost ratio.

4.4.2 Effect of the Total Number of Deployed Static Sensors

Fig. 4.8(a) shows the effect of the total number of deployed static sensors on the number of active mobile and static sensors. As more static sensors are deployed, fewer mobile sensors are utilized, because the static sensor deployment is better able to form a barrier at less cost on its own. When the number of deployed static sensors reaches a threshold (in this case 200), mobile sensors are no longer needed. Above this threshold, deploying more static sensors slowly reduces the total number of active sensors required. This is because more static sensors leads to more possible strategies that satisfy the barrier coverage requirements, and some of these additional strategies may use fewer active static sensors.



(a) Number of active sensors vs. total number of deployed static sensors.

(b) Sensor cost vs. total number of deployed static sensors.

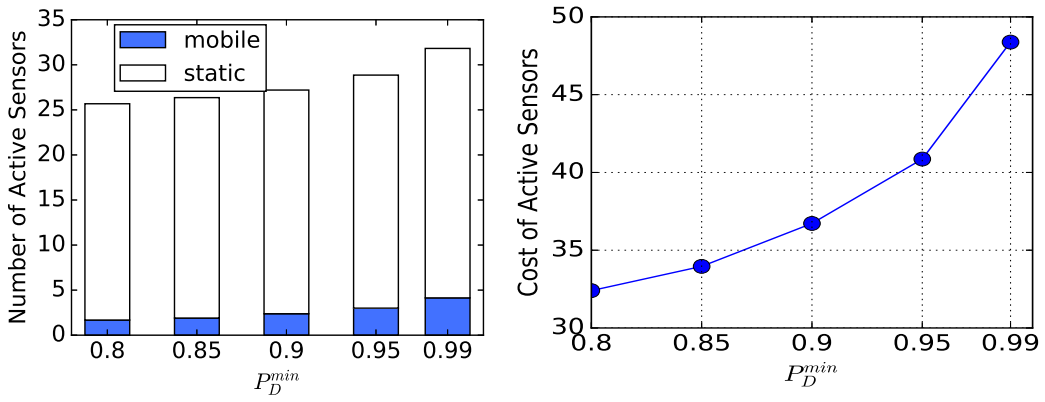
Figure 4.8 Effects of the total number of deployed static sensors.

Fig. 4.8(b) shows sensor costs as the total number of deployed static sensors increases. Two costs are evaluated. "Cost of active sensors" is the total cost of the active sensors, both mobile and static. "Cost of deployed sensors" is the total cost of all deployed sensors, both active and inactive. Note that the mobile sensors are deployed as needed and hence all of them are active.

In Fig. 4.8(b), the cost of deployed sensors first decreases and then increases. In the scenario demonstrated in Fig. 4.8(b), the minimum cost is reached on average when deploying 30 static sensors. When the number of deployed static sensors is below 30, more mobile sensors must be used, increasing the cost. When the number of deployed static sensors is higher than 30, the increased cost of deployed static sensors outweighs the decreased number of mobile sensors. Overall, we observe that the total number of deployed static sensors should be carefully chosen to minimize the total cost. On the other hand, the cost of active sensors, which is minimized by the proposed scheme, strictly decreases with the number of deployed static sensors. This aligns with the results in Fig. 4.8(a).

4.4.3 Effects of $P_D^{\min}$ and $P_F^{\max}$

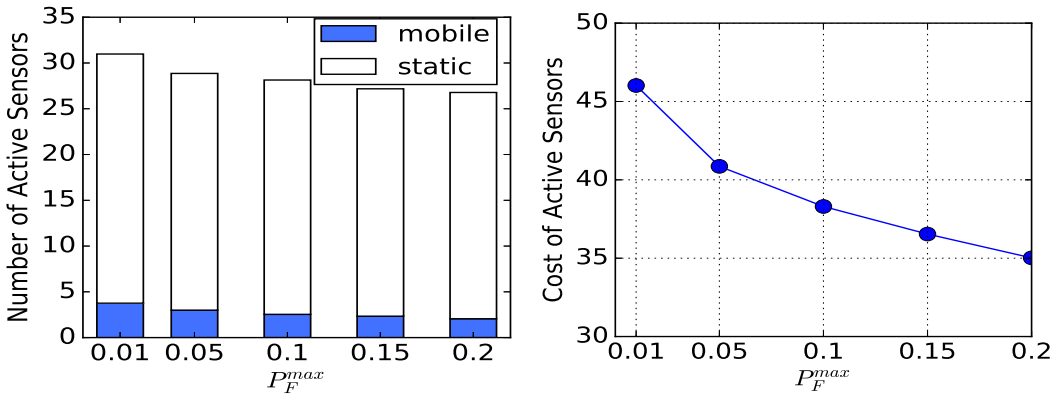
Fig. 4.9 shows the number of active sensors and their costs with different values of $P_D^{\min}$. The higher $P_D^{\min}$ is, the more sensors are needed to reach the required coverage level. The cost increases correspondingly.



(a) # active sensors vs. $P_D^{\min}$. (b) Cost of active sensors vs. $P_D^{\min}$.

Figure 4.9 Effect of $P_D^{\min}$.

Fig. 4.10 shows the number of active sensors and their costs with different values of $P_F^{\max}$. The higher $P_F^{\max}$ is, the fewer mobile and static sensors are needed. This is intuitive, as increasing $P_F^{\max}$ relaxes the constraint.



(a) # active sensors vs. $P_F^{\max}$. (b) Cost of active sensors vs. $P_F^{\max}$.

Figure 4.10 Effect of $P_F^{\max}$.

4.4.4 Effectiveness of N_A Skipping and Graph Pruning

Table 4.2 shows the number of iterations of different setups, compared with the number of sensors in the final solution $|S_{\text{final}}|$. As we can see, the number of iterations is less than $|S_{\text{final}}|$

since we skip some N_A values in the initialization module. The simulation results demonstrate the effectiveness of the skipping strategy.

Table 4.2 Number of iterations with different L and N_s^{total} ($W = 10$ m)

	$L = 100$ m			$L = 250$ m			$L = 500$ m		
N_s^{total}	50	100	200	50	100	200	50	100	200
$\lvert S_{\text{final}} \rvert$	26.6	28.9	27.1	61.3	66.2	73.7	118.3	124.7	135
# Iterations	8.7	10.9	9.1	10.4	15.3	22.8	10.3	16.9	27.1

Fig. 4.11 shows the number of edges in G throughout the iterations of the scheme, a measure of the effectiveness of the pruning process. Results are shown for 50 and 200 deployed static sensors. The number of edges is normalized to the number of edges in the initial, fully-connected graph. As the scheme iterates, the edges in G are gradually pruned. When the number of deployed static sensors is higher, more edges are pruned because fewer mobile sensors are used in the solutions, providing a tighter upper bound on the number of mobile sensors that can be included in an edge. Overall, the graph pruning process is shown to be effective. This helps expedite the scheme by reducing the computational complexity as the scheme runs.

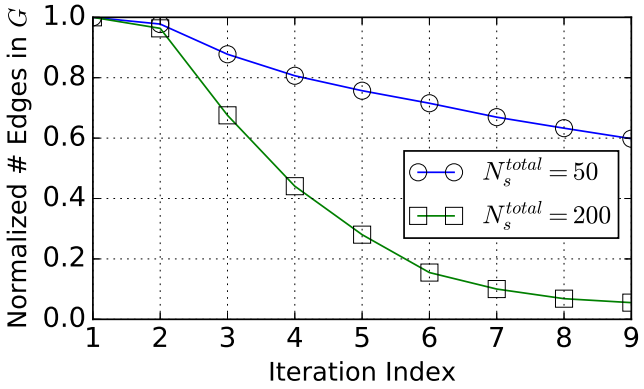


Figure 4.11 Number of edges in graph G, normalized to the initial, fully-connected graph, vs. iteration.

4.5 Conclusions

In this chapter, we proposed a scheme to solve the min-cost hybrid sensor deployment problem. We aimed to achieve $(P_D^{\min}, P_F^{\max})$-barrier coverage with a combination of mobile and static sensors while minimizing the total cost of active sensors. The problem was solved under an iterative framework, with the transformation to the hop-constrained shortest path problem in a graph. Simulation results show the proposed scheme can effectively choose the deployment strategy that achieves strong barrier coverage at a minimum cost.

CHAPTER 5. CONCLUSIONS AND FUTURE WORKS

5.1 Research Contributions

This dissertation explored the min-max sensor movement, min-num sensor selection, and min-cost hybrid sensor deployment problems arisen when providing barrier coverage in sensor networks. Main contributions of this dissertation are summarized as follows.

5.1.1 Min-max Sensor Movement Problem under Disk Model

We explored two min-max sensor movement problems under the disk model. We assumed mobile sensors were first randomly deployed in a 2D rectangular region, and then they relocated to designated destinations to form a strong horizontal barrier. With N mobile sensors available, the first problem selects the minimum number of sensors to form the barrier. The second problem may select any number out of the N available sensors to move to form the barrier. The challenge of the above two problems is the y-coordinate of sensor final positions, i.e., the barrier location, is unknown. To obtain the optimal barrier location, we proposed algorithms which first identify a set of discrete candidates for barrier locations, then check the candidates iteratively. For the first problem, our algorithm requires $O(N^4)$ iterations in the theoretical worst case, but in practice, it requires less than $O(N^2)$ iterations, as confirmed by simulation. For the second problem, our algorithm has a worst case time complexity of $O(\log \frac{\lambda_{\max}-\lambda_{\min}}{e} N^3 \lceil \frac{L}{2R} \rceil \log N)$ where $\lambda_{\min}$ and $\lambda_{\max}$ are the initial lower and upper bound of the optimal maximum moving distance.

5.1.2 Min-num Sensor Selection Problem under Probabilistic Model

We further investigated the min-num static sensor selection problem under the probabilistic model. The system false alarm probability and detection probability were jointly considered, and a $(P_D^{\min}, P_F^{\max})$-barrier coverage was defined. Our analysis showed that with the constraint

on the system false alarm probability, the number of active sensors affects the detection capability of sensors. This distinguishes the min-num sensor selection problem under the probabilistic model from the min-num sensor selection problem under the disk model. Specifically, we studied two sensor selection problems under the probabilistic model, without and with decision fusion, respectively. Both of them were solved under an iterative framework where the number of active sensors is assumed and validated iteratively, while with different evaluation methods for the detection capabilities.

5.1.3 Min-cost Barrier Coverage in Hybrid Network under Probabilistic Model

Finally, we studied the min-cost barrier construction problem in a hybrid network under the probabilistic model. Hybrid network means the network is composed of mobile and static sensors. We considered a two-step deployment strategy; first, the static sensors are randomly deployed, and then the mobile sensors are deployed and relocate to merge the coverage gaps left by static sensors. We aimed to find the proper coverage gaps to deploy mobile sensors such that $(P_D^{\min}, P_F^{\max})$-barrier coverage is achieved, and the total cost of the barrier is minimized. Under the probabilistic model, we solved the problem by iteratively trying multiple assumptions of the number of active sensors, and for each assumed number of active sensors, the min-cost deployment strategy is obtained by solving a hop-constrained shortest path problem.

5.2 Future Works

5.2.1 Sensor Movement Problems

5.2.1.1 Min-sum Sensor Movement Problems under Disk Model

Though we investigated two 2D min-max sensor movement problems under the disk model, the corresponding 2D min-sum sensor movement problems are left unexplored. If considering constructing a horizontal strong barrier, similar to the min-max problems, the min-sum problems can be classified into several categories according to whether the final x-coordinates and y-coordinates of sensors are given. The corresponding "given-x, given-y" problem has been solved in [Ban et al. (2010)], while the other three problems lack research. The objective of

min-max and min-sum can also be jointly considered to prolong the total barrier lifetime in multi-round barrier construction.

5.2.1.2 Curve Barrier Construction with Mobile Sensors under Disk Model

For the two 2D min-max sensor movement problems solved in Chapter 2, the final barrier has to be horizontal. However, since the randomly deployed sensors scatter anywhere in the deployed region, they would move shorter distances if the final barrier is not required to be horizontal. There is a lack of research works for constructing curve barriers with mobile sensors with the objective of min-max or min-sum.

5.2.1.3 Sensor Movement Problems under Probabilistic Model

Since data fusion under a probabilistic model can expand the coverage regions of sensors, it is worth investigating the sensor movement problems under the data fusion model. Data fusion would help to reduce the sensor moving distances. However, since data fusion requires extra energy for data communication over the lifetime of the barrier, while the energy consumption for sensor relocation is an one-time expense, we can not conclude that data fusion will prolong the lifetime of the barrier and the problem needs further investigation.

5.2.2 Sensor Selection and Deployment Problems

5.2.2.1 Energy/Cost-efficient Sensor Selection with Value Fusion

We investigated the min-num sensor selection problem under the probabilistic model with and without decision fusion in Chapter 3. The other data fusion method, value fusion, outperforms decision fusion regarding the detection capability of sensors [Clouqueur et al. (2004)], but it requires more data to be transferred among sensors. It is valuable to compare value fusion with decision fusion regarding the overall cost and energy consumption of sensors when achieving barrier coverage. To do the comparison, the foremost thing might be finding a way to evaluate the detection capability of sensors for an intruder path under the value fusion model.

5.2.2.2 Barrier Coverage in Hybrid Network with Data Fusion

Similar to the static sensor selection problem, achieving barrier coverage in a hybrid network with mobile and static sensors under the data fusion model lacks investigation. The problem becomes complicated when it involves movement, sensor selection, and data fusion at the same time. The selection of active static sensors affects the locations of coverage gaps. To merge the coverage gaps, we can utilize either data fusion or mobile sensors. Schemes should be designed to find the best sensor selection and deployment strategy by weighing the various sensor collaboration and movement choices.

BIBLIOGRAPHY

Ahmed, N., Kanhere, S. S., and Jha, S. (2005). Probabilistic coverage in wireless sensor networks. In *Proc. IEEE LCN*.

Ammari, H. M. and Giudici, J. (2009). On the connected k-coverage problem in heterogeneous sensor nets: The curse of randomness and heterogeneity. In *Proc. IEEE ICDCS*.

Andrews, A. M. and Wang, H. (2017). Minimizing the aggregate movements for interval coverage. *Algorithmica*, 78(1):47–85.

Ban, D., Yang, W., Jiang, J., Wen, J., and Dou, W. (2010). Energy-efficient algorithms for k-barrier coverage in mobile sensor networks. *International Journal of Computers Communications & Control*, 5(5):616–624.

Benkoczi, R., Friggstad, Z., Gaur, D., and Thom, M. (2015). Minimizing total sensor movement for barrier coverage by non-uniform sensors on a line. In *Proc. Springer Symposium on ALGOSENSORS*.

Benkoczi, R., Gaur, D. R., and Thom, M. (2016). A 2-approximation algorithm for barrier coverage by weighted non-uniform sensors on a line. In *Proc. Springer Symposium on ALGOSENSORS*.

Bhattacharya, B., Burmester, M., Hu, Y., Kranakis, E., Shi, Q., and Wiese, A. (2009). Optimal movement of mobile sensors for barrier coverage of a planar region. *Theoretical Computer Science*, 410(52):5515–5528.

Cardei, M., Thai, M. T., Li, Y., and Wu, W. (2005a). Energy-efficient target coverage in wireless sensor networks. In *Proc. IEEE Infocom*.

Cardei, M., Wu, J., Lu, M., and Pervaiz, M. O. (2005b). Maximum network lifetime in wireless sensor networks with adjustable sensing ranges. In *Proc. IEEE WiMob*.

Chakrabarty, K., Iyengar, S. S., Qi, H., and Cho, E. (2002). Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers*, 51(12):1448–1453.

Chen, A., Kumar, S., and Lai, T. H. (2007). Designing localized algorithms for barrier coverage. *in Proc. of ACM MobiCom*.

Chen, D. Z., Gu, Y., Li, J., and Wang, H. (2013a). Algorithms on minimizing the maximum sensor movement for barrier coverage of a linear domain. *Discrete & Computational Geometry*, 50(2):374–408.

Chen, J., Li, J., and Lai, T. H. (2013b). Energy-efficient intrusion detection with a barrier of probabilistic sensors: Global and local. *IEEE Transactions on Wireless Communications*, 12(9):4742–4755.

Cheng, T. M. and Savkin, A. V. (2009). A distributed self-deployment algorithm for the coverage of mobile wireless sensor networks. *IEEE Communications Letters*, 13(11):877–879.

Clouqueur, T., Saluja, K. K., and Ramanathan, P. (2004). Fault tolerance in collaborative sensor networks for target detection. *IEEE Transactions on Computers*, 53(3):320–333.

Czyzowicz, J., Kranakis, E., Krizanc, D., Lambadaris, I., Narayanan, L., Opatrny, J., Stacho, L., Urrutia, J., and Yazdani, M. (2009). On minimizing the maximum sensor movement for barrier coverage of a line segment. In *Proc. Springer ADHOC-NOW*.

Czyzowicz, J., Kranakis, E., Krizanc, D., Lambadaris, I., Narayanan, L., Opatrny, J., Stacho, L., Urrutia, J., and Yazdani, M. (2010). On minimizing the sum of sensor movements for barrier coverage of a line segment. In *Proc. Springer ADHOC-NOW*.

Dobrev, S., Durocher, S., Eftekhari, M., Georgiou, K., Kranakis, E., Krizanc, D., Narayanan, L., Opatrny, J., Shende, S., and Urrutia, J. (2015). Complexity of barrier coverage with relocatable sensors in the plane. *Theoretical Computer Science*, 579:64–73.

Eftekhari, M., Kranakis, E., Krizanc, D., Morales-Ponce, O., Narayanan, L., Opatrny, J., and Shende, S. (2013). Distributed algorithms for barrier coverage using relocatable sensors. In *Proc. ACM symposium on PODC*.

Han, X., Cao, X., Lloyd, E., and Shen, C.-C. (2008). Deploying directional sensor networks with guaranteed connectivity and coverage. *in Proc. of IEEE SECON*.

He, S., Chen, J., Li, X., Shen, X., and Sun, Y. (2012). Cost-effective barrier coverage by mobile sensor networks. In *Proc. IEEE Infocom*.

Kershner, R. (1939). The number of circles covering a set. *American Journal of Mathematics*, 61(3):665–671.

Kim, D., Wang, W., Son, J., Wu, W., Lee, W., and Tokuta, A. O. (2017). Maximum lifetime combined barrier-coverage of weak static sensors and strong mobile sensors. *IEEE Transactions on Mobile Computing*, 16(7):1956–1966.

Kumar, S., Lai, T. H., and Arora, A. (2005). Barrier coverage with wireless sensors. In *Proc. of ACM MobiCom*.

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

Lazos, L. and Poovendran, R. (2006). Stochastic coverage in heterogeneous sensor networks. *ACM Transactions on Sensor Networks*, 2(3):325–358.

Li, S. and Shen, H. (2015). Minimizing the maximum sensor movement for barrier coverage in the plane. In *Proc. IEEE Infocom*.

Liu, B., Brass, P., Dousse, O., Nain, P., and Towsley, D. (2005). Mobility improves coverage of sensor networks. In *Proc. ACM MobiHoc*.

Liu, B., Dousse, O., Wang, J., and Saipulla, A. (2008). Strong barrier coverage of wireless sensor networks. *in Proc. of ACM MobiHoc*.

Liu, B. and Towsley, D. (2004). A study of the coverage of large-scale sensor networks. In *Proc. IEEE MASS*.

Mehrandish, M., Narayanan, L., and Opatrny, J. (2011). Minimizing the number of sensors moved on line barriers. In *Proc. IEEE WCNC*.

Mostafaei, H. (2015). Stochastic barrier coverage in wireless sensor networks based on distributed learning automata. *Computer Communications*, 55:51–61.

Punnen, A. P. and Nair, K. (1994). Improved complexity bound for the maximum cardinality bottleneck bipartite matching problem. *Discrete Applied Mathematics*, 55(1):91–93.

Saipulla, A., Liu, B., Xing, G., Fu, X., and Wang, J. (2010). Barrier coverage with sensors of limited mobility. In *Proc. ACM MobiHoc*.

Saipulla, A., Westphal, C., Liu, B., and Wang, J. (2009). Barrier coverage of line-based deployed wireless sensor networks. In *Proc. IEEE Infocom*.

Shen, C., Cheng, W., Liao, X., and Peng, S. (2008). Barrier coverage with mobile sensors. In *Proc. IEEE I-SPAN*.

Tan, R., Xing, G., Wang, J., and Liu, B. (2011). Performance analysis of real-time detection in fusion-based sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 22(9):1564–1577.

Wang, G., Cao, G., and Porta, T. L. (2004). Movement-assisted sensor deployment. *Proc. of IEEE Infocom*.

Wang, H. and Zhang, X. (2015). Minimizing the maximum moving cost of interval coverage. In *Proc. Springer ISAAC*.

Wang, J. and Zhong, N. (2006). Efficient point coverage in wireless sensor networks. *Journal of Combinatorial Optimization*, 11(3):291–304.

Wang, W., Srinivasan, V., Chua, K.-C., and Wang, B. (2007). Energy-efficient coverage for target detection in wireless sensor networks. In *Proc. ACM/IEEE IPSN*.

Wang, Z., Chen, H., Cao, Q., Qi, H., and Wang, Z. (2014a). Fault tolerant barrier coverage for wireless sensor networks. In *Proc. of IEEE Infocom*.

Wang, Z., Liao, J., Cao, Q., Qi, H., and Wang, Z. (2014b). Achieving k-barrier coverage in hybrid directional sensor networks. *IEEE Transactions on Mobile Computing*, 13(7):1443–1455.

Wu, P., Cao, X., Wu, X., and Chen, G. (2012). Energy-efficient stochastic target coverage in sensor surveillance systems. In *Proc. IEEE ICCCN*.

Xing, G., Tan, R., Liu, B., Wang, J., Jia, X., and Yi, C.-W. (2009). Data fusion improves the coverage of wireless sensor networks. In *Proc. ACM MobiCom*.

Xing, G., Wang, X., Zhang, Y., Lu, C., Pless, R., and Gill, C. (2005). Integrated coverage and connectivity configuration for energy conservation in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 1(1):36–72.

Xu, B., Kim, D., Li, D., Lee, J., Jiang, H., and Tokuta, A. O. (2014). Fortifying barrier-coverage of wireless sensor network with mobile sensor nodes. In *Proc. WASA*.

Xu, X. and Sahni, S. (2007). Approximation algorithms for sensor deployment. *IEEE Transactions on Computers*, 56(12):1681–1695.

Yang, G. and Qiao, D. (2009). Barrier information coverage with wireless sensors. In *Proc. IEEE Infocom*.

Yang, G., Zhou, W., and Qiao, D. (2007). Defending against barrier intrusions with mobile sensors. In *Proc. WASA*.

Zhang, H. and Hou, J. C. (2005). Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc & Sensor Wireless Networks*, 1(1-2):89–124.

Zhang, X., Wymore, M. L., and Qiao, D. (2015). Optimized barrier location for barrier coverage in mobile sensor networks. In *Proc. IEEE WCNC*.

Zou, Y. and Chakrabarty, K. (2004). Sensor deployment and target localization in distributed sensor networks. *ACM Transactions on Embedded Computing Systems*, 3(1):61–91.

Zou, Y. and Chakrabarty, K. (2005). A distributed coverage-and connectivity-centric technique for selecting active nodes in wireless sensor networks. *IEEE Transactions on Computers*, 54(8):978–991.