BARRIER COVERAGE WITH WIRELESS SENSOR NETWORKS

Mohsen Eftekhari Hesari

A thesis

In the Department

of

Computer Science & Software Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy (Computer Science) at

Concordia University

Montréal, Québec, Canada

April 2014

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Mohsen Eftekhari Hesari**
Entitled: **Barrier Coverage with Wireless Sensor Networks**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Sheldon S. Williamson

_____ External Examiner
Dr. Binay Bhattacharya

_____ External to Program
Dr. Chadi Assi

_____ Examiner
Dr. Hovhannes A. Harutyunyan

_____ Examiner
Dr. Thomas G. Fevens

_____ Thesis Supervisor
Dr. Lata Narayanan

_____ Thesis Supervisor
Dr. Jaroslav Opatrny

Approved by

_____
Dr. Volker Haarslev, Graduate Program Director

_____ 2014

_____
Dr. Christopher W. Trueman, Interim Dean
Faculty of Engineering and Computer Science

Abstract

Barrier Coverage with Wireless Sensor Networks

Mohsen Eftekhari Hesari, Ph.D.

Concordia University, 2014

We study the problem of *barrier coverage* with a wireless sensor network. Each sensor is modelled by a point in the plane and a sensing disk or coverage area centered at the sensor's position. The barriers are usually modelled as a set of line segments on the plane. The barrier coverage problem is to add new sensors or move existing sensors on the barriers such that every point on every barrier is within the coverage area of some sensors. Barrier coverage using sensors has important applications, including intruder detection or monitoring the perimeter of a region.

Given a set of barriers and a set of sensors initially located at general positions in the plane, we study three problems for relocatable sensors in the centralized setting: the *feasibility* problem, and the problems of minimizing the maximum or the average relocation distances of sensors (*MinMax* and *MinSum* respectively) for barrier coverage. We show that the MinMax problem is strongly NP-complete when sensors have arbitrary ranges and can move to arbitrary positions on the barrier. We also study the case when sensors are restricted to use perpendicular movement to one of the barriers. We show that when the barriers are parallel, both the MinMax and MinSum problems can be solved in polynomial time. In contrast, we show that even the feasibility problem is strongly NP-complete if two perpendicular barriers are to be covered.

For the barrier coverage problem in distributed settings, we give the first distributed local algorithms for fully synchronous unoriented sensors. Our algorithms achieve barrier coverage for a line segment barrier when there are enough sensors to cover the entire barrier. Our first algorithm is oblivious and terminates in $\Theta(n^2)$ time, whereas our second one uses two bits of memory at each sensor, and takes $\Theta(n)$ steps, which is asymptotically optimal. However, if the sensors are semi-synchronous, and do not share the same orientation, we show that no algorithm exists that always terminates within finite time. Finally, for sensors that share the same orientation we give an algorithm that terminates within finite time, even if all sensors are fully asynchronous.

Finally, we study barrier coverage with multi-round random deployment using stationary sensors. We analyze the probability of barrier coverage with uniformly dispersed sensors as a function of parameters such as length of the barrier, the width of the intruder, the sensing range of sensors, as well as the density of deployed sensors. We propose two specific deployment strategies and analyze the expected number of deployment rounds and deployed sensors for each strategy. We present a cost model for multi-round sensor deployments, and for each deployment strategy we find the optimal density of sensors to be deployed in each round that minimizes the total expected cost. Our results are validated by extensive simulations.

Acknowledgments

I wish to express my gratitude to my supervisors, Prof. Narayanan and Prof. Opatrny. Their encouragement, guidance, and deep support during my study enabled me to develop an understanding of the subject. Their wide knowledge and detailed and constructive comments have been a great value for me.

I am grateful to all my student colleagues for their help and support. I would like to specially thank Sirma Altay, Puspal Bhabak, Hayk Grigoryan, and Louisa Harutyunyan for their friendship and help.

I owe my best friends, Abolfazl, Arash, Atefeh, Behnam, and Nona a lot for their continued moral support.

Lastly, and most importantly, I wish to thank my parents and my sisters, Samaneh and Mahsa, for their understanding, support and encouragement. They have lost a lot due to my studies abroad. To them I dedicate this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A wireless ad hoc network is an infrastructure-less network consisting of many wireless nodes. The data propagation in a wireless ad hoc network does not rely on any pre-existing infrastructure e.g. routers or access points; instead, wireless nodes forward data from other nodes. A wireless sensor network is an ad hoc network in which every node has a microcontroller, a communication module and sensing modules. For convenience, in the rest of this thesis, we refer to the nodes of a wireless sensor network simply as *sensors*. The goal of a wireless sensor network is to monitor some aspects of the environment. Each sensor is equipped with a set of sensing module (e.g. motion, light, temperature and humidity) and gathers information from its surrounding environment. Each sensor is also equipped with a communication module that enables it to communicate with other sensors or base stations. Every sensor in the network gathers information from its surrounding environment and passes it on to the base station(s) possibly through other sensors. The applications of wireless sensor networks are broad, ranging from temperature or humidity control in a building to habitat monitoring or surveillance of a restricted area and localization and tracking of an entity of interest (for examples see [ZG04]).

Surveillance, also called intrusion detection, is an important application of wireless

sensor networks. The goal of a surveillance system is to detect the entrance of entities of interest in a restricted area. For simplicity we use the terms *intruder* and *intrusion detection* to refer to the entity of interest and its detection by the network respectively. However this usage of the terms does not imply any illegal or unfavorable entrance or activity of the entity of interest. The literature on intrusion detection using wireless sensors can be classified into two major categories:

Area Coverage The goal of area coverage is the monitoring of an entire region [HT03, KLB04, MKPS01], on the assumption that the intruder might appear at any point in the region and must be detected within a fixed time delay.

Barrier Coverage In contrast to area monitoring, the focus of barrier coverage is on monitoring the entire boundary or a part of the boundary of a given region [BBSK07, BBH+09, CKK+09, CKK+10, KLA05].

Assuming that an intruder cannot enter a region without crossing its boundary, monitoring the boundary of the region is sufficient to detect all intruders with possibly fewer sensors. Wireless sensors can be deployed to monitor the perimeter of a restricted area, thereby creating a virtual barrier. Compared to wired alternatives, a wireless sensor-based monitoring system can provide a cost-effective solution for surveillance and intrusion detection.

The focus of this thesis is on the barrier coverage problem with wireless sensors. In the following we first present the models that we use to study barrier coverage and then we present the problems that we tackle in this thesis.

1.1 Model and Definitions

We proceed to define three essential entities: barrier, sensor, and intruder. We go on to discuss the various mobility models we use in this thesis, the possible sensor

deployment policies, and coverage redundancy. A barrier is the boundary, or a part of the boundary, of a restricted area that we wish to monitor. Examples are the border between two countries with the purpose of monitoring and regulating movement of people and vehicles between the two countries or a border between two sides of a jungle to study the movements of animals between the two sides. The sensors are the small pieces of equipment that are scattered along the barrier to monitor it. Finally the intruder is an entity of interest that crosses the barrier. To formulate the problem we need to first define model(s) for each of these three entities. We start with the barrier:

1.1.1 Barrier

There are two common models for a barrier in the literature:

- A barrier can be modelled as a long narrow band with a relatively small width compared to its length [KLA05, CKL07, LDWS08]
- The barrier can be modelled as a line segment, given that its width is zero [CKK+09, CKK+10, MNO11].

The problem may involve more than one barrier segment. For example arbitrary curved barriers can be modelled as a set of barrier segments. In fact in many problems we consider a set of barriers as an input.

1.1.2 Sensors

In a monitoring system each sensor is equipped with the essential sensing module and may or may not have other optional modules. Here are the list of the modules that we need for our problem formulation:

Sensing module: The sensing module may be a motion detector, heat detector, or detector of any other type of property that can be used to detect intruders. In this thesis, we do not go into details of the sensing module but look at it as a provided interface. For any sensor there is a neighborhood in which the sensor is capable of detecting intruders. We call this neighborhood the sensing area. In general, the sensing area of a sensor can have any shape and the event detection probability could be non-homogeneous throughout the sensing area of the sensor (depending on the sensor module, the environment properties, etc.) but in this thesis we only consider sensors with *unit disk sensing areas* and we assume that a sensor can detect an intruder if and only if it is within the sensing disk of the sensor. This is a model which is commonly used in the literature of wireless sensor networks.

Mobility module: In general, mobility of sensors may help barrier coverage by allowing sensors to move from overly monitored areas to those areas that do not have sufficient sensors for intended coverage. The mobility of sensors may be restricted in different ways, for example, the direction or the maximum distance that a sensor can move. Throughout this thesis we are using different mobility assumptions which are explained in more detail later on in this chapter.

Visibility module: The visibility module enables the sensor to locate other neighboring sensors when they are close enough. Visibility is different from sensing: the visibility module enables detection of other *sensors*, while the sensing module detects *intruders*. As in the case of sensing module, we define a visibility range for each sensor and assume that a sensor u can see another sensor v if and only if v lies within u's visibility range. We generally assume that when u sees sensor v, it can also determine its distance from v.

Communication module: A communication module enables sensors to communicate to each other or predefined base stations, for example to report collected sensor data or collaboratively compute or aggregate a function of this data. However in this thesis we are only concerned with the establishment of barrier coverage and not the collection of sensor data. In general communication gives more power to sensors e.g. local or even global knowledge of network topology, possibility of having centralized algorithms etc. However, the algorithms in this thesis *do not use any communication between sensor nodes*. Nevertheless, one way to achieve visibility of nearby sensors could be via carrier sensing which utilizes the radio transceiver of the communication module.

1.1.3 Intruder

An intruder can be modelled as a 2-dimensional shape in the plane whose trajectory is a curve that intersects the barrier. In this thesis we consider the intruder as an object with non-zero area. We assume that a sensor detects an intruder if and only if the intersection of the sensing range of the sensor and the intruder area is non-zero.

As mentioned earlier, different mobility capabilities can be considered for sensors. In the following we discuss the models that we use throughout this thesis in more detail.

1.1.4 Mobility Model

As stated earlier, the mobility module enables a sensor to move from its initial deployment to a new position. Sensor networks can be divided into 3 categories based on the ability of sensors to move:

Stationary sensors : sensors have no movement module and do not have any mobility capability.

Relocatable sensors : All or some sensors can relocate to new positions after the initial deployment but after reaching their final positions, they stay stationary. During the border monitoring phase sensors do not move.

Mobile sensors : All or some sensors are moving constantly (patrolling) after the deployment and during the border monitoring phase.

Relocatable and mobile sensors can increase the performance of the barrier coverage by covering the previously uncovered areas of the barrier. Our set of problems consider sensors that are either all stationary or all relocatable.

Even for mobile and relocatable sensors, the mobility can be limited in many ways. Limitations on both the distance and the direction that a sensor moves can be assumed.

1.1.5 Sensor Deployment Policies

A key decision in the design and implementation of a wireless sensor monitoring system is choosing the method of deploying the sensors.

There are three major sensor deployment strategies:

1. Deterministic deployment
2. Multi-round random deployment
3. Ad hoc deployment using relocatable sensors

In a deterministic deployment, sensors are deployed in predetermined positions. Whereas in a multi-round random deployment, sensors are dispersed randomly on the barrier in rounds. A new round of deployment is necessitated if the intended barrier coverage is not achieved in the previous rounds [YQ10]. Finally, in an ad hoc deployment with mobile sensors, sensors are initially located at arbitrary positions

and then some of the sensors may relocate to new positions such that the entire barrier is covered [SCLP08, CKK+09, CKK+10, CGLW12]. It is not difficult to see that with a deterministic deployment or the use of relocatable sensors, possibly fewer sensors are needed. However, deterministic deployment or use of relocatable sensors may not be feasible in many situations (e.g. hazardous or mountainous areas). On the other hand, with random deployment it is hard if not impossible to guarantee achieving barrier coverage in finite number of deployment rounds. Finally, the installation cost of the system as a combination of deployment cost and sensors cost should also be taken into account when deciding the deployment strategy.

In this thesis we consider two of the deployment strategies: multi-round random deployment and ad hoc deployment using relocatable sensors.

1.1.6 Coverage Redundancy

One challenge in the design of wireless sensor monitoring systems is errors in intrusion detection. There are errors related to the sensing modules of the sensors: false positive and true negative. Also it is possible that although a sensor correctly detects an intruder the information cannot reach a base station due to communication problems. In general, one way to reduce the rate of errors in a sensor network is deploying redundant sensors on the barrier. We say a border is (k, w)-covered if any intruder with a width greater than or equal to w is detected by at least k distinct sensors. The concept is similar to weak k-barrier coverage defined in [KLA05], and differs only in generalizing the notion of intruders considered there.

1.2 Problem Statement

The problems that we tackle in this thesis can be categorized into three sets. In the following we describe each set of problems separately.

1.2.1 Centralized Barrier Coverage of Multiple Barriers with Relocatable Sensors

In this set of problems, we assume that there exists a centralized entity that has global knowledge about the entire barrier and the sensors on it. Therefore this entity can determine if any portion of the barrier is not covered and then, if necessary, make the decision about which sensor to move to which position such that the intended barrier coverage is achieved. In this thesis, we study the problem with respect to different constraints:

- Feasibility problem: Given an input of sensors and barriers the problem is to decide whether there exists a set of final positions (one for each sensor) such that all the barriers are fully covered.
- MinSum problem: Given the input the problem is to find a feasible solution such that the sum of all movements (displacements of sensors) is minimized.
- MinMax problem: Similar to MinSum but instead of the sum of movements the maximum movement is the subject of minimization.

The study of the MinSum and the MinMax problems, is motivated by the minimization of the overall energy consumption of sensors, and the minimization of the deployment time (e.g. the time that the last sensor takes to reach its final position), respectively.

1.2.2 Distributed Barrier Coverage of A Single Barrier with Relocatable Sensors

For this set of problems, we assume that sensors are autonomous robots, each with localized information gathered from within its visibility radius. Here again depending on the assumed properties of sensors (whether sensors have a local or global sense of orientation, sensing model of sensors, time synchronization between sensors and the barrier shape) the results vary from giving optimal algorithms to proving non-existence of any distributed algorithm for the problem.

1.2.3 Barrier Coverage of A Single Barrier with Stationary Sensors and Random Deployment

Finally, in the last set of problems, we assume that sensors are dispersed randomly along the barrier and they cannot move once they are deployed. Assuming sensors with fixed sensing ranges much less than the barrier length, and finite number of deployed sensors, the intended barrier coverage is achieved with some probability p less than 1. This implies that with probability $1 - p$ the barrier is not covered and since sensors cannot move, more sensors needed to be deployed on the barrier. However if these new sensors added in the second round are also randomly deployed, still there may be a chance that the barrier is not covered after the second round of deployment and therefore more rounds of deployment may be needed until intended coverage is achieved. In this thesis we study several *multi-round random deployment* strategies.

1.3 Contributions of Thesis

In this section we summarize our contributions in each of the sub-problems introduced in the previous section:

1.3.1 Centralized Algorithms for Barrier Coverage of Multiple Barriers with Relocatable Sensors

We consider several variations of the problem of covering a set of barriers (modelled as line segments) using relocatable sensors. Unlike the previous studies that only considered initial positions of sensors being on the line containing the barrier, we consider sensors initially located at general positions in the plane. Given a set of barriers and a set of sensors located in the plane, we study three problems: (i) the feasibility of barrier coverage, (ii) the MinMax problem, and (iii) the MinSum problem. When sensors are permitted to move to arbitrary positions on the barrier, the MinMax problem is shown to be strongly NP-complete for sensors with arbitrary ranges. We also study the case when sensors are restricted to use perpendicular movement to one of the barriers. We show that when the barriers are parallel, both the MinMax and MinSum problems can be solved in polynomial time. In contrast, we show that even the feasibility problem is strongly NP-complete if two perpendicular barriers are to be covered, even if the sensors are located at integer positions, and have same sensing ranges. On the other hand, we give polytime algorithms for special cases of the feasibility problem. We also present several approximation algorithms for the problem.

1.3.2 Distributed Algorithms for Barrier Coverage of A Single Barrier with Relocatable Sensors

We study barrier coverage with relocatable sensors that have same sensing ranges and are initially located at arbitrary positions on the barrier and can relocate along the barrier. We study the problem in the distributed settings for the first time. We assume that each sensor has a *constant visibility range* and can move only a *constant distance* in every cycle. Furthermore we consider three different synchronization schemes for the sensors: *fully synchronous (FSYNC)*, *semi-synchronous (SSYNC)* and *asynchronous (ASYNC)* sensors. The results vary extensively based on the model used.

For fully synchronous sensors, we give the first two distributed algorithms that achieve barrier coverage for a line segment barrier when there are enough sensors in the network to cover the entire barrier. Our algorithms are local in the sense that sensors make their decisions independently based only on what they see within their constant visibility range. One of our algorithms is oblivious whereas the other uses two bits of memory at each sensor to store the type of move made in the previous step. We show that our oblivious algorithm terminates within $\Theta(n^2)$ steps with the barrier fully covered, while the constant-memory algorithm is shown to take $\Theta(n)$ steps to terminate in the worst case. Since any algorithm that can only move a constant distance in one step requires $\Omega(n)$ steps on some inputs, our second algorithm is asymptotically optimal.

We show that if there is no agreement between sensors on a global orientation (sensors are unoriented), then there is no algorithm for barrier coverage that always terminates. Obviously, the non-existence results also hold true for asynchronous sensors.

Finally, assuming that sensors share a global orientation, we give an algorithm

for barrier coverage that terminates after finite time even if sensors are fully asynchronous.

1.3.3 Barrier Coverage of A Single Barrier with Stationary Sensors and Random Deployment

We study multi-round wireless sensor deployment on a border modelled as a line segment. We present two different classes of deployment strategies: *complete* and *partial*. In complete strategies, in every round, sensors are deployed over the entire border segment, while in partial strategies, sensors are deployed over only some part(s) of the border. First, we analyze the probability of (k, w)-coverage for any complete strategy as a function of parameters such as length of barrier to be covered, the width of the intruder, the sensing range of sensors, as well as the density of deployed sensors. Second, we propose two specific deployment strategies - *Fixed-Density Complete* and *Fixed-Density Partial* - and analyze the expected number of deployment rounds and expected total number of deployed sensors for each strategy. Next, we present a model for cost analysis of multi-round sensor deployment and calculate, for each deployment strategy, the expected total cost as a function of problem parameters and density of sensor deployment. Finally we find the optimal density of sensors in each round that minimizes the total expected cost of deployment for each deployment strategy. We validate our analysis by extensive simulation results.

1.4 Outline of Thesis

The rest of this thesis is organized as follows: In Chapter 2 we summarize the related work on the barrier coverage problem. Centralized algorithms for barrier coverage with relocatable sensors are studied in Chapter 3. In Chapters 4 we study distributed

algorithms for barrier coverage with fully synchronous sensors while Chapter 5 is devoted to barrier coverage problem using semi-synchronous and asynchronous sensors. In Chapter 6 we study the multi-round random deployment of stationary sensors and finally in Chapter 7 we conclude this thesis and give directions for future research.

Chapter 2

Related Work

In this chapter we review the current state of the research in the area of barrier coverage with wireless sensors. The concept of barrier coverage with sensor networks was first introduced in [Gag92] as one possible application for WSNs. Based on the mobility capabilities of sensors, we classify the literature on barrier coverage into three categories: barrier coverage with stationary sensors, barrier coverage with relocatable sensors and barrier coverage with mobile sensors.

Sensors with mobility can possibly increase the number of solutions to a barrier coverage problem. Mobile sensors, if redundant in their initial position, can relocate to cover gaps on the barrier which require new deployments otherwise [BJY+10, CGLW12, CKK+09, CKK+10, MLC+12, MNO11, SLX+10]. Also continuously moving sensors (patrolling) can increase the chance of an intruder being detected [HCL+12, KLZ10]. However in this thesis we do not study mobile sensors (see [KLZ10] and [HCL+12] as examples on barrier coverage with mobile sensors).

Also another solution to covering the gaps on the barrier can be obtained by increasing the sensing ranges of currently deployed sensors. In [MLC+12] the sensing range of sensors which results in minimum total power consumption of sensors as a function of the sensing range is calculated. However, in this thesis we do not consider

changing the initial sensing range of sensors.

In the following we summarize the previous literature on barrier coverage with relocatable and stationary sensors:

2.1 Barrier Coverage with Relocatable Sensors

Relocating the sensors after the initial deployment is studied in many previous works [BBH+09, CKK+09, CKK+10, SLX+10, MNO11]. As mentioned in Chapter 1, relocating sensors is one way to achieve barrier coverage after the initial deployment. Given initial positions of the sensors, the goal of a sensor relocating algorithm is to determine how the sensors should move to new positions such that the entire barrier is fully covered. These algorithms can be classified into two major categories: centralized versus distributed algorithms. In the following we study the literature in each category separately:

2.1.1 Centralized Algorithms

The centralized version of the problem has been studied with respect to different constraints: minimizing the maximum distance traveled by every sensor (MinMax), minimizing the sum of the distances traveled by all sensors (MinSum), or minimizing the number of sensors that moved (MinNum).

In [SLX+10] barrier coverage using sensor relocatable sensors is studied. The deployment strategy is as follows: In the first phase, sensors are dropped aiming for some pre-determined locations on parallel lines (k parallel lines to achieve k-barrier coverage). Because of the deployment error the drop location might be within an error range of the pre-determined location. In the second phase each sensor moves to its final position within its moving range and stays stationary afterward. Based on

this model, the authors introduced an algorithm to minimize the maximum moving distance of sensors while keeping strong k-barrier coverage property. Their algorithm is basically discretizing the problem and solving it by a mixture of binary search technique and a maximum flow algorithm.

In [DHM+09] different optimization problems in the area of mobile sensors are studied. Minimizing the maximum, sum or number of movements to obtain a specific property (e.g. obtain a connected network, or a connected path) is studied. Although the studied problems is not directly related to barrier coverage, the measures and techniques are used in barrier coverage problem as well.

In [BBH+09] the problem of relocating sensors which were initially deployed inside a circular or simple polygon area is studied. The objective is to solve the MinMax and MinSum problems. The authors assumed that the sensing range of the sensors are large enough to cover the entire perimeter of the area. Different variations of the problem are studied. The authors provided polynomial time algorithms for the MinMax problem when sensors are located inside or on the perimeter of a circle or simple polygon. Also they provided approximation algorithms for the MinSum problem. The question whether the MinSum problem is NP-complete or not is left as an open problem. It was also shown that there exists a trivial solution to the problem of MinSum when sensors are located initially on a line segment barrier or on the perimeter of a circular barrier.

In [CKK+09, CKK+10, MNO11] the authors considered a model in which sensors are initially located on a line segment. The sensors are relocated (by moving left or right) to new positions to achieve coverage of the whole barrier. In contrast to [BBH+09], in all these papers the sensing range of the sensors are also taken into account.

In [CKK+09] different variations of the MinMax (minimizing the maximum movement of sensors to achieve coverage) problem are studied. The authors considered two cases: 1) sensors have same sensing range 2) sensors have different sensing range. A quadratic time algorithm is introduced for the MinMax problem when sensors have the same sensing range. Also a linear time 2-approximation algorithm is introduced to find the solution to the MinMax problem when the coverage is feasible (sum of the sensing range of sensors is greater than the length of the barrier). The complexity of the general MinMax problem with sensors of different sensing range was left as an open problem. But a variation of the problem where the position of one sensor is fixed is shown to be NP-complete.

Recently, in [CGLW12] it was shown that the MinMax is polytime solvable even when sensors have different sensing ranges. Also authors introduced an $O(n \log n)$ algorithm for the problem when sensors have the same sensing range. Furthermore a linear time algorithm is provided for the case where sensors are initially located on the barrier itself.

In [CKK+10] the authors studied the problem of minimizing the sum of the movements to achieve coverage (MinSum) when sensors are initially located on a line and the barrier is a portion of the same line. The authors took into account the sensing range of the sensors and the possibility of a sensor being initially positioned outside the barrier. The problem is shown to be NP-complete when sensors have different sensing ranges and polynomial time algorithms are introduced for the case where sensors have the same sensing range. Considering the MinSum problem where sensors have identical sensing ranges, for the case where total sensing range of the sensors is equal to the length of the barrier, linear time algorithms are given to calculate the optimal solution. Also for the case where sensing range of the sensors is greater than the length of the barrier an algorithm of $O(n^2)$ running time is introduced.

In [MNO11] the problem of minimizing the *number* of relocated sensors to achieve coverage (Min-Num) of a linear barrier when sensors are located on the barrier is studied. The authors showed that the problem is NP-hard when sensors have different sensing range. Also the Min-Num problem is shown to have a polynomial solution when all sensors have the same sensing range. All the cases when sensors have the same sensing range, are studied and for each case a polynomial algorithm is introduced. The authors also studied the case where the barrier is the perimeter of a circle and sensors are located on it. It is shown that the problem is NP-hard if sensors have different sensing ranges and a polynomial time algorithm is provided for the case where all sensors have the same sensing range. The Min-Num problem when the barrier consists of more than one line segment or when there are constant number of different sensing ranges, is left as an open problem.

2.1.2 Distributed Algorithms

While to our knowledge, the barrier coverage problem has not been studied in the distributed setting, there is a large body of recent work on the capabilities of *autonomous mobile robots* that is related to our work; see for example [FPS12, Mat94, PS06, SY99]. Initially the robots are assumed to be at arbitrary positions on the plane. Their goal is to collectively solve a given task; a typically studied task is the formation of some kind of pattern in the plane. Each robot repeatedly performs a *Look-Compute-Move* cycle. First it looks at the positions of the other robots, then it computes its own next position, and finally it moves to this new position. The robots are anonymous and identical, have no centralized coordination, nor do they communicate with each other; their decisions are made solely based on their observations of their surroundings. Different variations of the model exist based on whether or not the robots are synchronous, have agreement on a coordinate system, and have a local memory.

The computation model we use in Chapters 4 and 5 falls into the same general framework. Indeed, our sensors follow a Look-Compute-Move cycle, and have partial agreement on a coordinate system. However, our sensors have two important restrictions compared to the standard model of autonomous mobile robots. The sensors in our model have a *constant visibility range*, and in each cycle, they move a *constant distance*. Both these restrictions are more realistic for sensors. There are two other differences between our sensors and the typical autonomous mobile robot. First, the sensors may not be identical; for example, their sensing ranges could be different. Second, the sensors are not points on the plane; indeed they are similar to the *fat robots* studied in [CGP09, DDCM13], in the sense that they can detect each other's presence when their sensing ranges overlap.

Limited visibility is indeed an important restriction, and some papers have studied the impact of this restriction. As mentioned in [ASY95], there is no deterministic algorithm for the *point formation* or *gathering* problem, even for two robots under limited visibility, if they don't see each other at first. The authors give a synchronous algorithm that solves point formation for sensors in the same visibility graph, even if they don't share a coordinate system. In [FPSW01], an asynchronous algorithm for the same problem is given under limited visibility and a common coordinate system. In terms of movement capabilities, in the autonomous mobile robot literature, while the distance a robot can move in one cycle is assumed to be finite, there is no fixed bound on this distance.

The two problems that are closest to the barrier coverage problem and have been studied in the context of autonomous robots are the *line or circle formation problem* and the *spreading problem*. In the line formation problem, n robots are initially placed at arbitrary positions on the plane, and they must move to place themselves on a line, which is not specified in advance. The problem can be solved with total or

partial agreement on the coordinate system and unlimited visibility. Our problems in Chapter 4 and 5 differ in that the sensors are already on the barrier (a line segment or a circle); they must move to achieve complete coverage of the barrier. In the spreading problem [CP06], n robots are initially placed on a line, and must move to equidistant positions between the leftmost and rightmost robots. Clearly in a fully synchronous model and transparent sensors, where each robot can see all the other robots, the problem can be solved in one step, and is similar to the line formation problem. However, the authors of [CP06] make an assumption of a particular type of limited visibility: a robot can only see the robots that are closest to it (e.g. sensors are opaque). They show that the simple strategy of moving to the midpoint of the two neighbors converges to sensors with equidistant positions. Also a version of the algorithm in which sensors are fully synchronous and every sensor knows the number of other sensors on its left and right, is shown to terminate after $n-2$ time steps where n is the number of sensors. It is important to point out that in contrast to [CP06], in our model, a sensor only sees sensors in its visibility range, and therefore may not even see the sensor that is closest to it. Additionally, it cannot move an arbitrary distance in one cycle as in [CP06], it can move only a fixed distance independent of the distance between any two sensors.

The authors in [FPS08] studied the spreading of mobile sensors on a ring. They showed that if the ring is not directed (sensors do not share a common orientation of the ring) then there is no algorithm that can achieve exact spreading where the distance between every two consecutive sensors is the same. The negative result holds even if sensors have unlimited visibility range, unbounded memory and computational power and all their actions (computations and movements) are instantaneous. However when it comes to an oriented ring, the authors presented an exact algorithm for

spreading problem when sensors are aware of the final required distance between consecutive sensors. An approximation algorithm that converges to uniform spreading of sensors is presented when this distance is unknown to the sensors. The positive results hold true even if sensors are oblivious (have no memory of the past), asynchronous and have fixed visibility range.

In [EB09] authors extended the study of the spreading problem on a ring with oblivious agents. The authors assume that agents agree on a grid coordinate and they move only on the grid positions. Also every agent has a fixed visibility range of d^*. Let n and k denote the length of the ring and the number of sensors. The authors show that if $d^* < \lfloor n/k \rfloor$ and the ring is undirected (no global agreement between agents on the orientation of the ring), the spreading task is impossible. However an algorithm for the spreading problem is given when $d^* \geq \lceil n/k \rceil$ and the ring is directed. Note that in the given algorithm agents may never stop moving although spreading is achieved. However when agents are required to spread and stop (quiescent spread) authors showed that the task is impossible under the presented model. An algorithm which achieves quiescent and almost uniform spread is presented.

The barrier coverage problem has the same input as the spreading problem, but it differs in that the final positions are not required to be equidistant; instead they are required to achieve coverage. Equidistant positions are neither necessary nor sufficient for barrier coverage in general, though in the situation where the number and range of identical sensors is exactly enough to cover the barrier, the final positions of sensors would have to be equidistant. The difference in the problem requirements leads to different results. For example in Chapter 4 under certain conditions (e.g. extra sensor), we show how our algorithms can be extended for barrier coverage of a ring whereas the spreading problem in the same model is shown to be impossible. However our results conform to the previous results when the spreading and barrier

coverage requirements coincide (e.g. when the number of sensors are exactly enough to cover the barrier).

The focus of [FPS08] and [EB09] are sensors/agents located on a ring which is different than our main focus in this thesis. However some of our results are also extended for circular barriers.

2.2 Barrier Coverage with Stationary Sensors

The study of barrier coverage with random deployment was initiated in the seminal paper of Kumar et. al. [KLA05]. The authors introduced the notion of strong and weak k-barrier coverage for barriers modelled with narrow strips and intruders trajectories modelled as crossing paths. Strong k-barrier coverage guarantees the detection of any intruder that crosses the barrier by at least k distinct sensors. In contrast, weak k-barrier coverage only guarantees the detection of intruders along paths that are *orthogonal* to the barrier. It can be seen that strong coverage implies weak coverage but not vice versa. The authors studied the weak k-barrier coverage problem on narrow strips with width proportional to the inverse of the length. Critical conditions for weak k-barrier coverage are calculated when the length of the barrier goes to infinity. It was shown that to ensure barrier coverage with high probability, there should be at least $\log n$ active sensors on each orthogonal crossing path where n denotes the total number of sensors. Also an optimal deterministic deployment strategy to obtain k-barrier coverage (k parallel rows of sensors) with fewest possible number of sensors is presented.

In [CKL07] a new measure for barrier coverage called *L-local k-barrier coverage* is introduced. In contrast with strong barrier coverage, this new measure can be determined locally by sensors. A barrier is L-local k-barrier covered if and only if any crossing path that fits in a segment of length L of the barrier is detected by at least

k distinct sensors. Algorithms to determine whether the barrier is L-local k-barrier covered are introduced.

In [LDWS08] the work of [KLA05] is extended by removing the constraint on the width of the barrier and showing that strong barrier coverage can be achieved with high probability (whp) at a certain sensor density if width of the barrier is $\Omega(\log l)$, where l denotes the length of the barrier. They also proved that strong barrier coverage cannot be achieved whp if the width of the barrier is $o(\log l)$ regardless of the density of the sensors. The authors also studied the construction of sensor barriers with the help of so-called *vertical barriers*, which improve the chance of barrier coverage by dividing the barrier into smaller segments.

In [LZS+11] weak k-barrier coverage is studied when sensors are deployed randomly. Given a barrier of finite length and total number of deployed sensors, a lower bound on the probability that the barrier is weakly k-barrier covered is calculated. Simulation results show the tightness of the estimated lower bound when k is small. The authors also provided an algorithm that determines whether a given barrier is weakly k-barrier covered and if not, to calculate the percentage of the barrier which is not weakly k-barrier covered.

In [YQ10] a multi-round deployment strategy is studied. Randomness in the deployment is modelled using an error range R_{err}. At each round of deployment, the intended location of sensors is determined using the uncovered distances (gaps) and number of available sensors for that round. In the deployment phase, due to the randomness of the deployment, rather than falling at its intended location, each sensor might fall at a distance from it. However, this distance from the intended location is bounded by R_{err}. After the end of a round of deployment, the remaining new gaps are calculated and the procedure is repeated until all gaps are covered. The goal is to minimize the total number of sensors used to cover the entire barrier. The

authors derived the optimal solution for the general n-round deployment and showed that the optimal number of sensors can always be achieved by using only two rounds.

The probability of *path coverage* is studied in [NMA10]. The authors considered Poisson-distributed sensors on a closed curve (belt) of finite length and a given width. The probability of a barrier being 1-weak barrier-covered is calculated as a function of the number and sensing range of sensors. Also an estimate of the number of coverage gaps, and the length of gaps is derived. In their model, sensors are not necessarily located on the barrier but in a strip of some width.

Chapter 3

Centralized Algorithms Using Relocatable Sensors[1]

In this chapter we consider the algorithmic complexity of several natural generalizations of the barrier coverage problem with sensors of arbitrary ranges. We generalize the work in [CGLW12, CKK+09, CKK+10, MNO11] in two significant ways. First, we assume that the initial positions of sensors are arbitrary points in the two-dimensional plane, not necessarily on the line containing the barrier. This assumption is justified since in many situations, initial dispersal of sensors on the line containing the barrier might not be practical. Second, we allow more than one barriers that are parallel or perpendicular to each other. This generalization is motivated by barrier coverage of the perimeter of an area.

3.1 Computational Model and Preliminaries

Throughout this chapter, we assume that a set of sensors $S = \{s_1, s_2, \ldots, s_n\}$ is given where sensors $s_1, s_2, \ldots, s_n$ are located in the plane in positions $p_1, p_2, \ldots, p_n$

[1]Partial results of this chapter are also published in [DDE+13]

respectively, with $p_i = (x_i, y_i)$ for some real values x_i, y_i. The sensing ranges of the sensors are positive real values $r_1, r_2, \ldots, r_n$, respectively. A sensor s_i can detect any intruder in the closed circular area around p_i of radius r_i. We assume that sensor s_i is mobile and thus can relocate itself from its initial location p_i to another specified location p_i'. A *barrier* b is a line segment in the plane. We define an *arrangement* as an ordered pair $\mathcal{A} = (S, \mathcal{B})$ comprising a set S of sensors and a set $\mathcal{B}$ of barriers. Given an arrangement $\mathcal{A} = (S, \mathcal{B})$ with $\mathcal{B} = \{b_1, b_2, \ldots, b_k\}$, and $S = \{s_1, s_2, \ldots, s_n\}$ and sensors located at $p_1, p_2, \ldots, p_n$ in the plane, of sensing ranges $r_1, r_2, \ldots, r_n$, the *barrier coverage* problem is to determine for each s_i its final position p_i' on one of the barriers, so that all barriers are collectively covered by the sensing ranges of the sensors. We call such an assignment of final positions a *covering assignment*. Figure 3.1 shows an example of a barrier coverage problem and a possible covering assignment. We are also interested in optimizing some measure of the movement of sensors involved to achieve coverage, such as MinMax and MinSum. We use standard cost measures such as Euclidean or rectilinear distance. The distance between the initial position p and a final position p' of a sensor is denoted by $d(p, p')$.

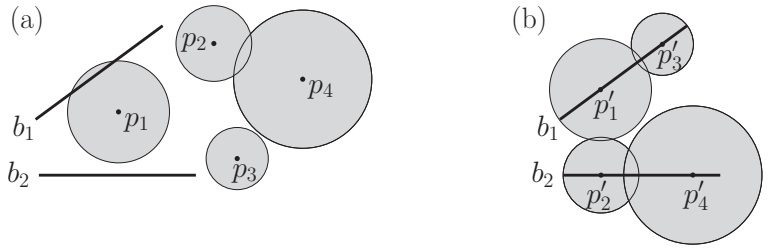


Figure 3.1: (a) A given barrier coverage problem (b) a possible covering assignment

We are interested in the algorithmic complexity of three problems:

Feasibility problem: Given an arrangement $\mathcal{A} = (S, \mathcal{B})$ with sensors in S located in the plane at $p_1, p_2, \ldots, p_n$, determine if there exists a valid covering assignment, i.e. determine whether there exist final sensor positions $p_1', p_2', \ldots, p_n'$ on the barriers such that all barriers in $\mathcal{B}$ are covered.

MinMax problem: Given an arrangement $\mathcal{A} = (S, \mathcal{B})$ with sensors in S located in the plane at $p_1, p_2, \ldots, p_n$, find final sensor positions $p'_1, p'_2, \ldots, p'_n$ on the barriers so that all barriers in $\mathcal{B}$ are covered and $\max_{1 \leq i \leq n}\{d(p_i, p'_i)\}$ is minimized.

MinSum problem: Given an arrangement $\mathcal{A} = (S, \mathcal{B})$ with sensors in S located in the plane at $p_1, p_2, \ldots, p_n$, find final sensor positions $p'_1, p'_2, \ldots, p'_n$ on the barriers so that all barriers in $\mathcal{B}$ are covered, and $\sum_{i=1}^{n} d(p_i, p'_i)$ is minimized.

3.2 Our Results

Throughout the chapter, we consider the barrier coverage problem with sensors of arbitrary ranges, initially located at arbitrary locations in the plane. In Section 3.3, we assume that sensors can move to arbitrary positions on any of the barriers. While feasibility is trivial in the case of one barrier, it is straightforward to show that it is NP-complete for even two barriers. The NP-completeness of the MinSum problem for one barrier follows trivially from the result in [CKK$^+$10]. In this chapter, we show that the MinMax problem is strongly NP-complete even for a single barrier. We show that this holds both when the cost measure is Euclidean distance and when it is rectilinear distance.

In light of these hardness results, in the rest of the chapter, we consider a more restricted but natural movement. We assume that once a sensor has been ordered to relocate to a particular barrier, it moves to the *closest point* on a line containing a barrier. We call this *perpendicular movement*. Note that it is possible for a sensor that is not located on the barrier to cover part of the barrier. However, we require final positions of sensors to be on the line containing the barrier. Section 3.4.1 considers the case of one barrier and perpendicular movement, while Section 3.4.2 considers the case of perpendicular movement and multiple *parallel* barriers. We show that all

three of our problems are solvable in polynomial time. Finally, in Section 3.5, we consider the case of perpendicular movement and two barriers perpendicular to each other. We show that even the feasibility problem is strongly NP-complete in this case. The NP-completeness result holds even in the case when the given positions of the sensors have integer values and the sensing ranges of sensors are limited to two different integer sensing ranges. In contrast, we give an $O(n^{1.5})$ algorithm for finding a covering assignment for a natural restriction of the problem that includes the case when all sensors are located in integer positions and the sensing ranges of all sensors are of diameter 1. Furthermore, we give a sufficient condition for the problem where a covering assignment can be calculated in linear time. Finally, we present three approximation algorithms for maximizing the fraction of the covered segments on the barriers. Our results are summarized in Table 3.1 below.

Barriers	Movement	Feasibility	MinMax	MinSum
one	Arbitrary	$O(n)$	NPC	NPC [CKK$^+$10]
two	Arbitrary	NPC	NPC	NPC
one	Perpendicular	$O(n)$	$O(n \log n)$	$O(n^2)$
k parallel	Perpendicular	$O(kn)$	$O(kn^{k+1})$	$O(kn^{k+1})$
2 perpendicular	Perpendicular	NPC	NPC	NPC

Table 3.1: Summary of our results. We assume sensors are ordered with respect to the x-coordinates of leftmost points in their coverage areas.

3.3 Arbitrary Final Positions

In this section, we assume that sensors are allowed to relocate to any final positions on the barrier(s). We consider first the case of a single barrier b. Without loss of generality, we assume that b is located on the x-axis between $(0, 0)$ and $(L, 0)$ for some L. The feasibility of barrier coverage in this case is simply a matter of checking

whether $\Sigma_{i=1}^{n} 2r_i \geq L$. For the MinSum problem, it was shown in [CKK+10] that even if the initial positions of sensors are on the line containing the barrier, the problem is NP-complete; therefore the more general version of the problem studied here is clearly NP-complete. Recently, it was shown in [CGLW12] that if the initial positions of sensors are on the line containing the barrier, the MinMax problem is solvable in polynomial time. We proceed to study the complexity of the MinMax problem when initial positions of sensors can be anywhere on the plane, and the final positions can be anywhere on the barrier. See Figure 3.2 for an example of the initial placement of sensors.

Theorem 3.1. *Consider an arrangement $\mathcal{A} = (S, \{b\})$. Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of sensors of ranges $r_1, r_2, \ldots, r_n$ initially located in the plane at positions $p_1, p_2, \ldots, p_n$. Let the barrier b be a line segment between $(0, 0)$ and $(L, 0)$. Given an integer k, the problem of determining if there is a covering assignment such that the maximum relocation distance (Euclidean/rectilinear) of the sensors is at most k is strongly NP-complete.*

Proof. The problem is trivially in NP; we give here a reduction from the 3-partition problem [GJ90]. We are given a multiset $A = \{a_1 \geq a_2 \geq \cdots \geq a_{3m}\}$ of $3m$ positive integers such that $B/4 < a_i < B/2$ for $1 \leq i \leq 3m$ and $\sum_{i=1}^{3m} a_i = mB$ for some B. The problem is to decide whether A can be partitioned into m triples $T_1, T_2, \ldots, T_m$ such that the sum of the numbers in each triple is equal to B. We create an instance of the barrier coverage problem as follows: Let $L = mB+m-1$, the barrier b be a line segment from $(0, 0)$ to $(L, 0)$, and let $k = L$. Place a sensor s_i of range $a_i/2$ at $-a_i/2$ where $1 \leq i \leq 3m$. In addition, place $m-1$ sensors $s_{3m+1}, s_{3m+2}, \ldots, s_{4m-1}$ of range $1/2$ at positions $(B+1/2, k), (2B+3/2, k), (3B+5/2, k), \ldots, ((m-1)B+(2m-3)/2, k)$. See Figure 3.2 for an example. Since $L = \sum_{i=1}^{4m-1} 2r_i$, all sensors must move to the barrier in any covering assignment. Observe that the distance from any of the $m-1$

sensors located above the barrier to the barrier is k, and when all of them move this distance, there are gaps of length B between these sensors on the barrier.

If there is a partition of S into m triples $T_1, T_2, \ldots, T_m$, the sum of each triple being B, then there is a solution to the movement of the sensors such that the three sensors corresponding to triple T_i are moved to fill the ith gap in the barrier b. The maximal move of the three sensors corresponding to T_i into ith gap is at most L, and the maximum of the moves of all sensors is k in this case. If such a partition does not exist, then any covering assignment for the barrier b corresponds to moving at least one of the sensors above the x-axis by $k+1$ (rectilinear distance), and by $\sqrt{k^2+1} > k$ (Euclidean distance).

It remains to show that the transformation from the 3-partition problem to the sensor movement problem is polynomial. Since 3-partition is strongly NP-complete [GJ90], we may assume that the values $a_1, a_2, \ldots, a_{3m}$ are bounded by a polynomial $c(3m)^j$ for some constants c and j. The 3-partition problem can be represented using $O(m \log m)$ bits. Therefore, $B \leq c_1 m^j$ and $k \leq c_2 m^{j+1}$ for some constants c_1 and c_2. Our reduction uses $n = 4m - 1$ sensors. In the corresponding barrier coverage problem we need $O(n \log n)$ bits for the positions and sizes of sensors $s_1, s_2, \ldots, s_n$ and we need $O(n \log n)$ bits to represent the position and size of each sensor of size 1. Thus we need $O(n \log n)$ bits to represent the corresponding barrier coverage problem, which shows that the transformation is polynomial.

By adding one additional sensor at distance $> k$ above the barrier, we can create an instance of the problem where $\sum_{i=1}^{4m-1} 2r_i > L$, and the proof remains exactly the same as that sensor cannot be involved in a covering assignment that has maximum relocation distance k.

Finally, it is straightforward to see that any given covering assignment can be verified in polynomial time, completing the proof of strong NP-completeness. □

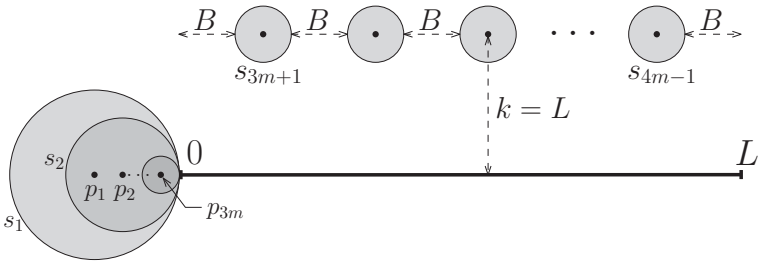


Figure 3.2: Reduction from 3-partition to the MinMax problem

It is easy to see that when there are two barriers to be covered, even feasibility of coverage is NP-complete. This can be shown by reducing the Partition problem [GJ90] to an appropriate 2-barrier coverage problem, as in [CKK+09]. It follows that k-barrier coverage is also NP-complete.

3.4 Perpendicular Movement: Parallel Barriers

In this section, we assume that all sensors use only perpendicular movement to a barrier. Furthermore, in the case of several barriers, the barriers are *parallel* to each other. Figure 3.3 illustrates an example of such a problem. Without loss of generality, we assume barriers $b_1, b_2, \ldots, b_k$, $k \geq 1$ are parallel to the x-axis. Thus, sensors may only move in a vertical direction. Let the set of n sensors $s_1, s_2, \ldots, s_n$ be initially located at positions $p_1, p_2, \ldots, p_n$ respectively, where $p_i = (x_i, y_i)$. We assume that the sensors are listed in the order of the leftmost x-coordinates they can cover, i.e., $x_1 - r_1 \leq x_2 - r_2 \leq \cdots \leq x_n - r_n$. For simplicity we assume all points of interest (sensor locations, left/right endpoints of sensor ranges and barriers) are distinct.

Since there are k barriers, there are up to k points on barriers with the same x-coordinate. We therefore speak of sensors being *candidates for x-coordinates*: a sensor s in position $p = (x, y)$ with sensing range r is a candidate sensor for x-coordinate x' if $x - r \leq x' < x + r$. Alternatively we say s *potentially covers* the x-coordinate x'. Notice that the sensor s potentially covers a *half-open interval* of x-coordinates;

this definition simplifies our algorithms. Furthermore, for any interval I on a barrier, we call it *k-coverable* iff for every point on I there are at least k distinct candidate sensors in S. We first consider the simpler case of $k = 1$.

3.4.1 One Barrier

Without loss of generality, let the barrier $b = b_1$ be the line segment between $(0, 0)$ and $(L, 0)$. Since the y-coordinate of all points on the barrier are the same, we sometimes represent the barrier or a segment of the barrier by an interval of x-coordinates. For technical reasons, we denote the segment of the barrier between the points $(i, 0)$ and $(j, 0)$ by the *half-open interval* $[i, j)$.

We first show a necessary and sufficient condition on the sensors for the barrier to be covered. We give a dynamic programming formulation for the MinSum problem. We denote the set of sensors $\{s_i, s_{i+1}, \ldots, s_n\}$ by S_i. If the barrier is an empty interval, then the cost is 0. If no sensor is a candidate for the left endpoint of the barrier, or if the sensor set is empty while the barrier is a non-empty interval, then clearly the problem is infeasible and the cost is infinity. If not, observe that the optimal solution to the MinSum problem either involves moving sensor s_1 to the barrier or it doesn't. In the first case, the cost of the optimal solution is the sum of $|y_1|$, the cost of moving the first sensor to the barrier, and the optimal cost of the subproblem of covering the interval $[x_1 + r_1, L)$ with the remaining sensors $S_2 = S - \{s_1\}$. In the second case, the optimal solution is the optimal cost of covering the original interval $[0, L)$ with

S_2. The recursive formulation is given below:

$$cost(S_i, [a, L)) = \begin{cases} 0 & \text{if } L < a \\ \infty & \text{if } x_i - r_i > a \text{ or } (S_i = \emptyset \text{ and } L > a) \\ min \begin{cases} |y_i| + cost(S_{i+1}, [x_i + r_i, L)), \\ cost(S_{i+1}, [a, L)) \end{cases} & \text{otherwise} \end{cases}$$

Observe that a subproblem is always defined by a set S_i and a left endpoint to the barrier which is given by the rightmost x-coordinate covered by a sensor. Thus the number of possible subproblems is $O(n^2)$, and it takes constant time to compute $cost(S_i, [a, L))$ given the solutions to the sub-problems. Using either a tabular method or memoization, the problem can be solved in quadratic time. The same dynamic programming formulation works for minimizing the maximum movement, except that in the case when the i-th sensor moves to the barrier in the optimal solution, the cost is the maximum of $|y_i|$ and $cost(S_{i+1}, [x_i + r_i, L))$ instead of their sum. A better approach is to check the feasibility of covering the barrier with the subset of sensors at distance at most d from the barrier in $O(n)$ time, and find the minimum value of d using binary search on the set of distances of all sensors to the barrier. This yields an $O(n \log n)$ algorithm for MinMax. This proves the following theorem:

Theorem 3.2. *Let $s_1, s_2, \ldots, s_n$ be n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ respectively, and let b be a barrier between $(0, 0)$ and $(L, 0)$. The MinSum problem using only perpendicular movement can be solved in $O(n^2)$ time, and the MinMax problem can be solved in $O(n \log n)$ time.*

3.4.2 Multiple Parallel Barriers

For simplicity, we explain the case of two barriers; the results generalize to k barriers as stated in Theorem 3.3. Assume without loss of generality that the two barriers

to be covered are b_1 between $(0,0)$ and $(L,0)$ and b_2 between (P,W) and (Q,W), $0 \leq P$. We shall assume that $L \leq Q$, the case $L > Q$ is very similar. We assume that the sensing ranges of sensors are smaller than half the distance W between the two barriers, and thus it is impossible for a sensor to simultaneously cover two barriers. See Figure 3.3 for an example of such a problem.

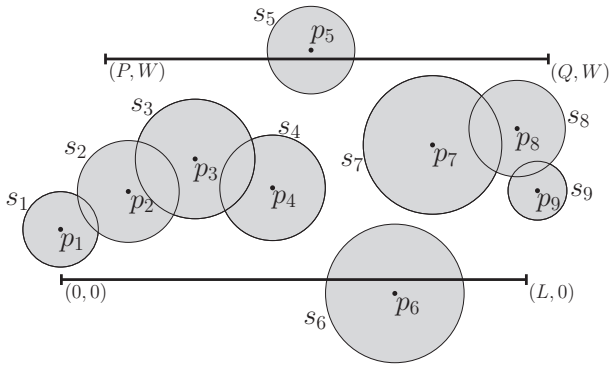


Figure 3.3: An example of a barrier coverage problem with two parallel barriers

Since there are two barriers, there are two points on barriers with the same x-coordinate. We first show a necessary and sufficient condition on the sensors for the two barriers to be covered. Clearly, since the sensing range of every sensor is smaller than half of the distance between the two barriers, the barrier coverage problem for the two parallel barriers b_1 and b_2 above is solvable by a set of sensors S *only if* the interval $[0, P)$ is 1-coverable, $[P, L)$ is 2-coverable, and $[L, Q)$ is 1-coverable by S. We proceed to show that this is also a sufficient condition, and give an $O(n)$ algorithm for finding a covering assignment for two parallel barriers.

Lemma 3.1. *Let $s_1, s_2, \ldots, s_n$ be sensors located at positions $p_1, p_2, \ldots, p_n$ respectively where $p_i = (x_i, y_i)$ and $x_1 - r_1 \leq x_2 - r_2 \leq \cdots x_n - r_n$. Let b_1 between $(0,0)$ and $(L,0)$ and b_2 between (P,W) and (Q,W), where $0 \leq P < L \leq Q$, be two parallel barriers to be covered. If intervals $[0,P)$ and $[L,Q)$ are 1-coverable, and interval $[P,L)$ is 2-coverable, then a covering assignment that uses only perpendicular movement of the sensors can be obtained in $O(n)$ time.*

Proof. We give an algorithm to find such a covering assignment. First we assign sensors to cover b_1 between $(0,0)$ and $(P,0)$ by repeatedly assigning an arbitrary candidate sensor to cover the leftmost uncovered point of this interval. Clearly this is possible, since the interval of x-coordinates $[0,P)$ is 1-coverable. Let s be the last sensor that was used in this assignment, of range r, and initially in position (x,y), so that its final position is $(x,0)$ where $x+r \geq P$.

$x+r \geq L$; Then we have a single barrier left and the interval of x-coordinates $[x+r,Q)$ is 1-coverable, so we can use the algorithm of the previous section.

$P < x+r < L$; Then since $[P,L)$ was initially 2-coverable, and s is the only unavailable sensor among all candidate sensors for this interval, it follows that the interval of x-coordinates $[P,x+r)$ is now 1-coverable and $[x+r,L)$ is 2-coverable. We now have a sub-problem of the same type as the original problem and proceed to solve it recursively.

$x+r = P$; Then there must be two other sensors that are candidates for the x-coordinate P. We arbitrarily pick one of these two candidate sensors and assign it to barrier b_1. It follows that the point (P,W) on barrier b_2 must be 1-coverable, and in fact the initial interval of b_2 is 1-coverable. Once again, the remaining sub-problem can be solved recursively.

Since at every step of the algorithm, one of the sensors is assigned to cover one of the barriers, in increasing order of the values $x_i - r_i$, the algorithm takes $O(n)$ time. □

It is easy to see that the lemma can be generalized for k barriers to show that the feasibility problem can be solved in $O(kn)$ time. We proceed to study the problem of minimizing the sum of movements required to perform barrier coverage.

The dynamic programming formulation given in Section 3.4.1 can be generalized for the case of two barriers. The key difference is that in an optimal solution, sensor s_i may be used to cover a part of barrier b_1 or barrier b_2 or neither. Let $xcost(S_i, [a_1, L), [a_2, Q))$ denote the cost of covering the interval $[a_1, L)$ of the barrier b_1 and the interval $[a_2, Q)$ of the second barrier with the sensor set $S_i = \{s_i, s_{i+1}, \ldots, s_n\}$. The optimal cost is given by the formulation below:

$$
\begin{aligned}
&xcost(S_i, [a_1, L), [a_2, Q)) = \\
&= \begin{cases}
cost(S_i, [a_2, Q)) & \text{if } L < a_1 \\
cost(S_i, [a_1, L)) & \text{if } Q < a_2 \\
\infty & \text{if } x_i - r_i > min\{a_1, a_2\} \text{ or } (S_i = \emptyset \text{ and } Q > min\{a_1, a_2\}) \\
min \begin{cases}
|y_i| + xcost(S_{i+1}, [x_i + r_i, L), [a_2, Q)), \\
|W - y_i| + xcost(S_{i+1}, [a_1, L), [x_i + r_i, Q)), \\
xcost(S_{i+1}, [a_1, L), [a_2, Q))
\end{cases} & \text{otherwise}
\end{cases}
\end{aligned}
$$

It is not hard to see that the formulation can be generalized to k barriers; a sensor s_i may move to any of the k barriers with the corresponding cost being added to the solution. Observe that a subproblem is now given by a set S_i, and a left endpoint of to each of the barriers to be covered. The total number of subproblems is $O(n^{k+1})$ and the time needed to compute the cost of a problem given the costs of the subproblems is $O(k)$. Thus, the time needed to solve the problem is $O(kn^{k+1})$. Clearly a very similar formulation as above can be used to solve the MinMax problem in $O(kn^{k+1})$ time as well. We have proved the following theorem.

Theorem 3.3. *Let $s_1, s_2, \ldots, s_n$ be n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ respectively where $p_i = (x_i, y_i)$ and $x_1 - r_1 \leq x_2 - r_2 \leq \cdots x_n - r_n$. Both the MinSum problem and the MinMax problem for k parallel barriers using only perpendicular movement can be solved in $O(kn^{k+1})$ time.*

The approach used to solve MinMax for a single barrier can be generalized for two barriers, as shown in the theorem below, getting a better time complexity.

Theorem 3.4. *Let $s_1, s_2, \ldots, s_n$ be n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ respectively, and let b_1 between $(0,0)$ and $(L,0)$ and b_2 between (P,W) and (Q,W) be the two parallel barriers to be covered. The MinMax problem for the two parallel barriers using only perpendicular movement can be solved in $O(n \log n)$ time.*

Proof. We first show that given a maximum distance d, we can decide in linear time whether a covering assignment exists so that every sensor relocates at most distance d to its final position. If $d < W/2$, the sets of candidate sensors for each of the two barriers are disjoint. We can verify independently the feasibility of covering each barrier with its set of candidate sensors, as shown in Lemma 3.1.

If $d \geq W/2$, we partition sensors in $\mathcal{S}$ that can cover any part of the barriers into the sets A, B, and C where A consists of sensors that are only candidates for barrier b_1 (that is, they are at distance $> d$ from barrier b_2), B consists of sensors that are only candidates for barrier b_2, and C consists of candidates for both barriers. We assign all sensors in set A to barrier B_1 and all sensors in set B to barrier B_2. This now leaves a set of uncovered intervals on each barrier. If there is a point x that is uncovered on either barrier and has no candidate sensors, then barrier coverage is impossible. If there is a point x that is only uncovered on one barrier and has a candidate sensor, then we assign the candidate sensor to the barrier. After this process is completed, we have a set of intervals that are 2-coverable. We now appeal to Lemma 3.1 to complete the proof.

The optimal value of d can be found using binary search on the set of distances of all sensors from each of the two barriers, and the algorithm in Lemma 3.1 takes $O(n)$ time. □

Note that our algorithms for k parallel barriers can be extended for k-barrier coverage of a single barrier that requires every point on the barrier to be covered by at least k distinct sensors.

3.5 Perpendicular Movement: Two Perpendicular Barriers

In this section we consider the problem of covering two perpendicular barriers. Once again, we assume that sensors can relocate to either of the two barriers, using only perpendicular movement. Figure 3.4 illustrates an example of such a problem. In contrast to the case of parallel barriers, we show here that even the feasibility problem in this case is NP-complete. For simplicity we assume that b_1 is a segment on the x-axis between $(0,0),(L_1,0)$ and b_2 is a segment on the y-axis between $(0,0),(0,L_2)$. Since the sensors can only employ perpendicular movement, the only possible final positions on the barriers for a sensor s_i in position $p_i = (x_i, y_i)$ are $p_i' = (0, y_i)$ or $p_i' = (x_i, 0)$.

We first show that the feasibility problem for this case is NP-complete by giving a reduction from the monotone 3-SAT problem [Gol87]. Recall that a Boolean 3-CNF formula $f = c_1 \wedge c_2 \wedge ... \wedge c_m$ of m clauses is called *monotone* if and only if every clause c_i in f either contains only unnegated literals or only negated literals. In order to obtain a reduction to a barrier coverage problem with two perpendicular barriers, we first put a monotone 3-SAT formula in a special form as given in the lemma below.

Lemma 3.2. *Let $f = f_1 \wedge f_2$ be a monotone 3-CNF Boolean formula with n clauses where f_1 and f_2 only contain unnegated and negated literals respectively, and every literal appears in at most m clauses. Then f can be transformed into a monotone formula $f' = f_1' \wedge f_2'$ such that f_1' and f_2' have only unnegated and negated literals*

respectively, and f' has the following properties:

1. *f and f' are equisatisfiable, i.e. f' is satisfiable if and only if f is satisfiable.*
2. *All clauses are of size two or three.*
3. *Clauses of size two contain exactly one variable from f and one new variable.*
4. *Clauses of size three contain only new variables.*
5. *Each new literal appears exactly once: either in a clause of size two or in a clause of size three.*
6. *Each variable x_i of f appears exactly in m clauses of f'_1, and exactly in m clauses of f'_2.*
7. *f' contains at most $3mn$ clauses.*
8. *The clauses in f'_1 (respectively f'_2) can be ordered so that all clauses containing the literal x_i ($\overline{x_i}$) appear before clauses containing the literal x_j (respectively $\overline{x_j}$) for $i < j$, and all clauses of size three are placed last.*

Proof. Let $f = f_1 \wedge f_2$ be a monotone 3-CNF Boolean formula, where f_1 only contains unnegated literals and f_2 only contains negated literals. Assume the clauses are numbered from 1 to n, and let m be the maximum number of occurrences of any literal in f. For each unnegated literal x_p that appears in the clause numbered i, we introduce a new variable $x_{p,i}$; suppose there are k such variables where $1 \leq k \leq m$. If $k < m$, we also introduce $m - k$ new variables $y_{p,1}, y_{p,2}, \ldots, y_{p,m-k}$. Similarly, for each negated literal $\overline{x_p}$ that appears in the clause numbered j in f_1, we introduce a new variable $x_{p,j}$; suppose there are k such variables where $1 \leq k \leq m$. If $k < m$, we also introduce $m - k$ new variables $z_{p,1}, z_{p,2}, \ldots, z_{p,m-k}$.

For each clause $c_i \in f_1$ of the form $(x_p \vee x_q \vee x_r)$, we put the collection of clauses $(x_p \vee x_{p,i}), (x_q \vee x_{q,i}), (x_r \vee x_{r,i})$ into f'_1 and the clause $(\overline{x_{p,i}} \vee \overline{x_{q,i}} \vee \overline{x_{r,i}})$ into f'_2. Similarly

for each clause $c_j \in f_2$ of the form $(\overline{x_p} \vee \overline{x_q} \vee \overline{x_r})$, we put the collection of clauses $(\overline{x_p} \vee \overline{x_{p,j}}), (\overline{x_q} \vee \overline{x_{q,j}}), (\overline{x_r} \vee \overline{x_{r,j}})$ into f_2' and the clause $(x_{p,j} \vee x_{q,j} \vee x_{r,j})$ into f_1'.

For every literal $x_p \in f_1$ that occurs $k < m$ times in f_1, we add clauses $(x_p \vee y_{p,1}) \wedge (x_p \vee y_{p,2}) \wedge \cdots (x_p \vee y_{p,m-k})$. Similarly, for every literal $\overline{x_p}$ that occurs $k < m$ times in f_2, we add clauses $(\overline{x_q} \vee \overline{z_{q,1}}) \wedge \cdots \wedge (\overline{x_q} \vee \overline{z_{q,m-k}})$. Finally, let $f' = f_1' \wedge f_2'$. From the construction of f' it is easy to verify that it has Properties 2 to 7 stated in the lemma. Property 8 follows from Property 2, 3, and 4.

Now we show that f and f' are equisatisfiable. First assume f is satisfiable, and let A be a satisfying assignment for f. We show how to obtain a satisfying assignment A' for f'. For every variable x_p in f, A' uses

(a) the same truth assignment for x_p as in A,

(b) the opposite truth value for all new variables $x_{p,i}$,

(c) the truth value **true** for every new variable of the type $y_{p,i}$, and

(d) the truth value **false** for every new variable of the type $z_{p,i}$.

To see that A' satisfies f', observe that all clauses of size two in f_1' are of the form $(x_p \vee x_{p,i})$ or $(x_p \vee y_{p,i})$ and are clearly satisfied. The only clauses of size three in f_1' are of type $(x_{p,i} \vee x_{q,i} \vee x_{r,i})$ and correspond to a clause $c_i = (\overline{x_p} \vee \overline{x_q} \vee \overline{x_r})$ in f_2. Since A satisfies c_i, one of x_p, x_q, x_r must be false. But then one of $x_{p,i}, x_{q,i}, x_{r,i}$ must be true in A', and hence the clause $(x_{p,i} \vee x_{q,i} \vee x_{r,i})$ is satisfied. A similar argument can be made about the clauses in f_2'.

Next assume that f' is satisfiable, and let A' be a satisfying assignment for f'. We claim that taking the assignment for the original variables x_p in A' will also satisfy f. To see this, consider the clause $c_i = (x_p \vee x_q \vee x_r)$ in f_1. In f_2' there is a corresponding clause $(\overline{x_{p,i}} \vee \overline{x_{q,i}} \vee \overline{x_{r,i}})$. Since A' satisfies this clause, at least one of $x_{p,i}, x_{q,i}, x_{r,i}$ must be false. Suppose $x_{p,i}$ is false. To satisfy the clause $(x_p \vee x_{p,i})$ in f_1', the truth value of

x_p in A' must be true. Thus the clause $c_i = (x_p \vee x_q \vee x_r)$ is satisfied in f_1. A similar argument can be made about the clauses in f_2. □

We give an example that illustrates the reduction and the ordering specified in Property 8.

Example 3.1. *Consider 3-CNF formula*

$$f = (x_1 \vee x_3 \vee x_4) \wedge (x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_4) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4)$$

An equisatisfiable formula f' satisfying the properties of Lemma 3.2 is:

$$\begin{aligned}
f' \quad &= (x_1 \vee x_{1,1}) \wedge (x_1 \vee x_{1,3}) \wedge (x_1 \vee y_{1,1}) \wedge (x_2 \vee x_{2,2}) \wedge (x_2 \vee x_{2,3}) \\
\wedge \quad &(x_2 \vee y_{2,1}) \wedge (x_3 \vee x_{3,1}) \wedge (x_3 \vee x_{3,2}) \wedge (x_3 \vee x_{3,3}) \wedge (x_4 \vee x_{4,1}) \\
\wedge \quad &(x_4 \vee x_{4,2}) \wedge (x_4 \vee y_{4,1}) \wedge (x_{1,4} \vee x_{2,4} \vee x_{4,4}) \wedge (x_{2,5} \vee x_{3,5} \vee x_{4,5}) \\
\wedge \quad &(\overline{x_1} \vee \overline{x_{1,4}}) \wedge (\overline{x_1} \vee \overline{z_{1,1}}) \wedge (\overline{x_1} \vee \overline{z_{1,2}}) \wedge (\overline{x_2} \vee \overline{x_{2,4}}) \wedge (\overline{x_2} \vee \overline{x_{2,5}}) \wedge (\overline{x_2} \vee \overline{z_{2,1}}) \\
\wedge \quad &(\overline{x_3} \vee \overline{x_{3,5}}) \wedge (\overline{x_3} \vee \overline{z_{3,1}}) \wedge (\overline{x_3} \vee \overline{z_{3,2}}) \wedge (\overline{x_4} \vee \overline{x_{4,4}}) \wedge (\overline{x_4} \vee \overline{x_{4,5}}) \wedge (\overline{x_4} \vee \overline{z_{4,1}}) \\
\wedge \quad &(\overline{x_{1,1}} \vee \overline{x_{3,1}} \vee \overline{x_{4,1}}) \wedge (\overline{x_{2,2}} \vee \overline{x_{3,2}} \vee \overline{x_{4,2}}) \wedge (\overline{x_{1,3}} \vee \overline{x_{2,3}} \vee \overline{x_{3,3}})
\end{aligned}$$

Lemma 3.3. *Let $s_1, s_2, \ldots, s_n$ be n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ respectively, and let b_1 between $(0,0)$ and $(L_1,0)$ and b_2 between $(0,0)$ and $(0,L_2)$ be the two perpendicular barriers to be covered. Then the problem of finding a covering assignment using perpendicular movement for the two barriers is strongly NP-complete.*

Proof. It is easy to see that any given covering assignment can be verified in polynomial time. Given a monotone 3-SAT formula f, we use the construction described in Lemma 3.2 to obtain a formula $f' = f'_1 \wedge f'_2$ satisfying the properties stated in Lemma 3.2 with clauses ordered as described in Property 8. Let f_1 have i_1 clauses,

and f_2 have i_2 clauses, and assume the clauses in each are numbered from $1, \ldots, i_1$ and $1, \ldots, i_2$ respectively. We create an instance P of the barrier coverage problem with two barriers b_1, the line segment between $(0, 0)$ and $(2i_1, 0)$ and b_2, the line segment between $(0, 0)$, and $(0, 2i_2)$.

For each variable x_i of the original formula f we have a sensor s_i of sensing range m located in position $p_i = ((2i-1)m, (2i-1)m)$, i.e., on the diagonal. See Figure 3.4 for an illustration of the instance of barrier coverage corresponding to the monotone 3-SAT formula from Example 3.1 above. Each of the variables $x_{i,j}, y_{i,j}, z_{i,j}$ is represented by a sensor of sensing range 1, denoted $s_{i,j}$, $s'_{i,j}$, and $s''_{i,j}$ respectively, and is placed in such a manner that the sensors corresponding to variables associated with the same s_i collectively cover the same parts of the two barriers as covered by sensor s_i. Furthermore, sensors corresponding to variables that appear in the same clause of size three cover exactly the same segment of a barrier. A sensor corresponding to a new variable $x_{i,j}$ that occurs in the pth clause in f'_1 and in the qth clause in f'_2 is placed in position $(2p-1, 2q-1)$. For example the sensor $s_{1,3}$ corresponding to the variable $x_{1,3}$ appears in the second clause of f'_1 and the fifteenth clause of f'_2, and hence is placed at position $(3, 29)$. Similarly, the sensor $s_{2,4}$ corresponding to the variable $x_{2,4}$ appears in the thirteenth clause of f'_1 and the fourth clause of f'_2, and hence is placed at position $(25, 7)$. A sensor corresponding to variable $y_{i,j}$ which occurs in the ℓth clause in f'_1 is placed in position $(2\ell-1, -1)$ and sensor corresponding to variable $z_{i,j}$ which occurs in the ℓth clause of f'_2 is placed in position $(-1, 2\ell-1)$. It is easy to see that the reduction is polynomial, and the sensor sizes and border length are linear in the length of the input to the barrier coverage problem.

Observe that in this assignment of positions to sensors, for any i, there is a one-to-one correspondence between the line segments of length 2 in b_1 and b_2 and clauses in f'_1 and f'_2 respectively. In particular, the sensors that potentially cover the line

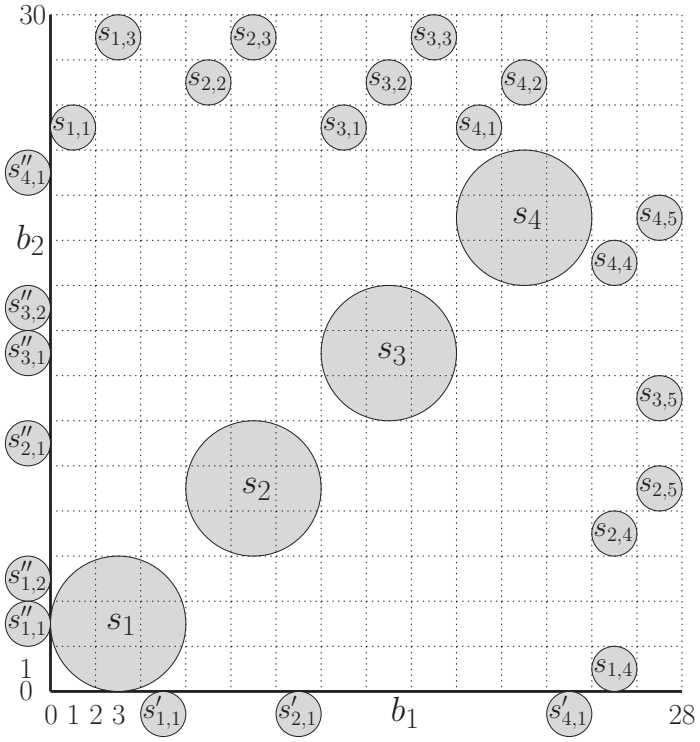


Figure 3.4: Barrier coverage instance corresponding to Example 3.1

segment from $(2i-2, 0)$ to $(2i, 0)$ on the barrier b_1 correspond to variables in clause i of f'_1. Similarly, the sensors that potentially cover the line segment from $(0, 2i-2)$ to $(0, 2i)$ on the barrier b_2 correspond to variables in clause i of f'_2. Thus, by associating the vertical move of a sensor with an assignment of **true** to the corresponding variable of f', and the horizontal move of a sensor with an assignment of **false** to the corresponding variable of f', f' is satisfiable if and only if for the corresponding instance P there exists a covering assignment assuming perpendicular movement. □

Since any instance of monotone 3-SAT problem can be transformed into an instance of monotone SAT problem in which no literal occurs more than three times, it follows from the proof that the problem is NP-complete even when the sensors are in integer positions and the ranges of sensors are limited to two different sizes 1 and $m \geq 3$. It is also clear from the proof that the perpendicularity of the barriers is not critical. The key issue is that the order of intervals covered by the sensors in

one barrier has no relationship to those covered in the other barrier. In the case of parallel barriers, this property does not hold. The exact characterization of barriers for which a polytime algorithm is possible remains an open question.

In the following we extend the NP-completeness results further to the case where sensors have the same sensing ranges. To achieve this, we use a technique that we call *binding* of sensors:

Definition 3.1. Given an arrangement $A = (S, B)$ and $S' \subseteq S$, sensors in S' are called *bound* together iff in any covering assignment of A, all sensors in S' move in the same direction (if any of them moves at all).

Lemma 3.4. *Consider the partial arrangement shown in Figure 3.5 and assume that barrier segments* $b_{1,1}, b_{1,2}, \ldots, b_{1,6}$ *are potentially covered only by* $\{s_1, t_2\}$, $\{s_2, t_1\}$, $\{s_3, t_1\}$, $\{s_3, t_4\}$, $\{s_4, t_4\}$, *and* $\{s_5, t_3\}$ *respectively. Also assume that* $b_{2,1}, b_{2,2}, \ldots, b_{2,6}$ *are potentially covered only by* $\{s_1, t_1\}$, $\{s_2, t_2\}$, $\{s_3, t_2\}$, $\{s_3, t_3\}$, $\{s_4, t_3\}$, *and* $\{s_5, t_4\}$ *respectively. Then sensors* $s_1, s_2, \ldots, s_5$ *are bound together.*

Proof. Consider a covering assignment c of the arrangement. First we consider the case where sensor s_1 moves down according to c to the bottom border. Since c is a covering assignment, $b_{2,1}$ must be covered by some sensor(s) in S moved according to c. However s_1 and t_1 are the only sensors that potentially cover $b_{2,1}$, and therefore according to c, sensor t_1 must move left. Similarly, since $b_{1,2}$ and $b_{1,3}$ are potentially covered only by $\{s_2, t_1\}$ and $\{s_3, t_1\}$ respectively and t_1 moves left, sensors s_2 and s_3 must move down in c to cover $b_{1,2}$ and $b_{1,3}$ respectively. Similarly it can be shown that sensors t_3 must move left to cover $b_{2,4}$, s_5 must move down to cover $b_{1,6}$, also t_4 must move left to cover $b_{2,6}$, and finally s_4 has to move down to cover $b_{1,5}$.

Similarly it can be shown that in any covering assignment, if s_1 moves left then all $s_2, ..., s_5$ need to move left as well. With similar arguments it can be shown that

if any sensor in $\{s_2, s_3, s_4, s_5\}$ moves then all sensors in $\{s_1, s_2, \ldots, s_5\}$ also move in the same direction and therefor the sensors are bound together. □

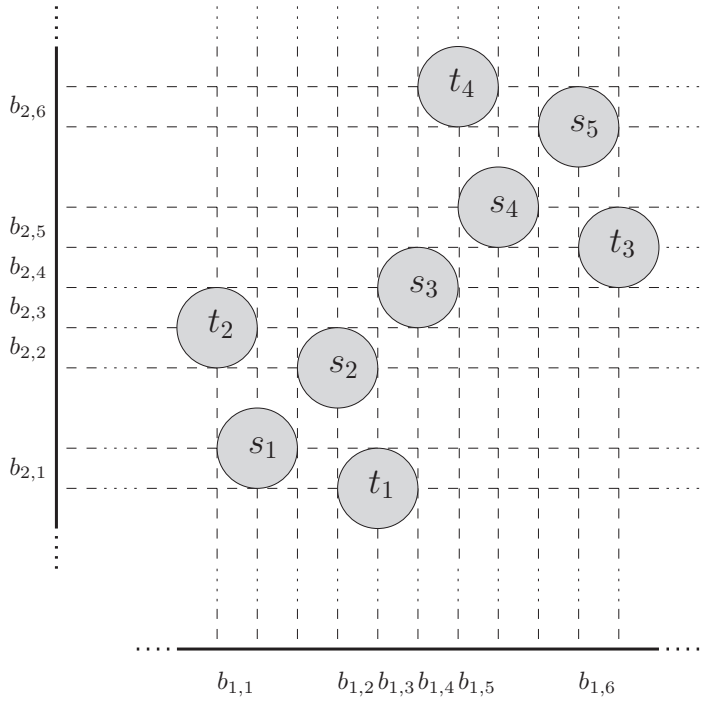


Figure 3.5: Bound sensors: In any covering assignment, sensors $s_1, s_2, \ldots, s_5$ always move in the same direction.

We use this binding gadget in the proof of the following theorem:

Theorem 3.5. *Let $s_1, s_2, \ldots, s_n$ be n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ respectively, and let b_1 between $(0, 0)$ and $(L_1, 0)$ and b_2 between $(0, 0)$ and $(0, L_2)$ be the two perpendicular barriers to be covered. Then the problem of finding a covering assignment using perpendicular movement for the two barriers is strongly NP-complete even if all sensors have unit sensing radius and are located at integer positions.*

Proof. In Lemma 3.3 we showed that the problem is NP-complete when sensors are allowed to have different sensing ranges. As mentioned before the problem is NP-complete even if sensors are limited to two sizes 1 and 3. Using the binding technique

presented in Lemma 3.4, we reduce the problem to one where all sensors have unit sensing ranges.

Note that in the construction of the clauses in the proof of Lemma 3.3, clauses are ordered with respect to the old variables and the clauses that contain only new variables (clauses of size three) are placed last. Therefore the area of the arrangement can be divided into five disjoint parts:

Area 1 : A rectangular area at the left bottom, that only contains sensors of size 3 located on the diagonal of the area.

Area 2 : The rectangular area on the top of Area 1 that only contains sensors of unit sensing range.

Area 3 : The rectangular area on the right of Area 1 that only contains sensors of unit sensing range.

Area 4 : The rectangular area on the bottom of Area 1 that only contains sensors of unit sensing range.

Area 5 : The rectangular area on the left of Area 1 that only contains sensors of unit sensing range.

Let $\mathcal{A}$ be an arrangement of the problem described above. In the following, we build an arrangement $\mathcal{A}'$, such that:

1. Every sensor in $\mathcal{A}'$ has unit sensing radius.

2. There exists a covering assignment for $\mathcal{A}$ iff there exists a covering assignment for $\mathcal{A}'$.

Let n' be the number of sensors in Area 1 of $\mathcal{A}$ and i_1, i_2 denote the number of clauses of size 3 in f'_1 and f'_2 respectively. Let $\mathcal{B}' = \{b'_1, b'_2\}$ be the set of barriers in

$\mathcal{A}'$ with b'_1 being the line segment between $(0,0)$ and $(20n'+2i_1,0)$ and b'_2 being the line segment between $(0,0)$, and $(0,20n'+2i_2)$. We put the sensors of unit sensing radius in $\mathcal{A}'$ as follows:

For every sensor s_i in Area 1 of $\mathcal{A}$ with sensing radius 3, and using the binding technique explained in Lemma 3.4, we place a set of sensors of unit sensing radius in $\mathcal{A}'$ that together replicate the behavior of s_i. More precisely, for every sensor s_i in Area 1 of $\mathcal{A}$ located at (x_i, y_i) we put 18 sensors in $\mathcal{A}'$:

- 4 sensors $e_{i,1}, e_{i,2}, e_{i,3}, e_{i,4}$ located at $(20i-19, 20i-19), (20i-19, 20i-1), (20i-1, 20i-19), (20i-1, 20i-1)$ respectively.

- 6 sensors $t_{i,1}, t_{i,2}, t_{i,3}, t_{i,4}, t_{i,5}, t_{i,6}$ located at $(20i-14, 20i-18), (20i-18, 20i-14), (20i-8, 20i-12), (20i-12, 20i-8), (20i-2, 20i-6), (20i-6, 20i-2)$ respectively.

- 8 sensors $v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4}, v_{i,5}, v_{i,6}, v_{i,7}, v_{i,8}$ located at $(20i-17, 20i-17), (20i-15, 20i-15), \ldots, (20i-3, 20i-3)$ respectively.

For every sensor $s_{i,j}$ in Area 2 of $\mathcal{A}$ that corresponds to a variable that occurs in the pth clause in f'_1 and qth clause of f'_2, we put a sensor $s_{i,j}$ in $\mathcal{A}'$ at $(2\lfloor\frac{p-1}{3}\rfloor+6p-2, 14n'+2q-1)$.

For every sensor $s_{i,j}$ in Area 3 of $\mathcal{A}$ that corresponds to a variable that occurs in the pth clause in f'_1 and qth clause of f'_2, we put a sensor $s_{i,j}$ in $\mathcal{A}'$ at $(14n'+2p-1, 2\lfloor\frac{q-1}{3}\rfloor+6q-2)$.

For every sensor $s'_{i,j}$ in Area 4 of $\mathcal{A}$ that corresponds to a variable that occurs in the pth clause in f'_1, we put a sensor $s'_{i,j}$ in $\mathcal{A}'$ at $(2\lfloor\frac{p-1}{3}\rfloor+6p-2, -1)$.

Finally, for every sensor $s''_{i,j}$ in Area 5 of $\mathcal{A}$ that corresponds to a variable that occurs in the qth clause of f'_2, we put a sensor $s''_{i,j}$ in $\mathcal{A}'$ at $(-1, 2\lfloor\frac{q-1}{3}\rfloor+6q-2)$.

See Figure 3.6 for a partial example that illustrates the positions of sensors in $\mathcal{A}'$

corresponding to sensors $s_1, s_{1,1}, s_{1,3}, s_{1,4}, s'_{1,1}, s''_{1,1}, s''_{1,2}$ in $\mathcal{A}$. The binding technique guarantees that all sensors $v_{i,1}$, $v_{i,2}, \ldots, v_{i,8}$ move in the same direction in a covering assignment.

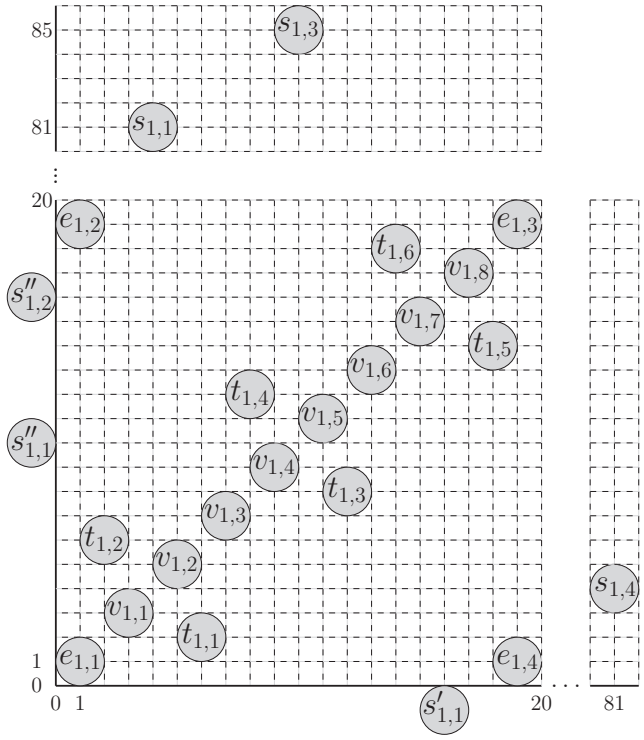


Figure 3.6: Partial example: Barrier coverage instance corresponding to Example 3.1

Next we show that there exists a covering assignment for $\mathcal{A}'$ iff there is a covering assignment for $\mathcal{A}$. First we assume that there exists a covering assignment c for $\mathcal{A}$ and show that a covering assignment for $\mathcal{A}'$ exists. We move the sensors in $\mathcal{A}'$ such that both borders b'_1 and b'_2 are covered as follows: For every sensor in $\mathcal{A}'$ if the sensor corresponds to a sensor in the Areas 2, 3, 4, or 5 of $\mathcal{A}$ we move the sensor as stated for its corresponding sensor in c. For every sensor $v_{i,j}$, we move it in the same direction as s_i in c. For sensors $t_{i,j}$ we move them in the opposite direction of s_i in c. We move all $e_{i,1}$ and $e_{i,3}$ to the bottom and all $e_{i,2}$ and $e_{i,4}$ to the left. It is easy to verify that every point on b'_1 and b'_2 is covered by some sensor in $\mathcal{A}'$ and therefore this is a covering assignment for $\mathcal{A}'$.

Second we show that if there exists a covering assignment c' for $\mathcal{A}'$ then there exists a covering assignment for $\mathcal{A}$: Since for every i with $1 \leq i \leq n'$ all sensors $v_{i,1}, v_{i,2}, \ldots, v_{i,8}$ are bound together they must move in the same direction in c'. We move the sensor s_i in $\mathcal{A}$ in the same direction as sensors $v_{i,j}$. For every other sensor in $\mathcal{A}$ we move them in the same direction as in c' for its corresponding sensor in $\mathcal{A}'$. Again it is easy to verify that this assignment is a covering assignment for $\mathcal{A}$.

It is clear from the construction that this reduction takes polynomial time and the theorem follows. □

3.5.1 Special Cases

We now turn our attention to restricted versions of barrier coverage of two perpendicular barriers for which we have polytime algorithms.

We call an arrangement $\mathcal{A} = (S, \{b_1, b_2\})$, a *non-overlapping* arrangement if for any two sensors $s_i, s_j \in S$, the intervals that are potentially covered by s_1 and s_2 on the barrier b_1 (and b_2) are either the same or disjoint. An example of a non-overlapping arrangement is shown in Figure 3.7. This would be the case, for example, if all sensor

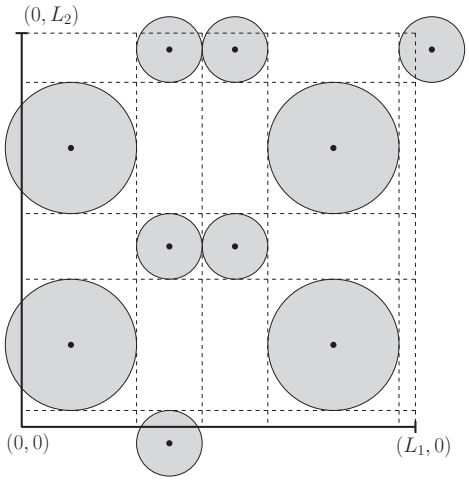


Figure 3.7: A non-overlapping arrangement of sensors. Each interval on the x-axis and y-axis delineated by dotted lines is represented by a sensor in the corresponding bipartite graph.

ranges are of the same diameter equal to 1 and the sensors are in integer positions. We show below that for a non-overlapping arrangement, the problem of finding a covering assignment is polynomial.

Theorem 3.6. *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of n sensors initially located in the plane at positions $p_1, p_2, \ldots, p_n$ and let b_1 and b_2 be two perpendicular barriers to be covered. If $\mathcal{A} = (S, \{b_1, b_2\})$ is a non-overlapping arrangement, then there exists an $O(n^{1.5})$ algorithm that finds a covering assignment for $\mathcal{A}$, using only perpendicular movement or reports that none exists.*

Proof. If there exists a segment of either of the barriers that is not covered by any of the sensors, then clearly there is no covering assignment. Otherwise, the problem of finding a covering assignment in this case can be reduced to the problem of maximum matching in a bipartite graph. Create one node for each sensor and one node for each segment of each barrier that is potentially covered by a sensor. Since $\mathcal{A}$ is a non-overlapping arrangement, the segments are disjoint and together they cover both barriers (see Figure 3.7). We put an edge between a node representing a barrier segment and a node representing a sensor if the sensor can cover the segment. Clearly, the problem of finding a covering assignment is equivalent to finding a matching in which each node representing a segment of the barrier is matched with a node representing a sensor. Since each node representing a sensor has degree two, this can be done in time $O(n^{1.5})$ using the Hopcroft-Karp algorithm. □

In the following we show another sufficient condition for an arrangement to have a covering assignment.

Theorem 3.7. *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of n sensors with same sensing ranges, initially located in the plane at positions $p_1, p_2, \ldots, p_n$ and let b_1 and b_2 be two perpendicular barriers to be covered. If b_1 and b_2 are both 3-coverable, then there exists a covering assignment for $\mathcal{A} = (S, \{b_1, b_2\})$ that can be calculated in $O(n)$ time.*

Proof. The basic idea is to convert $\mathcal{A}$ to a maximal matching problem on a bipartite graph. First, we re-label the sensors based on their x-coordinate. Let $s'_1, s'_2, \ldots, s'_n$ denote the new labels of the sensors. We divide b_1 into $m = \lfloor \frac{n}{2} \rfloor$ disjoint segments $b_{1,0}, b_{1,1}, \ldots, b_{1,m-1}$ as follows:

- $b_{1,0} = [\ell_{-1}, \ell_0)$ where $\ell_{-1} = 0$ and $\ell_0 = x'_1 + r$.
- For every $0 < i < m$, $b_{1,i} = [\ell_{i-1}, \ell_i)$ where $\ell_i = x'_{2i+1} + r$ (if $x'_{2i+1} + r > |b_1|$ the segment ends at $|b_1|$).

From the definition of $b_{1,0}, b_{1,1}, \ldots, b_{1,m-1}$ it can be seen that these segments are disjoint segments starting one after another on b_1. As mentioned in the definition of $b_{1,0}$ it starts from the origin. Also since $2(m-1)+1 = 2m-1 \geq n-2$, it can be seen that $b_{1,m-1}$ continues to at least the end of b_1. Therefore $\bigcup_{0 \leq i < m} b_{1,i} = b_1$.

Take a non-empty interval $b_{1,i}$ with $0 \leq i < m$. Since b_1 is 3-coverable, it is easy to see that ℓ_{i-1} is potentially covered by both s'_{2i+1} and s'_{2i+2}. Also ℓ_i must be potentially covered by s'_{2i+2}. Therefore any point in the interval $b_{1,i}$ is potentially covered by both s'_{2i+1} and s'_{2i+2} (see Figure 3.8)..

Similarly, we relabel sensors with respect to their y-coordinates and partition b_2 into segments $b_{2,0}, b_{2,1}, \ldots, b_{2,m-1}$. With a similar argument it can be shown that these segments are disjoint and $\bigcup_{0 \leq i < m} b_{2,i} = b_2$. Also any point in the interval $b_{2,i}$ with $0 \leq i < m$ is potentially covered by both s''_{2i+1} and s''_{2i+2}.

Similar to the proof of Theorem 3.6, we make a matching problem in a bipartite graph as follows: We create one node for each sensor and one node for each segment $b_{i,j}$ on b_1 or b_2. For every node corresponding to a segment $b_{1,i}$ we add two edges to nodes corresponding to sensors s'_{2i+1} and s'_{2i+2}. Also for every node corresponding to a segment $b_{2,i}$ we add two edges to nodes corresponding to sensors s''_{2i+1} and s''_{2i+2}. Note that according to this construction, the degree of every node in the graph, corresponding to a barrier segment in the arrangement is 2 and the degree of every

node in the graph, corresponding to a sensor in the arrangement is at most 2. A maximum matching that matches every nodes corresponding to barrier segments can be found in linear time and it is easy to see that any such matching corresponds to a covering assignment of $\mathcal{A}$. □

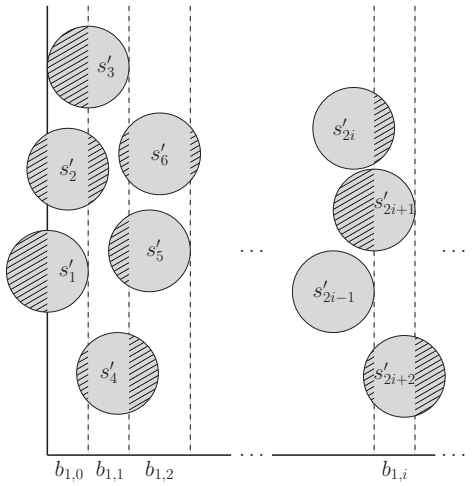


Figure 3.8: Sufficiency of potential 3-coverage.

As shown in Theorem 3.7, if both barriers in an arrangement are 3-coverable then there exists a covering assignment for the arrangement. In contrast, Figure 3.9 illustrates an arrangement where b_1 and b_2 are 2 and 3-coverable respectively, and yet we show that no covering assignment exists for this arrangement: To the contrary assume that c is a covering assignment of the mentioned arrangement. One of the sensors in $\{s_1, s_2, s_3\}$ must move left in c to cover $[0, 2]$ on b_2. Let s_i with $1 \leq i \leq 3$ denote a sensor that moves left. Hence, both $s_{i,1}$ and $s_{i,2}$ must move down to cover $[3i - 3, 3i - 1]$ on b_1. Consequently, sensors s_{i+3} and s_{i+4} must move left to cover $[2i, 2i + 2]$ on b_2. However, this results in $[i + 8, i + 9]$ on b_1 being left uncovered which contradicts the assumption that c is a covering assignment of the arrangement. Therefore there is no covering assignment for the arrangement in Figure 3.9.

The existence of a covering assignment for a general assumption of one border being k-coverable and the other border being k'-coverable, for other values of k and

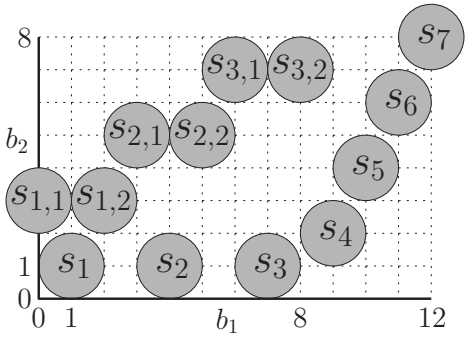


Figure 3.9: An example of an arrangement where b_1 and b_2 are 2 and 3-coverable and yet there exists no covering assignment for the arrangement.

k', is an open problem.

3.5.2 Approximation Algorithms

In the previous section we considered the barrier coverage of two perpendicular barriers with sensors limited to perpendicular movements and we showed that even the feasibility problem is NP-complete. The NP-completeness result implies that the optimization problem of maximizing the sum of lengths of the covered segments on the barriers is NP-hard. In this section we present three approximation algorithms for maximizing the sum of lengths of the covered segments on the barriers. Our first algorithm is a trivial observation which has a 1/2 approximation ratio:

Theorem 3.8. *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of n sensors initially located in the plane and let b_1 and b_2 be two perpendicular barriers to be covered. There exists an $O(n \log n)$ approximation algorithm with $1/2$ approximation ratio.*

Proof. Let c_1 and c_2 denote the sum of the lengths of segments that are 1-coverable on b_1 and b_2 respectively. We simply move all sensors in S down to b_1 if $c_1 \geq c_2$ and we move all sensors left to b_2 otherwise.

First, it is easy to see that c_1 can be calculated by first sorting sensors with respect to their x-coordinates in $O(n \log n)$ and then calculating the summation of

the lengths of segments on b_1 that are covered by sensors in $O(n)$. The value of c_2 can be calculated similarly and therefore the overall running time of the algorithm is $O(n \log n)$.

Second we show that the approximation ratio of the algorithm is 1/2: Let ℓ_1 and ℓ_2 respectively denote the sum of lengths of covered segments on b_1 and b_2 in an optimal assignment. Clearly, $c_1 \geq \ell_1$ and $c_2 \geq \ell_2$ and therefore $\max(c_1, c_2) \geq \frac{\ell_1+\ell_2}{2}$. However, $\max(c_1, c_2)$ is exactly the sum of the lengths of covered segments according to our algorithm, and therefore the approximation ratio is at least 1/2. □

Note that the above algorithm does not assume any special condition for the arrangement. However, assuming that both barriers are 1-coverable and furthermore one of the barriers is k-coverable, we give another algorithm that has an approximation ratio of $\frac{2k-1}{2k}$:

Theorem 3.9. *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of n sensors with sensing range r, initially located in the plane and let b_1 and b_2 be two perpendicular barriers to be covered with $|b_1| = |b_2| = L$. If one of the barriers is k-coverable and the other one is 1-coverable then an assignment of sensors that covers at least $\frac{2k-1}{k}L$ on the two barriers can be calculated in $O(kn + n \log n)$ time.*

Proof. Without loss of generality we assume b_1 and b_2 are k-coverable and 1-coverable respectively and that sensors are ordered with respect to their x-coordinates.

We partition sensors in S into k subsets $S_1, S_2, \ldots, S_k$ defined as follows

$$S_j = \{s_i \in S | i \equiv j \pmod{k}\}$$

Let $\overline{S}_i = S - S_i$ denote the subset of all sensors not in S_i and let c_i denote the sum of lengths of covered segments on b_2 if all sensors in $\overline{S}_i$ move to b_2. Let c_j be the

maximum among all c_i with $1 \leq i \leq k$. We move all sensors in S_j down to b_1 and all sensors in $\overline{S}_j$ left to b_2.

We show that using this algorithm b_1 is fully covered and at least $\frac{k-1}{k}L$ on barrier b_2 is covered. This leads to overall $\frac{2k-1}{k}L$ lengths of barriers being covered.

First we show that b_1 is fully covered if all sensors in S_j are moved down: Let $p \in [0, L]$ denote a point on b_1. Since b_1 is k-coverable, point p has at least k consecutive candidate sensors in the list $s_1, s_2, \ldots, s_n$. Obviously at least one of the sensors that potentially cover p is in S_j. Therefore if we move all sensors in S_j down, b_1 is fully covered.

Second we show that $c_j \geq \frac{k-1}{k}L$. Assume to the contrary that $c_j < \frac{k-1}{k}L$. Let c denote the sum of lengths of segments of b_2 that are covered if all sensors are moved to the left. It is easy to see that every sensor is considered $k-1$ times in $\sum_{0 \leq i < k} c_i$ and therefore $c \leq \frac{1}{k-1}\sum_{0 \leq i < k} c_i$. However since $c_j = \max_{1 \leq i \leq k} c_i$ and $c_j < \frac{k-1}{k}L$, we obtain $c < L$ which contradicts the assumption that b_2 is 1-coverable. Therefore $c_j \geq \frac{k-1}{k}L$.

Finally we show that this algorithm takes $O(kn + n \log n)$ time: To calculate the partitions $S_1, S_2, \ldots, S_k$ we need to sort the sensors according to their x-coordinates takes $O(n \log n)$ time. Calculating each c_i with $1 \leq i \leq k$ can be done by sorting all sensors in S with respect to their y-coordinates in $O(n \log n)$ time, and then calculating the summation of lengths of segments of b_2 that sensors in $\overline{S}_i$ cover in kn time. Finally, finding the maximum among all c_i values with $1 \leq i \leq k$ can be done in $O(k)$ time. Therefore the overall running time of the algorithm is $O(kn + n \log n)$. □

For an example of the algorithm consider the arrangement in Figure 3.10. According to our algorithm, $k = 2$ and $S_1 = \{s_1, s_3, \ldots, s_9\}$ and $S_2 = \{s_2, s_4, \ldots, s_{10}\}$. Theorem 3.9 implies that sensors in each of S_1 or S_2 are enough to cover b_1. Since $c_1 = 17$ and $c_2 = 20$, according to our algorithm sensors in S_2 are moved left and

sensors in S_1 are moved down. By this assignment 44 units are covered in total on b_1 and b_2. However, an optimal algorithm can cover both barriers by moving sensors in $\{s_2, s_3, s_6, s_7, s_9\}$ and $\{s_1, s_4, s_5, s_8, s_{10}\}$ down and left respectively. Therefore the approximation ratio of our algorithm, on this example is $\frac{44}{48} = \frac{11}{12}$.

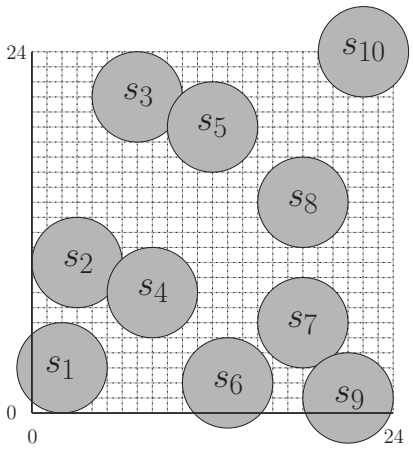


Figure 3.10: An arrangement where b_1 and b_2 are 2 and 1-coverable respectively.

Finally, given that there exists a covering assignment for the input arrangement, our third algorithm has an approximation ratio of at least 3/4.

Theorem 3.10. *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of n sensors with sensing range r, initially located in the plane and let b_1 and b_2 be two perpendicular barriers to be covered with $|b_1| = |b_2| = L$. Given that there exists a covering assignment for the arrangement $\mathcal{A} = (S, \{b_1, b_2\})$, then an assignment of sensors which covers at least $\frac{3}{2}L$ on the two barriers can be calculated in $O(n^2)$ time.*

Proof. Since we assumed that there exists a covering assignment for $\mathcal{A}$, therefore both b_1 and b_2 are 1-coverable. Our algorithm works as follows: In the first phase of the algorithm, for any point p with $p \in b_1$, if p has only one candidate sensor then we move that sensor down to b_1 and remove it from S. We continue this until every uncovered point on b_1 has at least 2 distinct candidate sensors in S. Let $\ell_1, \ell_2, \ldots, \ell_m$ denote the continuous maximal intervals on b_1 that are yet not covered. Consider an interval ℓ_i

and let S_i denote a subset of S such that every sensor in S_i potentially covers some points in ℓ_i. Since every point in ℓ_i with $1 \leq i \leq m$ has at least two candidate sensor in S_i, similar to our previous algorithm, the subset S_i can be partitioned into $S_{i,1}$ and $S_{i,2}$ such that the sensors in each subset can fully cover entire ℓ_i if moved down. In the second phase of the algorithm, we start with segment ℓ_1 and choose among $S_{1,1}$ and $S_{1,2}$, the subset that its sensors cover more length on the uncovered parts of b_2 and move them to the left. We move the other subset down to b_1. We do this for all segments ℓ_i and subsets $S_{i,1}$ and $S_{i,2}$ with $2 \leq i \leq m$ and update the uncovered segments on b_2 at the end of each iteration.

It is easy to see that this algorithm covers b_1 completely. Now we show that if sensors are moved according to the above algorithm at least $\frac{L}{2}$ is covered on b_2 and therefore the overall covered length on barriers is $\frac{3}{2}L$.

Let T denote any subset of S and let $c_{i,j}^{T}$ denote the length of b_2 that can be covered by sensors in $S_{i,j}$ and cannot be covered by sensors in T. Without loss of generality assume that in the second phase of the algorithm, for every iteration i with $1 \leq i \leq m$, sensors in subset $S_{i,1}$ are moved left and those in $S_{i,2}$ are moved down. Therefore:

$$c_{i,1}^{\cup_{1\leq j<i} S_{j,1}} \geq c_{i,2}^{\cup_{1\leq j<i} S_{j,1}}$$

Summing over all intervals l_i with $1 \leq i \leq m$, we get:

$$\sum_{1\leq i\leq m} c_{i,1}^{\cup_{1\leq j<i} S_{j,1}} \geq \sum_{1\leq i\leq m} c_{i,2}^{\cup_{1\leq j<i} S_{j,1}} \tag{3.1}$$

Also note that since we assumed there exists a covering assignment for $\mathcal{A}$, the sensors that are moved to b_1 in the first phase of the algorithm must also move down in any covering assignment of $\mathcal{A}$. Therefore sensors in $\cup_{1\leq i\leq m} S_i$ are enough to cover b_2 and

therefore:

$$\sum_{1\leq i\leq m} c_{i,1}^{\cup_{1\leq j<i}S_{j,1}} + \sum_{1\leq i\leq m} c_{i,2}^{\left(\cup_{1\leq j\leq m}S_{j,1}\right)\cup\left(\cup_{1\leq j<i}S_{j,2}\right)} = L$$

However it is easy to see that:

$$\sum_{1\leq i\leq m} c_{i,2}^{\cup_{1\leq j\leq m}S_{j,1}\cup_{1\leq j<i}S_{j,2}} \leq \sum_{1\leq i\leq m} c_{i,2}^{\cup_{1\leq j<i}S_{j,1}}$$

and therefore:

$$\sum_{1\leq i\leq m} c_{i,1}^{\cup_{1\leq j<i}S_{j,1}} + \sum_{1\leq i\leq m} c_{i,2}^{\cup_{1\leq j<i}S_{j,1}} \geq L$$

Combining this with Equation 3.1 we get:

$$\sum_{1\leq i\leq m} c_{i,1}^{\cup_{1\leq j<i}S_{j,1}} \geq \frac{L}{2}$$

This proves the lower bound on the length that is covered by sensors if they are moved according to the algorithm above.

Finally we show that the running time of the algorithm is $O(n^2)$. We first sort the sensors according to their x-coordinates. This takes $O(n \log n)$ time. Then in the first phase of the algorithm, for every sensor s_i with $1 \leq i \leq n$ we determine whether there is a point p on b_1 such that s_i is its only candidate sensor and if so we move s_i down. This phase can be done in $O(n)$. Partitioning of sensors into $S_{i,1}$ and $S_{i,2}$ with $1 \leq i \leq m$ can be done similar to the technique explained in the proof of Theorem 3.10 that takes $O(n)$ time. Finally, for calculating values of $c_{i,j}$, for every sensor in S since it only belongs to one $c_{i,j}$ we only calculate the portion it covers on b_2 once which takes $O(n)$. Therefore the overall calculation of all $c_{i,j}^{\cup_{1\leq j<i}S_{i,1}}$ takes $O(n^2)$ time and the total running time of the algorithm is $O(n^2)$. □

3.6 Conclusions

It was previously shown that the MinMax barrier coverage problem when the sensors are initially located on the line containing the barrier is solvable in polynomial time [CGLW12]. In contrast, our results in this chapter show that the same problem becomes strongly NP-complete when sensors of arbitrary ranges are initially located in the plane, and are allowed to move to any final positions on the barrier. It remains open whether this problem is polynomial in the case when there is a fixed number of possible sensor ranges. If sensors are restricted to use perpendicular movement, the feasibility, MinMax, and MinSum problems are all polytime solvable for the case of k parallel barriers. However, when the barriers are not parallel, even the feasibility problem is strongly NP-complete, even when sensor ranges are the same. We presented polynomial time solutions for some special cases and three approximation algorithms for the problem of covering the maximum possible fraction of the barriers. Our approximation algorithms assume that sensors have same sensing range, however they can be extended for the case where sensors have arbitrary sensing ranges.

Chapter 4

Synchronous Distributed Algorithms Using Relocatable Sensors[1]

Our focus in this chapter is on the barrier coverage problem using ad hoc deployment of relocatable sensors. We model a barrier with a line segment. We assume sensors are autonomous with limited visibility range and are initially located on the barrier. We present two distributed algorithms for barrier coverage.

The rest of this chapter is organized as follows: In Section 4.1 we present a summary of our results in this chapter. In Section 4.2 we present our model of the network and barrier, and introduce some terminology and basic facts. In Sections 4.3 and 4.4, we present our algorithms and prove their correctness and running times. Finally in Section 4.5, we briefly review our results and discuss some open problems.

[1]Results of this chapter are also published in [EKK+13]

4.1 Our results

We present two synchronous local distributed algorithms for the barrier coverage problem with identical sensors that have constant visibility range and constant movement per time step (the notion of time step is defined more precisely in Section 4.2). The first algorithm is an oblivious algorithm that achieves barrier coverage in $\Theta(n^2)$ time steps on a line segment barrier when sensors have identical sensing ranges and there are enough sensors to cover the entire barrier. In contrast to the first algorithm, our second algorithm uses two bits of memory for each sensor to store its state. It improves the running time of the first algorithm to $\Theta(n)$ time steps. Note that any algorithm that can move only a constant distance in one step requires $\Omega(n)$ time steps in the worst case, and so our second algorithm is asymptotically optimal. Our algorithms are self-stabilizing: if any sensors were to be removed after coverage has been achieved, the remaining sensors can re-establish coverage if sufficient number of sensors remains. We also study the behavior of our algorithms in cases where there are not enough sensors to cover the entire border.

4.2 Computational Model and Preliminaries

We model the barrier with a line segment of length $L \in Z$ covering the interval $[0, L]$ on the x-axis. A sensor network consists of a set of n sensors $S = \{s_1, s_2, \ldots, s_n\}$. Each sensor in the network is a mobile device equipped with a sensing module. We assume a sensor can sense an intruder or event if and only if it lies within the sensor's sensing range. Every sensor also has a communication module that can send and receive information within its communication range, and a movement module that enables the sensor to move along the barrier. Let $s_i^t = (x_i^t, y_i^t, r_i, R_i)$ denote a sensor s_i located at (x_i^t, y_i^t) at time t with sensing range r_i and communication range R_i. In this

chapter, we assume that all sensors have the same sensing range r, and therefore the *coverage length* of a sensor is $2r$. We assume that for every sensor $r \leq x_i^0 \leq L-r$, and that for $i \neq j$, we have $x_i^0 \neq x_j^0$. For convenience, we assume that $x_1^0 \leq x_2^0 \cdots \leq x_n^0$. We emphasize that while these names and positions of sensors facilitate our proofs, they are not known to any of the sensors, which are completely anonymous.

We assume a fully synchronous (FSYNC) model where time is divided into globally agreed time steps. In each step, every sensor executes a Look-Compute-Move cycle. We assume *limited visibility*: the visibility radius of the sensor is twice the sensing range (range in which the sensor can sense intruders). More precisely, sensor s_i^t is able to see all other sensors located in $[x_i^t - 2r, x_i^t + 2r]$. We say s_i^t sees s_j^t on its right if and only if $0 < x_j^t - x_i^t \leq 2r$ and s_i^t sees s_k^t on its left if and only if $0 < x_i^t - x_k^t \leq 2r$. Note that each sensor has its own conception of left and right, but there is no necessity for global agreement on this. Observe that a sensor is able to detect the next sensor iff there is no gap between them. For networked sensors, visibility could be achieved by exchanging *hello* messages so long as the communication range is twice the sensing range. An important additional assumption is that a sensor can sense an endpoint of the barrier if it lies within its sensing range. Finally, we also assume *limited movement*: in a single time step, a sensor can move at most one unit of distance and the time step is long enough such that this unit distance movement is achievable for sensors.

We study a discrete version of the problem where the coverage length $2r$ of every sensor is an integer greater than 1 ($2r \in \mathbb{N}_{>1}$). Note that an input with sufficient sensors to cover the border and sensors at distinct initial positions would necessarily already cover the border for $r = 0.5$. Initially every sensor's coverage area starts and ends on integer points in the interval $[0, L]$. In other words, at time $t = 0$ every sensor's position is in the form $x_i^0 = r + m$ for $0 \leq m \leq L - 2r$, and $m \in Z$. We say an algorithm A for barrier coverage *terminates* on input S at time t if and only

if when running A on S, no sensor in S moves at any time $t' \geq t$.

We start with some terminology and simple facts about the behavior of algorithms in our model. We define a *gap* as a maximal interval on $(0, L)$, where no point in this interval is within the sensing range of any sensor in the network. Informally, a *pile* is a subset of sensors that contains all sensors between two consecutive gaps, or between a gap and a barrier endpoint.

Definition 4.1. A *pile* at time t is a maximal set of $k \geq 1$ consecutive sensors $P^t = \{s_j \in S | i \leq j \leq i+k-1\}^t$, with $x^t_{j+1} - x^t_j \leq 2r$ for all $i \leq j < i+k-1$.

In Figure 4.2 at time $t = 2$, the piles in the network from left to right are $\{s_1\}^2$, $\{s_2, s_3\}^2$, $\{s_4, s_5, s_6, s_7\}^2$, and $\{s_8\}^2$. For illustration purposes, in the figures, a sensor's coverage length is represented as a rectangle of length $2r$ which shows the interval that the sensor with circular sensing area can cover on the barrier. Also for convenience, two sensors whose coverage lengths overlap are placed at different levels in the illustration; however, as stated earlier, all sensors are initially placed on the barrier and can only move on the barrier. Clearly, at any time, sensors in the network are partitioned into piles that are collectively exhaustive and mutually exclusive.

For any pile $P^t = \{s_i, s_{i+1}, \ldots, s_{i+k-1}\}^t$, we call s^t_i the *leftmost* sensor, s^t_{i+k-1} the *rightmost* sensor, the set

$\{s_{i+1}, s_{i+2}, \ldots, s_{i+k-2}\}^t$ the *middle* sensors and $|P^t| = k$ the *size* of P^t. Also we define $g_l(P^t) = x^t_i - r$ and $g_r(P^t) = x^t_{i+k-1} + r$ as the leftmost and rightmost points on $[0, L]$ that are covered by sensors in P^t respectively. The *length* of P^t can be calculated as $g_r(P^t) - g_l(P^t)$. See Figure 4.1 for an example of a pile.

Let $U^t = \{s_i, s_{i+1}, \ldots, s_j\}^t$ denote a subset of P^t. Then there is no gap between every two consecutive sensors in U^t. We define *excess* of U^t denoted $e(U^t)$ as the difference between the maximum length of the barrier that can be covered by sensors in U^t and the actual length of the barrier that is covered by sensors in U^t, that is,

$e(U^t) = 2r * (j - i) - (x_j^t - x_i^t)$.

Lemma 4.1. *For any pile $P^t = \{s_i, s_{i+1}, \ldots, s_j\}^t$ and integer k, if $i \leq k \leq j$ then:*

$$e(P^t) = e(\{s_i, s_{i+1}, \ldots, s_k\}^t) + e(\{s_k, s_{k+1}, \ldots, s_j\}^t)$$

Proof. Using the definition of excess of P^t:

$$\begin{aligned} e(P^t) &= 2r * (j - i) - (x_j^t - x_i^t) \\ &= 2r * (j - k + k - i) - (x_j^t - x_k^t + x_k^t - x_i^t) \\ &= 2r * (j - k) - (x_j^t - x_k^t) + 2r * (k - i) - (x_k^t - x_i^t) \\ &= e(\{s_k, \ldots, s_j\}^t) + e(\{s_i, \ldots, s_k\}^t) \end{aligned}$$

□

Based on their excess and length, we distinguish two special types of piles below.

Definition 4.2. A pile $P^t = \{s_i, s_{i+1}, \ldots, s_{i+k-1}\}^t$ is called a *heavy pile* if it has the following properties:

$$\begin{cases} e(P^t) \geq 2 & \text{if } g_l(P^t) > 0 \text{ and } g_r(P^t) < L \\ e(P^t) \geq 1 & \text{if } g_l(P^t) = 0 \text{ xor } g_r(P^t) = L \\ e(P^t) \geq 0 & \text{if } g_l(P^t) = 0 \text{ and } g_r(P^t) = L \end{cases}$$

Definition 4.3. A pile $P^t = \{s_i, s_{i+1}, \ldots, s_{i+k-1}\}^t$ is called a *medium pile* if $e(P^t) = 1$ and $|P^t| \geq 3$ and $g_l(P^t) > 0$ and $g_r(P^t) < L$.

In Figure 4.2, $\{s_4, s_5, s_6, s_7, s_8\}^0$ and $\{s_5, s_6, s_7, s_8\}^3$ are heavy piles, while $\{s_1, s_2, s_3\}^0$ is a medium pile. For a heavy or medium pile P^t, we define $o_l(P^t)$ and $o_r(P^t)$ as the leftmost and rightmost points on the barrier that are covered by more than one

sensor from P^t:

$$o_l(P^t) = \min\{x : \ 0 \leq x \leq L \text{ and } \exists s_i^t, s_j^t \in P^t \text{ and } x \text{ is covered by both } s_i^t \text{ and } s_j^t\}$$

$$o_r(P^t) = \max\{x : \ 0 \leq x \leq L \text{ and } \exists s_i^t, s_j^t \in P^t \text{ and } x \text{ is covered by both } s_i^t \text{ and } s_j^t\}$$

Figure 4.1 shows the values o_l, o_r, g_l, and g_r on a heavy pile. The following lemma shows that there exists at least one heavy pile in the network at any time, if there are enough sensors to cover the entire barrier.

Figure 4.1: An example of a (heavy) pile. The shaded areas are gaps.

Lemma 4.2. *If there are enough sensors to cover the entire barrier, the network contains at least one heavy pile.*

Proof. Assume at any time t the set of the sensors in the network S is partitioned into m piles $U_1^t, U_2^t, \ldots, U_m^t$. Since the number of sensors is enough to cover the entire barrier, it is easy to see that $\sum_{i=1}^{m} e(U_i^t)$ is at least equal to the number of gaps on the barrier. We also know that there is a gap between the covered areas of every two consecutive piles. Therefore there are at least $m-1$ gaps on the barrier between the piles. Consider the left and right endpoints of the barrier:

Both points are covered by sensors.

If both endpoints are covered by the same pile, we have a single pile in the network which is heavy. Without loss of generality assume U_1^t and U_m^t are two piles covering the border endpoints. If $e(U_1^t) > 0$ ($e(U_m^t) > 0$), then U_1^t (U_m^t) is

a heavy pile. If $e(U_1^t) = e(U_m^t) = 0$, since there are $m-1$ gaps on the barrier, $\sum_{i=2}^{m-1} e(U_i^t) \geq m-1$. Therefore there is at least one pile U_i^t, with $1 < i < m$ where $e(U_i^t) > 1$ and hence U_i^t is a heavy pile.

Only one of the points is covered by a sensor.

Without loss of generality let U_1^t be the heavy pile covering the endpoint of the barrier. If $e(U_1^t) > 0$, then U_1^t is a heavy pile. Otherwise $e(U_1^t) = 0$ and since there are m gaps on the barrier, $\sum_{i=2}^{m} e(U_i^t) \geq m$. Therefore there is at least one pile U_i^t, and $1 < i \leq m$ where $e(U_i^t) > 1$ and hence U_i^t is a heavy pile.

Neither of the points is covered by any sensor.

There are $m+1$ gaps on the barrier and therefore $\sum_{i=1}^{m} e(U_i^t) \geq m+1$. Therefore, we know there is at least one pile U_i^t and $1 \leq i \leq m$ where $e(U_i^t) > 1$ and hence U_i^t is a heavy pile.

□

4.3 An Oblivious Distributed Algorithm for Barrier Coverage

In this section, we describe an oblivious distributed algorithm for barrier coverage and prove that it terminates in $\Theta(n^2)$ steps. As shown in Algorithm 4.1, in each step, every sensor that senses another sensor on one side and a gap on the other side moves one unit toward the gap. An example of this algorithm is shown in Figure 4.2.

Algorithm 4.1 Oblivious algorithm for barrier coverage

Every sensor $s_i \in S$ at the beginning of every step does the following:
if s_i sees another sensor on one side and there is a gap on its other side **then**
  s_i moves one unit toward the gap during this step
end if

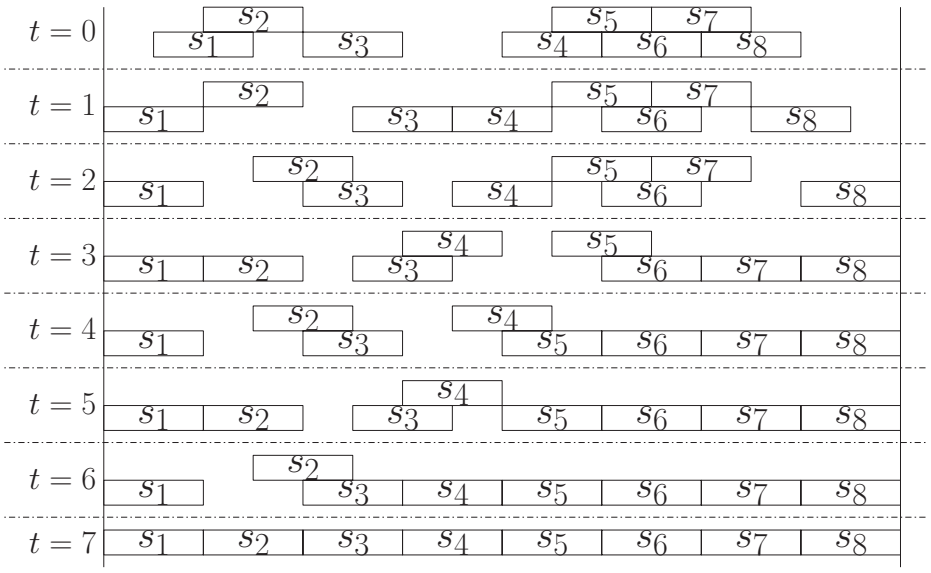


Figure 4.2: Algorithm 4.1 example, $r = 1$

Next we prove the correctness of Algorithm 4.1. We start with some lemmas that show the behavior of heavy and medium piles. First we establish a relationship between piles in consecutive steps.

Definition 4.4. We call U_1^{t-1} the *parent* of U_2^t and U_2^t is a *child* of U_1^{t-1} if and only if

$$\begin{cases} U_2^t \text{ contains all sensors in } U_1^{t-1} & \text{if } |U_1^{t-1}| \leq 2 \\ U_2^t \text{ contains all middle sensors in } U_1^{t-1} & \text{if } |U_1^{t-1}| > 2 \end{cases}$$

In Figure 4.2, the pile $\{s_5, s_6, s_7, s_8\}^3$ is a child of the pile $\{s_4, s_5, s_6, s_7\}^2$. Observe that a heavy or medium pile at time t can be the child of two or more heavy piles at time $t-1$. The following technical lemmas show that heavy and medium piles cannot appear *out of nowhere*, they must always have one or more parents in the previous step.

Lemma 4.3. *Let P^{t+1} be a heavy or medium pile at time $t+1$ and U^t be a pile at time t. If P^{t+1} contains at least two sensors from U^t then U^t is a heavy or medium pile and a parent of P^{t+1}.*

Proof. Let s_i and s_j be the leftmost and rightmost sensors in $U^t \cap P^{t+1}$ respectively. First we show that U^t is a parent of P^{t+1}. If $\{s_i, s_j\}$ are the leftmost and rightmost sensors of U^t then P^{t+1} contains all sensors of U^t. Otherwise one of $\{s_i, s_j\}$ is a middle sensor of U^t, and since middle sensors of a pile do not move P^{t+1} contains all middle sensors of U^t. In both cases, U^t is a parent of P^{t+1}.

Second we show U^t is either a heavy or medium pile. We consider the possibilities for movement of s_i^t and s_j^t. Since s_i^t and s_j^t are both are in the same pile U^t, either s_i^t moves to the left or it does not move. Similarly either s_j^t moves to the right or does not move at all. Therefore all possible cases are as follows:

s_i^t and s_j^t move to the left and right respectively:

It can be seen that $e(U^t) = e(\{s_k^t | i \leq k \leq j\}) = 2 + e(\{s_k^{t+1} | i \leq k \leq j\}) \geq 2$ and hence U^t is a heavy pile.

s_i^t moves to the left and s_j^t does not move:

Since s_i^t moves left $e(U^t) \geq e(\{s_k^t | i \leq k \leq j\}) = 1 + e(\{s_k^{t+1} | i \leq k \leq j\}) \geq 1$. Also since s_j^t does not move either $j = n$ (it is the rightmost sensor in the network and reached the right end of the barrier) or s_j^t is a middle sensor of U^t. If $j = n$ then U^t is a heavy pile. If not, then s_j^t is a middle sensor of U^t, and $|U^t| \geq 3$ and is a medium pile. Thus U^t is a heavy pile or a medium pile.

s_j^t moves to the right and s_i^t does not move:

This case is similar to the previous case and either U^t is either a heavy pile or a medium pile.

neither s_i^t nor s_j^t move:

Since s_i^t does not move, either it is the leftmost sensor in the network and reached the left end of the barrier or it is the leftmost middle sensor of U^t. In both cases it is the leftmost sensor of P^{t+1}. Similarly s_j^{t+1} is the rightmost

sensor of P^{t+1}. Therefore $P^{t+1} \subseteq U^t$ and U^t is either a heavy or medium pile.

We have shown that in all cases U^t is either a heavy or a medium pile and a parent of P^{t+1}. □

Lemma 4.4. *Any heavy or medium pile P^{t+1} at time $t+1$ with $t \geq 0$ is a child of one or more heavy or medium piles at time t.*

Proof. First consider the case where $g_l(P^{t+1}) > 0$ or $g_r(P^{t+1}) < L$. Since P^{t+1} is a heavy or medium pile, we have $|P^{t+1}| \geq 2$. Let s_i^{t+1} and s_{i+1}^{t+1} denote two consecutive sensors of P^{t+1} such that $e(\{s_i, s_{i+1}\}^{t+1}) \geq 1$. The existence of such two sensors is guaranteed since P^{t+1} is either a heavy or medium pile with $g_l(P^{t+1}) > 0$ or $g_r(P^{t+1}) < L$. Let U_1^t and U_2^t denote the piles containing s_i^t and s_{i+1}^t. One of the following two cases holds:

$U_1^t = U_2^t$**:**

Therefore $\{s_i, s_{i+1}\} \subseteq P^{t+1} \cap U_1^t$ and by Lemma 4.3, U_1^t is a heavy or medium pile and a parent of P^{t+1}.

$U_1^t \neq U_2^t$**:**

Since $e(\{s_i, s_{i+1}\}^{t+1}) \geq 1$, sensors s_i^t and s_{i+1}^t move to the right and left respectively and therefore it can be seen that $e(\{s_i, s_{i+1}\}^{t+1}) = 1$. Since P^{t+1} is a heavy or medium pile, it must contain at least one other sensor. Without loss of generality assume $s_{i+2}^{t+1} \in P^{t+1}$. Since s_{i+1}^t moves to the left therefore s_{i+1}^t and s_{i+2}^t belong to the same pile U_2^t. By Lemma 4.3, U_2^t is a heavy or medium pile and a parent of P^{t+1}.

Second consider the case where $g_l(P^{t+1}) = 0$ and $g_r(P^{t+1}) = L$. In this case P^{t+1} contains all sensors in the network. Take any heavy pile U^t (Lemma 4.2 guarantees the existence of such a pile); clearly U^t is a parent of P^{t+1}. □

Additionally, it follows from the definition of a parent pile that a pile can be a parent to at most one pile in the next step. Thus, the number of heavy and medium piles in the network can never increase. We now proceed to show that the total excess in the heavy and medium piles is guaranteed to decrease within $2n-1$ steps. The following technical lemma is useful in some of the proofs below.

Lemma 4.5. *Let U^t be a heavy or medium pile with a heavy or medium pile child P^{t+1}, and let s_i^t and s_k^{t+1} be the rightmost sensor of U^t and P^{t+1} respectively. If P^{t+1} is not a child of V^t, the next medium or heavy pile, if any, to the right of U^t, then:*

$$e(\{s_{i-1},\ldots,s_k\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t)$$

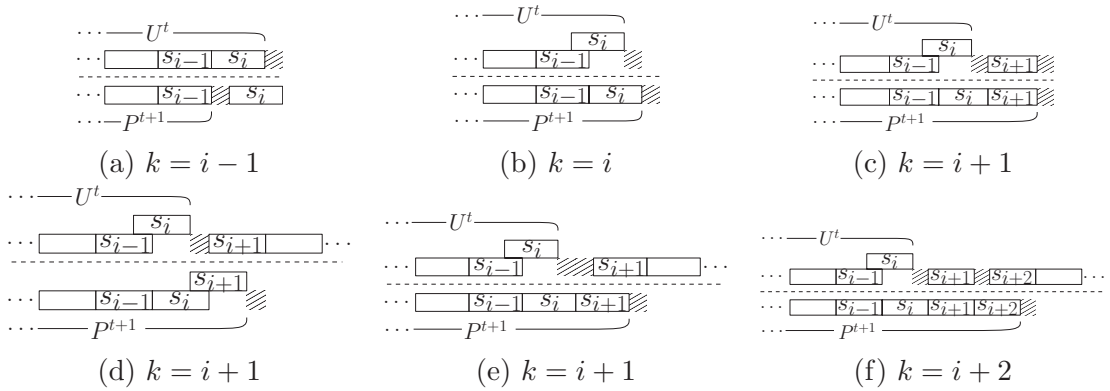
Proof. First note that sensor s_{i-1}^t does not move to the right. We consider the possibilities for k (see Figure 4.3):

$k = i-1$: Then s_i^t was dropped from the pile, and $e(\{s_{i-1},s_i\}^t) \geq e(\{s_{i-1}\}^{t+1}) = 0$.

$k = i$: Since s_i^t does not move to the left, therefore $e(\{s_{i-1},s_i\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t)$.

$k = i+1$: Then either the next pile after U^t was of size one and sensor s_{i+1}^t does not move, in which case $e(\{s_{i-1},s_i,s_{i+1}\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t) - 1$ or the next pile was of size at least two, and s_{i+1}^t separated from it and moved left to join P^{t+1}. If it did not create an overlap with s_i^{t+1} (recalling that s_i^t moved right) , then $e(\{s_{i-1},s_i,s_{i+1}\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t) - 1$. If instead it created an overlap with s_i^{t+1} then $e(\{s_{i-1},s_i,s_{i+1}\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t)$.

$k = i+2$: In this case, since P^{t+1} is not a child of the next heavy or medium pile to the right of U^t, it must be that s_{i+1} is a singleton pile, followed by a gap of size one and s_{i+2}^t separates from its pile to move left to join P^{t+1}. In this case, $e(\{s_{i-1},\ldots,s_{i+2}\}^{t+1}) \leq e(\{s_{i-1},s_i\}^t) - 1$.



(a) $k = i - 1$ (b) $k = i$ (c) $k = i + 1$

(d) $k = i + 1$ (e) $k = i + 1$ (f) $k = i + 2$

Figure 4.3: The different possibilities for the rightmost sensor s_k^{t+1}.

$k > i + 2$: Then P^{t+1} contains two sensors from V^t and by Lemma 4.3, is a child of V^t, a contradiction.

□

Clearly the same arguments hold for sensors at the left endpoint of U^t:

Corollary 4.1. *Let U^t be a heavy or medium pile with a heavy or medium pile child P^{t+1}, and let s_i^t and s_k^{t+1} be the leftmost sensor of U^t and P^{t+1} respectively. If P^{t+1} is not a child of V^t, the next medium or heavy pile, if any, to the left of U^t, then:*

$$e(\{s_k, \ldots, s_{i+1}\}^{t+1}) \leq e(\{s_i, s_{i+1}\}^t)$$

Next we show that if two or more heavy or medium piles *merge* into one child, the excess of the child is strictly less than the combined excess of the parents, while if a heavy or medium pile has a single parent, its excess cannot be more than that of its parent.

Lemma 4.6. *Let P^{t+1} be a heavy or medium pile. If P^{t+1} has $k \geq 1$ heavy or medium pile parents $\{U_1^t, \ldots, U_k^t\}$, then $e(P^{t+1}) \leq \Sigma_{j=1}^k e(U_j^t) - (k-1)$.*

Proof. Let s_i^t and s_j^t be the leftmost and rightmost sensors of U_1^t and U_k^t respectively. Lemma 4.5 and Corollary 4.1 imply that movements of sensors to the left of s_{i+1}^t

and s^t_{j-1} do not add to the excess of P^{t+1} relative to its parents. Thus, we consider only the difference in excess between $\{s_{i+1}, \ldots, s_{j-1}\}^{t+1}$ and $\{s_{i+1}, \ldots, s_{j-1}\}^t$. Recall that the excess of a set of sensors is the difference between capacity of the pile and the length of barrier covered by the set. The two sets contain the same sensors, therefore their capacities are the same. However, the length of the barrier covered by $\{s_{i+1}, \ldots, s_{j-1}\}^{t+1}$ is at least $k-1$ more than the length of barrier covered by $\{s_{i+1}, \ldots, s_{j-1}\}^t$, since the total lengths of gaps between the parents is at least $k-1$. The lemma follows. □

We claim that a heavy pile with a single parent cannot have a medium pile parent. For heavy piles with excess ≥ 2, this follows immediately from Lemma 4.6. A heavy pile with excess 0 contains all sensors in S and must have a heavy pile parent, since by Lemma 4.2, there exists a heavy pile in the previous step. Consider a heavy pile P^{t+1}, with excess 1 and a single parent that is a medium pile U^t. Assume without loss of generality that $g_r(P^{t+1}) = L$, and let s^t_i be the rightmost sensor of U^t. Using a case analysis similar to the proof of Lemma 4.5, it is easy to see that $g_r(P^{t+1}) = L$ implies that the rightmost sensor of P^{t+1} must be either s^{t+1}_i or s^t_{i+1} with s^t_{i+1} being a singleton sensor. In both these cases, it is easy to see that $e(P^{t+1}) = e(U^t) - 1 = 0$, which contradicts the fact that P^{t+1} is a heavy pile with excess one. We conclude that the single parent of a heavy pile must be a heavy pile itself.

We proceed to show that while the excess of a heavy pile with a single parent is not necessarily smaller than its parent, some notion of progress in terms of the excess is in fact achieved.

Lemma 4.7. *Let P^t be a heavy pile that does not cover the entire barrier. If P^t is the only heavy pile parent of heavy pile P^{t+1}, then one of the following must hold:*

(i) t is an excess-reducing step: $e(P^{t+1}) < e(P^t)$

(ii) t is a pile-maintaining step: $e(P^{t+1}) = e(P^t)$ and one of the following is true:

(a) $o_l(P^{t+1}) = o_l(P^t) < o_r(P^t) < o_r(P^{t+1})$ *and* $o_r(P^{t+1}) = g_r(P^{t+1}) - 2r + 1$

(b) $g_l(P^{t+1}) + 2r - 1 = o_l(P^{t+1}) < o_l(P^t) < o_r(P^t)$ *and* $o_r(P^t) = o_r(P^{t+1})$

(c) $g_l(P^{t+1}) + 2r - 1 = o_l(P^{t+1}) < o_l(P^t) < o_r(P^t)$ *and* $o_r(P^t) < o_r(P^{t+1}) = g_r(P^{t+1}) - 2r + 1$

(iii) t is a pile-shortening step: $e(P^{t+1}) = e(P^t)$ and $o_r(P^{t+1}) = o_r(P^t)$ and $o_l(P^{t+1}) = o_l(P^t)$ and $|P^{t+1}| < |P^t|$

Proof. Let s_i^t and s_j^t be the leftmost and rightmost sensors of P^t respectively, and let s_p^{t+1} and s_q^{t+1} be the leftmost and rightmost sensors of P^{t+1} respectively. Assume that $e(P^{t+1}) = e(P^t)$, that is, t is not an excess-reducing step. Then from the proof of Lemma 4.5, it can be seen that only one of two cases is possible for the rightmost sensor of P^{t+1}. Either $q = j - 1$, that is, s_j^t is dropped from the pile in which case $o_r(P^{t+1}) = o_r(P^t)$, or $q = j + 1$ and s_{j+1}^t is added to the pile so that there is an overlap between s_j^{t+1} and s_{j+1}^{t+1}, and $o_r(P^t) < o_r(P^{t+1}) = g_r(P^{t+1}) - 2r + 1$. Similarly at the left end of the pile, either $p = i+1$, that is, s_i^t is dropped from the pile in which case $o_l(P^{t+1}) = o_l(P^t)$, or $p = i - 1$ and s_{i-1}^t is added to the pile so that there is an overlap between s_{i-1}^{t+1} and s_i^{t+1} with $g_l(P^{t+1}) + 2r - 1 = o_l(P^{t+1}) < o_l(P^t)$. It is now easy to see that if a sensor is dropped at one endpoint, and not added at another, then t is a pile-shortening step, while if a sensor is added at either endpoint, then t is a pile-maintaining step. □

Theorem 4.1. *Given any input of n sensors with enough sensors to cover the barrier ($n \geq L/2r$), Algorithm 4.1 terminates in $O(2rn^2)$ steps with the whole barrier fully covered.*

Proof. Let $X(t)$ be the *total heavy and medium pile excess* at time t. Then it follows from Lemma 4.6 that $X(t + 1) \leq X(t)$. We prove that as long as the barrier is

not completely covered, this quantity must in fact decrease within at most $2n-1$ steps. Assume for the purpose of contradiction that t is a step when the barrier is not completely covered and $X(t+2n-1) = X(t)$. Then Lemma 4.6 implies that the number of heavy and medium piles at time $t+2n-1$ is the same as the number of heavy and medium piles at time t. Additionally, every medium or heavy pile at time $t+2n-1$ has a unique ancestor at every step in the time interval $[t, t+2n-2]$. Let P^{t+2n-1} be a heavy pile; we denote its unique heavy pile ancestor at time i as P^i for $t \leq i \leq t+2n-2$. Since $X(t+2n-1) = X(t)$, it follows from Lemma 4.6 that $e(P^t) = e(P^{t+i})$ for $0 \leq i \leq 2n-1$. Lemma 4.7 implies that for every t' such that $t \leq t' < t+2n-1$, step t' is either a pile-maintaining step or a pile-reducing step. Since $|P^t| \leq n$, there must exist at least one pile-maintaining step; let T be the first pile-maintaining step in the time interval $[t, t+2n-2]$.

Without loss of generality assume that $o_r(P^{T+1}) > o_r(P^T)$. Then by Lemma 4.7, $o_r(P^{T+1}) = g_r(P^{T+1}) - 2r + 1$. Then $T+1$ cannot be a pile-reducing step, and by assumption, it cannot be an excess-reducing step. Therefore $T+1$ is also a pile-maintaining step. The same argument can be repeated for every step from T to $t+2n-2$, proving that they must be all be pile-maintaining steps. However, in each such pile-maintaining step t', $P^{t'}$ acquires a new rightmost sensor, and this can happen at most $n-2$ times. We conclude that $T \geq t+n+1$. Since T was the first pile-maintaining step in the time interval $[t, t+2n-2]$, it must be that every t' such that $t \leq t' < T$ was a pile-shortening step. However, since $|P^t| \leq n$, there can be at most n consecutive pile shortening steps. Thus, $T \leq t+n$, a contradiction. We have proved that $X(t+2n-1) < X(t)$ for any time t that the barrier is not completely covered.

Therefore, either the barrier is completely covered at time $2n\ i$ or $X(2n\ i) \leq X(0) - i$. On any input of n sensors, $X(0)$ is $O(2rn)$. It follows that the barrier must

be completely covered in $O(2rn^2)$ steps. □

Theorem 4.2. *There exist inputs of n sensors with $n \geq \frac{L}{2r}$ where the running time of Algorithm 4.1 is $\Omega(n^2)$.*

Proof. We claim that Algorithm 4.1 takes $\Omega(n^2)$ time on the input shown in Figure 4.4.

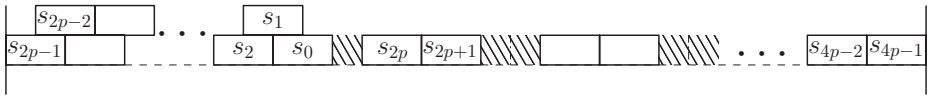


Figure 4.4: A worst case input for Algorithm 4.1

Call S_i the arrangement of $n = 4p$ sensors defined as follows:

- Start with a heavy pile of $2p - i$ sensors with their left ends starting at consecutive positions 0 to $2p - i - 1$.
- then a gap of size 1 followed by i piles of two sensors of excess 1 separated by gaps of size 1.
- then a gap of 1 followed by (a) $p - i/2$ piles of two sensors and zero excess, separated by gaps of size 2, if i is even and (b) $p - (i-1)/2$ piles of two sensors and zero excess, separated by gaps of size 2, and finally a single sensor if i is odd.

The arrangement S_0 is shown in Figure 4.4. We consider the time taken by Algorithm 4.1 on input S_0. It is easy to see that the final arrangement must be a pile of $4p$ sensors of excess 0, and the sensor initially at position 1 (the second left-most sensor) must move in any solution. However, we claim that in Algorithm 4.1, this sensor cannot move until time $8p^2 - 10p + 2$, thus giving a $\Omega(n^2)$ lower bound for Algorithm 4.1. For convenience we number the sensors in the heavy pile thus: the leftmost sensor is labelled $2p - 1$, the next sensor is labelled $2p - 2$ and so forth,

until the rightmost sensor in the heavy pile is labelled 0. We prove by induction that sensor i in the heavy pile moves for the first time at time $t_i = 2i^2 + 3i$ and the arrangement at this time is S_i. The basis is clearly true since sensor 0 moves at step 0 and the initial arrangement is S_0. For the inductive step, after sensor i moves at time t_i, observe that there will be $2i + 1$ pile-maintaining steps, followed by an excess-reducing step, followed by $2i+3$ pile-reducing steps to reach arrangement S_{i+1}, thus $t_{i+1} = t_i + 4i + 5 = 2(i + 1)^2 + 3(i + 1)$ as desired. □

So far we considered the case where initially there exist enough sensors to cover the border. In the following we study the behaviour of Algorithm 4.1 in cases where the number of sensors is not sufficient to cover the entire border.

First we define some new notation. A *non-singular pile (NSP)* is a pile which contains more than one sensor while a *non-singular gap (NSG)* is a gap with length greater than one. Note that if all piles in the network are singular Algorithm 4.1 terminates.

As shown in the next lemma, for n in a certain range, Algorithm 4.1 never terminates.

Lemma 4.8. *Given any input of n sensors with $\frac{L+1}{2r+1} < n < \frac{L}{2r}$, Algorithm 4.1 never terminates.*

Proof. First we show that at any time $t \geq 0$ there exists an NSP. Assume to the contrary that at some time t every pile contains exactly one sensor. Since there is a gap between any two sensors, the distance between the leftmost end of s_1^t and the rightmost end of s_n^t is at least $2rn + n - 1 > \frac{2r(L+1)}{2r+1} + \frac{L+1}{2r+1} - 1 = L$, a contradiction, since all sensors are initially entirely in the interval $[r, L - r]$ and can never move outside this interval.

Let P^t denote an NSP at time t. Clearly, since $n < L/2r$, there must be a gap either to the right or to the left of P^t. Let s_i^t be the rightmost sensor of P^t with a

gap on its right (the case where there is a gap on the left of P^t is symmetric). Thus, in Algorithm 4.1, sensor s_i^t must move to the right and thus the algorithm does not terminate at time t. □

Now we consider the case where $n \leq \frac{L+1}{2r+1}$. We show that Algorithm 4.1 terminates in this case. Another notation that we need in our proofs is *identically arranged* time steps:

Definition 4.5. Let t_1 and t_2 with $t_1 \neq t_2$ denote two distinct time steps. t_1 and t_2 are called *identically arranged* iff for every sensor s_i its position at time t_1 and t_2 are the same; or more formally:

$$\forall s_i \in S \; x_i^{t_1} = x_i^{t_2} \tag{4.1}$$

Lemma 4.9. *On any input of sensors if Algorithm 4.1 does not terminate, there exist two distinct identically arranged time steps t_1 and t_2 such that arrangement of sensors at t_1 (and t_2) contains at least one NSP.*

Proof. Clearly, if the algorithm does not terminate, there must be an NSP in every time step. The existence of identically arranged time steps is straightforward since the number of sensors and grid positions on the border are finite. □

Lemma 4.9 implies that for any sensor $s_i \in S$ and a time step t with $t_1 \leq t < t_2$ if s_i^t moves to the right (left) then there exists a time step t' such that $t_1 \leq t' < t_2$ and $s_i^{t'}$ moves to the left (right).

Lemma 4.10. *Let t_1 and t_2 denote two identically arranged time steps with $t_2 > t_1$ and let g^{t_1} be a gap at time t_1. Also let sensors $s_i^{t_1}$ and $s_{i+1}^{t_1}$ denote the rightmost and leftmost sensors on the left and right of g^{t_1} respectively (if any). If there exists an NSP in the arrangement at time t_1, then there exists a time step t' with $t_1 \leq t' < t_2$ such that either $s_i^{t'}$ moves to the right or $s_{i+1}^{t'}$ moves to the left.*

Proof. We write the proof for the case where there exists an NSP on the left of $s_i^{t_1}$. The proof for the other case is symmetric. For the sake of contradiction we assume that s_{i+1} does not move to the left in any time step between t_1 and t_2, also s_i does not move to the right at any time step before t' and show that $s_i^{t_1}$ moves to the right at t'.

Let $P_1^{t_1}$ denote an NSP on the left of $s_i^{t_1}$ (see Figure 4.5). Let $s_{i_1}^{t_1}$ denote the rightmost sensor of P^{t_1}. Obviously $1 \leq i_1 \leq i$ and there exists a gap on the right of $s_{i_1}^{t_1}$. Therefore $s_{i_1}^{t_1}$ moves to the right. Lemma 4.9 implies that $s_{i_1}^{\tau_1}$ moves to the left for some time step τ_1 with $t_1 \leq \tau_1 < t_2$. Consider time step τ_1. According to Algorithm 4.1, a sensor moves to the left iff it is the leftmost sensor of some NSP. Let $P_2^{\tau_1}$ denote the NSP that contains $s_{i_1}^{\tau_1}$ and let $s_{i_2}^{\tau_1}$ denote the rightmost sensor of $P_2^{\tau_1}$. It is easy to see that $i_1 < i_2 \leq i$. Therefore $s_{i_2}^{\tau_2}$ moves to the right. Inductively it can be seen that at some time $\tau_k = t'$ sensor s_i becomes the rightmost sensor of $P_k^{\tau_k}$ and since there is a gap to its right it moves to the right and the proof is complete. □

Figure 4.5: An example of sensors at time t_1 with a gap and an NSP.

Lemma 4.11. *Running Algorithm 4.1, if $n \leq \frac{L+1}{2r+1}$ then there exists no pair of time steps t_1, t_2 with $t_1 < t_2$ such that t_1 and t_2 are identically arranged and arrangement of sensors at t_1 contains at least one NSP.*

Proof. We consider the following exhaustive cases at time t_1:

- There exists a gap of size greater than two

- At least one of the border endpoints is not covered by any sensor

- All gaps are of size either one or two and both border endpoints are covered

Consider the first case. Let g^{t_1} be a gap of size greater than two at time t_1 and let $s_i^{t_1}$ and $s_{i+1}^{t_1}$ denote the rightmost and leftmost sensors on the left and right of g^{t_1} respectively. There exists an NSP either on the left or on the right of g^{t_1} therefore Lemma 4.10 implies that at some time step t' with $t_1 \leq t' < t_2$ either $s_i^{t'}$ moves to the right or $s_{i+1}^{t'}$ moves to the left and therefore $x_{i+1}^{t'} - x_i^{t'} < x_{i+1}^{t_1} - x_i^{t_1}$ (the length of g reduces). Observe that in running Algorithm 4.1, the length of a gap cannot increase to any value more than 2. Therefore it is not possible that length of g^{t_2} is the same as the length of g^{t_1} which implies that t_1 and t_2 are not identically arranged.

Considering the second case, similar to the first case the length of the gap adjacent to the uncovered endpoint at time t_1 will reduce at some time $t_1 \leq t' < t_2$ and according to Algorithm 4.1 it cannot increase after that. Therefore the length of the gap at time t_2 is strictly less than the length of the gap at time t_1 and t_1 and t_2 are not identically arranged.

Finally consider the case where all gaps at time t_1 are of size either one or two and both border endpoints are covered. Let m_1 and m_2 denote the number of gaps of length 1 and 2 respectively. Then the total length of gaps is clearly $m_1 + 2m_2$. Note that $n \leq \frac{L+1}{2r+1}$ implies that $n - 1 \leq L - 2rn \leq$ total length of gaps and therefore:

$$n - 1 \leq 2m_2 + m_1 \tag{4.2}$$

Also let p and p' denote the number of piles and NSPs at time t_1. It can easily be seen that $p \leq n - p'$ (equality holds when all NSPs contain exactly two sensors). Also since there is exactly one gap (either singular or non-singular) between every two piles:

$$m_1 + m_2 = p - 1$$

and hence

$$m_1 + m_2 \leq n - p' - 1 \tag{4.3}$$

Finally, Eq. 4.2 and Eq. 4.3 implies that:

$$p' \leq m_2 \tag{4.4}$$

Now that we proved that the number of NSGs in the network is at least equal to the number of NSPs, two sub-cases are possible at time t_1:

- There exists an NSG in the network such that either there is no NSP to its left or no NSP to its right
- There exist two NSGs in the network such that there is no NSP between them

In the following we show that in both cases t_1 and t_2 cannot be identically arranged. Assume to the contrary.

First consider the sub-case where there exists an NSG g at time t_1 with no NSP on the left of it (proof for the case where there exists no NSP on the right of g^{t_1} is symmetric). Let $s_{i+1}^{t_1}$ denote the leftmost sensor on the right of g^{t_1}. Since there is an NSP on the right of $s_{i+1}^{t_1}$, Lemma 4.10 implies that $s_{i+1}^{t'}$ moves to the left with $t_1 \leq t' < t_2$. Therefore sensor $s_{i+1}^{t''}$ should move to the right at some time t'' with $t < t'' < t_2$. For $s_{i+1}^{t''}$ to move right there should be no gap between $s_i^{t''}$ and $s_{i+1}^{t''}$ at time t'' and therefore:

$$x_i^{t''} > x_i^{t_1}$$

Therefore $s_i^{t''}$ is located at $x_i^{t_1} + 1$ at some time between t_1 and t''. However for this move to happen s_{i-1} must be located at $x_i^{t_1} - 2r$ and therefore at $x_{i-1}^{t_1} + 1$ at some time step between t_1 and t_2. Inductively, this implies that s_1 must be located at $x_1^{t_1} + 1$ at some time step between t_1 and t_2 which is impossible since according to

Algorithm 4.1 sensor s_1 never moves to the right.

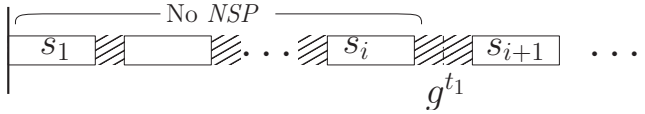


Figure 4.6: Sub-case where there exists an NSG with no NSP to its left

Second we consider the case where there exist two consecutive NSGs with no NSP between them. Let g_1 and g_2 denote two NSGs with no NSPs between them and g_1 is on the left of g_2. Let $s_i^{t_1}, s_{i+1}^{t_1}, \ldots, s_j^{t_1}$ with $i \leq j$ denote the sensors between g_1 and g_2. Since there exists an NSP on the left of $s_i^{t_1}$ Lemma 4.10 implies that either s_{i-1} moves to the right or s_i moves to the left at some time step between t_1 and t_2. In both cases there must exist a time step between t_1 and t_2 such that s_i is located at $x_i^{t_1} - 1$.

Similarly since there is an NSP on the right of s_j, sensor s_j should be located at $x_j^{t_1} + 1$ at some time step between t_1 and t_2. This implies that s_{j-1} must be at $x_{j-1}^{t_1} + 1$ at some time step between t_1 and t_2. Inductively, s_i must be at $x_i^{t_1} + 1$ at some time step between t_1 and t_2 (see Figure 4.7.

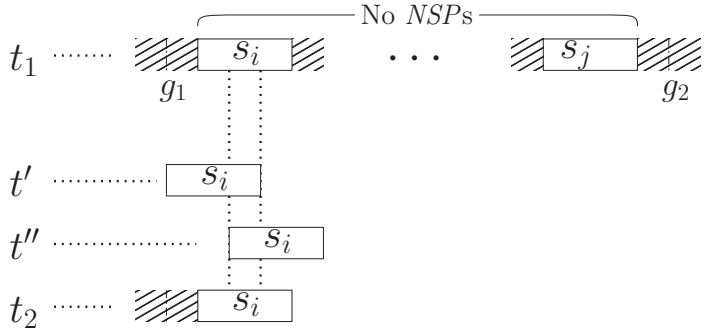


Figure 4.7: Sub-case where there exists two NSGs with no NSP between them

Now let t' and t'' denote the last time steps before t_2 that s_i is located at $x_i^{t_1} - 1$ and $x_i^{t_1} + 1$ respectively. Obviously $t' \neq t''$. If $t' < t''$, for $s_i^{t''}$ to be located at $x_i^{t_1} + 1$ sensor s_{i-1} must be located in $x_{i-1}^{t_1} + 2$ at some time step $\geq t'$. For $s_{i-1}^{t_2}$ to be located at $x_{i-1}^{t_1}$ sensor s_i must be located at $x_i^{t_1} - 1$ at some time step between $t' + 1$ and t_2

which contradicts the assumption that t' is the last time step that $s_i^{t'}$ is located at $x_i^{t_1} - 1$. The case where $t'' < t'$ similarly leads to a contradiction. □

Theorem 4.3. *For any input of n sensors, Algorithm 4.1 terminates if and only if* $n \leq \frac{L+1}{2r+1}$ *or* $\frac{L}{2r} \leq n$.

Proof. Theorem 4.1 and Lemma 4.8 describe the behavior of the algorithm when $n > \frac{L+1}{2r+1}$. The proof of termination for the case where $n \leq \frac{L+1}{2r+1}$ is straightforward from Lemma 4.9 and Lemma 4.11. □

Finally we show that there exists an input arrangement where Algorithm 4.1 takes $\Omega(n)$ steps to terminate while clearly an optimal algorithm needs only 1 step; see Figures 4.8.

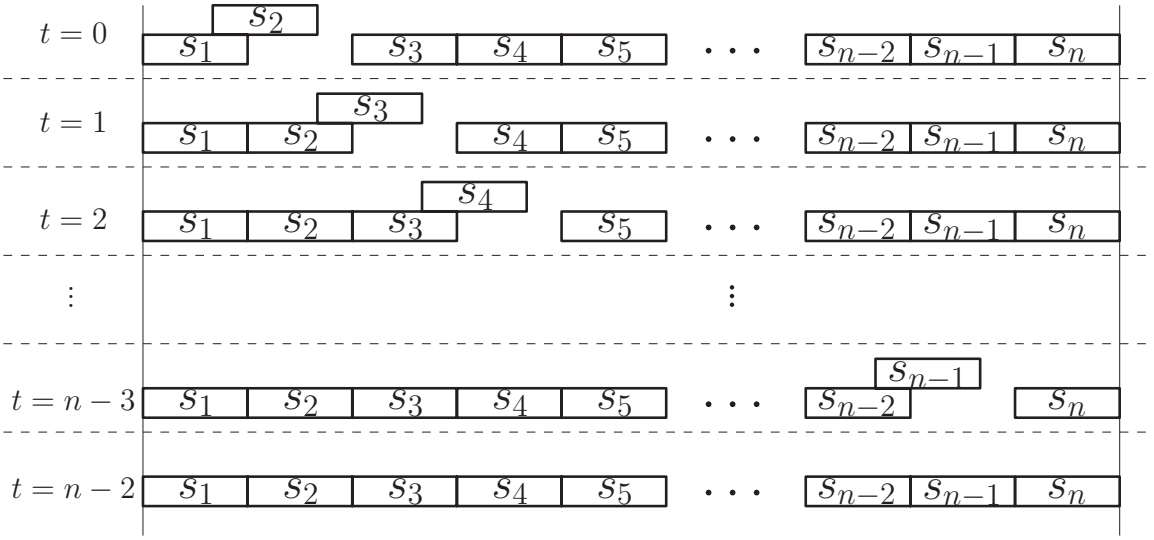


Figure 4.8: An example where Algorithm 4.1 takes $\Omega(n)$ time to terminate while an optimal algorithm terminates in one step.

4.4 A Constant-Memory Algorithm for Barrier Coverage

In this section, we describe Algorithm 4.2, a non-oblivious algorithm for barrier coverage and prove its correctness and complexity. In Algorithm 4.2, each sensor remembers whether it moved and the direction of the move in the previous step. Initially, or if not moving in the previous step, a sensor behaves like in Algorithm 4.1: if it senses a gap on one side and a sensor on the other it moves in the direction of the gap. However, the key difference is that once a sensor makes a move in one direction, it keeps moving in that direction in the subsequent steps as long as there is a gap next to it in that direction. Once it is "blocked" by a sensor in that direction, it stops and waits for one time step, before making a new decision as in Algorithm 4.1 again.

Thus, two bits of memory are needed in Algorithm 4.2 to remember the type of the previous move, however, it achieves full barrier coverage in linear time. Note that in Algorithm 4.2, although every sensor needs to distinguish between its own left and right sides, no global agreement among sensors on the concept of left and right is assumed.

In the proofs below we use the following concept of a *collection* of sensors.

Definition 4.6. For every heavy pile $P = U^0$ at time 0, let $C^t(P)$ denote the *collection* of P at time t. $C^t(P)$ is defined recursively as follows:

$$\begin{cases} C^0(P) = P & \text{if } t = 0 \\ C^t(P) = C^{t-1}(P) \cup A_l^t(P) \cup A_r^t(P) & \text{if } t > 0 \end{cases}$$

where $A_l^t(P)$ and $A_r^t(P)$ are the piles at time t that contains the leftmost and rightmost sensors of $C^{t-1}(P)$ respectively.

Algorithm 4.2

Every sensor $s_i \in S$ initially does the following:
  $s_i.STATE$ =NO-MOVE
Every sensor $s_i \in S$ at the beginning of every time step does the following:
switch $s_i.STATE$
  case RIGHT-MOVE:
    if s_i senses a gap on its right **then**
      s_i moves one unit to the right during this step
    else
      $s_i.STATE$=NO-MOVE
  case LEFT-MOVE:
    if s_i senses a gap on its left **then**
      s_i moves one unit to the left during this step
    else
      $s_i.STATE$=NO-MOVE
  case NO-MOVE:
    if s_i senses a sensor on its left and a gap on its right **then**
      s_i moves one unit to the right during this step
      $s_i.STATE$=RIGHT-MOVE
    else
      if s_i senses a sensor on its right and a gap on its left **then**
        s_i moves one unit to the left during this step
        $s_i.STATE$=LEFT-MOVE
end switch

See Figure 4.9 for an illustration of a collection. Observe that while there are no gaps between the sensors of a pile, there can be gaps between the sensors of a collection.

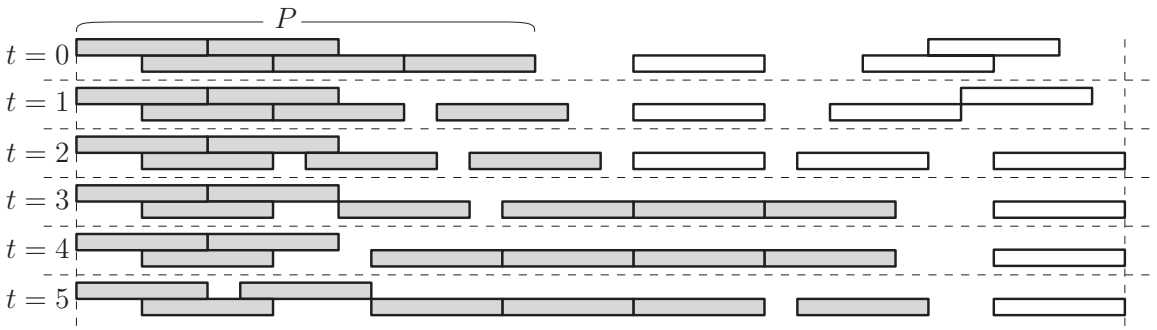


Figure 4.9: The shaded areas show the collections of the pile P at consecutive steps, $r = 2$.

Lemma 4.12. *Let $P = U^0$ and P^t be two heavy piles at time 0 and $t \geq 0$ respectively. If P^t is a descendant of P, then $P^t \subseteq C^t(P)$.*

Proof. We present an inductive proof. The basis follows directly from the definition of $C^0(P)$. Assume $P^t \subseteq C^t(P)$ for some $t \geq 0$. Let P^{t+1} denote a descendant of P at time $t+1$, then it is the only child of P^t. We show that $P^{t+1} \subseteq C^{t+1}(P)$. Let s_i and s_j be the leftmost and rightmost sensors of $C^t(P)$. If $s_i \in P^{t+1}$, then $P^{t+1} = A_l^{t+1}(P) \subset C^{t+1}(P)$. Similarly if $s_j \in P^{t+1}$, then $P^{t+1} = A_r^{t+1}(P) \subset C^{t+1}(P)$. If neither s_i nor s_j is in P^{t+1}, then $P^{t+1} \subset C^t(P) \subseteq C^{t+1}(P)$. □

Lemma 4.13. *Let $P = U^0$ and P^t denote two heavy piles at time 0 and $t \geq 0$ respectively. If P^t is a descendant of P then $C^t(P)$ has the following properties:*

(i) Either $C^t(P)$ is a pile (no gap between its leftmost and rightmost sensors), or

(ii) Between the leftmost and the rightmost sensor of $C^t(P)$ there are only gaps of length one and for each such gap between $s_i^t \in C^t(P)$ and $s_{i+1}^t \in C^t(P)$ we have:

- *if s_{i+1}^t is to the right of the rightmost sensor of P^t, then sensor s_i^t moves one unit to the right and s_{i+1}^t does not move to the left.*
- *if s_i^t is to the left of the leftmost sensor of P^t, then s_{i+1}^t moves one unit to the left and s_i^t does not move to the right.*

Proof. We present an inductive proof. It is clear that $C^0(P) = P$ and therefore it is a pile. Assume the argument is true for $C^t(P)$. We show that it is also true for $C^{t+1}(P)$. We assume $C^{t+1}(P)$ is not a pile and show that all gaps in $C^{t+1}(P)$ have unit length and all sensors on both sides of the gaps behave as stated.

Take any gap between $s_i^{t+1}, s_{i+1}^{t+1} \in C^{t+1}(P)$ to the right of the rightmost sensor of P^{t+1}. Observe that $s_{i+1}^t \in C^t(P)$; if not, $s_{i+1}^{t+1} \notin C^{t+1}(P)$ either, since there is a gap between s_i^{t+1} and s_{i+1}^{t+1} . Since s_i^t is to the left of s_{i+1}^t, clearly $s_i^t \in C_t(P)$ as well. Since s_i^t and s_{i+1}^t are both in $C^t(P)$, one of the following cases holds at time t:

There is no gap between s_i^t and s_{i+1}^t:

Therefore s_i^t cannot move to the right, and even if there is a gap between s_{i-1}^t and s_i^t, by the inductive hypothesis, s_i^t does not move to the left. We conclude that s_i^t does not move. However, since there *is* a gap between s_i^{t+1} and s_{i+1}^{t+1}, it must be that s_{i+1}^t moved one unit to the right, thereby creating the gap. Observe that this gap must be of length one, as needed for the induction. Since s_{i+1}^t moved to the right, s_{i+1}^{t+1} cannot move to the left, as needed. We now show that s_i^{t+1} moves to the right. If there is a gap between s_{i-1}^t and s_i^t, by the inductive hypothesis, s_{i-1}^t moves right and as already observed s_i^t does not move, so there is no gap between s_{i-1}^{t+1} and s_i^{t+1}. If there is no gap between s_{i-1}^t and s_i^t, by the inductive hypothesis, s_{i-1}^t does not move to the left and as already observed s_i^t does not move, so there is no gap between s_{i-1}^{t+1} and s_i^{t+1}. Since (a) there is no gap between s_{i-1}^{t+1} and $s_i t + 1$, (b) there is a gap between s_i^{t+1} and s_{i+1}^{t+1}, and (c) s_i^t does not move, according to the algorithm, s_i^{t+1} moves to the right as needed.

There is a gap of unit length between s_i^t and s_{i+1}^t:

By the inductive hypothesis, s_i^t moves one unit to the right and s_{i+1}^t does not move to the left. Since there is a gap between s_i^{t+1} and s_{i+1}^{t+1}, we conclude that both s_i^t and s_{i+1}^t move one unit to the right and the gap between s_i^{t+1} and s_{i+1}^{t+1} is of size one, as needed. Furthermore, since s_{i+1}^t moved to the right, s_{i+1}^{t+1} cannot move to the left, and since s_i^t moved to the right, and s_i^{t+1} has a gap on its right, s_i^{t+1} moves to the right, as needed.

We have shown that the gap between s_i^{t+1} and s_{i+1}^{t+1} is of length one, and the two sensors behave as stated. The proof of the case where s_i^{t+1} and s_{i+1}^{t+1} are to the left of the leftmost sensor of P^{t+1} is similar. This completes the proof by induction. □

Theorem 4.4. *Given any input of n sensors with enough sensors to cover the barrier ($n \geq L/2r$), Algorithm 4.2 terminates in at most $(4r+1)n$ steps with the whole barrier fully covered. Furthermore Algorithm 4.2 is asymptotically optimal.*

Proof. Let P denote the heavy pile ancestor of a heavy pile that exists at time $(4r+1)n$ (Lemma 4.2 guarantees the existence of at least one heavy pile at all times). We denote the heavy pile descendant of P at time t by P^t. First we show that the right endpoint of $C^{t+2}(P)$ is to the right of the right endpoint of $C^t(P)$. Let s_j^t be the rightmost sensor of $C^t(P)$ and assume it does not touch the endpoint of the barrier. Observe that even if there is a gap to the left of s_j^t, by Lemma 4.13, s_j^t cannot move to the left. We claim that if s_j^t does not move, then there is no gap between s_{j-1}^{t+1} and s_j^{t+1}. If there is no gap between s_{j-1}^t and s_j^t, and s_j^t does not move, then since by Lemma 4.13, s_{j-1}^t cannot move to the left, there is no gap between s_{j-1}^{t+1} and s_j^{t+1}. If instead there is a gap between s_{j-1}^t and s_j^t then by Lemma 4.13, this gap is of length one, and s_{j-1}^t moves to the right and closes the gap so that there is no gap between s_{j-1}^{t+1} and s_j^{t+1}. This proves the claim. Therefore either s_j^t does not move and there is no gap between s_{j-1}^{t+1} and s_j^{t+1} or s_j^t moves to the right. Now consider s_j^{t+1}: if there

is gap to its right, it moves to the right, and if not, s_j^{t+1} does not move, but clearly the rightmost sensor of $C^{t+1}(P)$ is a sensor s_k with $k > j$. In both cases, the right endpoint of $C^{t+2}(P)$ is at least one unit further to the right than that of $C^t(P)$. A similar argument can be made about the left endpoint of $C^{t+1}(P)$. We conclude that $C^{4rn}(P)$ must span the entire barrier.

If there is no gap in $C^{4rn}(P)$, then $C^{4rn}(P)$ is a pile and the algorithm terminates leaving the entire barrier covered. Assume instead that $C^{4rn}(P)$ contains some gaps. Let s_i^{4rn} and s_{i+1}^{4rn} be the sensors that surround the rightmost gap in $C^{4rn}(P)$ to the right of P^{4rn}. As shown in Lemma 4.13, s_i^{4rn} moves to the right while s_{i+1}^{4rn} does not move to the left. Furthermore, no sensor to the right of s_{i+1}^{4rn} can move since there are no gaps to the right of s_{i+1}^{4rn}. Thus the rightmost gap in $C^{4rn}(P)$ is consumed in step $4rn$ and the rightmost gap in $C^{4rn+1}(P)$ is between s_k and s_{k+1} where $k \leq i-1$. A similar argument holds for the leftmost gap in $C^{4rn}(P)$ to the left of P^{4rn}. Thus the distance between the rightmost and leftmost gap reduces by at least one sensor in every step, and after n steps, the algorithm terminates with the barrier completely covered.

To show that Algorithm 4.2 is asymptotically optimal take for example an input where all the sensors are piled up at distinct positions at the left end of the barrier and there is a gap of size $\Omega(n)$. Any algorithm that can only move sensors a constant distance in each step, including Algorithm 4.2 takes $\Omega(n)$ steps to terminate and therefore Algorithm 4.2 is asymptotically optimal. □

In the following we show that when the number of sensors is insufficient to cover the barrier, Algorithm 4.2 may not terminate.

Theorem 4.5. *Given any input of n sensors with $3 < n < \frac{L}{2r}$ and at least one NSP in the input arrangement, Algorithm 4.2 does not terminate.*

Proof. Assuming $n < \frac{L}{2r}$ first we show that if a sensor s_i^t with $1 < i < n$ moves, then

there exists a sensor s_j with $1 < j < n$ such that s_j^{t+a} moves for some a such that $1 \leq a \leq 2r+1$. Without loss of generality assume s_i^t moves to the right. According to Algorithm 4.2 if there is a gap between s_i^{t+1} and s_{i+1}^{t+1} then s_i^{t+1} moves to the right and the claim is proved. Now consider the case where there is no gap between s_i^{t+1} and s_{i+1}^{t+1}. Let P^{t+1} denote the pile containing s_i^{t+1} and let s_k^{t+1} and s_ℓ^{t+1} denote the leftmost and rightmost sensors of P^{t+1} respectively.

Case 1: $k > 1$**.** Since s_1 never moves to the right and there exists a gap on the left of s_k^{t+1} either s_{k-1}^{t+1} moves to the right or s_k^{t+1} moves to the left or s_k^{t+2} moves to the left.

Case 2: $\ell < n$**.** This case is symmetric to the previous case.

Case 3: $k = 1$ **and** $\ell = n$**.** Clearly, either s_1^{t+1} or s_n^{t+1} must move, and we claim that at least one of them will eventually detach from the pile. Since $n < \frac{L}{2r}$, therefore either $x_2^{t+1} > 3r$ or $x_{n-1}^{t+1} < L - 3r$. Assume $x_2^{t+1} > 3r$ and let $m = 2r - x_2^{t+1} + x_1^{t+1} < 2r$ (the proof of the other case is symmetric). Assuming no sensor s_j with $1 < j < n$ moves between time steps $t+1$ and $t+m+1$, at each time step $t' \in \{t+1, t+2, \ldots, t+m+1\}$ sensor $s_1^{t'}$ moves one unit to the left and therefore $x_1^{t+m+2} < x_2^{t+m+2} - 2r$ and according to Algorithm 4.2, sensor s_2^{t+m+2} moves to the left.

In all cases we showed that if $n < \frac{L}{2r}$ and any sensor s_i^t with $1 < i < n$ and $t < \infty$ moves then Algorithm 4.2 never terminates.

We only need to show that if $n < \frac{L}{2r}$ some sensor s_i^t with $1 < i < n$ and $t < \infty$ moves. Take time step $t = 0$. We know that the arrangement of sensors contains at least one NSP. Let P^0 denote an NSP at time $t = 0$ and let s_i^0 and s_j^0 denote the leftmost and rightmost sensors of P^0. If $i > 1$ then s_i^0, then s_i^0 moves to the left. Similarly if $j < n$ then s_j^0 moves to the right. Now consider the case where $i = 1$ and

$j = n$. Similar to Case 3 of the inductive step, some sensor $s_k^{t'}$ with $1 < k < n$ and $t' \leq 2r$ must move. □

Similar to Algorithm 4.1 we show that there exists an input arrangement where Algorithm 4.2 takes $\Omega(n)$ steps to terminate while an optimal algorithm needs only 1 step; see the example in Figure 4.10.

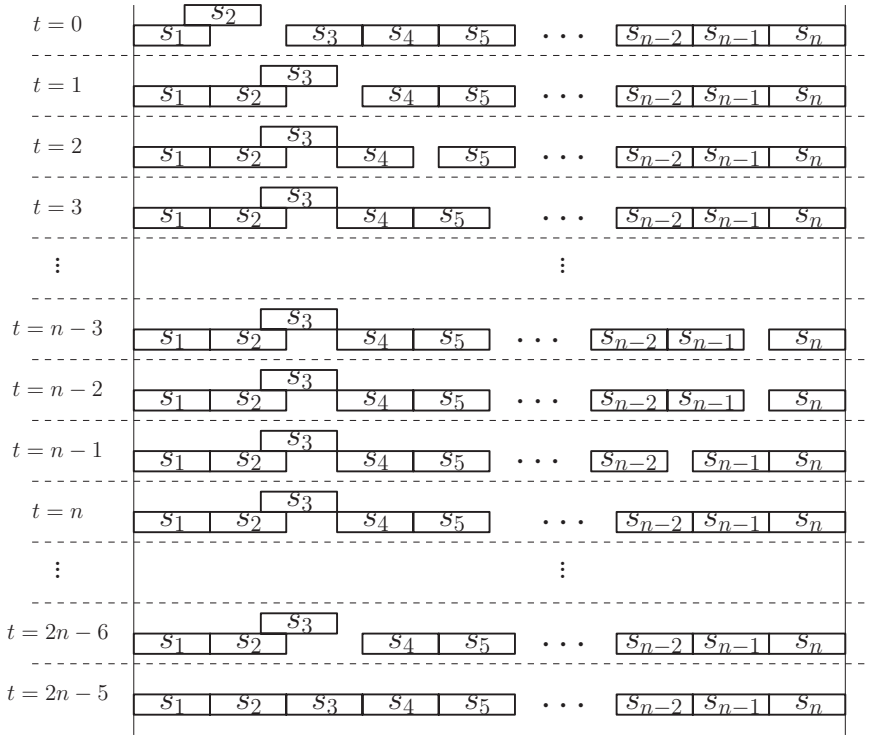


Figure 4.10: An example where Algorithm 4.2 takes $\Omega(n)$ time to terminate while an optimal algorithm terminates in one step.

4.5 Conclusions

In this chapter we studied the barrier coverage problem when the barrier consists of a single line segment, mobile sensors are located on the barrier and the sensing range of all sensors are the same. We presented two local distributed algorithms that achieve barrier coverage when there are enough sensors to cover the entire barrier.

Our first algorithm is oblivious and the worst case running time of this algorithm is $O(n^2)$. Our second algorithm uses two bits of memory to store the state of a sensor. We showed that our second distributed algorithm terminates after $O(n)$ steps and is asymptotically optimal.

Many open questions remain. Does there exist a linear-time oblivious algorithm for barrier coverage in our model? Would some generalization of our model, such as a larger visibility range, help? Is there an asynchronous algorithm for barrier coverage with restrictions of constant visibility and constant movement per unit step?

Chapter 5

Asynchronous Distributed Algorithms Using Relocatable Sensors

In this chapter we study distributed algorithms for the barrier coverage problem with semi-synchronous and asynchronous sensors. We again assume sensors with same limited visibility and sensing ranges are initially located on the barrier modelled as a line segment. First we show that if sensors do not share a common sense of orientation, and sensors are semi-synchronous or asynchronous then there exists no algorithm for the barrier coverage problem. Second in contrast to the non-existence results, if sensors have same orientation, we give an algorithm that terminates with the barrier fully covered if there are enough sensors on the barrier.

5.1 Computational Model and Preliminaries

As in Chapter 4, we model the barrier with a line segment of length L covering the interval $[0, L]$ on the x-axis. We assume all sensors are initially located on the barrier.

Furthermore all sensors have the same sensing range r and visibility range $2r$. Let s_i^t with $1 \leq i \leq n$ denote a sensor s_i at time t located at x_i^t. We assume sensors are labeled such that for every i and j with $1 \leq i < j \leq n$, we have $x_i^0 < x_j^0$. We emphasize that while labels and positions of sensors facilitate our proofs, they are not known to any of the sensors, which are completely anonymous, and completely identical.

We say s_i^t sees s_j^t on its right if and only if $0 < x_j^t - x_i^t \leq 2r$ and s_i^t sees s_k^t on its left if and only if $0 < x_i^t - x_k^t \leq 2r$. Furthermore we assume sensors are opaque: if there are multiple sensors on the right (left) and within visibility range of a given sensor, it only sees the leftmost (rightmost) sensor. Observe that with $2r$ visibility range, a sensor is able to detect whether there is a gap between its own and the next sensor's sensing areas.

Also every sensor has a conception of left and right, but there is no assumption for global agreement on this unless stated otherwise. We say sensors are *oriented* if and only if all sensors agree on a global left and right. Sensors are called *unoriented*, if the orientation of each sensor can change at any time independently of the other sensors. Obviously, when sensors are unoriented they do not necessary agree on left and right.

In addition, we assume there are two special sensors s_0 and s_{n+1} that are immobile, and are always located at $-r$ and $L+r$. While these special sensors do not require any sensing capabilities or visibility, the other sensors in the network cannot distinguish these special sensors from any other sensors. The entire set of sensors is denoted by $S = \{s_0, s_1, \ldots, s_n, s_{n+1}\}$.

Note that as in the previous chapter, in our figures, each sensor is represented by a rectangle which shows the interval that the sensor covers on the line barrier. Also for convenience, two sensors whose coverage lengths overlap are placed at different

levels in the illustration; however in our assumptions, all sensors have circular sensing area and are initially placed on the barrier and can only move on the barrier.

5.1.1 Synchronization Models of Sensors

As in the Chapter 4, each sensor works in a Look-Compute-Move cycle. In the *Look* phase a sensor looks at the arrangement of other sensors within its visibility range. Then in the *Compute* phase it calculates its next position based on the information it gathered in the Look phase and possibly its state. Finally, a sensor in its *Move* phase, moves toward the position it calculated in its Compute phase. Each sensor repeats the Look-Compute-Move cycle endlessly.

The literature on autonomous robots considers three major models for synchronization of sensors, which we use throughout this chapter.

Fully Synchronous Sensors (FSYNC): At any time all sensors are in the same phase (Look, Compute, or Move). Furthermore every sensor is active in every cycle.

Semi-Synchronous Sensors (SSYNC): Similar to FSYNC model, at any time all sensors are in the same phase. However not all sensors are necessarily active in every cycle. Every sensor, independently of the other sensors, may be inactive for an arbitrary but finite number of cycles.

Asynchronous Sensors (ASYNC): For every Look-Compute-Move cycle of a sensor, there may be an arbitrary but finite time delay between the phases as well as between the cycles. Observe that this implies that sensors can be deactivated at any time for an arbitrary time of finite duration.

As in the previous chapter, we define a *gap* as a maximal interval on $[0, L]$, where no point in this interval is within the sensing range of any sensor in the network. An

overlap is a maximal interval on $[0, L]$ such that every point in the interval is within the sensing range of more than one sensor in the network. We say a sensor s_i with $1 \leq i \leq n$ has a gap or overlap on its right if $x_{i+1} - x_i > 2r$ or $x_{i+1} - x_i < 2r$ respectively. Similarly a sensor s_i with $1 \leq i \leq n$ has a gap or overlap on its left if $x_i - x_{i-1} > 2r$ or $x_i - x_{i-1} < 2r$ respectively. Two consecutive sensors are called *attached* if there is neither a gap nor an overlap between them.

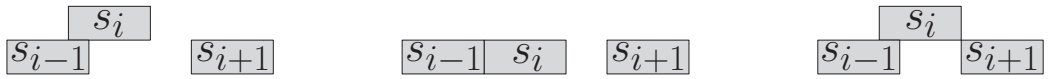
As in the previous chapter, we say an algorithm *terminates*, if there is a time t where no sensor moves at any time after t. Furthermore, if there are enough sensors initially on the barrier, an algorithm for barrier coverage must always terminate with the barrier being fully covered.

5.2 Impossibility of Algorithms for Barrier Coverage in SSYNC

In this section we consider the case where sensors are unoriented. We show that there is no algorithm for barrier coverage in the SSYNC and therefore in ASYNC models.

We give an adversary argument, by creating input arrangements and activation schedules that force any algorithm in the SSYNC model to either not terminate, or terminate without coverage. All movements will be assumed to be rigid; a sensor can always reach the destination it has computed. We focus on three types of sensors (a) sensors that have an overlap on one side, and a gap on the other side, (b) sensors that are attached to the next sensor on one side and a gap on the other side and (c) sensors that have an overlap on one side and are attached to the next sensor on the other side (see Figure 5.1). Any algorithm for barrier coverage must specify rules for movement in each of these situations. Note that with $2r$ visibility range, sensors can only determine whether there exists a gap with a neighboring sensor but cannot

determine anything about the length of such a gap. Thus, the magnitude of the movement of a sensor can only be a function of an overlap, if any, with a neighboring sensor, and cannot be a function of the length of an adjacent gap. We show that there exist arrangements and activation schedules for the sensors that defeat all possible combinations of these rules.



(a) (b) (c)

Figure 5.1: The three types of sensors under consideration

First we study the behavior of a sensor s_i with $1 \leq i \leq n$ that has an overlap of e with the sensor on its left, and has a gap on its right, as in Figure 5.1(a). We show that such a sensor must move right; if the gap is at least as big as the overlap, the sensor must eventually move so as to exactly remove the overlap, and if the gap is smaller than the overlap, the sensor must move at least enough distance to remove the gap.

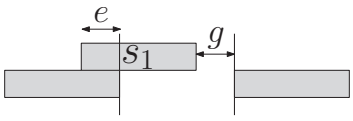


Figure 5.2: Arrangement for proof of Lemma 5.1

Lemma 5.1. *Consider an algorithm $\mathcal{A}$ for barrier coverage in SSYNC model and a sensor s_i^t with $dist(s_{i-1}^t, s_i^t) = 2r - e$ and $dist(s_i^t, s_{i+1}^t) = 2r + g$, with $e, g > 0$. If s_{i-1} and s_{i+1} are deactivated and only s_i is activated, there exists a time step $t' > t$ such that:*

(a) $x_i^{t'} = x_i^t + e$ if $g \geq e$ and

(b) $x_i^t + g \leq x_i^{t'} \leq x_i^t + e$ if $g < e$.

Proof. First we observe that the sensor s_i must eventually move at least distance $min(g, e)$ to the right. If not, the algorithm $\mathcal{A}$ does not terminate with barrier coverage on the arrangement shown in Figure 5.2, since s_1 is the only sensor that can move in the arrangement. Next we show that $x_i^{t'} \leq x_i^t + e$ for some $t' > t$. For the sake of contradiction, assume that there is a value of overlap e, such that according to $\mathcal{A}$, sensor s_i moves more than e; that is s_i moves $e + a$ to the right, with $a > 0$. Then we can construct an activation schedule such that $\mathcal{A}$ never terminates on the input shown in Figure 5.3: A single sensor is activated in each step. The sensors s_p to s_1 are activated in consecutive steps, followed by the sensors s_{p+1} to s_{p+q-1}. Then the sequence is reversed. It is easy to verify that at the end of the activation schedule, the initial arrangement is repeated again. The schedule can be repeated *ad infinitum*, forcing non-termination of the algorithm. It follows that in the case when $g \geq e$, whatever the overlap e with s_{i-1}, we can force s_i to move exactly e to the right.

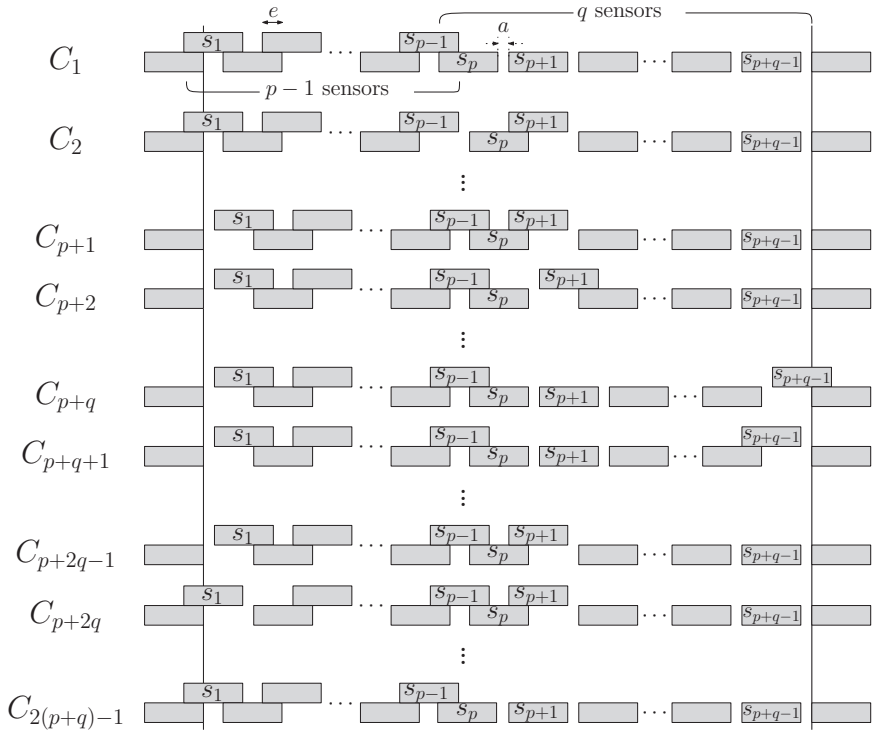


Figure 5.3: Arrangement for proof of Lemma 5.1 ($p, q \in \mathbb{N}$ are chosen so that $p\,e = q\,a$)

□

Next we consider the behavior of a sensor s_i that is attached to its neighbor on its left, and has a gap on its right as in Figure 5.1(b). We activate s_i and keep s_{i-1} and s_{i+1} deactivated. If s_i moves left, it creates an overlap with s_{i-1} and by Lemma 5.1(a), it will eventually move to the right to remove that overlap, and return to the same position. Alternatively, s_i may not move at all, or may move to the right. If it moves to the right, since it does not know the distance of the gap with s_{i+1} and has no overlap with s_{i-1}, it can only move a fixed constant distance, say b. The lemma below is a consequence of the preceding discussion.

Lemma 5.2. *Consider an algorithm $\mathcal{A}$ for barrier coverage and a sensor s_i with $dist(s_{i-1}^t, s_i^t) = 2r$ and $dist(s_i^t, s_{i+1}^t) > 2r$. If s_{i-1} and s_{i+1} are both kept deactivated and s_i is activated, there exists a time $t' > t$ such that $x_i^{t'} = x_i^t + h$ with $h \geq 0$.*

Finally, we consider the behavior of a sensor s_i that has an overlap e with s_{i-1} and is attached to sensor s_{i+1}, as shown in Figure 5.1(c). As before, we activate only s_i and keep both s_{i-1} and s_{i+1} deactivated. If s_i moves left, it creates a gap with s_{i+1}. By Lemma 5.1(b), s_i must eventually move right, either returning to its initial position, or moving further right. If it moves right by more than the value of the overlap, then it creates a gap to its left, and once again by Lemma 5.1(b) , it must move back left until the gap is removed. If for all values of the overlap, s_i makes a move to the right that does not eliminate the overlap, then we show below that the algorithm cannot achieve barrier coverage, leading to the conclusion that there must exist some value of overlap such that such a sensor will either not move, or move to exactly eliminate the overlap.

Lemma 5.3. *Consider an algorithm $\mathcal{A}$ for barrier coverage. There exists an overlap c with $0 < c < 2r$ such that for any sensor s_i with $dist(s_{i-1}^t, s_i^t) = 2r - c$ and*

$dist(s_i^t, s_{i+1}^t) = 2r$, if s_i is the only one of $\{s, s_{i-1}, s_{i+1}\}$ to be activated, there exists a time step t' with $t' > t$ such that either $x_i^{t'} = x_i^t + c$ (s_i moves right to exactly eliminate the overlap) or $x_i^{t'} = x_i^t$ (s_i returns to the same position).

Proof. Assume the contrary. By the discussion preceding the lemma, we can conclude that for any overlap e, there exists a time step t' such that $x_i^{t'} = x_i^t + d$ with $0 < d < e$. Consider the arrangement of sensors shown in Figure 5.4. We first activate s_1 until it moves distance d to the right. By assumption, there remains an overlap of $e - d$ between s_0 and s_1, and now there is an overlap of d between s_1 and s_2. We now keep s_1 deactivated, and activate s_2. Lemma 5.1 implies that sensor s_2 eventually moves exactly d to the right and eliminates the overlap completely. Observe that at this point, the arrangement repeats with only a different value of overlap. The new value of the overlap between s_0 and s_1 is strictly greater than zero and and the distance between s_1 and s_2 is exactly $2r$. This activation schedule can be repeated *ad infinitum*, and algorithm $\mathcal{A}$ never terminates with barrier coverage. □

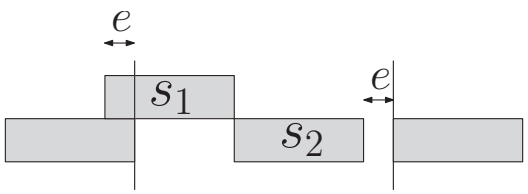


Figure 5.4: Arrangement for proof of Lemma 5.3

Now we proceed to prove our main result:

Theorem 5.1. *Let $s_1, s_2, \ldots, s_n$ be n sensors with sensing range r initially placed at arbitrary positions on a line segment. If the sensors are unoriented and visibility radius $2r$, there is no algorithm for barrier coverage in the SSYNC model.*

Proof. Consider the arrangement of sensors shown in Figure 5.5 with c chosen as in Lemma 5.3. If the value of h as specified in Lemma 5.2 is zero, then choose b to be

of any arbitrary non-zero length, otherwise, choose $b = h$. We create an activation schedule with four phases with a different set of sensors being activated in each phase, such that the sensors return to arrangement C_1 at the end of each phase. At each phase we only activate a subset of sensors and all other sensors are kept deactivated. We first activate only the set of sensors $\{s_1, s_3, \ldots, s_{2p-1}\}$. By Lemma 5.2, there is a future time step when either these sensors are in the same position (if $h = 0$), or they move distance $b = h$ to the right to yield arrangement C_2. In the second case, since sensors are unoriented, they will subsequently return to arrangement C_1. Next we activate only the set of sensors $\{s_2, s_4, \ldots, s_{2p}\}$. Using the same logic, they must return to the same arrangement, possibly via arrangement C_3. In the third phase, we activate only the sensors $\{s_{2p+1}, s_{2p+3}, \ldots, s_{2p+2q-1}\}$. By Lemma 5.3, there is a future time when either these sensors return to arrangement C_1, or they have moved left by a distance c to reach arrangement C_4. In the second case, they will eventually return to arrangement C_1. In the fourth phase, we activate only the set of sensors $\{s_{2p+2}, s_{2p+4}, \ldots, s_{2(p+q)}\}$. Using the same logic, they will return to arrangement C_1, possibly via arrangement C_5. Observe that all sensors have been activated at least once during the schedule. By repeating the above schedule *ad infinitum*, we can force sensors to repeatedly return to the arrangement C_1, thus completing the proof. □

Since an adversary in the ASYNC model has at least the power it has in the SSYNC model, obviously our non-existence results also hold for the ASYNC model.

5.3 Sensors with Orientation

In the previous section we showed that no algorithm for barrier coverage exists when sensors are unoriented. In this section, we give an algorithm for barrier coverage for oblivious *oriented* sensors in the ASYNC model. We assume sensors have *limited*

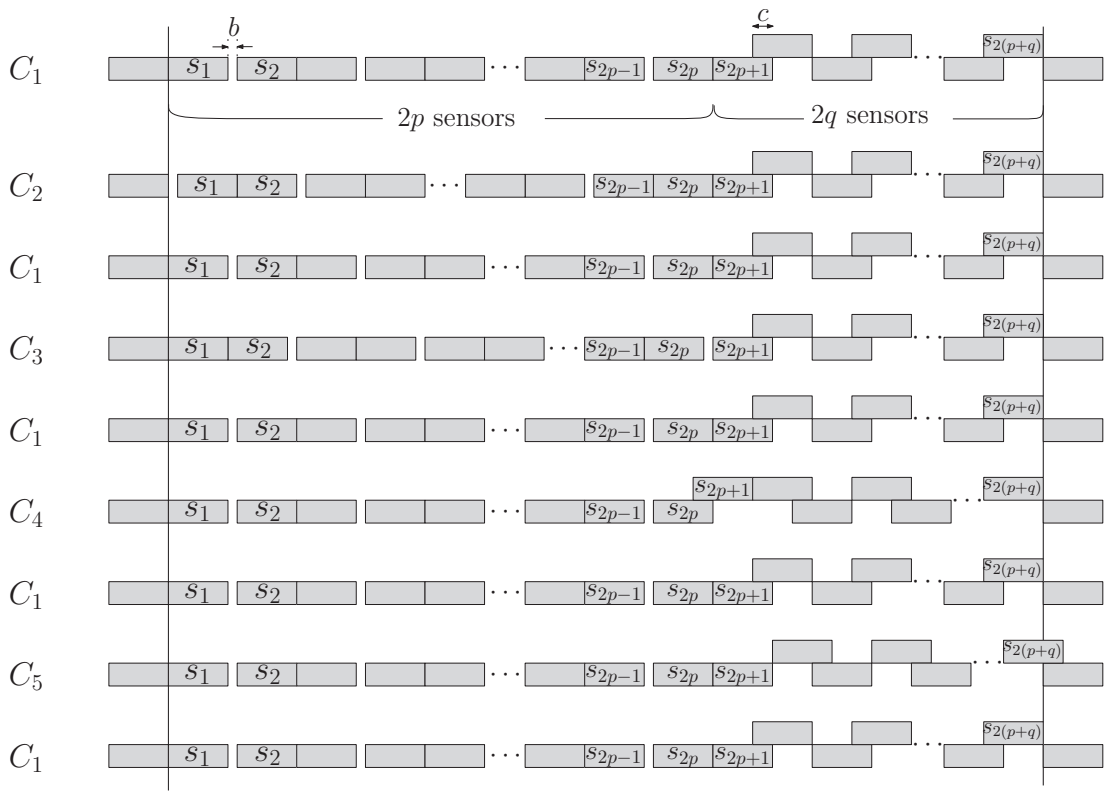


Figure 5.5: Arrangement for proof of Theorem 5.1 ($p, q \in \mathbb{N}$ are chosen so that $p\,b = q\,c$

mobility, in particular, in any move, a sensor can move at most distance r. As per the standard *rigidity* assumption in the literature, we assume that there is an (arbitrarily small) fixed constant δ such that if the destination point is at most δ away, the sensor will reach it. Otherwise, it moves at least δ towards the destination. This assumption is necessary for termination of any algorithm for autonomous mobile robots, otherwise there will be no guarantee that any robot ever reaches any chosen destination point. Our algorithm for barrier coverage in this model is given below:

Algorithm 5.1 Oblivious algorithm for barrier coverage with ASYNC oriented sensors

Every sensor $s_i \in S$ in its Look-Compute-Move cycle does the following:
$\epsilon < r$ is a fixed positive (arbitrarily small) constant
if s_{i-1} is not visible to s_i (there is a gap to its left) **then**
  s_i moves distance r to the left.
else
  $a := 2r - dist(s_{i-1}, s_i)$ (amount of overlap with previous sensor's range)
  if $dist(s_i, s_{i+1}) \geq 2r$ (no overlap from right) and $a > 0$ **then**
    s_i moves distance $min(r - \epsilon, a)$ to its right.
  else
    do nothing
  end if
end if

We proceed to prove the correctness of the algorithm. Note that the concept of time used in this section is more general than that used in the previous sections. In FSYNC and SSYNC models, we had the concept of a global time step, whereas in the ASYNC model sensors may be at different phases at any given time. Therefore when we talk about a time t in the ASYNC model, every sensor may be in a different phase of its Look-Compute-Move cycle at that time t. For example, if we consider a time t that is the beginning of the Look phase of a particular sensor s, we cannot conclude anything about the state of other sensors at that time t. We first show that the algorithm above is *collision-free* and *order-preserving*. In the context of autonomous mobile robots, a collision happens if two distinct robots move to exactly the same position. Since

robots are identical with no id, from the time a collision of two robots happens, they cannot be distinguished and will behave exactly the same if they have the same activation schedule. Therefore a collision is fatal for a barrier coverage algorithm, as for other distributed algorithms for autonomous mobile robots, and must be avoided.

Lemma 5.4. *Algorithm 5.1 is a collision-free and order-preserving protocol.*

Proof. Note that for any two sensors, in order for them to change order, there must exist a time where they collide. Therefore we only need to show that the algorithm is collision-free. For the sake of a contradiction assume that a collision happens. Let t denote the first time that two sensors collide and let s_i and s_{i+1} denote the two sensors that collide. Let t' with $0 \leq t' < t$ denote the last time where either s_i or s_{i+1} performed a Look before the collision and this Look lead to a Compute-Move of a non-zero distance. The existence of t' is essential for any collision. Since t is the first time that two sensors collide, therefore $x_i^{t'} < x_{i+1}^{t'}$. Based on the positions of s_i and s_{i+1} at times t and t', one of the following cases holds:

$x_i^{t'} \leq x_i^t = x_{i+1}^t \leq x_{i+1}^{t'}$**:** First consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) \geq 2r$. Since after this time s_i or s_{i+1} will perform at most one Move each therefore $x_i^t \leq x_i^{t'} + r - \epsilon$ and $x_{i+1}^t \geq x_{i+1}^{t'} - r$ and therefore $x_i^t < x_{i+1}^t$ which contradicts the assumption that $x_i^t = x_{i+1}^t$. Second consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) < 2r$. Therefore based on the Look at t', neither s_i nor s_{i+1} computes a Move of non-zero distance towards x_i^t that contradicts the assumption about t'.

$x_i^{t'} < x_{i+1}^{t'} < x_i^t = x_{i+1}^t$**:** First consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) \geq 2r$. Therefore $x_i^{t'} \leq x_i^t - 2r$. However since at any Move to right, a sensor moves at most $r - \epsilon$ therefore $x_i^t < x_i^{t'} + r$ which contradicts $x_i^t \geq x_{i+1}^{t'}$. Second consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) < 2r$. If s_i is in a Look phase at time t' it does not move right which contradicts $x_i^{t'} < x_i^t$. Therefore the last time s_i is in a Look

phase, that leads to a Compute-Move of a non-zero distance, is before t'. Let t'' with $t'' < t'$ denote the last Look of s_i that lead to a Compute-Move of a non-zero distance before the collision. It is easy to see that $x_i^t \leq x_i^{t''} + r - \epsilon$ and $x_{i+1}^{t''} \geq x_i^{t''} + 2r$. Also since t'' is the last Look of s_i before the collision, $x_i^\tau \geq x_i^{t''}$ at any time τ with $t'' \leq \tau \leq t'$. Now look at the location of s_{i+1} after t''. Since s_{i+1} moves left at most r in a Look-Compute-Move cycle only if there is a gap on its left when it performs a Look, it is easy to see that $x_{i+1}^{t'} > x_i^{t''} + r > x_i^t$ which contradicts our assumption.

$x_i^t = x_{i+1}^t < x_i^{t'} < x_{i+1}^{t'}$**:** It is very similar to the previous case: First consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) \geq 2r$. Therefore $x_{i+1}^{t'} \geq x_{i+1}^t + 2r$. However since at any Move to left, a sensor moves at most r therefore $x_{i+1}^t \geq x_{i+1}^{t'} - r$ which contradicts $x_{i+1}^t < x_i^{t'}$. Second consider the case where $dist(s_i^{t'}, s_{i+1}^{t'}) < 2r$. If s_{i+1} is in a Look phase at time t' it does not move left which contradicts $x_{i+1}^t < x_{i+1}^{t'}$. Therefore the last time s_{i+1} is in a Look phase, that leads to a Compute-Move of a non-zero distance, is before t'. Let t'' with $t'' < t'$ denote the last Look of s_{i+1} that lead to a Compute-Move of a non-zero distance before the collision . It is easy to see that $x_{i+1}^t \geq x_{i+1}^{t''} - r$ and $x_i^{t''} < x_{i+1}^{t''} - 2r$. Also since t'' is the last Look of s_{i+1} before the collision, $x_{i+1}^\tau \leq x_{i+1}^{t''}$ at any time τ with $t'' \leq \tau \leq t'$. Now look at location of s_i after t''. Since s_i moves right at most $r - \epsilon$ in a each Look-Compute-Move cycle given that there is no overlap on its right when it performs a Look, it is easy to see that $x_i^{t'} < x_{i+1}^{t''} - r \leq x_{i+1}^t = x_i^t$ which contradicts our assumption.

□

Next we show that running Algorithm 5.1, there is a time after which no sensor moves left, and after this time, the sensors provide contiguous coverage of $[0, x_n + r]$.

Lemma 5.5. *Running Algorithm 5.1, for every sensor $s_i \in S$ with $1 \leq i \leq n$ there is a time t_i such that s_i never moves left at any time after t_i. Furthermore, there is no coverage gap between s_0 and s_i at any time after t_i.*

Proof. We prove the claim inductively. Clearly it is true for s_0. Suppose there is a time t_i such that s_i never moves left at any time after t_i, and there is no gap between s_0 and s_i at any time after t_i. Consider any Look of s_{i+1} after time t_i. If there is a gap between s_i and s_{i+1}, then s_{i+1} moves at least δ towards s_i. Let t_{i+1} with $t_{i+1} \geq t_i$ be the first time that s_{i+1} performs a Look and there is no gap between s_i and s_{i+1}. If at this time there is an overlap with s_i, then s_{i+1} moves right, but observe that this Move can never create a gap between s_i and s_{i+1} since s_i does not move left by the inductive assumption, and s_{i+1} moves right by at most the amount of the overlap. It follows that after time t_{i+1}, the sensor s_{i+1} will never move left, and furthermore, there is no gap between s_0 and s_{i+1}. □

The next lemma shows that if there is an overlap between two sensors s_i and s_{i+1}, there comes a time when either there is an overlap between every two consecutive sensors on the right of s_{i-1} or s_{i+1} moves right and removes the overlap completely.

Lemma 5.6. *Running Algorithm 5.1, for every sensor $s_i \in S$ with $0 \leq i < n$ if there is an overlap between s_i and s_{i+1} at any time t with $t \geq t_n$, then there is a time t' with $t' \geq t$ where:*

(a) s_i and s_{i+1} are attached, or

(b) there is an overlap between every two consecutive sensors in $\{s_i, s_{i+1}, \ldots, s_{n+1}\}$.

Proof. We give an inductive proof. First assume that there is an overlap e between s_{n-1} and s_n at time t with $t \geq t_n$. Lemma 5.5 implies that s_{n-1} never moves left. Also from the algorithm it is clear that s_{n-1} does not move right as long as there is an

overlap between s_{n-1} and s_n. Therefore the amount of the overlap between s_{n-1} and s_n does not increase before first decreasing to zero (attached position). Consider the first time s_n is activated: either there is an overlap between s_n and s_{n+1} or s_n moves right and reduces the overlap by at least $min(e, \delta)$. Therefore there is a time t' with $t' \geq t$ where either there is an overlap between s_n and s_{n+1} (case (b)) or s_{n-1} and s_n become attached (case (a)). Second we show that if the lemma statement holds for all sensors in $\{s_i, s_{i+1}, \ldots, s_{n-1}\}$ for some $0 < i \leq n-1$, then it also holds for s_{i-1}. Imagine there is an overlap e between s_{i-1} and s_i at some time t with $t \geq t_n$ and consider sensors s_i and s_{i+1} at time t. One of the following cases holds:

(i) s_i and s_{i+1} are attached: Lemma 5.5 implies that s_i and s_{i+1} stay attached until sensor s_i is activated. At this time, according to Algorithm 5.1, sensor s_i moves right to reduces the overlap by at least $min(e, \delta)$.

(ii) there is an overlap between s_i and s_{i+1}: the inductive hypothesis implies that there is a time t_2 with $t_2 \geq t$ where either there is an overlap between every two consecutive sensors in $\{s_i, s_{i+1}, \ldots, s_{n+1}\}$ which implies (b) for s_{i-1} or sensor s_i and s_{i+1} become attached which is similar to (i).

In both cases after finite time either (b) holds true for s_{i-1} or the overlap between s_{i-1} and s_i goes down by $min(e, \delta)$. Therefore it can be seen that there exists a time t' with $t' \geq t$ where either s_{i-1} and s_i are attached or there is an overlap between every two consecutive sensors in $\{s_{i-1}, s_i, \ldots, s_{n+1}\}$ and the lemma follows. □

Next we show that regardless of the number of sensors, Algorithm 5.1 terminates:

Theorem 5.2. *Let $s_1, s_2, \ldots, s_n$ be n sensors with sensing range r initially placed at arbitrary positions on a line segment with length L. If the sensors have the same orientation and visibility radius of $2r$, Algorithm 5.1 always terminates in the ASYNC*

model. Furthermore, if $2rn \geq L$ the algorithm terminates with the barrier being fully covered.

Proof. Lemma 5.5 implies that there is a time t_n where there is no gap between sensors in $\{s_0, s_1, \ldots, s_n\}$. Consider the two sensors s_0 and s_1: Lemma 5.5 and Lemma 5.6 imply that there exists at time t'_0 with $t'_0 \geq t_n$ where either:

(a) s_0 and s_1 are attached: since no sensor moves left they stay attached at any time after t'_0, or

(b) there is an overlap between every two consecutive sensors in $\{s_0, s_1, \ldots, s_{n+1}\}$: therefore the barrier is fully covered and no sensor moves at any time after t'_0 and the algorithm terminates at time t'_0.

Inductively it can be seen that for any sensor s_i with $1 \leq i \leq n$ either s_i and s_{i+1} become attached at time t'_i or the algorithm terminates at time t'_i.

Regardless of the number of sensors, at the time t'_n, either every two consecutive sensors are attached or there exists i with $0 \leq i \leq n$ where every two consecutive sensors in $\{s_0, s_1, \ldots, s_i\}$ are attached and every two consecutive sensors in $\{s_i, s_{i+1}, \ldots, s_{n+1}\}$ have an overlap. In both cases Algorithm 5.1 terminates. Furthermore if there are enough sensors to cover the entire barrier, both cases imply that the barrier is fully covered. □

5.4 Conclusions

In this chapter we considered distributed local algorithms for barrier coverage with semi-synchronous and asynchronous sensors. We assumed that sensors have limited visibility radius of $2r$ where r is the sensing radius of sensors. We showed that if sensors are semi-synchronous and do not share a global orientation (unoriented

sensors), no algorithm exists for the problem. Obviously the results also hold for asynchronous sensors. In contrast, for the case where sensors have same orientation, we gave an distributed algorithm that achieves barrier coverage even if all sensors are asynchronous. Finally, the existence of algorithms for sensors that each have an orientation that does not change through time, but do not necessarily agree on a global orientation, remains open.

Chapter 6

Multi-Round Random Deployment Using Stationary Sensors[1]

In this chapter, we study the problem of barrier coverage when stationary sensors are deployed randomly on the barrier. We define two natural classes of sensor deployment strategies: *complete* and *partial* deployments. In complete strategies, sensors are deployed over the entire border in every round, while in partial strategies, the deployment may be over only part of the border. For any complete deployment strategy, we calculate the probability that a border is (k, w)-covered as a function of parameters such as the length of the border, the width of the intruder, and the sensing range and density of deployed sensors. Simulation results show that our analysis is tighter than that provided in [LZS+11]. Next we propose and study two specific deployment strategies called Fixed-Density Complete Deployment, and Fixed-Density Partial Deployment. For each strategy, we calculate the expected number of deployment rounds and expected total number of sensors employed by the strategy to achieve (k, w)-coverage. The number of rounds and total number of sensors are indeed the two main factors determining the total cost of deployment. We propose a model

[1]Results of this chapter are also published in [ENO13]

of total deployment cost and for each of the two studied strategies, we calculate the density of deployment that minimizes the expected total deployment cost.

The rest of this chapter is organized as follows. In Section 6.1 we define our model of the network. We study complete deployments and give a lower bound estimate of the probability that the border is (k, w)-covered after each round in Section 6.2. In Section 6.3 we propose two specific deployment strategies and and analyze their expected cost. Section 6.4 contains our simulation results. Finally in Section 6.5 we conclude our chapter and present directions for future work.

6.1 Network Model

We model a border of length l as an interval starting at 0 and ending at l. For simplicity, we assume that there are initially k sensors deployed at point 0 and at point l, the beginning and end of the border respectively. All other sensors are deployed randomly on the border, possibly in multiple rounds. We model an intruder with a line segment of width w. The position of the intruder is the center of this line segment.

A sensor can detect intruders within its sensing range r. The sensing can be defined more formally as follows: Let $\tau = 2r + w$. A sensor at $x \in \Re$ can detect an intruder of width w at position p iff $|p - x| \leq \frac{\tau}{2}$ or equivalently $p \in [x - \frac{\tau}{2}, x + \frac{\tau}{2}]$. We now give the definition of (k, w)-coverage:

Definition 6.1. A point p is called (k, w)-covered if any intruder of width greater or equal to w positioned at p is detected by at least k distinct sensors. A border modeled by an interval $[0, l]$ is (k, w)-covered if and only if every point on $[0, l]$ is (k, w)-covered.

We assume sensors are dispersed on some parts of the border in multiple rounds.

We define a *deployment interval* as follows:

Definition 6.2. In a given round of deployment, an interval $D \subseteq [0, l]$ is called a *deployment interval* if and only if for every subinterval $D' \subseteq D$, the probability that a sensor is deployed on D' is greater than zero and D is maximal.

A deployment strategy is called *complete* if in each deployment round, the entire border $[0, l]$ is a deployment interval and is called *partial* otherwise. In this chapter we assume that sensors are always dispersed over deployment intervals according to a Poisson distribution. We call the parameter of the Poisson distribution the *density* of deployment. Since the parameter of the Poisson distribution according to which sensors are dispersed can be different in every round, observe that there is an infinite number of complete deployment strategies possible. Partial deployment strategies can vary not only in the above parameters, but also in the deployment intervals used in every round.

6.2 Probability of (k, w)-coverage for Complete deployment Strategies

In this section we calculate the probability that a border is (k, w)-covered if sensors are dispersed according to a Poisson distribution over the entire border in R rounds with parameters $\lambda_1, \lambda_2, \ldots, \lambda_R$ respectively. Clearly, for complete strategies, if sensors are dispersed in R rounds the distribution of the sensor positions is the same as if they were deployed in a single round with parameter λ where $\lambda = \sum_{i=1}^{R} \lambda_i$

Consider a point $p \in [0, l]$ and the interval $I = [p - \frac{\tau}{2}, p + \frac{\tau}{2}]$. It can be easily seen that a sensor at $x \in \Re$ covers p if and only if $x \in I$. It follows that a border is (k, w)-covered if and only if for every point p on the border there are at least k distinct sensors in $[p - \frac{\tau}{2}, p + \frac{\tau}{2}]$.

Let $C_{p_1}^{p_2}$ denote the event that every point p with $p_1 + \frac{\tau}{2} \leq p \leq p_2 - \frac{\tau}{2}$ is (k, w)-covered. Therefore assuming there are k sensors at 0 and k sensors at l, the probability that a border with length l is (k, w)-covered can be denoted as:

$$Pr\big[[0, l] \text{ is } (k, w)\text{-covered}\big] = Pr\big[C_0^l\big]$$

Let x_i and x_j denote the positions of the i^{th} and j^{th} sensors on the border respectively. In the following we give a recursive approach for calculating $C_{x_i}^{x_j}$.

Lemma 6.1. *Let $x_i \leq x_{i+1} \leq x_{i+2} \leq \ldots \leq x_j$ with $j - i + 1 > k$ denote the positions of $j - i + 1$ consecutive sensors on the border. Then:*

$$C_{x_i}^{x_j} = (x_{i+k} - x_i \leq \tau) \cap C_{x_{i+1}}^{x_j}$$

Proof. It can be seen from the definition of $C_{x_i}^{x_j}$ that:

$$C_{x_i}^{x_j} = C_{x_i}^{x_{i+1}+\tau} \cap C_{x_{i+1}}^{x_j}$$

Therefore we just need to show that the event $C_{x_i}^{x_{i+1}+\tau}$ is equal to the event $(x_{i+k} - x_i \leq \tau)$. First if $x_{i+k} - x_i > \tau$ then there are fewer than k sensors in $[x_i, x_i + \tau]$ and therefore $x_i + \frac{\tau}{2}$ is not (k, w)-covered and $C_{x_i}^{x_{i+1}+\tau}$ does not occur. Second if $C_{x_i}^{x_{i+1}+\tau}$ does not occur, then there must exist a point p such that $p \in [x_i + \frac{\tau}{2}, x_{i+1} + \frac{\tau}{2}]$ and p is not k-covered. Note that $p \leq x_{i+1} + \frac{\tau}{2}$ and therefore $(x_i, p - \frac{\tau}{2})$ contains no sensor. Also since p is not (k, w)-covered there are at most $k - 1$ sensors in $[p - \frac{\tau}{2}, p + \frac{\tau}{2}]$. Therefore there are at most $k - 1$ sensors in $(x_i, p + \frac{\tau}{2}]$ and

$$x_{i+k} - x_i > p + \frac{\tau}{2} - x_i \geq \tau$$

Therefore $C_{x_i}^{x_{i+1}+\tau}$ is the same as the event $x_{i+k} - x_i \leq \tau$ and the statement of the

lemma is proved. □

Using symmetric arguments, we have the following:

Corollary 6.1.

$$C_{x_i}^{x_j} = (x_j - x_{j-k} \leq \tau) \cap C_{x_i}^{x_{j-1}} \tag{6.1}$$

It follows that for $j - i + 1 > k$:

$$\begin{aligned} Pr\big[C_{x_i}^{x_j}\big] &= Pr\big[(x_{i+k} - x_i \leq \tau) \cap C_{x_{i+1}}^{x_j}\big] \\ &= Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big]\ Pr\big[C_{x_{i+1}}^{x_j}\big] \end{aligned} \tag{6.2}$$

The next lemma gives a lower bound on $Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big]$:

Lemma 6.2. *Let $x_i \leq x_{i+1} \leq x_{i+2} \leq \ldots \leq x_j$ with $j - i + 1 > k$ denote the positions of $j-i+1$ consecutive sensors on the border. Assume the distance between consecutive sensors are iid exponential random variables. Then:*

$$Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big] \geq Pr\big[x_{i+k} - x_i \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big]$$

Proof. Let A, B and C denote three independent random variables with $A \geq 0$ and also let E denote any event. It can be seen that:

$$Pr\big[A \leq B | E \cap (A \leq C)\big] \geq Pr\big[A \leq B | E\big] \tag{6.3}$$

We use this fact to prove our lemma. Corollary 6.1 implies that

$$C_{x_{i+1}}^{x_j} = C_{x_{i+1}}^{x_{j-1}} \cap (x_j - x_{j-k} \leq \tau)$$

First we show that for any $j - i > k + 1$:

$$Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big] \geq Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{j-1}}\big] \tag{6.4}$$

Consider the case where $j - i \geq 2k$ and hence $j - k \geq i + k$. Since the distances between every two consecutive sensors are independent:

$$\begin{aligned} Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big] &= Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{j-1}} \cap (x_j - x_{j-k} \leq \tau)\big] \\ &= Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{j-1}}\big] \end{aligned} \tag{6.5}$$

Now consider the case where $k + 1 < j - i < 2k$. Let $A = x_{i+k} - x_{j-k}$, $B = \tau - (x_{j-k} - x_i)$, $C = \tau - (x_j - x_{i+k})$ and $E = C_{x_{i+1}}^{x_{j-1}}$. Note that A, B and C are independent, and $A \geq 0$ therefore (6.3) implies:

$$\begin{aligned} Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big] &= Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{j-1}} \cap (x_j - x_{j-k} \leq \tau)\big] \\ &\geq Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{j-1}}\big] \end{aligned} \tag{6.6}$$

In both cases (6.4) holds true.

By repeatedly applying the argument in (6.4) we conclude for any integer $j - i > k + 1$:

$$Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_j}\big] \geq Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{i+k+1}}\big] \tag{6.7}$$

Second we show that $Pr\big[x_{i+k} - p_{i+1} \leq \tau | C_{x_{i+1}}^{x_{i+k+1}}\big] \geq Pr\big[x_{i+k} - p_{i+1} \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big]$. Note that $C_{x_{i+1}}^{x_{i+k+1}} = (x_{i+k+1} - x_{i+1} \leq \tau)$. Let $E = (x_{i+k} - x_{i+1} \leq \tau)$, $A = x_{i+k} - x_{i+1}$, $B = \tau - (x_{i+1} - x_i)$, and $C = \tau - (x_{i+k+1} - x_{i+k})$. Note that A, B and C are independent, and $A \geq 0$ therefore (6.3) implies:

$$Pr\big[x_{i+k} - x_i \leq \tau | C_{x_{i+1}}^{x_{i+k+1}}\big] \geq Pr\big[x_{i+k} - x_i \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big] \tag{6.8}$$

The lemma follows from (6.7) and (6.8). □

From Eq. (6.2) and Lemma 6.2 we get for any $j - i + 1 > k$:

$$Pr\big[C_{x_i}^{x_j}\big] \geq Pr\big[x_{i+k} - x_i \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big] Pr[C_{x_{i+1}}^{x_j}] \tag{6.9}$$

To calculate $Pr\big[x_{i+k} - x_i \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big]$ note that $x_{i+k} - x_{i+1}$ is the sum of $k-1$ i.i.d. exponential random variables with parameter λ. Therefore $x_{i+k} - x_{i+1}$ has an Erlang distribution with shape parameter $k-1$ and rate λ and:

$$\begin{aligned} &Pr\big[x_{i+k} - x_i \leq \tau | x_{i+k} - x_{i+1} \leq \tau\big] \\ &\qquad = Pr\big[x_{i+k} - x_{i+1} \leq \tau - (x_{i+1} - x_i) | x_{i+k} - x_{i+1} \leq \tau\big] \\ &\qquad = \frac{\gamma\Big(k-1, \lambda\big(\tau - (x_{i+1} - x_i)\big)\Big)}{\gamma(k-1, \lambda\tau)} \end{aligned} \tag{6.10}$$

Eqs. (6.9) and (6.10) provide a recursive formula to calculate $Pr[C_{x_i}^{x_j}]$ when x_i and x_j are positions of i^{th} and j^{th} sensors when $j - i + 1 > k$.

Assuming that the distance between p_1 and the leftmost sensor in $[p_1, p_2]$ and the distance between p_2 and the rightmost sensor in $[p_1, p_2]$ are also exponentially distributed with parameter λ we use Eq. 6.9 as an approximation for $Pr\big[C_{p_1}^{p_2}\big]$.

Observe that for any pair of points (p_1, p_2), if sensors are dispersed along the entire interval $[p_1, p_2]$ the probability of coverage $Pr\big[C_{p_1}^{p_2}\big]$ is a function of $p_2 - p_1$ and not the absolute values of p_1 or p_2. Therefore, let $p(l)$ denote the probability of a border with length l being (k, w)-covered when sensors are dispersed with Poisson parameter λ. Using (6.9) and (6.10) if $l > \tau$ and there are at least k sensors on l:

$$p(l) \geq \int_{t=0}^{\tau} \frac{\gamma\big(k-1, \lambda(\tau - t)\big)}{\gamma(k-1, \lambda\tau)} f_d(t) p(l-t)\, dt \tag{6.11}$$

where $f_d(t) = \lambda e^{-\lambda t}$ is the distance between two consecutive sensors. Also $p(l) = 0$ if $l > \tau$ and there are less than k sensors on l. Finally $p(l) = 1$ if $l \leq \tau$.

We use the recursive equation in (6.11) as an estimate for $p(l)$ when $l > \tau$ with the initial condition $p(l) = 1$ when $l \leq \tau$. We could not calculate a closed formula for $p(l)$ from (6.11). Instead we calculate the Laplace transform of $p(l)$ and the values of $p(l)$ for different parameters λ can be calculated by numerical inverse Laplace transform.

Theorem 6.1. *Let $p(l)$ be the probability of a border of length l be (k, w)-covered when sensors are deployed according to a Poisson distribution with parameter λ. The Laplace transform of $p(l)$ denoted by $P(s) = \mathcal{L}\{p(l)\}$ can be calculated as:*

$$P(s) = \frac{1 - G(s) + \big(G(0) - 1\big)e^{-s\tau}}{s\big(1 - G(s)\big)} \tag{6.12}$$

where

$$G(s) = \begin{cases} \frac{-e^{-(s+\lambda)\tau}\lambda^k f^{(k-2)}(s)}{\gamma(k-1,\lambda\tau)(s+\lambda)} + \frac{\lambda}{s+\lambda} & k \geq 2 \\ \frac{-e^{-(s+\lambda)\tau}}{s+\lambda} + \frac{\lambda}{s+\lambda} & k = 1 \end{cases} \tag{6.13}$$

and $f(s) = \frac{e^{s\tau}-1}{s}$.

Proof. Let $g(t) = \frac{\gamma\big(k-1,\lambda(\tau-t)\big)\lambda e^{-\lambda t}}{\gamma(k-1,\lambda\tau)}$ when $t \leq \tau$ and $g(t) = 0$ when $t \geq \tau$. Also let $G(s) = \mathcal{L}\{g(t)\}$ denote the Laplace transform of $g(t)$. We start by taking Laplace transform of both sides of (6.11):

$$\begin{aligned} P(s) &= \int_{x=0}^{\tau} e^{-sx}dx + \int_{x=\tau}^{\infty} e^{-sx}\int_{t=0}^{x} p(x-t)g(t)dtdx \\ &= \frac{1-e^{-s\tau}}{s} + \int_{x=0}^{\infty} e^{-sx}\int_{t=0}^{x} p(x-t)g(t)\,dt\,dx - \int_{x=0}^{\tau} e^{-sx}\int_{t=0}^{x} p(x-t)g(t)\;dt\,dx \\ &= \frac{1-e^{-s\tau}}{s} + P(s)G(s) - \int_{t=0}^{\tau} g(t)\int_{x=t}^{\tau} e^{-sx}dxdt \\ &= \frac{1-e^{-s\tau}}{s} + P(s)G(s) + \frac{e^{-s\tau}G(0) - G(s)}{s} \end{aligned} \tag{6.14}$$

Solving (6.14) for $P(s)$ we get (6.12).

Now we calculate $G(s)$ as follows:

$$G(s) = \frac{\lambda}{\gamma(k-1, \lambda\tau)} M(s')|_{s'=s+\lambda} \tag{6.15}$$

where $m(t) = \gamma\big(k-1, \lambda(\tau - t)\big)\ u(\tau - t)\ u(t)$, $M(s) = \mathcal{L}\{m(t)\}$ and $u(t)$ is the Heaviside step function. Therefore:

$$M(s) = \int_{t=0}^{\tau} e^{-st} \gamma\big(k-1, \lambda(\tau - t)\big)\ dt \tag{6.16}$$

Eq. 6.13 can be obtained by solving (6.16) and replacing in (6.15). □

We have shown how to calculate $p(l) = \mathcal{L}^{-1}\{P(s)\}$ numerically.

6.2.1 Expected Minimum Number of Sensors for (k, w)-Coverage

Given a border of length l and parameter k, let $p(l, \lambda)$ denote the probability that a border of length l is (k, w)-covered if sensors are randomly deployed with density λ, and let η denote the minimum number of sensors that covers the entire border if sensors are deployed one at a time. The expected value of η can be calculated as follows:

$$\begin{aligned} E[\eta] &= \lim_{\lambda \to 0^+} \sum_{i=0}^{\infty} \big(1 - p(l, i\lambda)\big)\lambda i \\ &= \int_{\lambda=0}^{\infty} 1 - p(l, \lambda)\ d\lambda \end{aligned} \tag{6.17}$$

In the next section we use the value of $E[\eta]$ to calculate the expected number of sensors and rounds when more than one sensor is deployed in each round.

6.3 Cost Analysis of Multi-Round Sensor Deployment

In this section we define a model for cost analysis of multi-round sensor deployment. We present two strategies of sensor deployment and for each strategy based on our cost model we calculate the total cost of deployment. Finally, for each strategy of sensor deployment the Poisson parameter that minimizes the total cost is calculated based on other problem parameters such as length of the border, sensing ranges of sensors and width of the intruder.

First we define our cost model. Let N and R denote the total number of sensors and number of necessary deployment rounds until the entire border is (k, w)-covered. We assume that each round of deployment has a fixed cost C_r which is in addition to the cost of deployed sensors in that round. Furthermore each sensor that is deployed has a fixed cost C_n. We model the *total cost* of sensor deployment as:

$$C_{tot} = R * C_r + N * C_n \tag{6.18}$$

Since we are concerned with random deployments, R and N are random variables. One reasonable estimate of the total cost is its expected value (mean). Regardless of the correlation between R and N, for any strategy of random deployment the average total cost of deployment can be calculated as follows:

$$E[C_{tot}] = E[R]C_r + E[N]C_n \tag{6.19}$$

where $E[C_{tot}]$, $E[R]$, and $E[N]$ denote the expected values of C_{tot}, R and N respectively.

In the following sections we present two different deployment strategies.

6.3.1 Fixed-Density Complete Deployment

In this section, we propose a simple complete deployment strategy called Fixed-Density Complete Deployment. According to this strategy, in each round, the deployment interval is the entire border $[0, l]$ and the same density of deployment is utilized in every round. In other words, in every round, sensors are deployed on the entire border with parameter λ until the border is entirely (k, w)-covered. This is a realistic strategy if we cannot determine the exact location of the coverage gaps on the border. Let N_A and R_A denote the total number of deployed sensors and the number of deployment rounds until the border is fully covered. As stated in Section 6.2.1, let η denote the minimum number of sensors that need to be deployed until the border is covered (if one sensor is deployed at a time). Let a_i denote the number of sensors deployed in the i^{th} round. Therefore a_i with $1 \leq i \leq R_A$ are i.i.d. Poisson random variables with parameter λl and:

$$\sum_{i=1}^{R_A-1} a_i < \eta \leq \sum_{i=1}^{R_A} a_i$$

The problem can be solved using results from discrete time renewal theory. In a renewal process with Poisson inter-arrival times with parameter λl, R_A is the number of renewals until time η plus one and N_A is η plus the excess time (time until next arrival). The elementary renewal theorem implies that if $\eta \to \infty$, the value of $E[N_A]$ can be calculated as follows:

$$E[N_A|\eta = n] = n + \frac{\lambda l}{2} \tag{6.20}$$

We use this as an estimate for $E[N_A|\eta = n]$ when $n \gg \lambda l$. Then the estimate for

$E[R_A|\eta = n]$ when $n \gg \lambda l$ can be expressed as:

$$E[R_A|\eta = n] = \frac{n}{\lambda l} + \frac{1}{2} \tag{6.21}$$

Taking expected value from both sides of (6.21) and (6.20) we get:

$$E[R_A] = \frac{E[\eta]}{\lambda l} + \frac{1}{2} \tag{6.22}$$

and

$$E[N_A] = E[\eta] + \frac{\lambda l}{2} \tag{6.23}$$

Finally substituting from (6.22) and (6.23) into (6.19) we get our estimate for the expected total cost:

$$E[C_{tot}] = \left(\frac{E[\eta]}{\lambda l} + \frac{1}{2}\right) C_r + \left(E[\eta] + \frac{\lambda l}{2}\right) C_n \tag{6.24}$$

where η is the minimum number of sensors that cover the border entirely if sensors are deployed one at a time and is independent of the value of λ

To calculate the value λ^* that minimizes the average total cost we equate the first derivative of Eq. 6.24 with respect to λ to zero to obtain

$$\lambda^* = \frac{1}{l}\sqrt{\frac{2E[\eta]C_r}{C_n}} \tag{6.25}$$

6.3.2 Fixed-Density Partial Deployment

In the previous strategy in each round many sensors may be deployed on the portions of the border which are already covered. This is inevitable if we cannot determine the location of coverage gaps. However, if by some means, one can determine the position of uncovered portions of the border after a round of deployment, a more

efficient deployment can be done in the next round.

In this section, we assume one can determine the position of the gaps after each round of deployment and propose a partial deployment strategy called Fixed-Density Partial Deployment. For simplicity of analysis we assume that sensors are relabeled according to their x-coordinates after each round. The deployment interval for any round is specified as follows: For any two consecutive sensors at x_i and x_{i+1} we say $[x_i, x_{i+1}]$ is a deployment interval in the next round if and only if $x_{i+1} - x_i > \tau$. In addition, in every round, the deployment is done with the same density λ over the specified deployment intervals for the round.

In the following theorem we show the relationship between our specified deployment intervals and coverage gaps:

Theorem 6.2. *The interval $[x_i, x_{i+1}]$ is a deployment interval if and only if there is a point $p \in [x_i, x_{i+1}]$ such that p is not $(1, w)$-covered.*

Proof. First assume $[x_i, x_{i+1}]$ is a deployment interval. Then, according to our specifications of a deployment interval, $x_{i+1} - x_i > \tau$ and hence the interval $(x_i + \frac{\tau}{2}, x_{i+1} - \frac{\tau}{2})$ is not empty. For any point $p \in (x_i + \frac{\tau}{2}, x_{i+1} - \frac{\tau}{2})$ we have $x_i < p - \frac{\tau}{2}$ and $p + \frac{\tau}{2} < x_{i+1}$ therefore p is not $(1, w)$-covered.

Second assume point $p \in [0, l]$ is not $(1, w)$-covered. Since there is one sensor at 0 and one sensor at l, therefore $\frac{\tau}{2} < p < l - \frac{\tau}{2}$. Let x_i and x_j denote the position of rightmost sensor to the left of $p - \frac{\tau}{2}$ and leftmost sensor to the right of $p + \frac{\tau}{2}$. Since p is not $(1, w)$-covered, there is no sensor in $[p - \frac{\tau}{2}, p + \frac{\tau}{2}]$ and therefore $j = i + 1$. Also since $x_i < p - \frac{\tau}{2}$ and $x_{i+1} > p + \frac{\tau}{2}$ therefore $x_{i+1} - x_i > \tau$ and $[x_i, x_{i+1}]$ is a deployment interval. □

Compare our strategy with the case where in a round of deployment sensors are dispersed according to a Poisson distribution with parameter λ over the entire border. For those sensors that fall in the deployment intervals they fall according to Poisson

distribution with density λ. Also for any sensor s that falls in a non-deployment interval since the distance of s to any coverage gap is bigger than $\frac{\tau}{2}$ therefore it does not cover any point in any coverage gap. Therefore if sensors are dispersed over the entire border or only over the deployment interval the new coverage gaps have the same distribution (both position and length). In fact the only difference between our two strategies is the number of sensors used in each round (and therefore the total number of sensors). We use this fact to calculate $E[R_B]$ and $E[N_B]$ as follows:

$$E[R_B] = E[R_A] = \frac{E[\eta]}{\lambda l} + \frac{1}{2} \tag{6.26}$$

Let $D(i)$ denote the sum of length of deployment intervals after round $i > 0$ if sensors are dispersed on the entire border at each round. At any round $i > 0$ distances between two consecutive sensors are i.i.d. random variables with exponential distribution with parameter λi. Let $d(i)$ denote the distance between two consecutive sensors at round i. Therefore:

$$\begin{aligned} E[D(i)] &= l\frac{E\big[\mathbb{1}_{\{d(i)>\tau\}} * d(i)\big]}{E\big[d(i)\big]} \\ &= l * (i\lambda\tau + 1)e^{-i\lambda\tau} \end{aligned} \tag{6.27}$$

where $\mathbb{1}_{\{d(i)>\tau\}}$ is the indicator function of the event $d(i) > \tau$.

Let b_i denote the number of newly deployed sensors in round $i > 0$ on the deployment intervals from the previous round. The average total number of sensors $E[N_B]$

can be calculated as follows:

$$\begin{aligned} E[N_B] &= \sum_{i=1}^{\infty} E[b_i] \\ &= l\lambda + \sum_{i=1}^{\infty} \lambda E[D(i)] \\ &= l\lambda \left(1 + \sum_{i=1}^{\infty} (i\lambda\tau + 1)e^{-i\lambda\tau}\right) \\ &= l\lambda \left(\frac{\lambda\tau e^{-\lambda\tau}}{(1-e^{-\lambda\tau})^2} + \frac{1}{1-e^{-\lambda\tau}}\right) \end{aligned} \tag{6.28}$$

The minimum of $E[C_{tot}]$ can be calculated numerically using the calculated $E[R_B]$ and $E[N_B]$.

6.4 Simulation Results

In this section we present the results of our computer simulation using MATLAB. The positions of sensors are generated according to a Poisson distribution in a given interval. In the results each point is calculated as an average of 10000 simulations.

Figure 6.1 shows the probability of (k, w)-coverage as a function of the Poisson parameter λ. We used the estimated lower bound in [LZS+11] as a benchmark to compare with our results. Simulations confirm that our analysis provides a tighter lower bound. As k grows, the difference between analysis and simulation results also increases which is also observed in [LZS+11].

Figure 6.2 shows the expected number of dispersed sensors until the border is (k, w)-covered as a function of λ. It can be seen that for sufficiently small values of λ ($\lambda \leq 0.2$) the simulation results are very close to our analytical estimate. As λ increases, the difference between simulation results and analysis for $E[N_A]$ grows. It can be seen that when $\lambda > 0.7$ the number of necessary sensors converges to λl. The

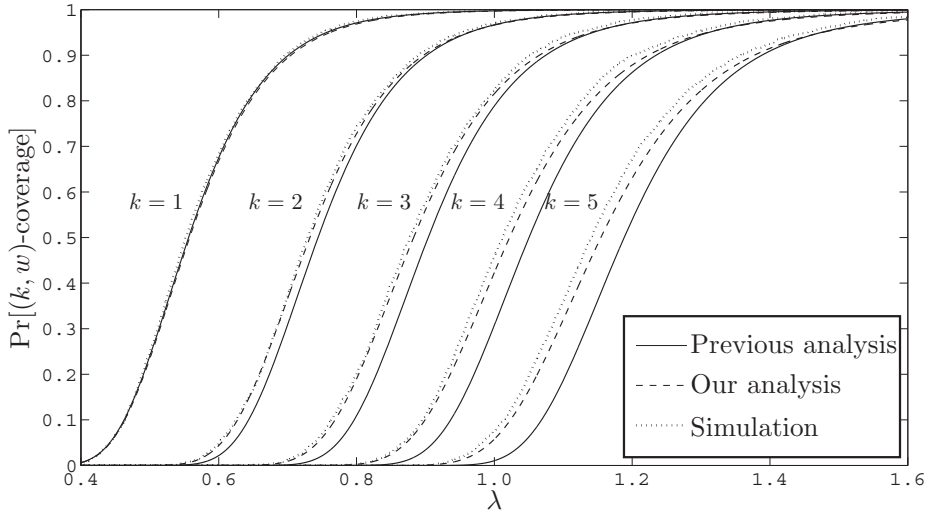


Figure 6.1: Simulation results for the probability of (k, w)-coverage with complete deployments. $l = 5000$, $\tau = 15$.

explanation is that when $\lambda \to \infty$ one round of deployment is necessary and sufficient and therefore $E[N_A] \to \lambda l$.

Figure 6.3 shows the expected number of deployment rounds in our complete deployment strategy until the entire border is (k, w)-covered. As expected, the number of needed rounds decreases as λ increases. Figure 6.4 shows the same simulation results when the range of λ is in $[0.14, 1.2]$. Note that when $0.2 < \lambda < 0.8$, a similar pattern as for $E[N_B]$ is observed in simulation results which is harder to see in Figure 6.3 due to the scaling.

For every pair of (C_r, C_n) the expected total cost of deployment as a function of λ can be calculated using Eq.6.19. Dividing both sides of Eq. 6.19 by C_n and replacing for $E[R]$ and $E[N]$ from (6.23) and (6.22) one can get $\frac{C_{tot}}{C_n}$ as a function of λ and $\frac{C_r}{C_n}$. Figure 6.5 shows simulation results vs analysis for three different values of $\frac{C_r}{C_n}$. As explained before analysis is valid when λ is sufficiently small. It can be seen from Eq. 6.25 when $\frac{C_r}{C_n}$ increases the λ^* also increases. But the equations are valid only if $\lambda \ll \frac{E[\eta]}{l}$. As can be seen in the middle and right figures, the optimal density calculated by simulations is getting farther from the value calculated analytically.

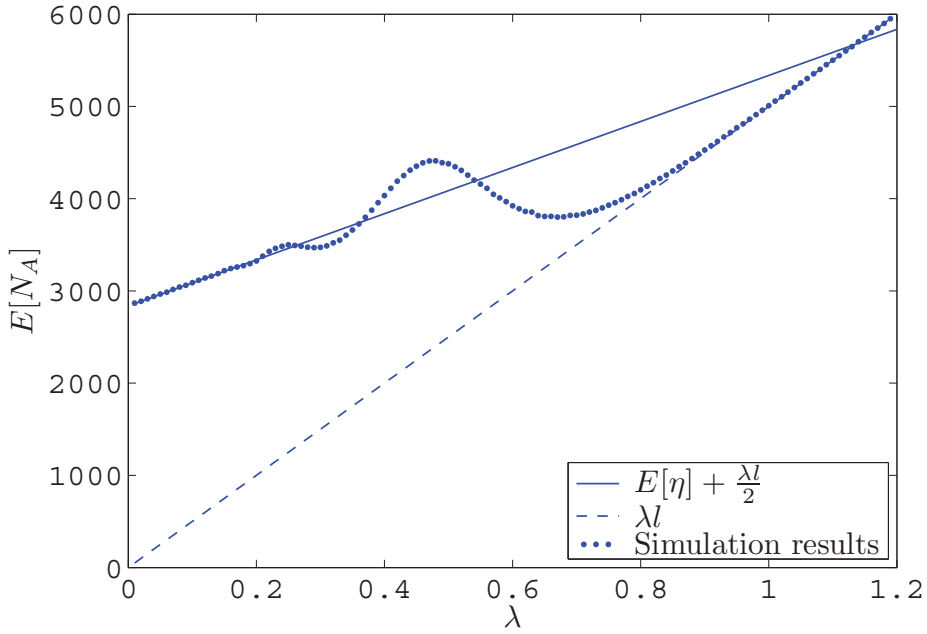


Figure 6.2: Fixed-Density Complete Deployment: Expected number of dispersed sensors until $(1, w)$-coverage is achieved. $l = 5000$ and $\tau = 15$.

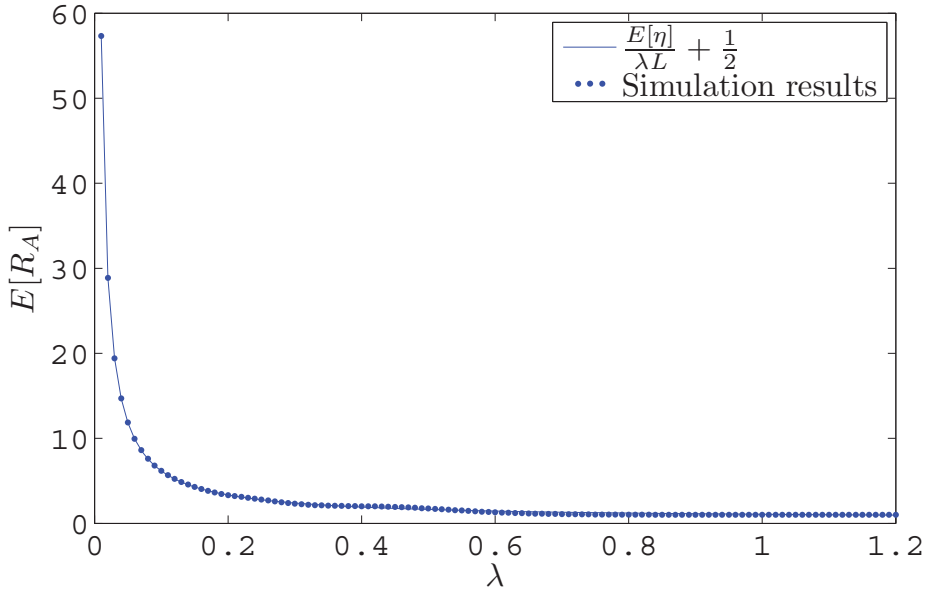


Figure 6.3: Fixed-Density Complete Deployment: Expected number of rounds until $(1, w)$-coverage is achieved. $l = 5000$ and $\tau = 15$.

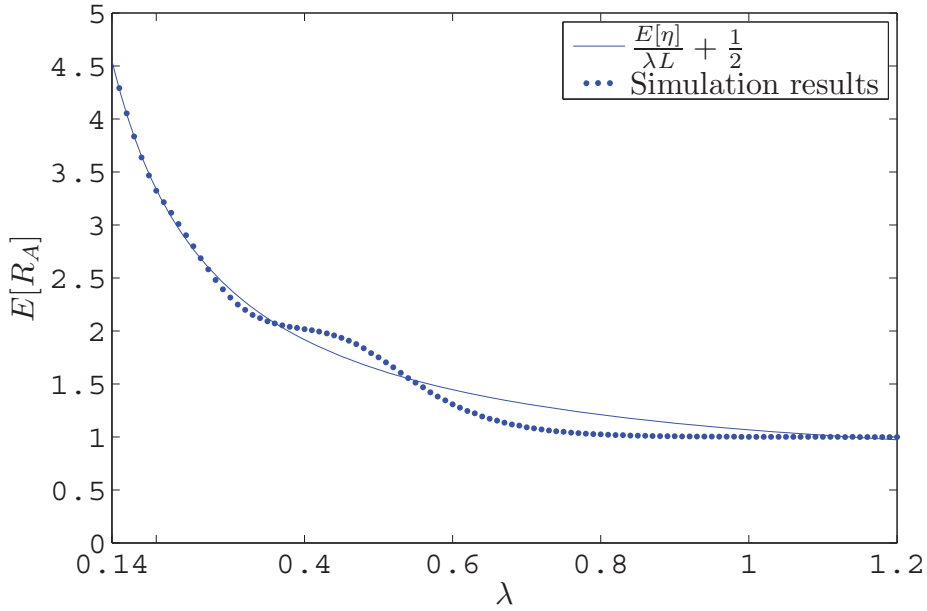


Figure 6.4: Fixed Density Complete Deployment: Expected number of rounds until $(1, w)$-coverage is achieved. $l = 5000$ and $\tau = 15$.

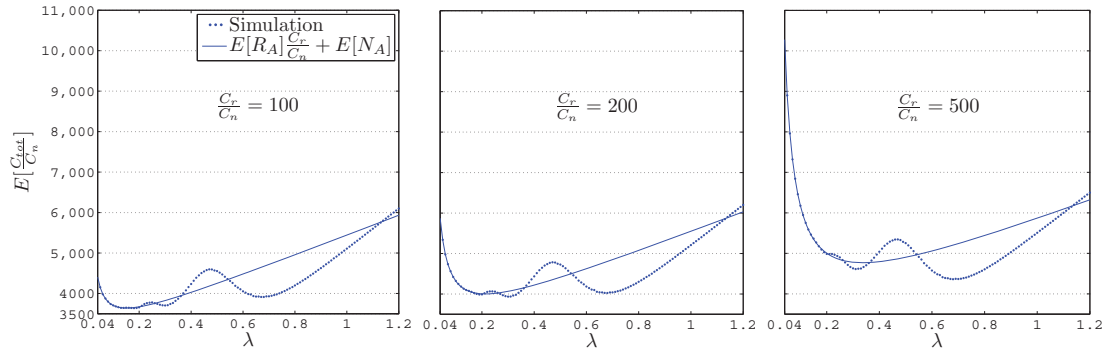


Figure 6.5: Fixed Density Complete Deployment: $E[\frac{C_{tot}}{C_n}]$ as a function of λ for different values of $\frac{C_r}{C_n}$. $l = 5000$, $\tau = 15$, $k = 1$

Figure 6.6 shows the minimum total deployment costs $\frac{C_{tot}}{C_n}$ as a function of $\frac{C_r}{C_n}$. For the simulation results, for any fixed value of $\frac{C_r}{C_n}$, different deployment densities were tried, and the minimum cost over all values of deployment densities tried is plotted in the graph. For analytical results, the optimal λ^* is calculated from Eq. 6.25 and the total cost is calculated by simulations using the mentioned deployment rate λ^* in each round. It can be seen that when $\frac{C_r}{C_n}$ is small the simulation results are close to the analysis.

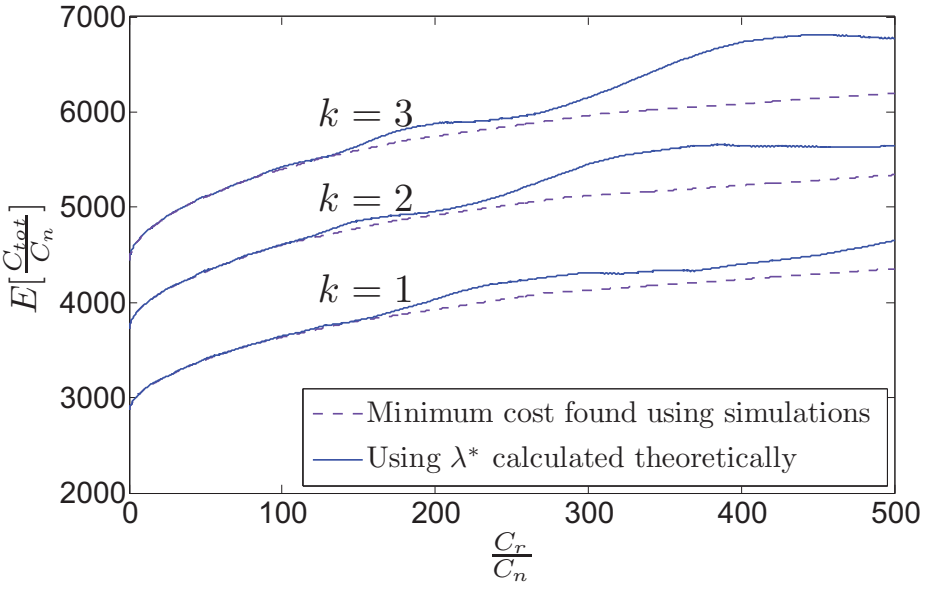


Figure 6.6: Fixed Density Complete Deployment: comparing total cost given by simulations and by analysis. For every $\frac{C_r}{C_n}$ optimal λ is calculated from Eq. 6.25 $l = 5000$, $\tau = 15$

As seen in Figure 6.7, when $\frac{C_r}{C_n}$ is small, by using the optimal λ^* obtained by our analysis, the total cost estimate would be quite accurate. However, even when $\frac{C_r}{C_n}$ is larger, the optimal cost as given by the analysis differs from the optimal cost obtained via simulations by at most 11%. Thus, for all values of $\frac{C_r}{C_n}$, prior to deployment, our analytical model can be used to obtain a deployment rate that is close to optimal in terms of cost.

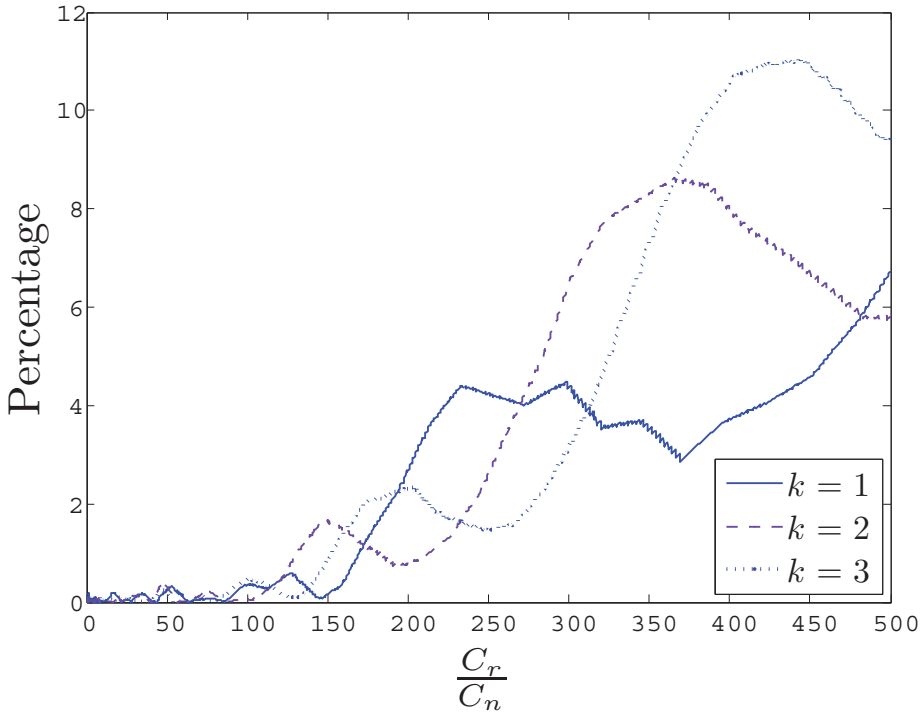


Figure 6.7: Fixed Density Complete Deployment: Percentage increase in total cost using λ^* given by analysis versus optimal cost given by simulations. $l = 5000$, $\tau = 15$

In the subsequent simulations we study the partial deployment strategy. In Figure 6.8 the total number of sensors using the partial deployment strategy is calculated as a function of density of deployed sensors in each round. It can be seen that the simulation results match the analysis very closely. Note that in contrast to complete deployment, the analytical results for the expected number of sensors using partial deployment are valid for all values of λ.

Figure 6.9 compares the total number of necessary sensors with different deployment strategies: complete vs. partial deployment. It can be seen that partial deployment always uses fewer number of sensors. This is expected, as in the partial deployment in contrast to complete deployment, we only deploy sensors on the coverage gaps. As λ increases ($\lambda > 0.7$), in both strategies, the total number of deployed sensors converges to λl. The reason is that in both strategies when $\lambda \to \infty$ one round of deployment is necessary and sufficient and therefore the expected number

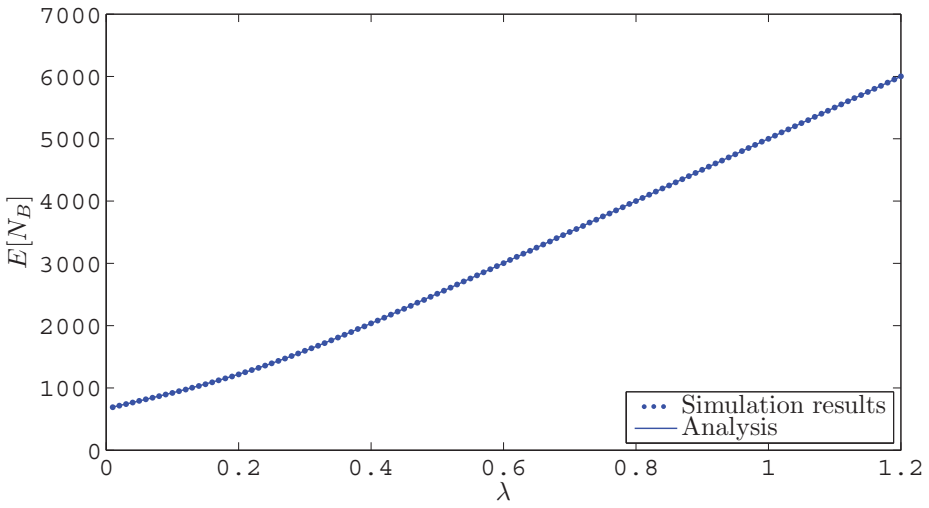


Figure 6.8: Fixed Density Partial Deployment: Expected number of necessary sensors for $(1, w)$-coverage. $l = 5000$ and $\tau = 15$.

of sensors converges to λl

In Figure 6.10 the minimum expected total cost of the two deployment strategies is compared. As mentioned before, in both strategies total number of rounds have the same distribution. Since the partial strategy uses fewer sensors on average, the total cost of partial deployment is expected to be lower. This is confirmed by the simulation results.

6.5 Conclusions

In this chapter we analyzed the barrier coverage problem with multi-round sensor deployment. We introduced two classes of multi-round deployment strategies: complete and partial deployment. For complete strategies, we calculated the probability of a border being (k, w)-covered as a function of length of the border, density of sensors, sensing range of sensors and width of the intruder. We also defined a simple cost model for evaluation of the total cost of a deployment strategy. Finally we studied two specific deployment strategies: Fixed-Density Complete and Fixed-Density

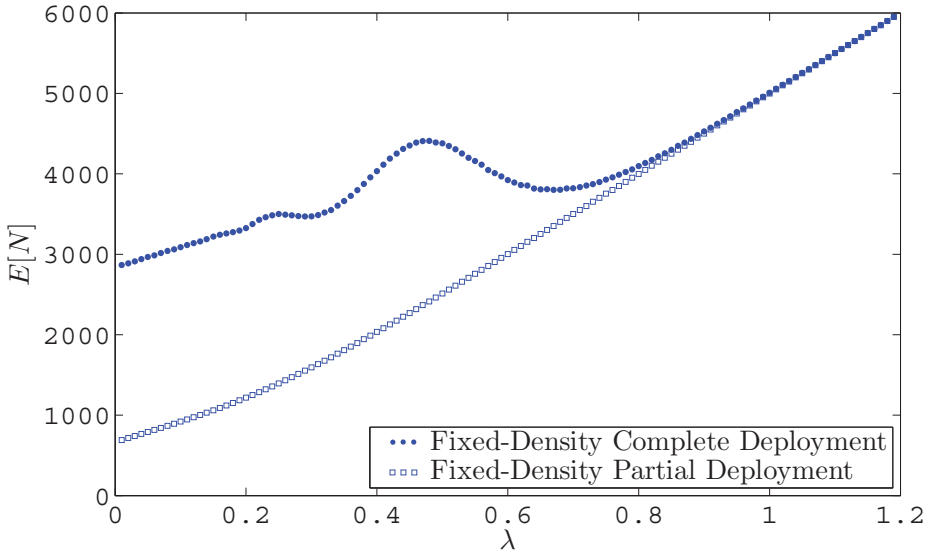


Figure 6.9: Fixed Density Complete vs Partial Deployment: Expected number of necessary sensors for $(1, w)$-coverage. $l = 5000$ and $\tau = 15$.

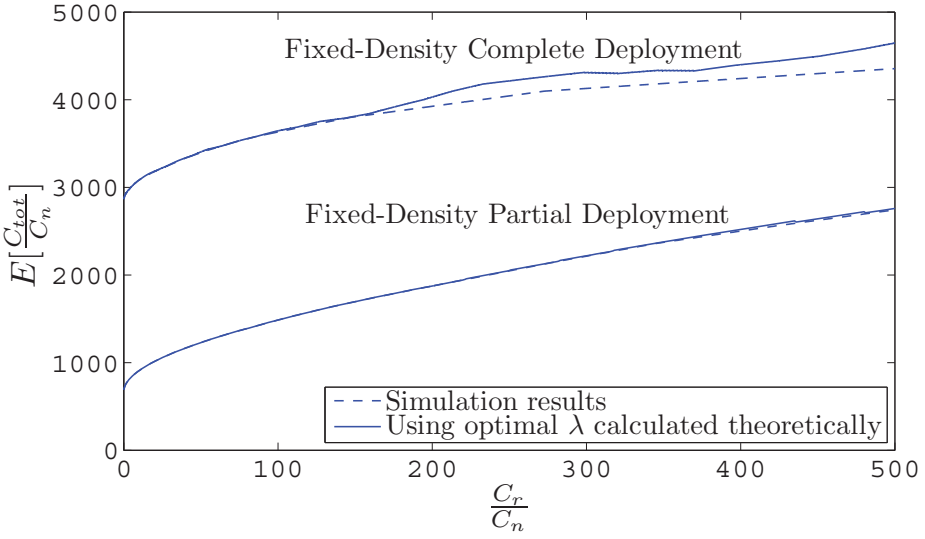


Figure 6.10: Fixed Density Complete vs Partial Deployment: Minimum expected total cost of deployment. $l = 5000$, $\tau = 15$ and $k = 1$.

Partial. For each strategy we calculated the expected total cost as a function of the density of sensors in each round and estimated the optimal density that minimizes the total expected cost. We performed extensive computer simulations to validate our analysis. Our analysis of fixed-density partial deployment matches the simulation results closely. Also the results for fixed-density complete deployment show that our analytical model can always be used prior to deployment to obtain a deployment rate that is close to optimal in terms of cost. In fact, in our simulations, when per-round cost is at most two hundred times the per-sensor cost, the deployment rate given by the analysis yields an optimal cost.

The deployment strategies that we introduced in this chapter can be extended to minimize the expected total cost further. We only studied fixed density strategies, where in every round, sensors are dispersed with the same density. One can consider deployment strategies with different densities in each round. Also the analysis of our partial deployment strategy for $k > 1$ is left as an open problem.

Chapter 7

Conclusions and Future Directions

In this thesis we studied several problems on barrier coverage with wireless sensor networks. We considered both centralized and distributed algorithms with sensors with different capabilities. First we studied centralized algorithms for relocatable sensors, where each sensor is capable of moving from an ad hoc initial position to a final position where it remains afterward. The MinMax problem was shown to be NP-complete when sensors with different ranges are initially located in the plane and are allowed to move to arbitrary final positions. We considered a natural constraint movement that we call perpendicular movement. Polytime algorithms are given for the case where barriers are parallel and we showed that the problem is NP-complete when there are two perpendicular barriers. Furthermore, polynomial time algorithms are given for special cases of the problem. We also presented two approximation algorithm for maximizing the sum of the lengths of covered segments of the barriers.

Then we considered the distributed algorithms for barrier coverage of a barrier modelled with a single line segment. We assumed that sensors behave as autonomous robots and have local information of the network. The results are highly dependent on the time synchronization between sensors and whether sensors share some global

information. We gave two polytime distributed algorithms that achieve barrier coverage if fully synchronous unoriented sensors are initially located on grid positions. Our first algorithm takes $O(n^2)$ time and is oblivious while our second algorithm assumes sensors with 2 bits of memory that can remember their previous actions and terminates in linear time. We showed that no distributed algorithm for barrier coverage exists when sensors are semi-synchronous and do not share a common orientation. In contrast, we gave an algorithm for the case where sensors share the same orientation. Our algorithm achieves barrier coverage within finite time, even if all sensors are asynchronous.

Finally, we considered the barrier coverage with stationary sensors and multi-round random deployment. Sensors are assumed to be deployed uniformly at random on a barrier modelled as a stripe. We presented two different sensor deployment strategies and for each strategy we gave an estimate for the probability of barrier coverage at the end of each round. The expected number of necessary rounds and sensors to achieve barrier coverage is calculated for each strategy. The overall cost of barrier deployment can be estimated as sum of the costs of sensors plus the cost of deployment rounds. Using this metric for each of our deployment strategies, we calculated the best parameters that give the minimum cost. We validated our analytical results with extensive computer simulations.

Several directions can be suggested for future work. For centralized algorithms, characterizing the situations when the feasibility of barrier coverage can be determined in polynomial time remains an interesting open problem. Considering perpendicular movement, existence of faster algorithms for parallel barriers is an open question. Also the existence of better approximation algorithms as well as study of different movement models are of interest. Furthermore, more realistic models where final positions of sensors are not required to be on the barriers can be considered.

For distributed algorithms, we considered three different existing timing models for autonomous robots. However other models such as a model where sensors have almost but not fully synchronized clocks, is suggested as a future direction. Also withing existing timing models, there are still many open problems such as: Is there any algorithm for barrier coverage with sensors that each have sense of orientation but do not necessarily agree on a global orientation, in the semi-synchronous/asynchronous models? Can our fully synchronous algorithms be extended for sensors that are not necessarily on grid positions? How about algorithms that do not terminate, but eventually sensors positions coverage to those in a covering assignment? What is the average-case performance of the algorithms? Also the question whether a visibility range larger than $2r$ can help with designing of algorithms for barrier coverage with unoriented sensors in SSYNC model remains unanswered.

For barrier coverage with multi-round random deployment with stationary sensors, we introduced two classes of deployment strategies but only analyzed one strategy example from each class. Also we only considered the model where sensors are dispersed uniformly at random on the barrier. It would be of interest to study other possible strategies as well as other random models.

Finally, investigating the applicability of the barrier coverage algorithms for solving area coverage is an interesting avenue for further research.

Bibliography

[ASY95] H. Ando, I. Suzuki, and M. Yamashita. Formation and agreement problems for synchronous mobile robots with limited visibility. In *Proceedings of IEEE International Symposium on Intelligent Control*, pages 453–460, 1995.

[BBH+09] B. Bhattacharya, M. Burmester, Y. Hu, E. Kranakis, Q. Shi, and A. Wiese. Optimal movement of mobile sensors for barrier coverage of a planar region. *Theoretical Computer Science*, 410(52):5515–5528, 2009.

[BBSK07] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length. In *Proceedings of MobiCom '07*, pages 75–86, 2007.

[BJY+10] D. Ban, J. Jiang, W. Yang, W. Dou, and H. Yi. Strong k-barrier coverage with mobile sensors. In *Proceedings of International Wireless Communications and Mobile Computing Conference*, pages 68–72, 2010.

[CGLW12] D. Z. Chen, Y. Gu, J. Li, and H. Wang. Algorithms on minimizing the maximum sensor movement for barrier coverage of a linear domain. In *Proceedings of SWAT'12*, pages 177–188, 2012.

[CGP09] J. Czyzowicz, L. Gasieniec, and A. Pelc. Gathering few fat mobile robots in the plane. *Theoretical Computer Science*, 410(6-7):481–499, 2009.

[CKK+09] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, and M. Yazdani. On minimizing the maximum sensor movement for barrier coverage of a line segment. In *Proceedings of ADHOC-NOW, LNCS v. 5793*, pages 194–212, 2009.

[CKK+10] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, and M. Yazdani. On minimizing the sum of sensor movements for barrier coverage of a line segment. In *Proceedings of ADHOC-NOW, LNCS v. 6288*, pages 29–42, 2010.

[CKL07] A. Chen, S. Kumar, and T. H. Lai. Designing localized algorithms for barrier coverage. In *Proceedings of MobiCom '07*, pages 63–74, 2007.

[CP06] R. Cohen and D. Peleg. Local algorithms for autonomous robots systems. In *Proceedings of SIROCCO'06, LNCS v. 4056*, pages 29–43, 2006.

[DDCM13] S. Datta, A. Dutta, S. C. Chaudhuri, and K. Mukhopadhyaya. Circle formation by asynchronous transparent fat robots. In *Proceedings of ICDCIT'13*, pages 195–207, 2013.

[DDE+13] S. Dobrev, S. Durocher, M. Eftekhari, K. Georgiou, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, S. Shende, and J. Urrutia. Complexity of barrier coverage with relocatable sensors in the plane. In *CIAC*, pages 170–182, 2013.

[DHM+09] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, and M. Zadimoghaddam. Minimizing movement. *ACM Transactions on Algorithms*, 5(3):1–30, 2009.

[EB09] Y. Elor and A. M. Bruckstein. Multi-a(ge)nt deployment and patrolling on a ring graph. Technical Report CIS-2009-16, Computer Science Department, Technion, Israel, 2009.

[EKK+13] M. Eftekhari, E. Kranakis, D. Krizanc, O. Morales-Ponce, L. Narayanan, J. Opatrny, and S. Shende. Distributed local algorithms for barrier coverage using relocatable sensors. In *Proceeding of ACM PODC Symposium*, pages 383–392, 2013.

[ENO13] M. Eftekhari, L. Narayanan, and J. Opatrny. On multi-round sensor deployment for barrier coverage. In *Proceedings of 10th IEEE MASS*, pages 310–318, 2013.

[FPS08] P. Flocchini, G. Prencipe, and N. Santoro. Self-deployment algorithms for mobile sensors on a ring. *Theoretical Computer Science*, 402(1):67–80, 2008.

[FPS12] P. Flocchini, G. Prencipe, and N. Santoro. *Distributed Computing by Oblivious Mobile Robots: Synthesis Lectures on Distributed Computing Theory*. Morgan and Claypool Publishers, 2012.

[FPSW01] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous mobile robots with limited visibility. In *Proceedings of STACS'01, LNCS v. 2010*, pages 247–258, 2001.

[Gag92] D. Gage. Command control for many-robot systems. In *Proceedings of the Nineteenth Annual AUVS Technical Symposium (AUVS '92)*, 1992.

[GJ90] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990.

[Gol87] E. M. Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1987.

[HCL+12] S. He, J. Chen, X. Li, X. Shen, and Y. Sun. Cost-effective barrier coverage by mobile sensor networks. In *Proceedings of INFOCOM'12*, pages 819–827, 2012.

[HT03] C. F. Huang and Y. C. Tseng. The coverage problem in a wireless sensor network. In *Proceedings of WSNA'03*, pages 115–121, 2003.

[KLA05] S. Kumar, T. H. Lai, and A. Arora. Barrier coverage with wireless sensors. In *Proceedings of MobiCom '05*, pages 284–298, 2005.

[KLB04] S. Kumar, T. H. Lai, and J. Balogh. On k-coverage in a mostly sleeping sensor network. In *Proceedings of MobiCom '04*, pages 144–158, 2004.

[KLZ10] Y. Keung, B. Li, and Q. Zhang. The intrusion detection in mobile sensor networks. In *Proceedings of MobiHoc'10*, 2010.

[LDWS08] B. Liu, O. Dousse, J. Wang, and A. Saipulla. Strong barrier coverage of wireless sensor networks. In *Proceedings of MobiHoc'08*, 2008.

[LZS+11] L. Li, B. Zhang, X. Shen, J. Zheng, and Z. Yao. A study on the weak barrier coverage problem in wireless sensor networks. *Computer Networks*, 55:711–721, 2011.

[Mat94] M. Matarić. *Interaction and Intelligent Behavior*. PhD thesis, MIT, 1994.

[MKPS01] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of IEEE INFOCOM'01*, pages 1380–1387, 2001.

[MLC+12] H. Ma, D. Li, W. Chen, Q. Zhu, and H. Yang. Energy efficient k-barrier coverage in limited mobile wireless sensor networks. *Computer Communications*, 35(14):1749–1758, 2012.

[MNO11] M. Mehrandish, L. Narayanan, and J. Opatrny. Minimizing the number of sensors moved on line barriers. In *Proceedings of IEEE WCNC'11*, pages 1464–1469, 2011.

[NMA10] M. Noori, S. Movaghati, and M. Ardakani. Characterizing the path coverage of random wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2010(716565):1–11, 2010.

[PS06] G. Prencipe and N. Santoro. Distributed algorithms for autonomous mobile robots. In *Proceedings of* 5^{th} *IFIP International Conference on Theoretical Computer Science (TCS'06)*, volume 209, pages 47–62, 2006.

[SCLP08] C. Shen, W. Cheng, X. Liao, and S. Peng. Barrier coverage with mobile sensors. In *Proceedings of I-SPAN'08*, pages 99–104, 2008.

[SLX+10] A. Saipulla, B. Liu, G. Xing, X. Fu, and J. Wang. Barrier coverage with sensors of limited mobility. In *Proceedings of MobiHoc'10*, pages 201–210, 2010.

[SY99] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999.

[YQ10] G. Yan and D. Qiao. Multi-round sensor deployment for guaranteed barrier coverage. In *Proceedings of IEEE INFOCOM'10*, pages 2462–2470, 2010.

[ZG04] F. Zhao and L. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann Publishers Inc., 2004.