Weak Coverage of a Rectangular Barrier *

S. Dobrev[1], E. Kranakis[2], D. Krizanc[3], M. Lafond[4], J. Maňuch[5], L. Narayanan[6], J. Opatrny[6], S. Shende[7] and L. Stacho[8]

[1]Institute of Mathematics, Slovak Academy of Sciences, Bratislava, Slovakia
[2]School of Computer Science, Carleton University, Ottawa, Canada
[3]Dept. of Mathematics and Computer Science, Wesleyan University, Middletown CT, USA
[4]Department of Mathematics and Statistics, University of Ottawa, Ottawa, Canada
[5]Department of Computer Science, University of British Columbia, Vancouver, Canada
[6]Department of Computer Science and Software Engineering, Concordia University, Montreal, QC, Canada
[7]Department of Computer Science, Rutgers University, Camden, NJ, USA
[8]Department of Mathematics, Simon Fraser University, Burnaby BC, Canada

Abstract

Assume n wireless mobile sensors are initially dispersed in an ad hoc manner in a rectangular region. They are required to move to final locations so that they can detect any intruder crossing the region in a direction parallel to the sides of the rectangle, and thus provide *weak barrier coverage* of the region. We study three optimization problems related to the movement of sensors to achieve weak barrier coverage: minimizing the *number* of sensors moved (MinNum), minimizing the *average* distance moved by the sensors (MinSum), and minimizing the *maximum* distance moved by the sensors (MinMax). We give an $O(n^{3/2})$ time algorithm for the MinNum problem for sensors of diameter 1 that are initially placed at integer positions; in contrast we show that the problem is NP-hard even for sensors of diameter 2 that are initially placed at integer positions. We show that the MinSum problem is solvable in $O(n \log n)$ time for homogeneous range sensors in arbitrary initial positions for the Manhattan metric, while it is NP-hard for heterogeneous sensor ranges for both Euclidean and Manhattan metrics. Finally, we prove that even very restricted homogeneous versions of the MinMax problem are NP-hard.

1 Introduction

Intruder detection is an important application of wireless sensor networks. Each sensor monitors a circular area centered at its location, and can immediately alert a monitoring station if it detects the presence of an intruder. Collectively the sensors can be deployed to monitor the entire region, providing so-called *area coverage*. However, for many applications, it is sufficient, and much more cost-effective, to simply monitor the boundary of the region, and provide so-called *barrier coverage*.

Barrier coverage was introduced in [17], and has been extensively studied since then [2, 3, 13, 18, 20, 8, 9]. The problem was posed as the deployment of sensors in a narrow belt-like rectangular region in such a way that any intruder crossing the belt would be detected. A sensor network is said to provide *strong barrier coverage* if an intruder is detected regardless of the path it follows

*Research supported by NSERC, Canada

across the given barrier (see Figure 1 (a)). In contrast, a sensor network provides *weak coverage* if an intruder is detected when it follows a straight-line path across the width of the barrier. If the location of the sensors is not known to a trespasser, weak coverage is often sufficient, and is more cost-effective.

In this paper, we consider a more general notion of weak coverage than previously considered. Given a rectangular barrier, we aim to detect intruders who cross the region in a straight-line path parallel to *either of the axes* of the rectangle (see Figure 1 (b)).

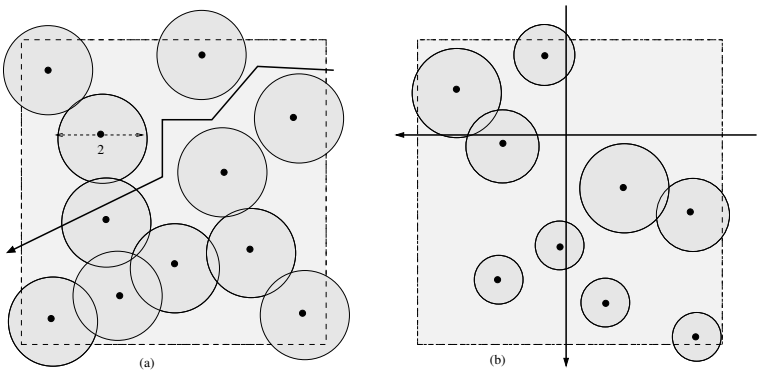

Figure 1: (a) Strong coverage of the shaded square area by a homogeneous network, (b) Weak coverage of the shaded square area by a non-homogeneous network for paths perpendicular to the axes.

A sensor network can be deployed for the given barrier in several different ways. In *deterministic deployment*, sensors are placed in pre-defined locations that ensure intruder detection. However, when the deployment area is very large, or the terrain in the area is difficult or dangerous, a deterministic deployment might be costly or even impossible. In those instances a *random or ad hoc deployment* of sensors can be done [2]. However, this type of deployment might leave some gaps in the coverage of the area. Two approaches have been considered in order to deal with this problem. One is a *multi-round random deployment* in which the random dispersal is repeated until the coverage of the area is assured with very high probability [20]. The other approach is to use *mobile (or relocatable) sensors* [6]. After the initial dispersal, some or all sensors are instructed to *relocate to new locations* so that the desired barrier coverage is achieved. Clearly, the relocation of the sensors should be performed in the most efficient way possible. In particular, we may want to minimize the time or energy needed to perform the relocation, or the number of sensors to be relocated.

1.1 Notation and problem definition

We assume that n sensors are initially located in an axis-parallel rectangular area R of size $a \times b$ in the Cartesian plane. The n *sensors* $S_1, S_2, \ldots, S_n$ have sensing ranges $r_1, r_2, \ldots, r_n$ respectively. The *diameter* of a sensor is equal to twice its range. We assume that $\sum_{k=1}^{n} 2r_k \geq \max\{a, b\}$; this ensures that placed in appropriate locations, the sensors can achieve weak barrier coverage. A sensor network is called *homogeneous* if the sensing ranges of all sensors in the network is the same. Otherwise the network is called *heterogeneous*.

A *configuration* is a tuple $(R, p_1, p_2, \ldots, p_n)$ where R is the rectangle to be weakly barrier-covered and $\{p_1, p_2, \ldots, p_n\}$ are the positions of the sensors. We say a configuration is a *blocking configuration* if any straight line, perpendicular to either x or y axes, crossing the rectangle R,

crosses the sensing area of at least one sensor. In other words, a blocking configuration achieves weak coverage of the rectangle R (abbreviated WCR). A non-blocking configuration is said to have *gaps* in the coverage. A given configuration $(R, p_1, p_2, \ldots, p_n)$ is said to be an *integer configuration* if $p_i = [k_i, j_i]$ for some integers k_i, j_i, for every i in the range $1 \leq i \leq n$. We consider both the Euclidean and Manhattan metrics for distance and denote it by $d(x.y)$.

Given an initial configuration $(R, p_1, p_2, \ldots, p_n)$, we study three problems related to finding a blocking configuration:

- **MinSum-WCR problem**: Find a blocking configuration $\{R, p'_1, p'_2, \ldots, p'_n\}$ that minimizes $\sum_{k=1}^{n} d(p_k, p'_k)$, i.e., the sum of all movements.
- **MinMax-WCR problem:** Find a blocking configuration $\{R, p'_1, p'_2, \ldots, p'_n\}$ that minimizes $\max\{d(p_1, p'_1), d(p_2, p'_2), \ldots, d(p_n, p'_n)\}$, i.e., the size of the maximal move among the sensors.
- **MinNum-WCR problem:** Find a blocking configuration $\{R, p'_1, p'_2, \ldots, p'_n\}$ which minimizes the number of indices for which $d(p_k, p'_k) \neq 0$, $1 \leq k \leq n$, i.e., minimizes the number of relocated sensors.

1.2 Our Results

For the MinNum-WCR problem, we show that the problem is NP-complete, even when the initial configuration is an integer configuration, and even when all sensors have range 1. However when the initial configuration is an integer configuration, all sensors have range 0.5, we give an $O(n^{3/2})$ algorithm for solving the MinNum-WCR problem.

When all sensors have the same range, regardless of their initial positions, we give an $O(n \log n)$ algorithm to solve the MinSum-WCR problem using the Manhattan metric. However, the problem is shown to be NP-complete for both Manhattan and Euclidean metrics when the sensors can have different ranges.

Finally, we show that the decision version of the MinMax-WCR problem is NP-complete even for a very restricted case. More specifically, given an integer configuration, with all sensor ranges equal to 0.5, the problem of deciding whether there is a blocking configuration with maximal move at most 1 (using either the Manhattan or Euclidean metric) is NP-complete. This is in sharp contrast to the one-dimensional barrier coverage case where the MinMax problem can be solved in polynomial time for arbitrary initial positions, and heterogeneous sensor ranges.

1.3 Related Work

Barrier coverage using wireless sensors was introduced as a cost-effective alternative to area coverage in [17]. The authors introduced and studied the notions of both *strong* and *weak* barrier coverage in this paper, and studied coverage of a narrow belt-like region. Since then the problem has been extensively studied, for example, see [2, 3, 13, 18, 20].

The problem of achieving barrier coverage using mobile or relocatable sensors was introduced in [8]. The authors studied a line segment barrier and gave a polynomial time algorithm for the MinMax problem when all sensors have the same range, and are initially placed on the line containing the barrier. For the same setting, the case of heterogeneous sensors was shown to be also solvable in polynomial time in [7], and the algorithm of [8] for the homogeneous case was also improved. An $O(n^2)$ algorithm for the MinSum problem with homogeneous sensors is given in [9], and an improved $O(n \log n)$ algorithm is presented in [1]. It was proved in [9] that the MinSum problem is NP-hard when sensors have heterogeneous ranges. The MinNum problem is considered in [19], and shown to be NP-hard for heterogeneous sensors and poly time for homogeneous sensors.

In [10], the complexity of the MinMax and MinSum problems when sensors are initially placed in the plane and are required to relocate to cover parallel or perpendicular barriers is studied. The authors show that while MinMax and MinSum can be solved using dynamic programming in polynomial time if sensors are required to move to the closest point on the barrier, even the feasibility of covering two perpendicular barriers is NP-hard to determine.

A stochastic optimization algorithm was considered in [15]. Distributed algorithms for the barrier coverage problem were studied in [11, 12]. Further, [16] provides algorithms for deciding if a set of sensors provides k-fault tolerant protection against rectilinear attacks in both one and two dimensions. To the best of our knowledge, the problem of weak coverage of a rectangular region (in two directions) has not been studied previously.

2 MinNum-WCR Problem

For a line segment barrier, the MinNum problem was shown to be NP-complete if sensors have different ranges, but if all sensors have the same range, a polynomial-time MinNum algorithm is given in [19]. In this section, we study the MinNum-WCR problem.

2.1 Hardness result

We show that MinNum-WCR is NP-complete, even when all sensors have sensing range 1 and the initial configuration is an integer configuration. We give a reduction from a restricted satisfiability problem, shown to be NP-complete in [4], and defined below:

`3-Occ-Max-2SAT` Problem:
Input: An integer t, a set of boolean variables $x_1, \ldots, x_n$ and a set of clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$, each consisting of a conjunction of two literals, such that each variable appears in *exactly* 3 clauses, and no variable occurs only positively in $\mathcal{C}$, nor only negatively in $\mathcal{C}$.
Question: Does there exist an assignment of $x_1, \ldots, x_n$ that satisfies at least t clauses of $\mathcal{C}$?

Theorem 1. *MinNum-WCR is NP-complete even for integer configurations in which all sensing diameters are equal to* 2.

Proof. Given a `3-Occ-Max-2SAT` instance with variables $x_1, \ldots, x_n$ and clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$, we construct a corresponding instance of the MinNum-WCR problem consisting of a set of sensors S each having radius $r = 1$, and a rectangle R to be covered. Note that $m = \frac{3}{2}n$, since there are $3n$ literals in $\mathcal{C}$ and each clause contains two literals.

R is defined to be a $(6n + 2t) \times (6n + 2t)$ square. The sensor set S contains one sensor s_{i,C_j} for each literal x_i that appears in a clause C_j. We also need two sensors α_i and β_i for each $i \in [n]$. Formally, $S = \{s_{i,C_j} : x_i \text{ occurs in clause } C_j, i \in [n], j \in [m]\} \cup \{\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n\}$. We first describe how the sensors of S are laid out on the y-axis, then on the x-axis. For a sensor $s \in S$, denote by $y(s)$ and $x(s)$ the y and x coordinate of its center, respectively. Figure 2 illustrates the y and x positioning of the sensors.

Each variable x_i has a corresponding gadget Y_i on the vertical axis, where a gadget is simply a set of sensors positioned in a particular manner. Each gadget Y_i covers the vertical range $[6(i-1)..6i]$. Let C_{j_1}, C_{j_2} and C_k be the three clauses in which x_i occurs. Choose j_1 and j_2 such that x_i occurs in C_{j_1} and C_{j_2} in the same manner (either positively in both, or negatively in both), and so that it occurs in C_k differently. Let $z = 6(i-1)$ and let $y(\alpha_i) = z + 1$, $y(s_{i,C_{j_1}}) = z + 2$, $y(s_{i,C_k}) = z + 3$,

$y(s_{i,C_{j_2}}) = z + 4$ and $y(\beta_i) = z + 5$. Observe that $Y_1, \ldots, Y_n$ cover the range $[0..6n]$ on the y-axis, which leaves the range $(6n..6n + 2t]$ uncovered.

Also, note that moving α_i or β_i creates a gap in Y_i. Moreover, s_{i,C_k} can be moved, or both $s_{i,C_{j_1}}$ and $s_{i,C_{j_2}}$ can be moved. However, moving both s_{i,C_k} and one of $s_{i,C_{j_1}}$ or $s_{i,C_{j_2}}$ creates a gap (see Figure 2).

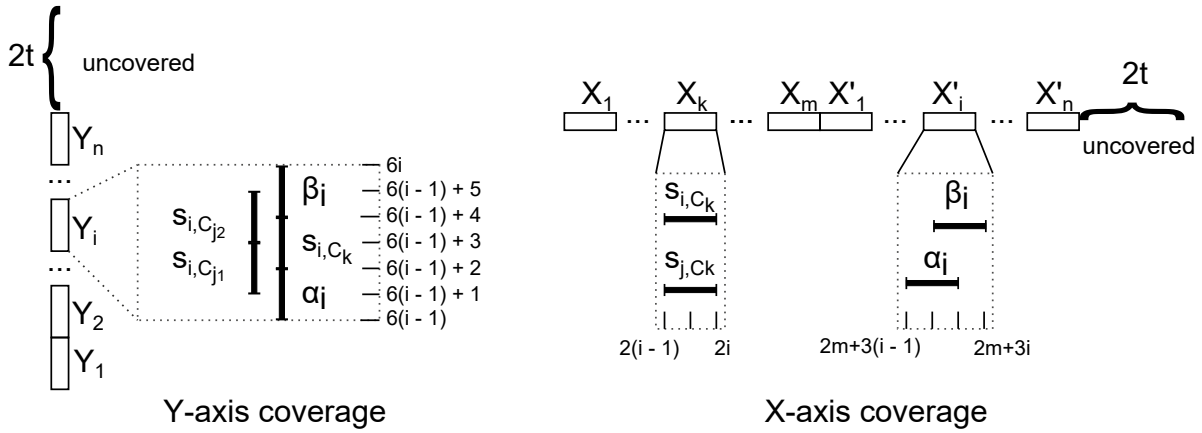

Figure 2: An illustration of the y and x positioning of the sensors in S. The Y-axis subfigure only shows the coverage of the sensors on the Y axis, and does not depict the x coordinates of the sensors (and the X-axis subfigure does not depict the y coordinates). The depicted Y_i gadget corresponds to variable x_i occurring in clauses C_{j_1}, C_{j_2} and C_k, where x_i occurs in the same manner in the first two. An example of these clauses might be $C_{j_1} = (x_i \vee x_1), C_{j_2} = (x_i \vee x_2)$ and $C_k = (\overline{x_i} \vee x_3)$. The depicted X_k gadget corresponds to a clause C_k containing the two variables x_i and x_j. Finally, the depicted X'_i gadget contains α_i and β_i, which are partially overlapping.

We now describe how the sensors are laid out on the x axis. Each clause $C_k \in \mathcal{C}$ has a corresponding gadget X_k that covers the range $[2(k-1)..2k]$, and is constructed as follows. Let x_i and x_j be the two variables occurring in C_k. Then X_k contains the sensors s_{i,C_k} and s_{j,C_k}, and we set $x(s_{i,C_k}) = x(s_{j,C_k}) = 2k - 1$. Thus X_i covers the $[2(k-1)..2k]$ range and $X_1, \ldots, X_m$ cover the range $[0..2m]$. Note that so far every sensor s_{i,C_j} for $i \in [n], j \in [m]$ has been placed. We finally create one gadget X'_i for each pair (α_i, β_i). More precisely, for each $i \in [n]$, let X'_i contain the α_i, β_i sensors, and set $x(\alpha_i) = 2m + 3(i-1) + 1$ and $x(\beta_i) = 2m + 3(i-1) + 2$. Then X'_i covers the range $[2m + 3(i-1)..2m + 3i]$, and $X_1, \ldots, X_m, X'_1, \ldots, X'_n$ cover the range $[0..2m + 3n]$. Recall that $m = \frac{3}{2}n$, and so $2m + 3n = 6n$. Therefore, the x-axis also has the range $(6n..6n + 2t]$ uncovered.

It is not hard to see that this construction can be carried out in polynomial time. We now show that the `3-Occ-Max-2SAT` instance admits an assignment of $x_1, \ldots, x_n$ that satisfies t clauses of $\mathcal{C}$ if and only if it is possible to cover the square R by moving t sensors in the corresponding `MinNum` instance.

($\Rightarrow$): Suppose there is a truth assignment of $x_1, \ldots, x_n$ that satisfies t clauses $C'_1, \ldots, C'_t$ of $\mathcal{C}$. We claim that it is possible to move t sensors and completely cover R. Observe that the uncovered portion R' of the MinNum-WCR instance consists of a $2t \times 2t$ area. For each $k \in [t]$, let x_{i_k} be a variable whose value in the assignment satisfies C'_k (either true or false). Then we move the sensors in the set $S' = \{s_{i_1,C'_1}, s_{i_2,C'_2}, \ldots, s_{i_t,C'_t}\}$, and place them diagonally in R', that is at positions $(6n + 2(i-1) + 1, 6n + 2(i-1) + 1)$ for $i \in \{1, \ldots, t\}$. It is easy to see that R' is now covered. To show that the rest of R remains covered, it suffices now to show that no position that was covered before moving S' has become uncovered.

On the x-axis, each sensor of S' belongs to a clause gadget X_i, and no two sensors of S' belong to the same clause gadget (since we picked exactly one sensor per satisfied clause in S'). Since, for each clause gadget X_i, there are two sensors covering the same x range, moving only one cannot leave a portion of X_i uncovered on the x axis. We deduce that the x-axis is completely covered. As for the y-axis, suppose there is an uncovered position in some gadget Y_i after moving S'. Let C_{j_1}, C_{j_2} and C_k be the clauses containing x_i, where x_i appears in the same manner in C_{j_1} and C_{j_2}. As α_i and β_i were not moved and yet moving S' leaves uncovered positions within Y_i, it must be the case that s_{i,C_k} and one of $\{s_{i,C_{j_1}}, s_{i,C_{j_2}}\}$, say $s_{i,C_{j_1}}$ without loss of generality, were moved. By the construction of S', this implies that the assignment of x_i satisfies clauses C_k and C_{j_1}. However, x_i appears differently in the two clauses (positively in one, negatively in the other), contradicting the validity of the initial assignment. We deduce that the y-axis is completely covered.

($\Leftarrow$): Suppose that the `MinNum` instance allows coverage of the square by moving a set S' of sensors such that $|S'| \leq t$. We claim that the following assignment of $x_1, \ldots, x_n$ is valid and satisfies at least t clauses of $\mathcal{C}$: for each sensor $s_{i,C_j} \in S' \setminus \{\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n\}$, assign to x_i the value that makes it satisfy C_j. If there are unassigned variables afterward, assign them arbitrarily. Note that a variable x_i might be assigned multiple times, so we must show that it is not assigned both true and false.

First note that as each axis has an uncovered range of length $2t$ and each sensor can only cover a range of 2 on both axes, we must have $|S'| = t$. Also note that, by the same argument, each sensor of S' must be moved to R', that is, the $[6n \ldots 6n + 2t]$ range on both the x and y-axes. In particular, no sensor is moved inside the $6n \times 6n$ area that was initially covered before moving S'. Thus, the solution S' cannot contain two sensors s_{i,C_j} and s_{i,C_k} such that x_i occurs differently in C_j and C_k, as this would create a gap in the Y_i gadget on the y axis. This shows that x_i cannot be assigned to both true and false, and so our assignment is valid. Also, no α_i or β_i sensor can belong to S', since otherwise a gap would be created either on the x or y axis. Moreover, S' does not contain two sensors s_{i,C_k} and s_{j,C_k} from the same clause gadget X_k, as this would create a gap on the x axis. We deduce that each sensor of S' belongs to a distinct clause gadget X_k of S'. Combined with the facts that $|S| = t$ and $s_{i,C_j} \in S'$ implies that x_i satisfies C_j, it follows that the constructed assignment satisfies at least t clauses of $\mathcal{C}$. □

2.2 An efficient algorithm for integer configurations

We now show that there is a polynomial algorithm to solve the MinNum-WCR problem for integer configurations when all sensor diameters are equal to 1, and the rectangle to be covered is displaced by 0.5 from the integer grid that contains sensor positions (see Figure 3). It is not hard to see that there always exists an optimal solution to the MinNum-WCR problem which produces a final blocking configuration which is also an integer configuration.

Consider an integer configuration $(R, p_1, p_2, \ldots, p_n)$ as specified above. The position of a sensor is a pair (i, j) where i is said to be the *row* and j is the *column* in which the sensor is located. A row i (or column j) so that no sensor is located in it is called an *row gap* (resp. *column gap*), and some sensor needs to be moved to cover such a row or column. Let r and c be the number of rows and columns gaps in the initial configuration. Clearly in the final blocking configuration, there are no uncovered rows or columns. By moving a sensor, we can cover a row or column gap or both. For example, if row i and column j are both gaps, moving a sensor to position (i, j) covers both row i and column j. However, moving a sensor may also create a new row or column gap. To understand better the net effect of moving a sensor based on the other sensors in its row and column, we introduce the following classification of sensors (see Figure 3 for an illustration).

Definition 1. *Let S_k be a sensor in position (i, j). We say that*

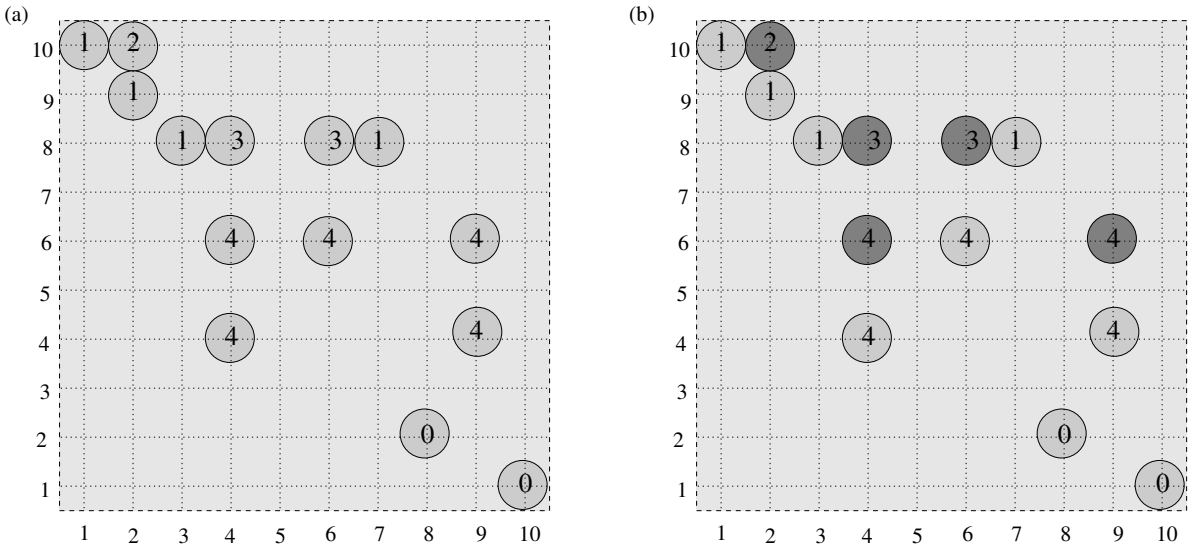

Figure 3: (a) Classification of sensors for the MinNum algorithm. Notice that at most 2 free sensors can be removed from row 6 or column 4 without creating a new gap. (b) A maximum free set of sensors is shown in dark gray.

1. S_k *is* free *if there is at least one other sensor located in row* i *and at least one other sensor in row* j*.*
2. S_k *is of* type 0 *if* S_k *is the only sensor located in row* i *and the only sensor in column* j*.*
3. S_k *is of* type 1 *if either there is another sensor located in row* i *but no other sensor located in column* j *or there is no other sensor located in row* i *but there is another sensor located in column* j*.*
4. S_k *is of* type 2 *if it is a free sensor, and there is at least one sensor of type* 1 *located in row* i*, and also at least one sensor of type 1 in column* j*.*
5. S_k *is of* type 3 *if it is a free sensor, and if there is at least one sensor of type 1 located in row* i *(column* j*) and only free sensors in column* j *(row* j*).*
6. S_k *is of* type 4 *if it is a free sensor, and the only sensors located in row* i *and column* j *are all free sensors.*

We call a move of a sensor a *sliding move* if the final position of the sensor is either in the same row or column as its initial position. We call a move a *jumping* move if the final position is in a different row *and* column from its initial position.

Consider a sensor of type 0. Any sliding move of such a sensor creates an additional row or column gap and can cover at most one other gap. Any jumping move of this sensor creates both a row and a column gap, and it can cover at most one previous row gap and one column gap. Thus the total number of row and column gaps cannot decrease by moving a sensor of Type 0. In what follows, we can therefore assume that sensors of type 0 and the rows and columns in which they reside are not considered any further.

A sensor of Type 1 which has another sensor in its row can make a sliding move in its column and cover a row gap. Any jumping move of this sensor can cover one row and one column gap, but

it creates a column gap. Thus the total number of row and column gaps decreases by at most 1 by moving a sensor of Type 1

However, if there is a row gap i and a column gap j, then a free sensor (of Type 2, 3, or 4) can make a jumping move to position $[i, j]$ and cover both row i and column j, without creating any new row or column gaps. Therefore, moving a free sensor can reduce the total number of row and column gaps by 2. However, moving a free sensor can change the types of sensors in its row or column, and moving *multiple* free sensors can create new empty rows or columns as for example removing all free sensors from row 6 in Figure 3.

Denote the set of free sensors by F. We define a *maximum free set* to be a maximum-sized subset M of free sensors that can all be removed at the same time without creating new row and column gaps (see Figure 3(b) for an example). The following theorem shows that a maximum free set can be found in polynomial time.

Theorem 2. *Given an integer configuration as input, a maximum free set M can be found in $O(n^{3/2})$ time.*

Proof. Define X to be the set of rows and columns that contain only the sensors in F, and let $B \subseteq F$ be a minimum-sized subset of F so that every row (or column) in X contains a sensor in B. We call B a minimum blocking set for X. Then clearly M is a maximum free set if and only if $F - M$ is a minimum blocking set. Therefore, to find a maximum free set, we proceed by finding a minimum blocking set.

Consider a graph $G = (V, E)$ defined as follows. The vertex set V contains a vertex corresponding to every row and column in X; we call the vertex c_i if it corresponds to column i and r_j if it corresponds to row r_j. We also introduce two extra vertices x and y. For every Type 4 sensor at position (i, j), we introduce an edge e_{ij} between the vertices r_i and c_j. For every Type 3 sensor that has only free sensors in its row i (resp. column i), we introduce an edge e_{ix} between vertex r_i (resp. c_i) and the vertex x. Finally, we add the edge e_{xy} between vertices x and y.

We claim that B is a blocking set for X if and only if $E' \cup \{e_{xy}\}$ forms an edge cover in the above graph G, where E' is the set of edges corresponding to sensors in B. To see this, observe that if a sensor of type 4 at position (i, j) is in B, then the corresponding edge e_{ij} covers both vertices r_i and c_j in the graph. Similarly, if a type 3 sensor at position (i, j) that has free sensors only in its row (resp. column) is in B then the corresponding edge e_{ix} covers vertices r_i (resp. c_i) and x. Furthermore e_{xy} covers both x and y. Since B blocks all rows and columns in X, it follows that all vertices in G are covered by $E' \cup \{e_{xy}\}$. Conversely, consider an edge cover in G. It must include the edge $\{x, y\}$ since it is the only edge incident on y. Additionally, any set of edges that covers the remaining vertices in G must be incident on all vertices corresponding to the set X, and therefore corresponds to a set of sensors that blocks X. This completes the proof of the claim.

Since an edge cover can be found via a maximum matching in $O(\sqrt{V}E) = O(n^{3/2})$ time [14], we can find a minimum blocking set, and thereafter, a maximum free set M in $O(n^{3/2})$ time. □ □

Assume that in the given MinNum-WCR configuration there are r row gaps and c column gaps. We can assume without loss of generality that $r \geq c$. We now give an algorithm to solve the MinNum-WCR problem:

Algorithm 1: The MinNum-WCR Algorithm
Input: I an integer configuration with r row gaps and c column gaps, with $r \geq c$

Construct the maximum free set M for I.
Recalculate the types of sensors after removing the sensors in M.
Let k be the number of free sensors in M.

Case $k \geq r$**:** Move c sensors from M using jumping moves to cover the first c row gaps and all c column gaps. Then $r-c$ sensors from M use sliding moves to cover the remaining row gaps. The total number of sensors moved is $c + (r - c) = r$.

Case $c \leq k < r$**:** Move c sensors from M using jumping moves to cover the first c row gaps and all c column gaps. Move the remaining $k - c$ sensors from M to cover $k - c$ row gaps. Finally $r - k$ sensors of Type 1 use sliding moves to cover the remaining row gaps. The total number of sensors moved is $c + (k - c) + (r - k) = r$.

Case $k < c$ **:** Move k sensors from M using jumping moves to cover k row and column gaps. Then we use sliding moves of sensors of Type 1 to cover the remaining row and column gaps. In total $k + (c - k) + (r - k) = r + c - k$ sensors are moved.

Theorem 3. *Given an integer configuration, where all sensor ranges are* 0.5*, and the rectangle R to be covered is displaced by 0.5 in both axes, Algorithm 1 solves the MinNum-WCR problem in $O(n^{3/2})$ time.*

Proof. First we prove that Algorithm 1 produces a blocking configuration. Clearly, the sensors in M can be moved without creating new row or column gaps. Assume now that all sensors in M have been used by our algorithm to cover the gaps , but there still remains a row gap (or column gap). Recall that the number of sensors is assumed to be at least as large as the longer side of the rectangle. Thus, by the pigeonhole principle, there exists a row (or column, respectively) that contains more than one sensor. Such sensors must be of Type 1. One of them can make a sliding move along its column (or row) to cover the row gap, without creating any column gap or any other row gap. This shows that as long as there are gaps, there are sensors of Type 1 available to fill them as needed in the algorithm.

Next we show that Algorithm 1 moves an optimal number of sensors. Given an input configuration with r empty rows and c empty columns, assume without loss of generality that $r \geq c$. Observe that at least r sensors need to be moved to cover all empty rows, thus r is a lower bound on the number of sensors to be moved. If $k \geq c$, Algorithm 1 moves exactly r sensors, and it is optimal in the first two cases.

Suppose instead that $k < c$. At most k row and column gaps can be covered with sensors from M. It follows from the maximality of M that all remaining sensors are not free, that is, moving any of them must be done using a sliding move, which reduces either the number of row gaps or the number of column gaps by at most 1. Thus, in total we need to move $k + (c - k) + (r - k)$ sensors. Therefore Algorithm 2 moves an optimal number of sensors in this case as well.

Given a list of sensors with their coordinates, we can calculate in $O(n)$ time a list of the number of nodes in each row and a list of the number of nodes in each column. By an $O(n)$ scan of these lists we can find all nodes of Type 0 or 1, and we can mark the rows and columns containing sensors of Type 1. Now, an additional $O(n)$-time scan of the lists determines the types of all other nodes. By Theorem 2, a maximum free set of nodes can be constructed in $O(n^{3/2})$ time. After removing

the max free set of sensors M, we can update the types of nodes in $O(n)$ time. Row and column gaps can be found in $O(n)$ time. New positions can be calculated in $O(n)$ time. Thus the total time taken by Algorithm 2 is $O(n^{3/2})$. □

Remark: Notice that the algorithm described in this section assumes (a) that the sensors are initially at integer positions, and (b) that the rectangle R to be covered is displaced from the integer grid by 0.5 in each direction and (c) sensor diameters are 1. All these assumptions are necessary for the algorithm to produce a valid solution.

3 MinSum-WCR Problem

In this section, we study the MinSum-WCR problem. The MinSum problem is known to be NP-hard in the one-dimensional case of heterogeneous sensors on a line segment barrier [9]. The following theorem is therefore an immediate consequence:

Theorem 4. *For a heterogeneous sensor network, the MinSum-WCR problem is NP-complete for both Manhattan and Euclidean metrics.*

If the sensors are homogeneous and we consider the Manhattan metric, then to minimize the sum of the movements we can first minimize the sum of all horizontal movements by applying the known, one-dimensional, $O(n \log n)$ MinSum algorithm [9] to the x-coordinates of the sensors, and second we minimize the sum of all vertical movements by applying this $O(n \log n)$ MinSum algorithm to the y-coordinates of the sensors. It is easy to see that this gives an optimal solution. Thus we have the following result.

Theorem 5. *If all sensor ranges are equal, there is an $O(n \log n)$ algorithm to solve MinSum-WCR for the Manhattan metric.*

Remark: Unlike the algorithm for MinNum-WCR, the algorithm for MinSum-WCR works for arbitrary input configurations, and sensor ranges that are equal but of arbitrary size.

4 MinMax-WCR Problem

As seen in the previous section, the complexity of MinSum-WCR is very similar to the complexity of the MinSum barrier coverage of a line segment. However, this is not the case for the MinMax-WCR problem. For a line segment barrier, the MinMax problem can be solved using an $O(n \log n)$ algorithm even in the heterogeneous case [7]. Surprisingly, the MinMax-WCR problem is NP-hard even for a integer configuration with a very restricted possible move size, as shown in Theorem 6 below.

Throughout this section we will assume that the rectangle to be covered is displaced by 0.5 in both directions, that the given configurations are integer configurations and all sensor diameters are equal to 1 (i.e., ranges are equal to 0.5). A *line* will be used to represent an interval of length 1 (either vertical or horizontal), the line being in the center of the interval. We place a line at every integer coordinate inside the rectangle on both axes. We denote the x and y coordinates of the center of a sensor p by $x(p)$ and $y(p)$, respectively. We then say a sensor p *blocks* a vertical line x if $x(p) = x$, and p blocks a horizontal line y if $y(p) = y$ (i.e. p covers the whole interval corresponding to the line). For convenience, we may define a sensor p as a position, i.e. we may write $p = (x, y)$ to indicate that $x(p) = x$ and $y(p) = y$.

Notice that for integer configurations, with sensor ranges equal to 0.5, if every row and every column of the integer grid overlaying the given rectangle is blocked by a sensor, then this gives a blocking configuration (in particular, blocking line 1 on the x-axis covers the leftmost part of the rectangle, as it is displaced by 0.5). This will be used in the sequel, and in pictures we will only indicate the locations of sensors on the grid. First we need some additional definitions.

Definition 2. *Given configurations $C = (R, p_1, \ldots, p_k)$ and $C' = (R, q_1, \ldots, q_k)$, we define* max-$d$-distance *of C and C' as* $\max_{1 \le i \le k} d(p_i, q_i)$.

Definition 3. *An $a \times b$* integer rectangle*, denoted by $R_{a \times b}$, is the following subset of $\mathbb{Z}^2$:*

$$\{[i, j] \in \mathbb{Z}^2 : \ 1 \le i \le a, \ 1 \le j \le b\}.$$

A vertical line $x = i$, will be referred to as the vertical line i*. Similarly, a horizontal line $y = j$, will be referred to as the* horizontal line j*. An integer configuration C is called* blocking *if for every integer $1 \le i \le a$, vertical line i is blocked by a sensor in C and for every $1 \le j \le b$, horizontal line j is blocked by a sensor in C.*

The proof of NP-completeness of the MinMax-WCR problem will follow from the following results. We first show that the *MinMax (V, H)-Blocking Problem*, a more general version of MinMax-WCR, is NP-hard. We then present a reduction from the (V, H)-Blocking problem to MinMax-WCR.

Definition 4. *Let $R_{a \times b}$ be an integer rectangle, V be a subset of $\{1, \ldots, a\}$ and H a subset of $\{1, \ldots, b\}$. These sets will represent vertical and horizontal lines (V-lines and H-lines, respectively) that need to be blocked. A configuration $T = (R_{a \times b}, q_1, \ldots, q_k)$ is called* (V, H)-blocking *if for every $i \in V$ (respectively, $j \in H$), vertical line i (respectively, horizontal line j) is blocked in T.*

Problem 1 (MinMax (V, H)-Blocking Problem)**.** *Given a distance metric d, an integer rectangle $R_{a \times b}$, an integer configuration $I = (R_{a \times b}, p_1, \ldots, p_k)$, sets $V \subseteq \{1, \ldots, a\}$ and $H \subseteq \{1, \ldots, b\}$, and a real number $D > 0$, decide if there exists a (V, H)-blocking configuration $T = (R_{a \times b}, q_1, \ldots, q_k)$ at max-d-distance at most D from I.*

Lemma 1. *The MinMax (V, H)-Blocking Problem with Manhattan or Euclidean metric is NP-complete for $D = 1$. In addition, the problem remains NP-complete if the instance $I = (R_{a \times b}, p_1, \ldots, p_k)$ with given V and H satisfies:*

(VH1) no sensor of I lies on vertical line a and on horizontal line b,

(VH2) $a \notin V$ and $b \notin H$.

Proof. We will say that a sensor p *partially blocks* a line if p covers a non-empty subset of the interval represented by the line, but does not block it.

Now, for the proof of NP-completeness, it is easy to check that the problem is in NP. We will prove the lemma by reduction from 3-SAT(2,2), a SAT problem with 3-clauses only in which every variable has exactly two positive occurrences and two negated occurrences. This problem was shown to be NP-complete in [5]. Let S be an instance of 3-SAT(2,2) problem with variables $x_1, \ldots, x_n$ and clauses $c_1, \ldots, c_m$. Start with an integer configuration $I = (R_{a \times b})$ with no sensors, where $a = 16n + 4m$ and $b = 24n$, and $V = H = \emptyset$. For each clause and each variable of the instance, we will include in I a "gadget" consisting of a subset of horizontal and vertical lines, some of them included in V and H, some of them shared with other gadgets and a collection of sensors that lie at intersections of these lines. If in a solution T for configuration I, a sensor of a gadget

blocks a horizontal or vertical line, we say the line is blocked by the gadget. Given the initial instance $I = (R_{a\times b}, p_1, \ldots, p_k)$ and a solution $T = (R_{a\times b}, q_1, \ldots, q_k)$, we say that a sensor *moved up* if $y(q_i) < y(p_i)$, it *moved down* if $y(q_i) > y(p_i)$, it *moved left* if $x(q_i) < x(p_i)$ and it *moved right* if $x(q_i) > x(q_i)$.

Construction and properties of gadgets:

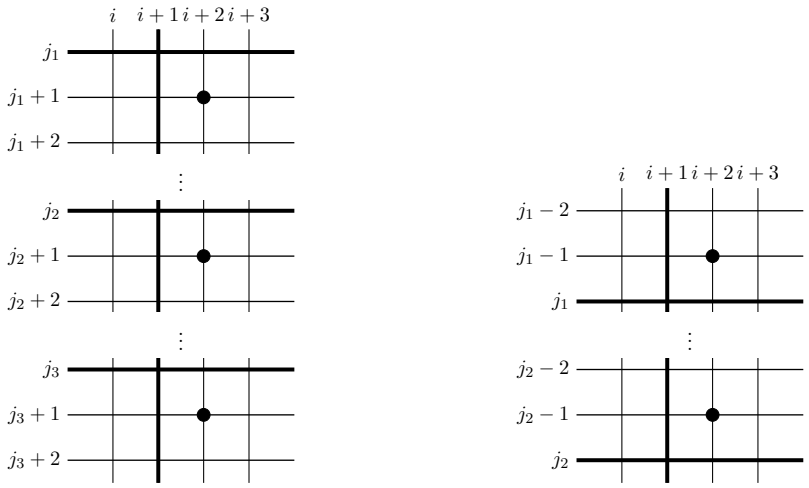

Figure 4: A 3-clause gadget (left) and a 2-clause gadget (right). The H-lines and V-lines of each gadget are depicted in bold. The sensors of the gadget that are added to configuration I are depicted by dots. Note that only the center point of the sensor is represented, and not its whole coverage area.

3-clause gadgets. For a clause of S we construct a *3-clause gadget* as follows. The gadget uses 4 consecutive vertical lines, say $i, i+1, i+2, i+3$ and 3 groups of 3 consecutive horizontal lines, say j_1, j_1+1, j_1+2, j_2, j_2+1, j_2+2 and j_3, j_3+1, j_3+2. The vertical lines are not shared with any other gadget, while only horizontal lines shared with other gadgets are j_1, j_2 and j_3 — each shared with exactly one variable gadget corresponding to one literal of the clause. We add $i+1$ to V, and j_1, j_2, j_3 to H. In addition, we add sensors $(i+2, j_1+1)$, $(i+2, j_2+1)$ and $(i+2, j_3+1)$ to I, cf. Figure 4. Since $i+1 \in V$ and there are no sensors on vertical lines i and $i+1$ (they are not shared with any other gadget) and only sensors on vertical line $i+2$ are the three sensors of this gadget, in any solution T, at least one of them has to move left by the full distance $D = 1$. The sensors of the gadget that do not move left by 1 will correspond to literals that we say are set to value FALSE by T, and remaining of the three sensors of the gadget to literals that are set to value TRUE by T (note that setting a negated occurrence $\neg x$ to TRUE corresponds to setting x to FALSE).

Claim 1. *In any solution T for configuration I, for each clause, one of its literals is set to* TRUE *by T.*

Proof. This follows since at least one sensor of the clause gadget moves left by 1 to block the V-line of the gadget. □

Note that each of the horizontal lines $j_1, j_2, j_3 \in H$, is either blocked by a sensor of the clause gadget (if the corresponding literal is FALSE), or needs to be blocked by a sensor of the corresponding variable gadget, which we now describe.

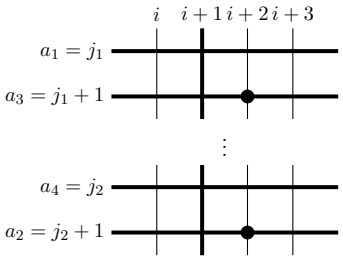

Figure 5: A switch gadget.

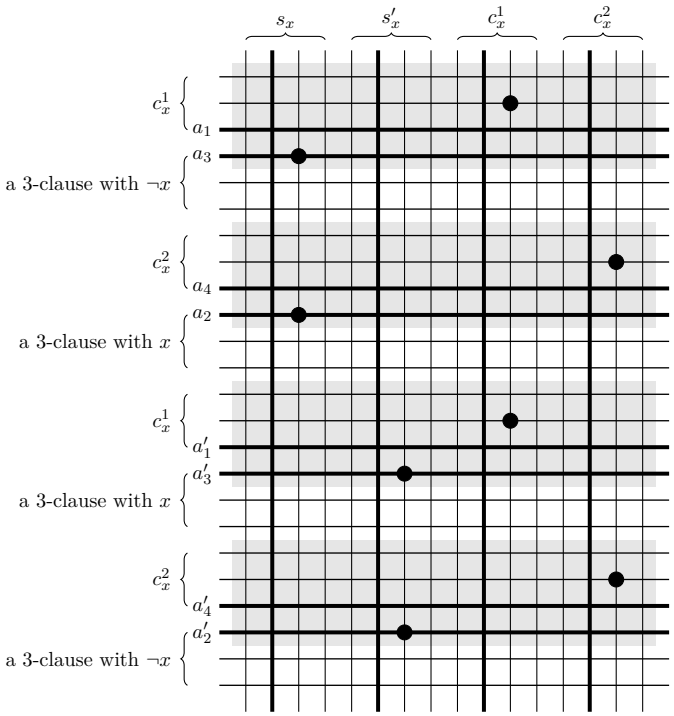

Figure 6: A variable gadget for x (shaded area) consisting of two switch gadgets s_x and s'_x and two 2-clause gadgets c^1_x and c^2_x. Note that some horizontal lines are shared with four 3-clause gadgets: a_2 and a'_3 with 3-clause gadgets containing positive occurrences of x, and a_3 and a'_2 with 3-clause gadgets containing negated occurrences of x. Horizontal lines through unshaded areas do not belong to this variable gadget, but belong one of the four 3-clause gadgets.

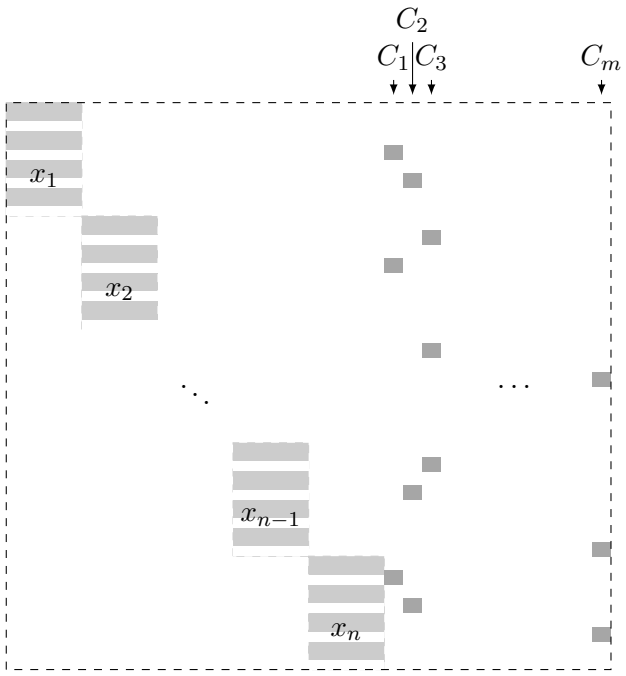

Figure 7: An illustration of an integer configuration for a given 3-SAT(2,2) instance containing clause $C_1 = x_1 \vee x_2 \vee \neg x_n$ and $C_m = x_1 \vee x_{n-1} \vee x_n$. On the left there are n variable gadgets followed by m clause gadgets.

Variable gadgets. Variable gadgets are constructed using *2-clause gadgets*, which are defined in the same way as 3-clause gadgets but have one less group of horizontal lines and are flipped vertically, cf. Figure 4, and *switch gadgets* described as follows. A switch gadget uses 4 consecutive vertical lines, say $i, i+1, i+2, i+3$ and 2 groups of 2 consecutive horizontal lines, say $a_1 = j_1, a_3 = j_1 + 1$ and $a_4 = j_2, a_2 = j_2 + 1$. The vertical lines are not shared with any other gadget. Each of the these horizontal lines is shared with a clause gadget such that $a_1, a_2, a_3, a_4 \in H$, and $j_1 - 1, j_1 + 2, j_2 - 1, j_2 + 2 \notin H$. We add $i + 1$ to V and sensors $(i + 2, j_1 + 1)$ and $(i + 2, j_2 + 1)$ to I (each sensor corresponds to a literal of the clause). As above, at least one of the two sensors has to move left by $D = 1$ in any solution. Consequently, in an integer blocking configuration there are five possibilities which of the horizontal lines $a_1, \ldots, a_4$ are blocked by the sensors of the switch gadget: $\{a_1, a_2\}$, $\{a_2\}$, $\{a_2, a_3\}$, $\{a_3\}$ and $\{a_3, a_4\}$.

We now introduce a variable gadget for each variable x of S. It will consists of two switch gadgets: s_x with horizontal lines $a_1, \ldots, a_4$ and s'_x with lines $a'_1, \ldots, a'_4$, and two 2-clause gadgets: c^1_x with H-lines a_1 and a'_1 and c^2_x with H-lines a_4 and a'_4. To specify the relative position of these four gadgets in $R_{a\times b}$, we will denote i, j_1 and j_2 of gadget g by $i(g)$, $j_1(g)$ and $j_2(g)$. Note that horizontal lines a_1, a'_1, a_4, a'_4 are shared between the two switch gadgets and two 2-clause gadgets. The remaining horizontal lines of the switch gadgets, a_2, a'_2, a_3, a'_3, are shared with four 3-clause gadgets for clauses of S containing x or $\neg x$, cf. Figure 6. Note also that we set $i(s'_x) = i(s_x) + 4$, $i(c^1_x) = i(s_x)+8$, $i(c^2_x) = i(s_x)+12$, $j_1(c^2_x) = j_1(c^1_x)+6$, $j_2(c^1_x) = j_1(c^1_x)+12$ and $j_2(c^2_x) = j_1(c^1_x)+18$. As a result all four gadgets will be placed into a 16×24 rectangle as depicted in Figure 6.

We have described all ingredients of the construction. To complete the construction we will now describe how the gadgets are placed relatively to each other in $R_{a\times b}$. We place the gadget for x_1 to the top left corner of $R_{a\times b}$, the gadget for x_2 diagonally below, and so on, cf. Figure 7. Horizontal lines of the clause gadgets are already determined by placement of variable gadgets (see

the description of variable gadget), so it is sufficient to choose vertical lines of these gadgets. We place them in the order as they appear in S, each taking next the four vertical lines. Note that the construction satisfies that $a \notin V$ and $b \notin H$ and there on sensors on vertical line a and horizontal line b, which justifies the second part of the statement of the lemma.

Before proceeding to the equivalence between instances, we need the following results. We first establish the existence of an integer blocking configuration, assuming that some solution exists, and show that the variable assignments made by such a configuration must be consistent.

Lemma 2. *Suppose that there exists a blocking configuration T for configuration I as constructed above, such that each sensor moves by distance at most $D = 1$ (under either the Manhattan or Euclidean metric). Then T can be transformed, in polynomial time, into an integer blocking configuration T'.*

Proof. We show how to transform T so that every sensor either stays put, or moves in exactly one direction by 1 or -1. We first claim that T can be transformed into a solution T' in which every sensor moves in only one direction, and that the sensors that move horizontally move left by exactly 1. First observe that for every sensor p, there is a unique vertical line $v(p) \in V$ that p can block (i.e. each sensor has a V-line on its left, but not on its right). Moreover, $v(p)$ is at horizontal distance exactly 1 from p. Thus for each $v \in V$, there must be at least one sensor p with $v(p) = v$ such that p moves left by 1 in order to block v (and p does not move vertically). Suppose now that a sensor p does not block $v(p)$, but p has a non-zero horizontal movement within $(-1, 1]$. Then there is another sensor p' that blocks $v(p)$, where p' moves left by 1. Since $v(p)$ is blocked and p has no V-line on its right, we may set the horizontal movement of p to 0 in T' without affecting the set of blocked lines. This proves our claim, since this can be applied to every sensor. Since this claim holds for either the Manhattan or Euclidean metric, the rest of the proof also applies to both since no sensor moves diagonally now.

We now assume that the above transformations have been applied and that solution T satisfies our above claim. We show how each sensor can be made to move by distance 0 or 1. Suppose that there is a sensor that moves by a fractional distance (i.e. in $(0, 1)$). By our assumption, this sensor must move vertically. Note that there are 3 types of sensors: those from 2-clause, 3-clause and switch gadgets. We handle each possible case separately.

First let p be a sensor of a 2-clause gadget c_x^1 or c_x^2. In I, p lies on a line $h \notin H$, with $h+1, h+2 \in H$ and $h-1, h+3, h+4 \notin H$. Moreover there are sensors q on line $h+2$ (from an s_x or s'_x gadget) and r on line $h+3$ (from the 3-clause gadget). Note that there are no other sensors placed on the lines from $h-1$ to $h+4$, and so these are the only three available to block $h+1, h+2$. Suppose that p moves vertically by a fractional amount. If p moves up, then it does not contribute to blocking any line, so its position can be reset and its movement becomes 0 (in both directions, since p is assumed to move only vertically). If p moves down, it only partially blocks line $h+1$, and so sensor q must move up to fill the gap. But then line $h+2$ is only partially blocked, and so sensor r must in turn move up. In other words, all three sensors p, q, r move vertically in solution T, and are not used to block any V-line. We can thus obtain an alternate solution T' by resetting the positions of q and r, and we move p down by 1 so that it covers the line $h+1$. Thus we may assume that each sensor p of a 2-clause gadget satisfies our claim.

Suppose now that a sensor q from a s_x or s'_x gadget moves vertically by a fractional amount. Let $h+2$ be the line that q lies on in I, and define p on line h and r on line $h+3$ as above. If q moves up, then p must move down to block the gap on $h+1$ and r must move up to block the gap on $h+2$. Thus all three of p, q, r move vertically, and the alternate solution obtained by not moving q and r and moving p down by 1 achieves the desired result. If q moves down, then p must

move down by 1 to block $h+1$ and r must move up by 1 to block the gap on $h+2$. Thus q does not contribute to blocking any line and its position can be reset to 0 movement. In this situation all sensors move by a unit amount. This covers the sensors from s_x or s'_x gadgets.

We now assume that all thee above transformations have been performed, so that sensors from 2-clause or switch gadget do not move fractionally. Suppose that a sensor r from a 3-clause gadget lying on line $h+3$ moves vertically by a fractional amount, and define the corresponding sensors p and q as above. If r moves down, it does not contribute to blocking a line and we may reset its position. If r moves up, then it only partially covers line $h+2$, and so q must cover the remaining gap. But as assumed above, q moves vertically by distance either 0 or 1, and so in order to block this gap it cannot move vertically at all. Thus r does not contribute to blocking line $h+2$, and its position can be reset to 0 vertical movement. This concludes the details of the transformation. The fact that all the necessary modifications can be done in polynomial time is straightforward and omitted. □

Note that for $D = 1$, the movement of a sensor in an integer blocking configuration is the same under both the Manhattan or Euclidean metric. Thus the following results hold in both cases.

Recall in the description of 3-clause gadget we set the truth value of each literal. The following claim will help us establish consistency of this assignment for each variable.

Claim 2. *For any integer blocking configuration T for I and any variable x in SAT instance S, the sensors of the variable gadget for x block either horizontal lines a_2, a'_3 and not horizontal lines a_3, a'_2, or vice versa.*

Proof. In an integer blocking configuration, the 2-clause gadget c^1_x can block at most one of the H-lines a_1 and a'_1. Without loss of generality assume that a_1 is not blocked by c^1_x. Then it has to be blocked by switch gadget s_x. Hence, by the discussion above, the set of H-lines blocked by s_x is $\{a_1, a_2\}$. Therefore, a_3 and a_4 are not blocked by s_x. Hence, a_4 must be blocked by 2-clause gadget c^2_x, and hence, a'_4 is not blocked by c^2_x. It follows that s'_x blocks a'_3 and a'_4, and does not block a'_1 and a'_2. This is the first case of the claim. We leave it to the reader to check that assuming a'_1 is not blocked by c^1_x leads to the second case. This concludes the proof of the claim. □

Corollary 1. *For any integer blocking configuration T for configuration I, for each variable x, it cannot happen that a positive literal of x is set to* TRUE *by T and simultaneously, a negative literal of x is set to* TRUE *by T.*

Proof. Assume for contrary that both a positive literal and a negated literal of x are set to TRUE by their 3-clause gadgets. Hence, the H-lines corresponding to these literals are not blocked by these 3-clause gadgets, and must be blocked by the variable gadget for x, which is not possible by Claim 2. □

Equivalence between instances. Let $k = 8n + 3m$. It remains to show that there exists a (V, H)-blocking configuration $T = (R_{a\times b}, q_1, \ldots, q_k)$ at max-d_M-distance at most 1 from I if and only if S is satisfiable. Assume that there exists such a configuration T'. Then by Lemma 2, we can obtain from T' an integer blocking configuration T for I. We define a variable assignment α as follows: $\alpha(x_i) =$ TRUE if at least one of the positive occurrences of x_i in S is set to TRUE by T and $\alpha(x_i) =$ FALSE if both positive occurrences of x_i are set to FALSE by T. This assignment has the following property: if a literal ℓ is set to TRUE by T, then $\alpha(\ell) =$ TRUE (note that it can happen that ℓ is set to FALSE by T and $\alpha(\ell) =$ TRUE). This is clear for positive occurrences of variables from the definition of α. Suppose by contradiction that for a variable x_i, a negated occurrence $\neg x_i$

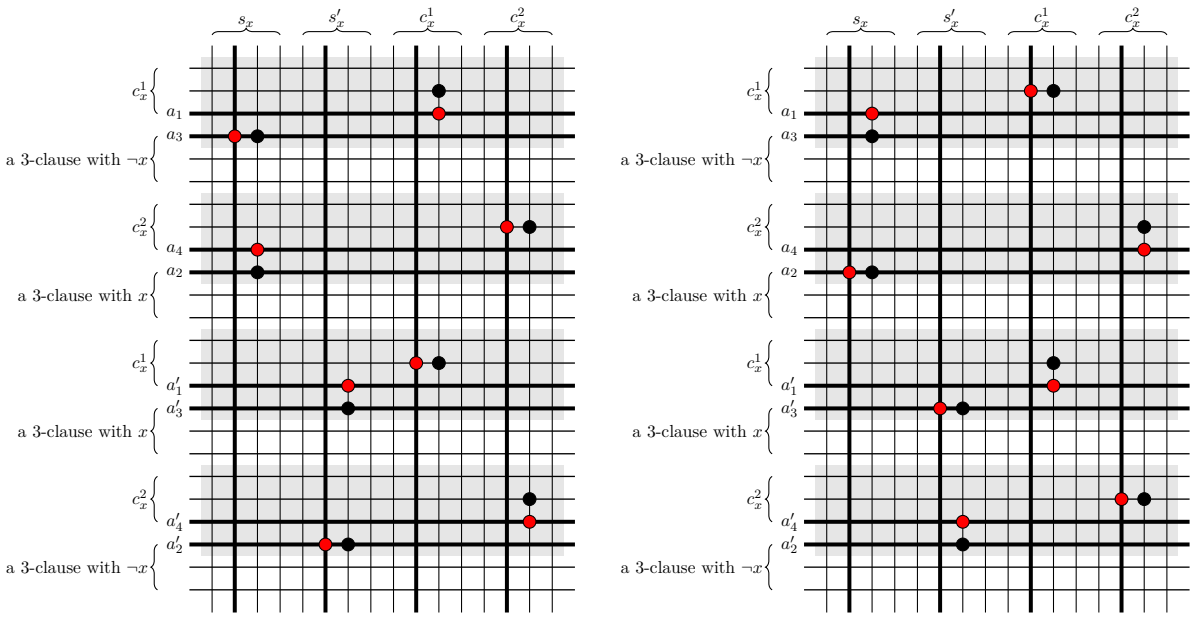

Figure 8: A part of configuration T depicted by red dots: on the left the case when $\alpha(x) = \text{FALSE}$ and on the right the case when $\alpha(x) = \text{TRUE}$.

is set to TRUE by T, but $\alpha(\neg x_i) = \text{FALSE}$. Then $\alpha(x_i) = \text{TRUE}$, and so there is a positive occurrence x_i is set to TRUE by T, which is a contradiction with Corollary 1. Now it follows by Claim 1 that α is a satisfiable assignment for S.

Next, assume that there exists a satisfiable assignment α for S. We will construct a (V, H)-blocking configuration T at maximum distance 1 from I by specifying new positions of sensors for each gadget. As it turns out, T will also be an integer blocking configuration. For the variable gadget of a variable x_i, we distinguish two cases depending on value of $\alpha(x_i)$ as depicted in Figure 8. As one can observe, all V- and H-lines of the gadget are blocked with exception of two H-lines a_2, a'_3 that correspond to the positive occurrences of x_i in case $\alpha(x_i) = \text{FALSE}$ (configuration depicted on the left) or two H-lines a'_2, a_3 that correspond to the negated occurrences of x_i in case $\alpha(x_i) = \text{TRUE}$ (configuration depicted on the right). Note that H-lines corresponding to occurrences of x_i with value TRUE are blocked by the variable gadget.

For each clause, pick a literal ℓ with $\alpha(\ell) = \text{TRUE}$. To obtain T move the sensor corresponding to this literal left and move the remaining two sensors up. The first sensor will block the V-line of the gadget and the remaining two sensors block the two corresponding H-lines. The H-line corresponding to ℓ is not blocked by the clause gadget, but since $\alpha(\ell) = \text{TRUE}$, it is blocked by the variable gadget corresponding to ℓ. Hence, all H-lines corresponding to literals are blocked. Therefore, T is (V, H)-blocking and clearly at maximum distance at most 1 from I. This shows that given I, V and H, it is NP-hard to decide if there exists an integer (V, H)-blocking configuration $T = (R_{a\times b}, q_1, \ldots, q_k)$ at maximum distance at most 1 from I. □

We now present the main result of this section:

Theorem 6. *The MinMax-WCR Problem with Manhattan or Euclidean metric is NP-complete for maximum distance $D = 1$.*

Proof. We will show that the MinMax Problem is NP-complete for $D = 1$ by a reduction from the MinMax (V, H)-Blocking Problem which is NP-complete by Lemma 1.

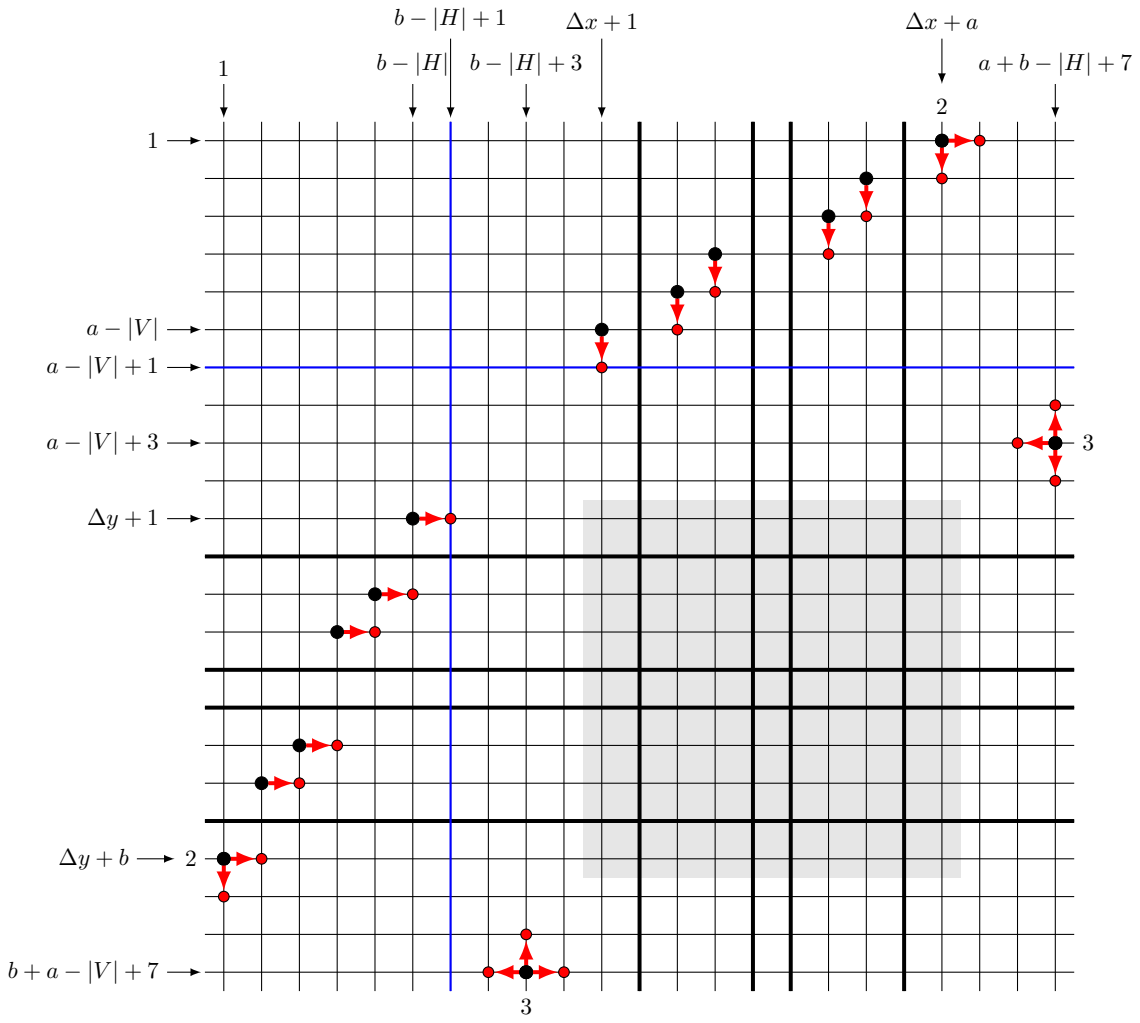

Figure 9: The reduction from (V, H)-blocking problem Movement MinMax. A dot with number c depicts c sensors at the same position. The shaded square represents the original instance I. As before the bold lines depict lines in V and H. The construction guarantees that the sensors around the shaded square have to block all lines not in V and H, and cannot block any lines in V and H, cf. the proof. Blue lines indicate the starting point of this proof.

Let $I = (R_{a\times b}, p_1, \ldots, p_k), H, V$ be an instance of the MinMax (V,H)-Blocking Problem satisfying properties (VH1) and (VH2). Let $R' := R_{(a+b-|H|+7)\times(b+a-|V|+7)}$. We construct a new instance

$$I' = (R', p'_1, \ldots, p'_k, v_1, \ldots, v_{a-|V|+4}, h_1, \ldots, h_{b-|H|+4})$$

of the MinMax Problem as follows. We add $\Delta x := b - |H| + 4$ vertical lines before rectangle $R_{a\times b}$ and 3 vertical lines after $R_{a\times b}$. Similarly, we add $\Delta y := a - |V| + 4$ horizontal lines above $R_{a\times b}$ and 3 horizontal lines below. This defines the rectangle R' of I'. Consequently, every sensor p_i of the original instance I is shifted by $(\Delta x, \Delta y)$, i.e., $p'_i = (x(p_i) + \Delta x, y(p_i) + \Delta y)$. Next, we define sensors $v_1, \ldots, v_{a-|V|+4}$. Let $x = c$ be the i-th vertical line of $R_{a\times b}$ not in V. We set $v_i = (c + \Delta x, a - |V| + 1 - i)$. This defines the first $a - |V|$ sensors of the sequence. Next, add another sensor $v_{a-|V|+1}$ at the same position as $v_{a-|V|}$. Finally, set the remaining 3 sensors of the sequence to $(a+b-|H|+7, a-|V|+3)$. The sensors $h_1, \ldots, h_{b-|H|+3D+1}$ can be set symmetrically, cf. Figure 9.

Next we will show that, under both the Manhattan or Euclidean metrics, there exists a configuration T at maximum distance at most 1 from I that is (V,H)-blocking if and only if there exits a configuration T' at maximum distance at most 1 from I'.

Assume such a $T = (R_{a\times b}, q_1, \ldots, q_k)$ exists. We construct

$$T' = (R', q'_1, \ldots, q'_k, v'_1, \ldots, v'_{a-|V|+4}, h'_1, \ldots, h'_{b-|H|+4})$$

as follows:

- to obtain q'_i move each sensor p'_i in the same direction p_i moved in T;
- for $i \in \{1, \ldots, a - |V|\}$ to obtain v'_i move sensor v_i 1 line down;
- move sensor $v_{a-|V|+1}$ 1 line right, sensor $v_{a-|V|+2}$ 1 line left, sensor $v_{a-|V|+3}$ 1 line up and sensor $v_{a-|V|+4}$ 1 line down;
- to obtain sensors $h'_1, \ldots, h'_{b-|H|+4}$ move sensors $h_1, \ldots, h_{b-|H|+4}$ symmetrically to corresponding cases above; cf. Figure 9.

Clearly, since each sensor in T moved by Manhattan/Euclidean distance at most 1, each sensor in T' has moved Manhattan/Euclidean distance at most 1. Therefore, T' is maximum distance at most 1 from I'. Next, we verify that each vertical line of R' is blocked. The vertical lines $1, \ldots, b - |H| + 1$ are blocked by sensors $h'_{b-|H+1}, \ldots, h'_1$. The next 3 vertical lines are blocked by sensors $h'_{b-|H|+2}, \ldots, h_{b-|H|+4}$. The next a vertical lines (the lines of the original rectangle $R_{a\times b}$) are blocked by either $q'_1, \ldots, q'_k$ or $v'_1, \ldots, v'_{a-|V|}$ depending on whether the line is in V or not. The remaining 3 vertical lines are blocked by sensors $v'_{a-|V|+1}, \ldots, v'_{a-|V|+4}$. Similarly, one can argue that every horizontal line is blocked. Hence, T' is blocking, as required.

For the converse, suppose such a $T' = (R', q'_1, \ldots, q'_k, v'_1, \ldots, v'_{a-|V|+4}, h'_1, \ldots, h'_{b-|H|+4})$ exists. We will first show that sensors $v'_1, \ldots, v'_{a-|V|+4}, h'_1, \ldots, h'_{b-|H|+4}$ do not block lines in V and H. In other words, the moves illustrated in Figure 9 are forced. To this end, consider horizontal line $a - |V| + 1$ (see the blue horizontal line in Figure 9). The only sensor of I' at (Manhattan and Euclidean) distance at most 1 from this line is v_1 (note that there are 3 horizontal lines separating this line from the h-sensors and p'-sensors). Therefore, in T', v'_1 has moved 1 line down. Now, consider horizontal line $a - |V|$. The only sensors at distance at most 1 are v_1 and v_2. Since v'_1 does not lie on this line, v'_2 must lie on it, i.e., it has moved 1 line down. Inductively, it follows that all sensors $v'_1, \ldots, v'_{a-|V|-1}$ have moved 1 line down. Now, consider the vertical line $\Delta x + a + 1$. Since

the instance I satisfies (VH2), the sensors $v_{a-|V|}, v_{a-|V|+1}$ (lying at the same position) lie on the last vertical line of rectangle $R_{a\times b}$. Since I satisfies (VH1), the vertical line $\Delta x + a + 1$ can only be blocked by either $v'_{a-|V|}$ or $v'_{a-|V|+D}$. This sensor must move at least one line right, therefore, it cannot block the horizontal line 2. It follows the remaining sensor of $\{v'_{a-|V|}, v'_{a-|V|+1}\}$ must move 1 line down to block this line. We have that none of these sensors in T' considered so far can block lines in V. The remaining v'-sensors and h'-sensors are at distance more than 1 from V-lines, therefore the V-lines must be blocked by q'-sensors. Symmetrical argument shows that also the H-lines must be blocked by q'-sensors.

Now, we are ready to construct $T = (R_{a\times b}, q_1, \ldots, q_k)$. For each $i \in \{1, \ldots, k\}$, obtain q_i from q'_i by adding vector $(-\Delta x, -\Delta y)$ i.e. set $x(q_i) := x(q'_i) - \Delta x$ and $y(q_i) := y(q'_i) - \Delta y$. Since sensors p'_i and q'_i have distance at most 1, T is at maximum distance at most 1 from I. It is also easy to see that T is (V, H)-blocking, since no sensor outside of $q'_1, \ldots, q'_k$ was used to block any line of the MinMax instance corresponding to V and H after shifting (even partially). Thus $q'_1, \ldots, q'_k$ block the shifted V-lines and H-lines of the instance I, implying that $q_1, \ldots, q_k$ must be (V, H)-blocking. This completes the □

5 Conclusion

In this paper we studied the complexity of establishing weak barrier coverage (WCR) in a given rectangular area using mobile sensors so that the network can detect any crossing of the area in a direction perpendicular to the sides of the rectangle. We considered the three typical optimization measures MinSum, MinMax, and MinNum for movements of sensors. For the MinNum-WCR problem, we show that the problem is NP-hard if sensors have sensing diameter 2, even if sensors are placed initially at integer locations. On the other hand, if sensors of sensing diameter 1 are placed at integer locations, we show an $O(n^{3/2})$ algorithm to solve the problem. For the MinMax-WCR problem, we show that the problem is NP-complete for both Euclidean and Manhattan metrics even when sensors are initially placed in integer locations. For the MinSum-WCR problem using the Manhattan metric, we showed that the problem is NP-complete for heterogeneous sensors, and solvable in $O(n \log n)$ time for homogeneous sensors. The complexity of the MinSum-WCR problem for the Euclidean metric remains unknown.

References

[1] A. M. Andrews and H. Wang. Minimizing the aggregate movements for interval coverage. *Algorithmica*, pages 1–39, 2016.

[2] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length. In *ACM International Conference on Mobile Computing and Networking*, pages 75–86, 2007.

[3] 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.

[4] P. Berman and M. Karpinski. On some tighter inapproximability results. In *International Colloquium on Automata, Lbanguages, and Programming*, pages 200–209. Springer, 1999.

[5] P. Berman, M. Karpinski, and A.D. Scott. Approximation hardness of short symmetric instances of MAX-3SAT. Technical Report TR03-049, ECCC, 2003.

[6] 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. *TCS*, 410(52):5515–5528, 2009.

[7] D. Chen, Y. Gu, J. Li, and H. Wang. Algorithms on minimizing the maximum sensor movement for barrier coverage of a linear domain. *Algorithm Theory–SWAT 2012*, pages 177–188, 2012.

[8] 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. *ADHONOW*, pages 194–212, 2009.

[9] 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. *ADHOCNOW*, pages 29–42, 2010.

[10] 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. *Theoretical Computer Science*, 579:64–73, 2015.

[11] M. Eftekhari, P. Flocchini, L. Narayanan, J. Opatrny, and N. Santoro. Distributed barrier coverage with relocatable sensors. In *Proceedings of Sirocco 2014 conference, LNCS*, volume 8576, pages 235–249, 2014.

[12] M. Eftekhari, E. Kranakis, D. Krizanc, O. Morales-Ponce, L. Narayanan, J. Opatrny, , and S. Shende. Distributed algorithms for barrier coverage using relocatable sensors. *Distributed Computing*, 29(5):361–376, 2016.

[13] M. Eftekhari, L. Narayanan, and J. Opatrny. On multi-round sensor deployment for barrier coverage. In *Proc. of 10th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, pages 310–318, 2013.

[14] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[15] M. Habib. Stochastic barrier coverage in wireless sensor networks based on distributed learning automata. *Computer Communications*, 55:51–0âĂŞ61, 2015.

[16] E. Kranakis, D. Krizanc, F. L. Luccio, and B. Smith. Maintaining intruder detection capability in a rectangular domain with sensors. In *Algorithms for Sensor Systems - 11th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2015, Patras, Greece, September 17-18, 2015*, pages 27–40, 2015.

[17] S. Kumar, T. H. Lai, and A. Arora. Barrier coverage with wireless sensors. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 284–298. ACM, 2005.

[18] 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.

[19] M. Mehrandish, L. Narayanan, and J. Opatrny. Minimizing the number of sensors moved on line barriers. In *IEEE WCNC*, pages 653–658, 2011.

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