On Minimizing the Sum of Sensor Movements for Barrier Coverage of a Line Segment

J. Czyzowicz[1], E. Kranakis[2], D. Krizanc[3], I. Lambadaris[4], L. Narayanan[5], J. Opatrny[5], L. Stacho[6], J. Urrutia[7], and M. Yazdani[4]

[1] Département d'informatique, Université du Québec en Outaouais, Gatineau, QC, J8X 3X7, Canada. Supported in part by NSERC grant.
[2] School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada. Supported in part by NSERC and MITACS grants.
[3] Department of Mathematics and Computer Science, Wesleyan University, Middletown CT 06459, USA.
[4] Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, K1S 5B6, Canada. Supported in part by NSERC and MITACS grants.
[5] Department of Computer Science, Concordia University, Montréal, QC, H3G 1M8, Canada. Supported in part by NSERC grant.
[6] Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby, British Columbia, Canada, V5A 1S6. Supported in part by NSERC grant.
[7] Instituto de Matemáticas, Universidad Nacional Autónoma de México, Área de la investigación cientifica, Circuito Exterior, Ciudad Universitaria, Coyoacán 04510, México, D.F. México. Supported in part by CONACYT grant.

Abstract. A set of sensors establishes barrier coverage of a given line segment if every point of the segment is within the sensing range of a sensor. Given a line segment I, n mobile sensors in arbitrary initial positions on the line (not necessarily inside I) and the sensing ranges of the sensors, we are interested in finding final positions of sensors which establish a barrier coverage of I so that the sum of the distances traveled by all sensors from initial to final positions is minimized. It is shown that the problem is NP complete even to approximate up to constant factor when the sensors may have different sensing ranges. When the sensors have an identical sensing range we give several efficient algorithms to calculate the final destinations so that the sensors either establish a barrier coverage or maximize the coverage of the segment if complete coverage is not feasible while at the same time the sum of the distances traveled by all sensors is minimized. Some open problems are also mentioned.

Key words and phrases: Mobile Sensor, Barrier Coverage, Line segment, Efficient Algorithm, NP-complete, Movement Optimization.

1 Introduction

An important application of wireless sensor networks involves the surveillance of a given region. This surveillance can be done in two different ways: either sensors are be placed throughout the region to monitor the activity in the entire region, or sensors are placed along the perimeter of a region where they establish a barrier that can detect intruders attempting to penetrate the region. Surveillance of a region by such a barrier is more efficient in comparison with complete coverage of the region, since it can be established with fewer sensors at a lower cost. When the perimeter of the region to be monitored is difficult to access or contaminated, it might not be feasible to place sensors right away on the perimeter of a region so that a barrier coverage of the perimeter is achieved. However, in such situation mobile sensors can be used, they can be dropped at some arbitrary initial positions and the mobile sensors are then instructed to move to some specific positions on the border to establish a barrier at the perimeter of the region. Since in sensor networks energy available to a sensor is very limited, one of the main considerations in deployment of sensor networks is the efficient use of energy. Thus, when using mobile sensors to establish a barrier at the perimeter of a region, one would be interested to determine for each sensor a specific position at the border so that sensors in these position establish a barrier coverage and the moves to these position can be done with the minimal possible energy consumption.

In a general setting of the barrier coverage problem, there is a predefined geometric planar region with a well defined boundary and a given set of mobile sensors. Each sensor, say S, has a pre-determined *sensing range* $r(S)$ (determined by the manufacturer). Thus when S is located at location u any other point p in the plane is within the sensing range of the sensor if and only if its Euclidean distance from u is at most $r(S)$. The sensors are initially placed in the plane in arbitrary locations either interior or exterior to the region. They are able to move in any direction in the plane and the energy consumption for movement is similar among the sensors and is proportional to the distance traveled. Starting from these initial positions we are interested in calculating final destination of each sensor so that the sensors in final destinations establish a barrier coverage of the region, i.e., no part of the boundary is outside the sensing range of all the sensors, and the sum of the distances traveled by all sensors is minimized. The above optimization problem, referred to as *MinSum*, represents the minimization of the total energy consumed by all the sensors needed to establish a barrier coverage of the boundary of the given region.

In this paper we restrict our study to the one dimensional barrier coverage problem. We are given a line and the barrier is represented by a finite segment on the line. The sensors are initially located on the line containing the barrier, possibly outside the given barrier. We consider the problem of minimizing the sum of movements of sensors within the line in order to achieve a barrier coverage. We assume that an intruder is a mobile agent that may cross the given barrier from any direction in the plane. As before an intruder can be detected only if it is within the sensing range (range for short) of at least one sensor of the

wireless sensor network and thus the sensor network establishes barrier coverage if every point of the barrier is within the sensing range of at least one sensor.

Although the problem is restricted to a simplified one-dimensional barrier version, it will become apparent in the sequel that it still contains both challenging algorithmic questions and interesting solutions that illustrate the complexity of MinSum barrier coverage in this setting. Clearly we have to have a good understanding of the one-dimensional version before considering the two-dimensional problem.

1.1 Preliminaries and notation

We now give several preliminary concepts and define more precisely several variants of the MinSum barrier coverage problem.

An instance of a barrier coverage problem consists of a closed line interval $I = [0, L]$, the *barrier* to be covered, on the real line with pre-defined endpoints 0 and $L > 0$. We also have n sensors $S_1, S_2, \ldots, S_n$ in initial positions $x_1 \leq x_2 \leq \cdots \leq x_n$ on the line (possibly outside the interval $[0, L]$), and the range of the i-th sensor is a given positive real number $r_i = r(S_i)$, $1 \leq i \leq n$.

Thus the set of points (not necessarily of $[0, L]$) which is within the range of sensor S_i in position x_i is the closed interval $I(S_i, x_i) = [x_i - r_i, x_i + r_i]$ of length $2r_i$. We call it the *covering interval* of S_i. The *total sensor range* of a given instance, denoted R, is the sum of lengths of covering intervals of all sensors, i.e., $R = \sum_{i=1}^{n} 2r_i$.

First of all observe that the barrier coverage problem is *feasible* if and only if the total sensor range R is at least as large as the interval $[0, L]$, i.e., $R \geq L$. In the sequel, we also consider the *non-feasible* case $R < L$. In this case we will be interested in optimizing the sensor movements so that the sensors in the final positions cover either a sub-interval or sub-intervals of I of total length R.

Given an instance of a barrier coverage problem, we call a *gap* a sub-interval of I none of whose points is within range of any sensor and which cannot be enlarged any further. Since the ranges of sensors are assumed to be closed intervals, a gap is an open sub-interval of $[0, L]$, except when one of the endpoints of the gap is either 0 or L. Thus if interval $[a, b]$ is a gap, we assume that a, or b is not a part of the gap unless $a = 0$ or $b = L$. We call an *overlap* either a sub-interval of I which is covered by more than one sensor and which cannot be enlarged any further, or a sub-interval of the line outside I which is covered by a sensor and which cannot be enlarged any further.

Optimization problems. Given an instance of the barrier coverage problem we investigate how to determine the final destinations of the sensors so that the barrier is covered by sensors and the sum of the distances traveled by the respective sensors to their final destinations in minimized. As mentioned before, the sum of distances traveled by sensors corresponds to the total energy needed by sensors to reach the final configuration. More formally, if the i-th sensor S_i moves by a distance m_i (a movement to the left, right will be indicated by $m_i < 0$, $m_i > 0$, respectively) from its original position x_i, the new position will

be $x_i + m_i$ and the new covering interval will be $I(S_i, x_i + m_i)$. If the problem is feasible we are interested in studying the following optimization problem.
MinSum optimization problem $R \geq L$:

$$\text{minimize } \{ \sum_{1 \leq i \leq n} |m_i| \} \text{ subject to } [0, L] \subseteq \bigcup_{i=1}^{n} I(x_i + m_i). \tag{1}$$

When $R < L$ and thus the problem is not feasible, we are interested in a *best effort* solution, i.e., an arrangement of sensors that attains the largest possible coverage of $[0, L]$, while at the same time achieving the MinSum requirements of the movements of the sensors. We call *contiguous* an arrangement of sensors that attains the largest possible coverage of size R as a contiguous sub-interval of $[0, L]$, and *non-contiguous* an arrangement of sensors that attains the largest possible coverage of size R as a collection of possibly disjoint sub-intervals.
Non-contiguous MinSum optimization problem for $R < L$:

$$\begin{aligned} \text{minimize } \{\sum_{i=1}^{n} |m_i|\} \text{ subject to } & \bigcup_{i=1}^{n} I(S_i, x_i + m_i) \subseteq [0, L] \text{ and} \\ & |\bigcup_{i=1}^{n} I(S_i, x_i + m_i)| = R. \end{aligned} \tag{2}$$

Contiguous MinSum optimization problem for $R < L$:

$$\begin{aligned} \text{minimize } \{\sum_{i=1}^{n} |m_i|\} \text{ subject to } & \bigcup_{i=1}^{n} I(S_i, x_i + m_i) \subseteq [0, L] \text{ and} \\ & \bigcup_{i=1}^{n} I(S_i, x_i + m_i) \text{ is an interval of size } R. \end{aligned} \tag{3}$$

1.2 Related work

Several recent papers in the area of sensor networks considered the problem of deployment of mobile sensors for coverage of a region, see for example [11], [12], and [13]. Unlike the problem considered in this paper, they aim to provide coverage of an entire two-dimensional region, and their algorithms do not consider the optimization problems stated above.

The problem studied in our paper is motivated by securing an area by ensuring its border surveillance and intruder detection with a wireless sensor system. [10] proposes efficient algorithms to determine, after sensor deployment, whether a region is barrier covered. It also establishes optimal deployment patterns to achieve barrier coverage when deploying sensors deterministically. In addition, they consider barrier coverage with high probability when sensors are deployed randomly. In [4] the problem of local barrier coverage is introduced and it is shown that it is possible for individual sensors to locally determine the existence of local barrier coverage, even when the region of deployment is arbitrarily curved. Techniques for deriving density estimates for achieving barrier coverage and connectivity in thin strips are introduced in [1], where sensors are deployed

as a barrier to detect moving objects and events. In all these instances the problem studied concerns *static* optimal sensor deployment patterns and there is no concept of mobility of the sensors.

Related to our study is the work in [7] but it does not consider the coverage problem. Also related is a supply and demand problem, known in the literature as *Earth Movers Problem* (or EMP for short), see [5], [3], [9]. Despite some similarities EMP differs from our problem in several respects and the results for EMP cannot be used to solve the barrier coverage problem studied here.

There are two papers which are closely related to our study. The first is [2], where a similar but simpler problem was introduced and studied. Their optimization problem differs from our model in that they do not specify the sensor ranges to be employed; unlike in our paper they seek algorithms to move the sensors to "equidistant" locations on the barrier so as to optimize the efficiency of the barrier coverage regardless of the initial coverage of the sensors. For example, according to their model the n sensors will move from their initial positions to the specific locations $0, \frac{L}{n-1}, \ldots, \frac{iL}{n-1}, \ldots, \frac{(n-2)L}{n-1}, L$, respectively. In our work the algorithms are sensitive to the predefined sensor ranges (which are given as input to the problem) thus accomplishing the same barrier coverage task with less movement than may be done in [2]. Similar observations apply to the other cases of the two dimensional versions of the problem considered in [2].

The second and most directly related research is done in [6] where the same geometric setting is being considered: n sensors on a line that want to establish a barrier coverage of a given line segment by moving the sensors to new positions, but a different optimization measure is being analyzed. Namely, the final positions of sensors that establish barrier coverage minimize "the maximum distance traversed" by any sensor, as opposed to the "sum of the distances covered" considered in the present paper. The motivation for the problem studied in [6] is to minimize the time required to attain coverage while in the problem studied here we minimize the total energy consumed. Despite the apparent similarity of the two problems the results and algorithms are quite different.

1.3 Results of the paper

In this paper we study several interesting variants of the barrier coverage problem obtained by changing assumptions on the sensors and final destinations, e.g., when (a) the sensors may have different ranges, (b) the sensors have identical ranges, (c) the resulting coverage is contiguous or non-contiguous and study the complexity of the proposed algorithms. Several instances of the problem are shown to have efficient algorithmic solutions while others are shown to be NP-complete even to approximate up to constant factor (see Remark 1 after the proof of Theorem 1). Our results are summarized in Table 1 below.

Section 2 presents NP completeness results for MinSum problems for sensors with non-identical ranges. Section 3 deals with sensors of identical ranges. Subsection 3.1 includes the ordering lemma which is basis for the remaining results of the paper. Subsection 3.2 gives algorithms for different versions of the MinSum barrier coverage. The paper concludes with several proposals for possible

	identical ranges		non-identical ranges
coverage	contiguous	non-contiguous	
$R < L$	$O(n)$	$O(n)$	NP-complete
$R = L$	$O(n)$	not applicable	NP-complete
$R > L$	$O(n^2)$	not applicable	NP-complete

Table 1. MinSum problem results for n sensors with barrier of length L and R the total sensor ranges.

extensions as well as related open problems. Due to lack of space, some details of the proofs can be found in the Appendix.

2 NP Completeness Results

In this section we consider the MinSum problems for sensors with non-identical ranges.

Theorem 1. *Let $S_1, S_2, \ldots, S_n$ be n sensors with ranges $r_1, r_2, \ldots, r_n$ located on a line containing segment $[0, L]$, in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$, $\sum_{i=1}^{n} r_i = R \geq L$, and k be a given number. The problem of calculating the movements of sensors on the line so that the sensors cover the segment $[0, L]$ and the sum of movement of the sensors is less than k is NP-hard.*

Proof. We give the proof only for the case $R = L$. The proof for the case $R > L$ is very similar. We prove it by reducing the 3-partition problem (see [8]) to the problem of covering the line segment $[0, L]$ with sensors such that the sum of the movements of the sensors is minimized. The 3-partition problem is defined as follows: we are given a multiset $S = \{a_1 \geq a_2 \geq \cdots \geq a_n\}$ of $n = 3m$ positive integers such that $B/4 < a_i < B/2$ for $1 \leq i \leq n$ and $\sum_{i=1}^{n} a_i = mB$ for some B. The problem is to decide whether S 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.

Let $L = mB + m - 1$ and $k = m(m+1)(B+1)$. Consider a sensor movement problem as shown in Figure 1. We have a sensor S_i of range $a_i/2$ for every $1 \leq i \leq n$ positioned at $-a_i/2$. In addition, we have $m-1$ blocks of sensors of range $1/(2k)$, each block containing k sensors. Each block of these sensors covers a subinterval of $[0, L]$ of size 1, leaving m gaps of size B on the line segment $[0, L]$. Clearly, any solution that covers the segment $[0, L]$ requires that all sensors are moved inside the segment without leaving there any gaps or overlaps, and any solution can be interpreted as a partition of S into subsets, with sensors with range $1/(2k)$ separating the subsets in the partition.

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 we only move sensors $S_1, S_2, \ldots, S_n$ and the three sensors corresponding to triple T_i are moved to fill the ith gap in the interval $[0, L]$. The sum of the moves of the three sensors corresponding to T_i into ith gap is less than $iB + (i-1)$, and the sum of the moves of all sensors for all triples is thus less than $m(m+1)(B+1)/2 = k/2$ in this case. If such a partition does not exist, then any solution to the coverage of the line segment $[0, L]$ corresponds to either: (a) a partition of S into m subsets

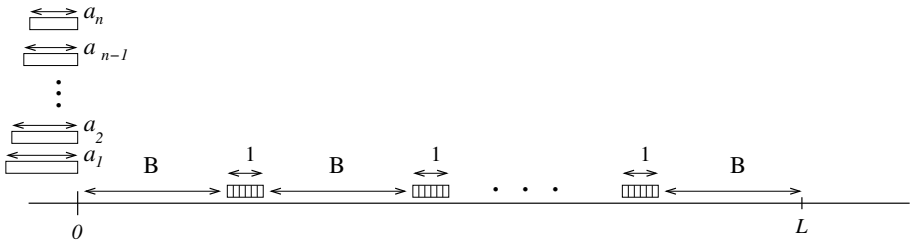

Fig. 1. Sensor arrangement for proving the NP completeness of the MinSum problem.

in which the sum of elements in at least two subsets differs from B by at least 1 in which case we need to move all the sensors in at least one block of sensors with range $1/(2k)$ at least distance 1; or (b) a partition of S into less than m or more than m subsets and this would require one to move at least k of the sensors with range $1/(2k)$ by a distance of 1 or more.

However, moving k of the sensors with range $1/(2k)$ by 1 increases the sum of movements of sensors by at least $k = m(m+1)(B+1)$. Thus the sum of movement of the sensors is less than $k/2$ if and only if the 3-partition problem has a solution. 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 [8], we may assume that the values $a_1, a_2, \ldots, a_n$ are bounded by a polynomial cn^j for some constants c and j. Therefore, $B \leq 3c_1 n^j$ and $k \leq c_2 n^{j+2}$ for some constants c_1 and c_2. Our reduction uses $n + k(m-1)$ sensors and $n + km \leq n + m^2(m+1)B \leq c_3 n^{j+3}$ for some constant c_3. The 3-partition problem can be represented using $O(n \log n)$ bits. In the corresponding sensor movement 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 $O(\log k) = O(\log n)$ bits to represent the position and size of each sensor of size $1/(2k)$. Thus we need $O(n^{j+3} \log n)$ bits to represent the corresponding sensor movement problem, which shows that the transformation is polynomial. □

One can similarly show that when $R < L$ the problem of calculating the movements of sensors on the line so that the sensors give a maximal coverage the segment $[0, L]$ and the sum of movement of the sensors is less than k is NP-hard.

Remark 1. The proof of the above theorem also shows that if $NP \neq P$ there is no polynomial 2-approximation algorithm for the MinSum problem, since the result of a 2-approximation algorithm for sensor movements would be less than k if and only if the corresponding 3-partition problem has a solution. Clearly, the proof can be modified to show the non-existence result for any constant factor approximation algorithm.

3 Sensors with Identical Ranges

In view of the NP-complete results of the previous section, we consider in this section the MinSum problem for sensors of identical range, say r.

3.1 Ordering Property of Optimal Configurations

An important observation that will be useful in the MinSum optimization problem concerns the order of final positions of sensors in an optimal configuration. It is shown below that there exists an optimal solution of the MinSum problem so that the final destinations of sensors preserve the initial ordering of sensors. In other words, two sensors on their way to the optimal locations do not have to cross paths.

Lemma 1. *Let $x_i \leq x_j$ and $y_i > y_j$ be real numbers.*

$$|x_i - y_i| + |x_j - y_j| \geq |x_i - y_j| + |x_j - y_i| \tag{4}$$

Proof. It can be easily proved by considering the five possible arrangements of values x_i, x_j, y_i, y_j. □

Lemma 1 implies that there exists an optimal solution of the barrier coverage problem which preserves the initial order of position of the sensors.

Corollary 1 (Order Preservation). *For any of the MinSum optimization problems, if $x_1 \leq x_2 \leq \cdots \leq x_n$ are the initial positions of sensors $S_1, S_2, \ldots, S_n$ of identical range then there exists an optimal solution of the problem such that the final destinations of sensors satisfy $y_1 \leq y_2 \leq \cdots \leq y_n$, respectively.*

According to the order preservation lemma the MinSum problem is trivial when $R = L$ and thus we consider below only the cases $R > L$ and $R < L$.

3.2 Algorithms for MinSum Barrier Coverage

We now propose several efficient algorithms for sensors with identical ranges. We start with the Contiguous MinSum problem, $R < L$. We first give an $O(n)$ algorithm for maximal contiguous coverage of the line with n sensors which minimizes the sum of the movements of the sensors.

We say that sensors S_i and S_{i+1} are in *attached position* if the difference between their positions is equal to $2r$, i.e., there is no gap or overlap between the two sensors.

Lemma 2 (On an infinite line). *Let $S_1, S_2, \ldots, S_n$ be n sensors with identical range r located on a line in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ with $R < L$. There is an $O(n)$ algorithm that calculates the movements of sensors on the line so that the sensors cover a segment of size $2rn$ and the sum of movements of the sensors is minimized.*

Proof. Let $y_1, y_2, \ldots, y_n$ be positions on the line such that $\sum_{i=1}^{n}(|x_i - y_i|$ is minimal among all such possible assignment of values. According to Lemma 1, there is an optimal solution such that $y_1 < y_2 < \ldots < y_n$. Furthermore, since the sensors cover a contiguous segment of the line, we have $y_i = y_1 + 2(i-1)r$ for $2 \leq i \leq n$. In fact, our algorithm determines a solution of this type.

Consider the possibility that the sensors $S_1, S_2, \ldots, S_n$ have moved to positions $y_1 = 0, y_2 = 2r, \ldots, y_n = 2(n-1)r$, respectively, on the line, i.e., the sensor S_1 is moved to location 0 and the other sensors are moved to attached positions following the initial order of sensors. Then the values $-x_1, 2r - x_2, \ldots, 2(n-1)r - x_n$ give the displacements of the sensors. Let l_1 be the number of sensors that move left, l_2 be the number of sensors that move right, l_3 be the number of sensors that remain stationary in this assignment, and $shifts_0$ be the sum of the absolute values of all shifts when S_1 is in position 0. If $l_1 > l_2 + l_3$ then consider the assignment of positions to sensors by shifting all positions of the sensor to the right by c where c is the smallest negative shift. In this assignment all left shifts of sensors are decreased by c, all the right shifts of sensors are increased by c and the zero shifts become c. Thus in this assignment the sum of the absolute values of all shifts, say $shifts_c$, is equal to $shifts_0 - c(l_1 - l_2 - l_3)$, which is smaller than $shifts_0$. Similarly, if $l_2 > l_1 + l_3$ then the assignment of positions by shifting positions of all sensors to the left by c, where c is the smallest positive shift, we obtain an assignment of positions to sensors in which the sum of the absolute values of all shifts is smaller than $shifts_0$. Thus we obtain an optimal assignment of positions to sensors when $l_1 \leq l_2 + l_3$ and $l_2 \leq l_1 + l_3$. By finding the median of $-x_1, 2r - x_2, \ldots, 2(n-1)r - x_n$ and shifting the configuration to the right or left by the median value so that the median of all the shifts becomes 0 we obtain an assignment that minimizes the sum of shifts. Clearly, the value of the median of n values can be calculated in $O(n)$ and so can the n shifted values of positions of the sensors. □

When sensors are in the positions determined by the algorithm of the previous lemma they give a maximal contiguous coverage of a segment of a line which minimizes the sum of all shifts, but not necessarily of the segment $[0, L]$. However, when $R < L$ we can easily modify the solution above so as to solve the MinSum contiguous problem by shifting the solution into the segment $[0, L]$ if the segment covered by the sensors from Lemma 2 is not in it already and we obtain the following theorem.

Theorem 2. *Let $I = [0, L]$ be a line segment and $S_1, S_2, \ldots, S_N$ be sensors with identical range r located on a line in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ and $R < L$. There is a $O(n)$ algorithm to solve the Contiguous MinSum problem for $R < L$.*

MinSum problem, $R > L$: The optimal solution of this problem is more difficult to obtain, since it does not correspond to sensors being in attached positions. We give below an algorithm for the MinSum problem that is of time complexity $O(n^2)$. This algorithm is more complex to state and to verify its correctness and thus we break it into several lemmas.

Given an instance of the barrier coverage problem with $R > L$ we enumerate the gaps in the interval $[0, L]$ as $g_1.g_1, \ldots g_l$ from the left. Informally, the algorithm considers the gaps in the given interval $[0, L]$ in the left to right order. It eliminates each gap by removing the overlaps to the left and right of the gap in the inside-out manner, removing at every step the overlap whose "cost" is the lowest among the available gaps. The cost is related to the number of sensors whose positions must be shifted when eliminating the overlap.

Let A be an algorithm that solves the MinSum problem. We say that the algorithm is *locally optimal* with respect to gaps $g_1, g_2, \ldots, g_k$, $1 \leq k \leq l$, if the sum of moves of the sensors needed to eliminate gaps g_1, g_2, g_k is minimal, without creating any new gap or increasing the size of the other gaps.

Consider an instance of the MinSum problem for $R > L$ with sensors of identical range. We enumerate the overlaps of the sensor ranges from left to right as $o_1, o_2, \ldots, o_k$, each overlap is either the interval corresponding to the nonempty intersection of the ranges of two consecutive sensors, if the intersection is inside I, or the nonempty intersection of the range of a sensor with $(-\infty, 0)$ or (L, ∞). Thus all of a sensor range outside of the interval $[0, L]$ it treated as an overlap. When moving a sensor, say S_i, in order to achieve a contiguous coverage of the interval $[0, L]$, we assign to it a real number d_i, indicating the difference between the present and initial position and we call it its *shift value*. Thus negative values correspond to moving sensors to the left, while positive values correspond to moving sensors to the right.

Clearly, at any stage of the algorithm the sum $\sum_{i=1}^{n} |d_i|$ gives the cost of the moves of sensors performed so far. See Figure 2 for a possible initial and final configuration, including the shifts and overlaps.

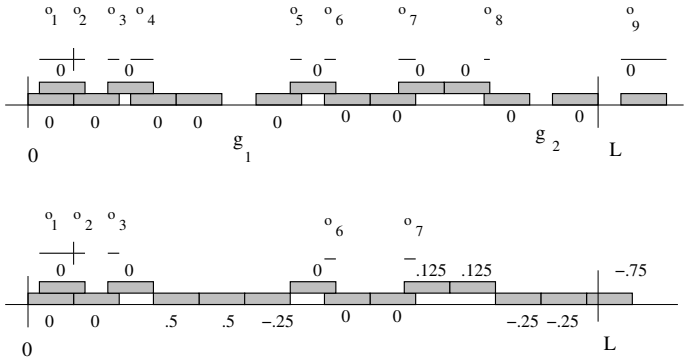

Fig. 2. Example of initial and final configurations

We first provide some claims concerning the necessary properties of any locally optimal solution of the problem, which will also form the foundations of our algorithm.

Lemma 3. *Consider a locally optimal solution with respect to gaps $g_1, g_2, \ldots, g_k$, $1 \leq k \leq l$. Then in this solution, no new overlaps are created. Furthermore, an*

Fig. 3. Forbidden sensor configurations in a locally optimal solution

initial overlap inside $[0, L]$ *cannot be moved left or right and its size cannot be increased. Thus any locally optimal algorithm can only eliminate an existing overlap or make it smaller by moving its left sensor to the left, or (and) the right sensor to the right, or it leaves the overlap exactly as it is initially.*

Proof. Let S_i and S_{i+1} be two overlapping sensors in a locally optimal solution. If $d_i > 0$, we could decrease the cost of the solution by moving S_i slightly to the left (see (a) of Figure 3), since S_i moved too much to the right. Similarly if $d_{i+1} < 0$, we could decrease the cost of the solution by moving S_{i+1} slightly to the right (see (b) of Figure 3). Thus the configurations in Figure 3 cannot occur in a locally optimal solution. However, creation of a new overlap necessarily corresponds to (a) or (b) of Figure 3. Similarly, moving an initial overlap left or right, or increasing its size necessarily corresponds to either (a) or (b) of Figure 3. □

Lemma 4. *Consider a locally optimal solution with respect to gaps* $g_1, g_2, \ldots, g_k$, $1 \leq k \leq l$. *Let* $S_i, S_2, \ldots, S_j$ *be the sensors in the portion of the solution which does not contain gaps any more and let* $d_i, d_{i+1}, \ldots, d_j$ *be the sequence of shift values of these sensors. If for some* m, $1 \leq m \leq j-1$ *we have* $d_m > 0$ *and* $d_{m+1} \geq 0$, *or* $d_m \leq 0$ *and* $d_{m+1} < 0$, *or* $d_m > 0$ *and* $d_{m+1} < 0$, *then* S_m *and* S_{m+1} *are in attached position in the solution.*

Proof. Since S_m and S_{m+1} are in the part where gaps were eliminated, they are either attached or they overlap. An overlap of these two sensors in either case would be one of the forbidden configurations in Figure 3. □

Lemma 5. *Let* $S_i, S_{i+1}, \ldots, S_j$ *be a sequence of sensors that in the initial configuration does not contain any gap but it contains overlaps* $o_k, o_{k+1}, \ldots, o_l$. *Let* m_t *be the integer such that* S_{m_t} *and* S_{m_t+1} *are the two sensors that form overlap* o_t, $k \leq t \leq j$. *If in a locally optimal solution overlap* o_t *is either eliminated or made smaller by moving the right sensor* S_{m_t+1} *to the right, then in this solution all overlaps* $o_{t+1}, \ldots, o_l$ *have been eliminated by moving the sensors* $S_{m_t+1}, S_{m_t+2}, \ldots, S_j$ *to the right into attached positions. If in a locally optimal solution overlap* o_t *is either eliminated or made smaller by moving sensor* S_{m_t} *to the left, then in this solution all overlaps* $o_k, o_{k+1} \ldots, o_{t-1}$ *have been eliminated by moving the sensors* $S_i, S_{i+1}, \ldots, S_{m_t}$ *to the left into attached positions.*

Proof. If overlap o_t is either eliminated or made smaller by moving S_{m_t+1} to the right, then all sensors to the right of it until the next right gap must be moved to the right so that we do not create a new overlap, which is forbidden by Lemmas 3. Since the shift values of $S_{m_t+1}, S_{m_t+2}, \ldots, S_j$ are all positive, they must be all in attached positions by Lemma 4. The proof of the second part of the lemma is analogous. □

Lemmas 3, 4, 5 above form the basis for the design of our MinSum algorithm.

Main Algorithm: MinSum algorithm for $R > L$:
Our algorithm proceeds by closing the gaps from left to right producing a locally optimal solution. For each gap, say g, we search to the left and to the right from the gap to find the "closest overlaps", say o_i and o_j of sensor ranges on each side of the gap that can be used to shrink the gap. For each of these two overlaps we calculate the cost of using o_i or o_j to shrink the gap, the cost being equal to the *number of sensors* that are being shifted. At any time the cheapest of the two overlaps is used to shrink the gap. The sensors that are moved in the shrinking process are put in attached position, unless the gap is smaller than the overlap. The pseudocode for the main algorithm is as follows.

Algorithm MinSum

Input: L and the initial positions $x_1, x_2, \ldots, x_n$ of sensors (assumed sorted).
Output: The final positions $y_1, y_2, \ldots, y_n$ of sensors for the contiguous coverage of the interval $[0, L]$ that minimize the sum of movements.

1: initialize array $d_1, d_2, \ldots, d_n$ of sensor shifts to 0;
2: scan $x_1, x_2, \ldots, x_n$ and calculate the sequence of overlaps $o_1, o_2, \ldots, o_k$, the sequence of gaps g_1, g_2, g_l, and their sizes;
3: for $i := 1$ to l do //eliminate Gap i
  repeat
    find o_j, the closest overlap left of g_i and its cost w.r.t. g_i;
      (if there is no such overlap, set the cost to ∞).
    find o_k, the closest overlap right of g_i and its cost w.r.t. g_i;
      (if there is no such overlap, set the cost to ∞).
    if ($cost(o_j) \leq cost(o_k)$) then //right shift is done
      { if $size(o_j) < size(g_i)$ then $c := size(o_j)$ else $c := size(g_i)$;
        $size(g_i) := size(g_i) - c$;
        $size(o_j) := size(o_j) - c$;
        add c to the values in array d of sensors between o_j and g_i;
      }
    else //left shift is done
      { if $size(o_k) < size(g_i)$ then $c := size(o_k)$ else $c := size(g_i)$;
        $size(g_i) := size(g_i) - c$;
        $size(o_k) := size(o_k) - c$;
        subtract c from the values in array d of sensors between g_i and o_k;
      }
  until $size(g_i) = 0$;
4: for $i := 1$ to n do
  $y_i = x_i + d_i$; //the final positions of the sensors.

Now we can state the main theorem.

Theorem 3. *Let $S_1, S_2, \ldots, S_n$ be sensors with identical range r located on a line in initial positions $x_1 \leq x_2 \leq \ldots \leq x_n$ (not restricted to lie inside the segment $[0, L]$) and $R > L$. Algorithm MinSum above solves this instance of the MinSum problem in time $O(n^2)$.*

When calculating the cost of a shift for the overlap o_i on the left of the present gap, we have to take into account the fact that shifting those sensors to the right whose shift is negative at present is actually equivalent to undoing a left shift that was done when removing another gap to the left of the present gap. Thus shifting these sensors with negative moves to the right is decreasing the cost of the sum of movements done so far. Another factor that needs to be considered is the difference between overlaps of sensors inside interval $[0, L]$ and overlaps that are outside this interval. Therefore, we need to define the cost of moving a portion p of overlap s_j to the right and left, respectively. Due to the page limit, these Definitions 1 and 2 and the detailed proof of Theorem 3 are given in the Appendix.

4 Conclusion and open problems

We have studied the barrier coverage problem for a wireless sensor network when the perimeter to be covered is a finite line segment. In view of the results, an interesting problem is to study the barrier coverage by sensors with limited number of different ranges. For the case of a one dimensional barrier, one could consider the problem of barrier k coverage, whereby each intruder should be detected by at least k different sensors, for some fixed $k > 1$. Also, the possibility that the perimeter consists of several line sub-intervals could be investigated. Another class of problems concerns extensions to higher dimensions.

The two dimensional version of the problem is wide open. Specifically, one might consider other geometric barriers, e.g., circular barriers, convex barriers or boundaries of simple polygons. Also one might consider other types of sensor movements, e.g., the movement of the sensors towards the globally optimal position on the circular barrier may proceed through the interior of the circle as opposed to only moving on the perimeter.

Another interesting class of problems would be to examine the above questions in light of a "decentralized" sensor communication model. Finally, it would be interesting to investigate how to optimize other more realistic energy consumption metrics, e.g., sum of squares of movements of all the sensors.

Bibliography

[1] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 75–86, 2007.

[2] 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, to appear. Also in proceedings of 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA'08) held August 21-24, 2008, in St. John's, Newfoundland, Canada. LNCS*, 2008.

[3] S. Cabello, P. Giannopoulos, C. Knauer, and G. Rote. Matching point sets with respect to the Earth Mover's Distance. *Computational Geometry: Theory and Applications*, 39(2):118–133, 2008.

[4] A. Chen, S. Kumar, and T.H. Lai. Designing localized algorithms for barrier coverage. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 63–74, 2007.

[5] S. Cohen. *Finding Color and Shape Patterns in Images.* PhD Thesis, Stanford University, Dept. of Computer Science, 1999.

[6] 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 8th International Conference on Ad Hoc Networks and Wireless, September 22-25, Murcia, Spain, 2009, SVLNCS*, 5793:194–212, 2009.

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

[8] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* WH Freeman, San Francisco, 1979.

[9] O. Klein and R.C. Veltkamp. Approximation Algorithms for Computing the Earth Mover's Distance Under Transformations. *LNCS*, 3827:1019, 2005.

[10] S. Kumar, T.H. Lai, and A. Arora. Barrier coverage with wireless sensors. *Wireless Networks*, 13(6):817–834, 2007.

[11] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Localized sensor self-deployment with coverage guarantee. *ACM SIGMOBILE Mobile Computing and Communications Review*, 12(2):50–52, 2008.

[12] Shuhui Yang, Minglu Li, and Jie Wu. Scan-based movement-assisted sensor deployment methods in wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 18(8):1108–1121, 2007.

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

Appendix

Proof of Theorem 3. First, we need to introduce some notation for the algorithm and state the properties of optimal solutions.

Definition 1. *(***Right-shift cost***) Let g be a gap, o be the closest overlap of sensors to the left of the gap g and $x_i, x_{i+1}, \ldots, x_j$ be the sequence of sensors starting with rightmost sensor of the overlap until the gap g. Let p be the size of o if there is no negative values in $d_i, d_{i+1}, \ldots, d_j$ or p be the absolute value of the smallest left shift in the sequence $d_i, d_{i+1}, \ldots, d_j$. If o is in the interval $[0, L]$ we define the cost(o) of size p with respect to g to be the* difference *between the number of sensors in $x_i, x_{i+1}, \ldots, x_j$ that have non-negative shifts so far and the number of sensors with negative shifts. If o is outside the interval $[0, L]$ we define the cost(o) of size p with respect to g to be the addition of the distance of o to 0 and of the* difference *between the number of sensors in $x_i, x_{i+1}, \ldots, x_j$ that have non-negative shifts so far and the number of sensors with negative shifts.*

In example in Figure 4 (a) we obtain that *cost*(o) of size .5 with respect to g is equal to $4 - 2$. In Figure 4 (b) overlap o is outside the interval $[0, L]$ at distance 0.5 to 0. Thus its cost to g of size 1 is equal to $.5 + 4 - 2$.

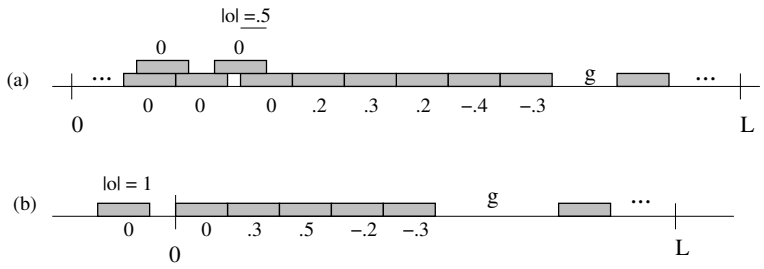

Fig. 4. Examples of right-shift costs

We include a similar cost increase for overlaps outside $[0, L]$ in the definition of the left-shift costs.

Definition 2. *(***Left-shift cost***) Let g be a gap, o be the closest overlap of size x to the right of g and i be number of sensors from g up to the overlap o and including the leftmost sensor of the overlap. If o is in the interval $[0, L]$ we define the cost(o) of size x with respect to g to be equal to $(j + 1 - i)$. If o is outside the interval $[0, L]$ we define the cost(o) of size x with respect to g to be equal to $(j + 1 - i) + b$ where b is the distance of o to L.*

In Figure 5 (a) we obtain that *cost*(o) of size 0.25 with respect to g is equal to 3. In Figure 5 (b) overlap o is outside the interval $[0, L]$ at distance 0.5 to L. Thus its cost to g of size 0.75 (g cannot use more of it) is equal to $0.5 + 4$.

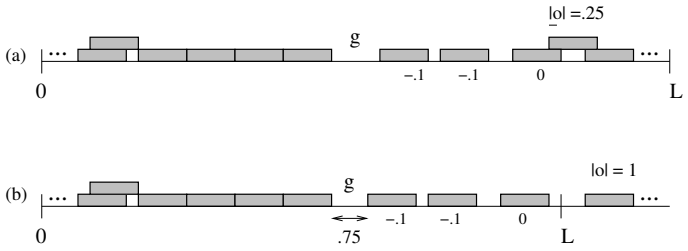

Fig. 5. Examples of left-shift costs

The difference in the definition of the left-shift and right-shift costs is due to the left-to-right processing of the gaps. The left shifts involve moves of sensors whose shift values are all zero or negative, while right shifts involve moves of sensors whose shift values are zero, positive, or negative. Also due to the left-to-right processing of the gaps, when a right shift is done, it is done over a sequence of sensors that are in attached positions and thus these sensors are shifted right as one block in attached positions. This does not need to be the case of a left shift in case when for a gap g_i the closest overlap to the right of it is located to the right of gap g_{i+1}. In such a case, since we are interested in a locally optimal solution with respect to gaps $g_1, g_2, \ldots, g_i$, we simply shift to the left all sensors from the overlap up to the gap by the same amount, thus leaving attached sensors in attached positions, and leaving gaps to the right of g_i of the same size.

Proof. (Theorem 3) We use below $o : p$ to denote a part of the overlap o of size p. We show the correctness of our MinSum algorithm by induction on the gap sequence. Consider the elimination of the first gap by the algorithm. The algorithm considers the relevant overlaps in the inside-out order, and at every step chooses the elimination of the overlap of the lowest cost and the shifted sensors are put in the attached positions as needed by Lemmas 4 and 5. Thus, the algorithm eliminates a subsequence $(o_{i_1} : p_{i_1}, o_{i_2} : p_{i_2}, \ldots)$ of overlaps $o_1, o_2, \ldots, o_k$ in this process, the overlaps at the ends of the subsequence could be eliminated only partially. The total size of overlaps in the subsequence is equal to $|g_1|$. In view of Lemmas 4 and 5, any locally optimal algorithm A would need to eliminate a subsequence $(o_{i'_1} : p_{i'_1}, o_{i'_2} : p_{i'_2}, \ldots)$ of overlaps $o_1, o_2, \ldots, o_k$ and the total of eliminated overlaps is equal to $|g_1|$. Assume that $i'_1 < i_1$, Then necessarily $i_2 < i'_2$ but our algorithm did not use $o_{i_1} - 1$ since its cost is higher than the cost of o_{i_2}, and clearly, cost of $o_{i'_1}$ is at least as large as the the cost of $o_{i_1} - 1$. Thus, we could improve the solution obtain by A by using o_{i_2} instead of $o_{i'_1}$ in the elimination of g_1. Therefore, the elimination of the first gap by our algorithm is locally optimal. Similarly a contradiction is obtained if $i'_2 < i_2$.

Assume that the algorithm eliminated gaps $g_1, g_1, \ldots, g_i$ so that the cost is locally optimal. Consider the elimination of the g_{i+1}. If no overlaps remain to the left of g_{i+1}, all the shifts are done from the right in inside out order and this is obviously optimal. Let $o_j, o_{j+1}, \ldots$ be overlaps that remain to the right of the

sensors that formed gap g_i after the elimination of $g_1, g_1, \ldots, g_i$. If the algorithm uses in the elimination of g_{i+1} only the overlaps $o_j, o_{j+1}, \ldots$ and overlaps to the right of g_{i+1}, then any overlap that was used previously to eliminate gaps $g_1, g_1, \ldots, g_i$ would be obviously of higher cost to g_{i+1} than o_j and the other way around. Thus the overlaps being used in the elimination of g_{i+1} were of no use for an optimal elimination of $g_1, g_1, \ldots, g_i$ and the overlaps being used in an optimal elimination of $g_1, g_1, \ldots, g_i$ are of no use for the elimination of g_{i+1}. This means that the moves done by our algorithm are locally optimal with respect to g_{i+1} and also to $g_1, g_1, \ldots, g_i, g_{i+1}$.

If while processing gap g_{i+1} after the elimination of overlap o_j there are no more overlaps to the left of g_{i+1} then, necessarily, overlaps to the right are used by the algorithm in the inside-out manner to eliminate it and this is again optimal.

It remains to consider the situation during the execution of the algorithm when the only remaining overlap located to the left of g_{i+1}, say o_k, is located to the left of the sensors that formed gap g_i and there are also some overlaps to the right of g_{i+1}.

If all the shift factors between o_k and g_{i+1} are positive, then all overlaps to the right of o_j up to g_{i+1} were eliminated using the right shifts. If o_k is of lower cost to g_{i+1} than any overlap to the right of g_{i+1}, then applying the right shift to o_k is the optimal move and all the previous moves done to eliminate $g_1, g_1, \ldots, g_i$ remain locally optimal by Lemma 5. Clearly o_k was not useful for other gaps to the left of it.

If in the elimination of a gap, say g_m, located between o_k and g_{i+1}, gap g_m was eliminated or diminished using a left shift involving overlap o_{i_m}, than it could be better, with respect to $g_1, g_1, \ldots, g_i, g_{i+1}$, to use the right shift of sensors from o_{i_m} up to g_{i+1} and use o_k to eliminate or diminish g_m using a right shift, since o_k is closer to g_{i+1}. Thus this reverses the left shift done previously. By Lemma 5, all sensors between the rightmost sensor of o_k and g_{i+1} are in attached position and removal or diminishing of o_k must maintain these sensors in attached positions. Thus the right shift of the rightmost sensor of o_k actually is done by shifting to the right the whole sequence of sensors in attached position up to g_{i+1}, which reverses the right shift of sensors from o_{i_m} up to g_m. In our algorithm, when we calculate the cost of o_k with respect of g_{i+1}, this cost is defined to be equal to be the difference between the number of sensors that have non-negative shifts so far and the number of sensors with negative shifts. Thus we are actually calculating the cost of using o_j for elimination of gap g_m by a right shift and use o_k to eliminate or diminish g_{i+1} using a right shift. This move would be done if its cost is cheaper than using an overlap to the right of g_{i+1}. Thus, the overall cost of moves done in the elimination of $g_1, g_1, \ldots, g_i, g_{i+1}$ is improved. However, we cannot claim anymore that we preserved the local optimality with respect to $g_1, g_1, \ldots, g_i$.

To show the optimality of the algorithm in this case, consider the sequence of overlaps that are used in the elimination of gaps between the position of o_k and g_{i+1}. As mentioned above, these overlaps form a subsequence in the sequence of

overlaps, and the sum of overlaps of the subsequence equals the sum of these gap sizes. Furthermore all sensors between the rightmost sensor of o_k and g_{i+1} are in attached position. Any different lower-cost process would have to use a different subsequence of overlaps for the elimination of the gaps between the position of o_k and g_{i+1}. This could be done only by using a subsequence or subsequences of overlaps that is obtained from the current one either by (a) adding some overlaps on its right end and removing some in the sequence or (b) adding some overlaps on its left end and removing some in the subsequence, or (c) adding some overlaps on its left and right end and removing some in the subsequence. We now consider these three cases in detail.

Case (a): Adding an overlap on the right end of the subsequence and removing o_k at the left end was exactly the move considered by the algorithm before o_k was used and it was rejected due to a higher cost. Any further continuation in this process of extending the sequence further to the right will increase the overall cost even further as any of these is of higher cost than the cost of o_k. Removing an overlap in the middle of the subsequence will restore a part of a gap either in one of $g_1, g_1, \ldots, g_i$ or in g_{i+1}. If it restores a part of a gap in one of $g_1, g_1, \ldots, g_i$ then this new gap must be covered using an overlap o_k, but the use of o_k was not optimal for $g_1, g_1, \ldots, g_i$ by the inductive hypothesis. If it restores a part of a gap in g_{i+1} then adding an overlap on the right end of the subsequence was considered by the algorithm and rejected.

Case (b): If we remove an overlap at the right end of the subsequence it restores a part of gap g_{i+1}, as this overlap was necessarily used in a left shift elimination of g_{i+1}. This part of gap g_{i+1} must be removed by using an overlap to the left of o_k. Our algorithm used an overlap at the right end because it was of lower cost than using o_k. Removing an overlap in the middle of the subsequence will restore a part of a gap either in one of $g_1, g_1, \ldots, g_i$ or in g_{i+1}. If it restores a part of a gap in one of $g_1, g_1, \ldots, g_i$ then this new gap must be covered using an overlap o_k, but the use of o_k was not optimal for $g_1, g_1, \ldots, g_i$ by the inductive hypothesis. If it restores a part of a gap in g_{i+1} then it would need to be covered by an overlap on the right end of the subsequence which was considered by the algorithm and rejected.

Case (c): of adding overlaps at the left and right end can be shown not to be optimal in the same way as (a) and (b).

Thus the subsequence of overlaps used by the algorithm is locally optimal with respect to the gaps removed by these overlaps, including g_{i+1}. The optimality of removal of gaps not involved in this subsequence of overlaps remains valid, and thus we obtain a locally optimal solution with respect to $g_1, g_1, \ldots, g_i, g_{i+1}$.

Consider now the time complexity of the algorithm. We have n sensors to consider. Any consecutive pair of sensors can create at most one gap. In addition, there can be one gap preceding the first sensor and following the last sensor. Thus, in total there at most $n + 1$ overlaps in the given instance. Inside the interval $[0, L]$, any consecutive pair of sensors can create at most one overlap, and a sensor can also create an overlap if its range is located outside the interval. Thus, in total there $O(n)$ overlaps in the given instance. For each gap, the

algorithm performs a sequence of left and right shifts. For each shift, we need to calculate its cost, which is proportional to n. After each shift is done, the algorithm updates the shift values of the sensors, which is proportional to n. Thus the total cost of each shift is of time complexity $O(n)$. Each left shift either eliminates an overlap or it completes the coverage of a gap. Thus, in total, we can do at most $n+1$ of these shifts. In view of the NP-complete results presented in this paper, an interesting problem is to study approximation algorithms for the barrier coverage by sensors with different ranges. Any right shift either eliminates an overlap, or it completes the coverage of a gap, or it could be a reversal of a left shift. In a reversal of a left shift we either make a right shift equal to the size of a left shift (and decrease the size of some left shifts), or we eliminate an overlap, or complete an elimination of a left shift that was decreased earlier. Thus, in total there can be at most $O(n)$ such shifts. Therefore, the total complexity of the algorithm is $O(n^2)$.