Optimal Movement of Mobile Sensors for Barrier Coverage of a Planar Region*

Binay Bhattacharya*¶ Mike Burmester† Yuzhuang Hu*
Evangelos Kranakis‡¶ Qiaosheng Shi* Andreas Wiese§

Abstract

Intrusion detection, area coverage and border surveillance are important applications of wireless sensor networks today. They can be (and are being) used to monitor large unprotected areas so as to detect intruders as they cross a border or as they penetrate a protected area. We consider the problem of how to optimally move mobile sensors to the fence (perimeter) of a region delimited by a simple polygon in order to detect intruders from either entering its interior or exiting from it. We discuss several related issues and problems, propose two models, provide algorithms and analyze their optimal mobility behavior.

1 Introduction

Monitoring and surveillance are two of the main applications of wireless sensor networks today. Typically, one is interested in monitoring a given geographic region either for measuring and surveying purposes or for reporting various types of activities and events. Another important application concerns critical security and safety monitoring systems. One is interested in detecting intruders (or movements thereof) around *critical* infrastructure facilities and geographic delimiters (chemical plants, forests, etc). Since the information security level of the monitoring system might change rapidly because of hostile attacks targeted at it, research efforts are currently underway to extend the scalability of wireless sensor networks so that they can be used to monitor international borders as well. For example, [11] reports the possibility of using wireless sensor networks for replacing traditional barriers (more than a kilometer long) at both the building and estate level. Also, "Project 28" concerns the construction of a virtual fence as a way to complement a physical fence that will include 370 miles of pedestrian fencing and 300 miles of vehicle barrier (see [8] which reports delays in its deployment along the U.S.-Mexico border).

To begin, we say that a point is *covered* by a sensor if it is within its range. In this paper we will use the concept of *barrier coverage* as used in [11], which differs from the more

*School of Computing Science, Simon Fraser University, Vancouver, BC, Canada.
†Department of Computer Science, Florida State University, Tallahassee, Florida, USA.
‡School of Computer Science, Carleton University, Ottawa, Ontario, Canada.
§Institut für Mathematik, Technische Universität Berlin, Berlin, Germany.
¶Research supported in part by NSERC and MITACS.
*A first version of this manuscript was presented at the 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA 2008), St John's, Newfoundland.

traditional concept of *full coverage*. In the latter case, one is interested in covering the entire region by the deployment of sensors, while in the former, all crossing paths through the region are covered by sensors. Thus, one is not interested in covering the entire deployment region but rather in detecting potential intruders by guaranteeing that there is no path through this region that can be traversed undetected by an intruder that crosses the border. Clearly, barrier coverage is an appropriate model of movement detection that is more efficient than full coverage, since it requires fewer sensors for detecting intruders. This is the case, for example, when the width of the deployment region is three times the range of the sensors.

In article [3], the authors consider the problem of how individual sensors can determine barrier coverage *locally*. In particular, they prove that it is possible for individual sensors to locally determine the existence of barrier coverage, even when the region of deployment is arbitrarily curved. Although local barrier coverage does not always guarantee global barrier coverage, they show that for thin belt regions, local barrier coverage almost always provides global barrier coverage. They also consider the concept of *L-local barrier coverage* whereby if the bounding box that contains the entire trajectory of a crossing path has length at most L, then this crossing path is guaranteed to be detected by at least one sensor.

1.1 Motivation, model and problem statement

Motivated by the works of [3] and [11], in this paper we go further by asking a question not examined by any of these papers. More precisely, given that the mobile sensors have detected the existence of a crossing path (e.g., using any of the above algorithms) how do they reposition themselves *most efficiently* within a specified region so as to repair the existing *security hole* and thereby prevent intruders.

Furthermore, we stipulate the existence of a geometric planar region (the critical region to be protected) delimited by a simple polygon, the *barrier*, and mobile sensors (or robots) that are lying in the interior of this polygon. The sensors can move autonomously in the plane. Each sensor has knowledge of the region to be barrier-covered, of its geographic location and can move from its starting position A to a new position A' on the perimeter of this polygon (see Figure 1). For each sensor A, we consider the distance $d(A, A')$ between the initial and

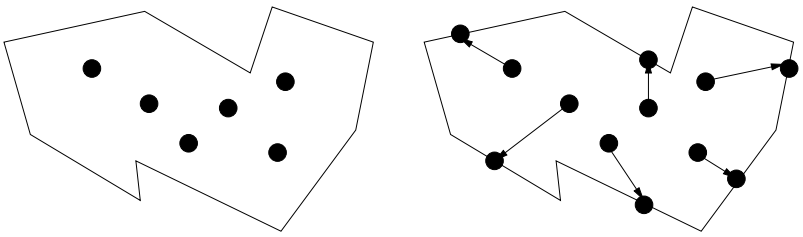

Figure 1: Sensors move from their initial position to positions on the perimeter of a simple polygon.

final positions of the sensors, respectively, and investigate how to move the sensors within this region so as to optimize either the *minimum sum* or the *minimum of the maximum* of the distances covered by the respective sensors. We call this the *barrier coverage problem*. In the sequel, we investigate the complexity of this problem for various types of regions and types of movement of the mobile sensors.

1.2 Related work

An interesting research article by [1] surveys the different kinds of holes that can form in geographically correlated problem areas of wireless sensor networks. The authors discuss relative strengths and short-comings of existing solutions for combating different kinds of holes, such as coverage holes, routing holes, jamming holes, sink/black holes, wormholes, etc. [2] looks at critical density estimates for coverage and connectivity of thin strips (or annuli) of sensors. In addition, [5] and [6] design a distributed self deployment algorithm for coverage calculations in mobile sensor networks, and consider various performance metrics, such as coverage, uniformity, time and distance traveled until the algorithm converges. Related research on art gallery theorems (see [14]) is concerned with finding the minimal number of positions for guards or cameras so that every point in a gallery is observed by at least one guard or camera.

In addition to the research on barrier coverage already mentioned, there is extensive literature on detection and tracking in sensor networks. In [12], the authors consider the problem of event tracking and sensor resource management in sensor networks and transform the detection problem into finding and tracking the cell that contains the point in an arrangement of lines. [9] addresses the problem of tracking multiple targets using a network of communicating robots and stationary sensors by introducing a region-based approach for controlling robot deployment. [16] considers the problem of accurate mobile robot localization and mapping with uncertainty using visual landmarks. Finally, related to the problem of detecting a path through a region that can be traversed undetected by an intruder is the paper [15] which gives necessary and sufficient conditions for the existence of vertex disjoint simple curves homotopic to certain closed curves in a graph embedded on a compact surface.

1.3 Outline and results of the paper

Section 2 gives the formal model on a disk and defines the *min-max* (minimizing the maximum) and *min-sum* (minimizing the sum) problems for a set of sensors within a disk or a simple polygon. Section 3 looks at the simpler one-dimensional case, and derives simple optimal algorithms for the case in which the sensors either all lie on a line or on the perimeter of the disk. Section 4 and Section 5 are the core of the paper and provide algorithms for solving the min-max and min-sum problems, respectively. In Section 4, an $O(n^{3.5} \log n)$-time algorithm for the min-max problem on a disk and an $O(mn^{3.5} \log n)$-time algorithm for the min-max problem on a simple polygon, are proposed (m is the number of edges of the simple polygon). Our approximation algorithms for min-sum problems on a disk or a simple polygon are presented in Section 5, where we give a PTAS with approximation $1 + \epsilon$ having running time $O(\frac{1}{\epsilon}mn^5)$ for a given constant ϵ. We also present experimental results on the min-sum problem, and Section 6 gives the conclusion.

2 Preliminaries and formal model

First we describe the formal model on a disk and provide the basic definitions and preliminary concepts.

2.1 Optimization on the unit disk

The simpler scenario we envision concerns n mobile sensors located in the interior of a unit-radius circular region (see Figure 2). We assume that the sensors are location aware (i.e.,

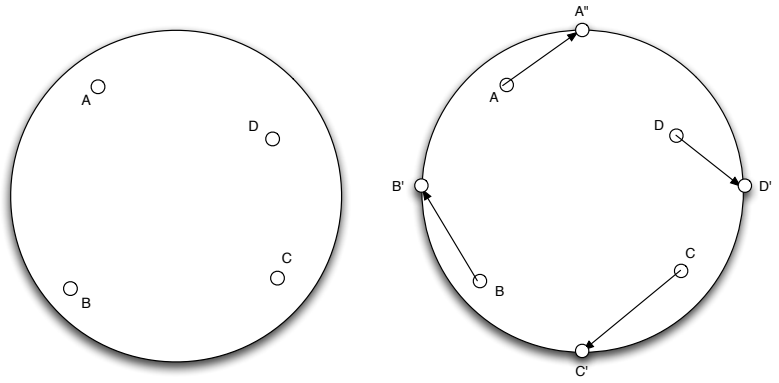

Figure 2: Four mobile sensors A, B, C, D located in the interior of a disk move to new positions A', B', C', D' on the perimeter of the disk so that $A'B'C'D'$ forms a regular 4-gon.

they know their geometric coordinates) and also know the location of the center of the disk. We would like to move all the sensors from their initial positions to the perimeter of the disk so as to:

1. Form a regular n-gon, and
2. Minimize the total/maximum distance covered (see Figure 2).

The motivation for placing the sensors on the perimeter is because it provides the most efficient way to protect the disk from intruders. Observe that when all n sensors lie on the perimeter and form a regular n-gon, then each sensor needs only cover a circular arc of size $2\pi/n$ so as to be able to monitor the entire perimeter. Using elementary trigonometry, it follows easily that the transmission range of each sensor must be equal to $r = \sin(\pi/n)$.

More formally, for n given sensors in positions $A_1, A_2, \ldots, A_n$, respectively, which move to new positions $A'_1, A'_2, \ldots, A'_n$ on the perimeter forming a regular n-gon, the total distance covered is:

$$\sum_{i=1}^{n} d(A_i, A'_i). \tag{1}$$

It is clear that the sum is minimized when each sensor A_i moves to its destination A'_i in a straight line.

The reason for having the sensors form a regular n-gon is because this is evidently the optimal final arrangement that will enable the sensors to detect intruders (i.e., by being equidistant on the perimeter). Since the final positions $A'_1, A'_2, \ldots, A'_n$ of the sensors form a regular n-gon it is clear that the final configuration can be parametrized by using a single angle $0 \leq \theta \leq 2\pi$. However, a difficulty arises in view of the fact that we must also specify a permutation $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ of the sensors such that the i-th sensor moves from position $A_{\sigma(i)}$ to the new position A'_i.

Let the n sensors have coordinates (a_i, b_i), for $i = 1, 2, \ldots, n$. Let us parametrize the regular polygon with respect to the angle of rotation, say, θ. The n vertices of the regular n-gon that lie on the perimeter of the disk can be described by

$$(a_i(\theta), b_i, (\theta)) = \left(\cos\left(\theta + \frac{(i-1)2\pi}{n}\right), \sin\left(\theta + \frac{(i-1)2\pi}{n}\right)\right), \text{ for } i = 1, 2, \ldots, n, \tag{2}$$

respectively, where $(a_i(\theta), b_i(\theta))$ are the vertices of the regular n-gon when the angle of rotation is θ.

2.1.1 Minimizing the sum

We are interested in minimizing the sum

$$S_n(\theta) := \sum_{i=1}^{n} \sqrt{(a_i - a_i(\theta))^2 + (b_i - b_i(\theta))^2}, \tag{3}$$

as a function of the angle θ. The optimization problem is:

$$\min_{\theta} S_n(\theta). \tag{4}$$

This of course assumes that the i-th sensor is assigned to position $(\cos(\theta+(i-1)2\pi/n), \sin(\theta+(i-1)2\pi/n))$ on the perimeter. In general, we have to determine the minimum over all possible permutations $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ of the sensors. If for a given permutation $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ we define the following sum

$$S_n(\sigma, \theta) := \sum_{i=1}^{n} \sqrt{\big(a_{\sigma(i)} - a_i(\theta)\big)^2 + \big(b_{\sigma(i)} - b_i(\theta)\big)^2} \tag{5}$$

then the general optimization problem is:

$$\min_{\sigma,\theta} S_n(\sigma, \theta). \tag{6}$$

2.1.2 Minimizing the maximum

The previous problem concerned minimizing the sum of the distance traveled by the robots. In view of the fact that the robots are moving simultaneously, it makes sense to consider the problem of minimizing the maximum of the distances traveled by each robot. More formally, given n sensors in positions $A_1, A_2, \ldots, A_n$, respectively, which move to new positions $A'_1, A'_2, \ldots, A'_n$ forming a regular n-gon on the barrier, the maximum distance covered is:

$$\max_{1 \le i \le n} d(A_i, A'_i). \tag{7}$$

It is clear that the maximum is minimized when each sensor A_i moves in a straight line to its destination A'_i. The min-max problem involves minimizing the maximum

$$M_n(\theta) := \max_{1 \le i \le n} \sqrt{(a_i - a_i(\theta))^2 + (b_i - b_i(\theta))^2}\ , \tag{8}$$

as a function of the angle θ. The optimization problem is therefore to compute

$$\min_{\theta} M_n(\theta). \tag{9}$$

This of course assumes that the i-th sensor is assigned to position $(\cos(\theta+(i-1)2\pi/n), \sin(\theta+(i-1)2\pi/n))$ on the perimeter. In general, we have to determine the minimum over all possible

permutations $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ of the sensors. If for a given permutation $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ we define the following maximum

$$M_n(\sigma, \theta) := \max_{1 \le i \le n} \sqrt{\big(a_{\sigma(i)} - a_i(\theta)\big)^2 + \big(b_{\sigma(i)} - b_i(\theta)\big)^2}\,, \tag{10}$$

then the general optimization problem is:

$$\min_{\sigma, \theta} M_n(\sigma, \theta). \tag{11}$$

2.2 Optimization on a simple polygon

We define the problem of minimizing the sum and minimizing the maximum on a simple polygon in a similar way, as follows.*

Let P be a simple polygon. (From now on, a polygon is always assumed to be simple.) We denote the boundary of P by ∂P. We assume that ∂P is oriented in the clockwise (also called positive) direction. For any two points $A, C \in \partial P$, $\hat{\pi}_P(A, C)$ denotes the set of all points $B \in \partial P$ such that when starting after A in positive direction along ∂P, B is reached before C. Let $P_1, P_2, \ldots, P_m$ denote the vertices of P ordered in the positive direction, and let the edges of P be $e_1, e_2, \ldots, e_m$, where edge e_i has endpoints P_i and P_{i+1}, $1 \le i \le m$ (the indices are computed modulo m: so $P_0 = P_m$). Denote by $\ell(e_i)$ the length of edge e_i, $1 \le i \le m$, and by $\hat{d}_P(A, B)$ the length of $\hat{\pi}_P(A, B)$ for any two points A and B on ∂P (called *polygonal distance* between A and B). Let $L(P) = \sum_{i=1}^{m} \ell(e_i)$.

We are given n mobile sensors, which are located in the interior or on the boundary of P. Each sensor has the knowledge of its geometric coordinates and the simple polygon (i.e., the coordinates of all vertices $P_i, 1 \le i \le m$ and the clockwise ordering of these vertices). The objective is to move all the sensors from their initial positions to ∂P such that:

1. The polygonal distance between any two consecutive sensors on the polygon is $L(P)/n$, and
2. The total/maximum distance covered is minimized.

Note that only straight-line moving distance is considered here. That is, we view sensors as mobile robots with unrestricted movement (therefore, it is allowed to move sensors outside the perimeter of the given region).

More formally, we are given n sensors located at positions $A_1, A_2, \ldots, A_n$, respectively. Let A'_i be the destination position of A_i on ∂P, $i = 1, 2, \ldots, n$. Without loss of any generality, assume that for any i, $1 \le i < n$, A'_{i+1} is the first position after A'_i in the positive direction along ∂P. The new positions $A'_1, A'_2, \ldots, A'_n$ should satisfy $\hat{d}_P(A'_i, A'_{i+1}) = L(P)/n, 1 \le i \le n$ (taken modulo n), and we consider the following two objectives:

1. **Minimizing the sum:** $\min \sum_{i=1}^{n} d(A_i, A'_i)$, and
2. **Minimizing the maximum:** $\min \max_{i=1}^{n} d(A_i, A'_i)$,

where $d(\cdot, \cdot)$ is the straight-line distance metric.

*Although the approach proposed later (parametric search) will also work for arbitrary simple curves, we refrain from such a generalization so as to avoid unnecessary complications.

3 Mobile sensors in one dimension

In this section, we look at the one-dimensional problem and provide efficient algorithmic solutions. In particular, since optimization for the min-max is similar (and simpler than the two dimensional analogue) we provide algorithms only for the min-sum.

3.1 Sensors on the unit line segment

In this model we suppose that the sensors can move on a line segment. Further, instead of protecting a circular range the sensor can now protect an interval of a given size centered at the sensor. Consider the min-sum optimization problem for the case of n sensors on a line. Without loss of generality assume the segment has length 1 and let the n sensors be at the initial locations $A_1 \leq A_2 \leq \cdots \leq A_n$, respectively. Then the destination locations must be at the positions $\frac{2i-1}{2n}$, for $i = 1, 2, \ldots, n$.

Theorem 1 *The optimal arrangement is obtained by moving point A_i to position $\frac{2i-1}{2n}$, for $i = 1, 2, \ldots, n$, respectively.*

Proof. It is clear that the destinations must be equidistant and cover the endpoints 0 and 1. The optimal configuration is therefore: $1/2n$,$3/2n$, $\ldots$, $(2n-1)/2n$. For any point X let $d(X)$ be its destination. Recall that our goal is to determine the destinations of each point X so that the sum

$$\sum_X |X - d(X)| \tag{12}$$

is minimized.

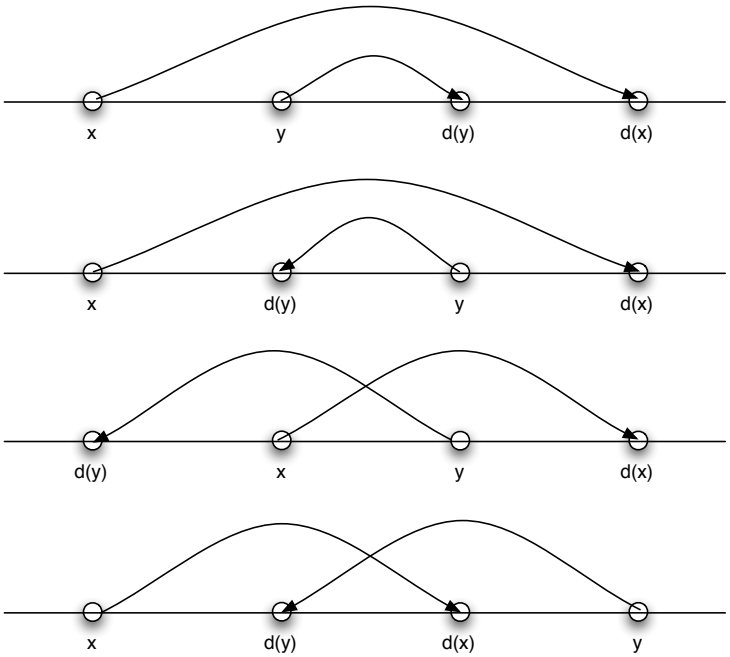

Figure 3: Four possible orderings of $x, y, d(x), d(y)$ where $x < y$ and $d(y) < d(x)$.

Suppose that there is an optimal assignment between $\{A_1, \ldots, A_n\}$ and $\{\frac{1}{2n}, \frac{3}{2n}, \ldots, \frac{2n-1}{2n}\}$, which is not the assignment resulting by moving point A_i to position $\frac{2i-1}{2n}$, for $i = 1, 2, \ldots, n$, respectively. Then, in this optimal assignment, there must exist at least one *inverse pair* $x = A_i$ and $y = A_j$ $(i < j)$ such that $d(x) > d(y)$. By considering all possible six orderings of $x, y, d(x)$, and $d(y)$ (Four of them are depicted in Figure 3. The other two are $d(y) \leq x \leq d(x) \leq y$ and $d(y) < d(x) \leq x < y$), we can see that the sum of moving distance is smaller or unchanged if we move x to $d(y)$ and y to $d(x)$ instead. After switching their destinations for each inverse pair, we obtain the assignment resulting by moving point A_i to position $\frac{2i-1}{2n}$, for $i = 1, 2, \ldots, n$, respectively. This completes the proof of the theorem. ■

3.2 Sensors on the perimeter of the unit disk

In this model, we suppose that the sensors can move on the perimeter of a disk. Further, instead of protecting a circular range, the sensor can now protect an arc on the perimeter of a given size centered at the sensor. The same idea as for a line segment should work for the case of the unit disk when the sensors lie on its perimeter. The main difficulty here is that we no longer have a unique destination. Instead, we can parametrize all possible destinations of the n points by $\phi + \frac{2j\pi}{n}$, for $j = 0, 1, \ldots, n-1$, using a fixed angle $0 \leq \phi < \frac{2\pi}{n}$.

Suppose we are given n points $A_1, A_2, \ldots, A_n$ in clockwise order along the perimeter of the disk. First of all, we prove the following lemma.

Lemma 2 *The minimal cost assignment of the destinations for the given points must be among the n assignments*

$$(A_1, \ldots, A_j, \ldots, A_n) \rightarrow \left(A_i + (i-1)\frac{2\pi}{n}, \ldots, A_i + (i-j)\frac{2\pi}{n}, \ldots, A_i + (i-n)\frac{2\pi}{n}\right),$$

for $i = 1, 2, \ldots, n$. Note that each such assignment has a fixed point A_i.

Proof. We denote by $d(A_i)$ the destination of A_i on the unit disk, $i = 1, 2, \ldots, n$, and by $\hat{\pi}(x, d(x))$ the arc traveled (clockwise or counter-clockwise) by a sensor from its initial position x to destination $d(x)$. Clearly, the length of $\hat{\pi}(x, d(x))$ is always no more than π.

First we prove that there exists an optimal assignment in which $d(A_1), d(A_2), \ldots, d(A_n)$ are also in clockwise order (called *Condition A*). Let Δ be a minimal cost (optimal) assignment. For any pair of initial positions $x = A_i$ and $y = A_j$ $(1 \leq i, j \leq n)$ in Δ, we consider the following two cases.

- Case 1: x and y move to their destinations in different directions. Without loss of any generality, we assume that x moves clockwise to its destination $d(x)$ and y moves counter-clockwise to its destination $d(y)$. In this case, $\hat{\pi}(x, d(x))$ and $\hat{\pi}(y, d(y))$ cannot overlap, since, otherwise, we can obtain a new assignment that has a smaller cost (a contradiction with that Δ is an optimal assignment).

- Case 2: x and y move to their destinations in a same direction. Then we can update the assignment Δ to satisfy Condition A among x and y without increasing its cost.

Now, in the assignment Δ, we have the following cases for each pair $A_i, A_{i+1}, i = 1, \ldots, n$ (note that $A_{n+1} = A_1$): (Assume that A_i clockwise moves to $d(A_i)$.)

- $\hat{\pi}(A_i, d(A_i))$ and $\hat{\pi}(A_{i+1}, d(A_{i+1}))$ are disjoint. See Figure 4(a) and (b) for reference;

- $\hat{\pi}(A_i, d(A_i))$ and $\hat{\pi}(A_{i+1}, d(A_{i+1}))$ are overlapping. Then $\hat{\pi}(A_i, d(A_i))$ contains A_{i+1} and $\hat{\pi}(A_{i+1}, d(A_{i+1}))$ contains $d(A_i)$. See Figure 4(c) for reference.

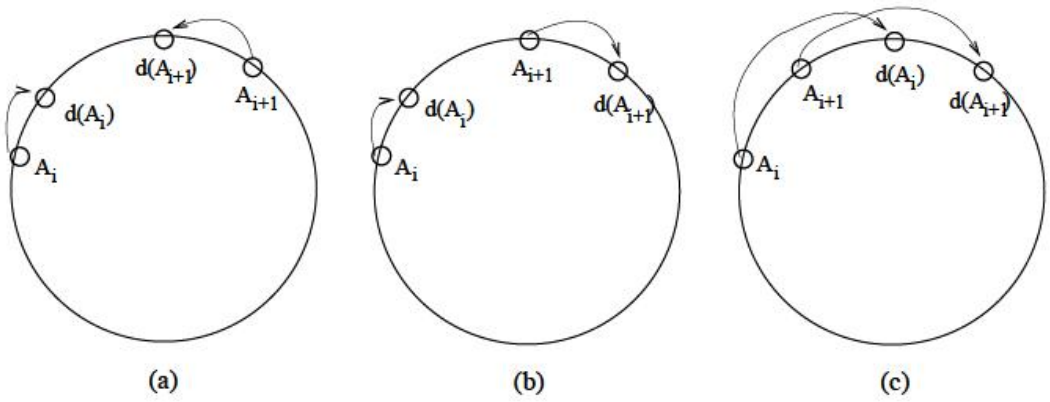

Figure 4: Possible orderings of $A_i, A_{i+1}, d(A_i), d(A_{i+1})$ in the updated assignment Δ.

It is not difficult to see that, with such property between each pair of $A_i, A_{i+1}, i = 1, \ldots, n$, the updated assignment Δ satisfies Condition A.

Second, we prove that there exists an optimal assignment among the n assignments

$$(A_1, \ldots, A_j, \ldots, A_n) \rightarrow \left(A_i + (i-1)\frac{2\pi}{n}, \ldots, A_i + (i-j)\frac{2\pi}{n}, \ldots, A_i + (i-n)\frac{2\pi}{n} \right),$$

for $i = 1, 2, \ldots, n$. Consider the above minimal cost assignment Δ. Suppose that, in the corresponding n-gon, none of the corner points is an initial point. Now rotate this n-gon and keep the assignment unchanged. Let n_1 (resp. n_2) be the number of initial points that move clockwise (resp. counter-clockwise) to their destinations. Without loss of any generality, we assume that $n_1 \leq n_2$. Then the clockwise rotation of the n-gon decreases the cost or remains the same until it touches an initial point. This completes the proof of the lemma. ■

Based on Lemma 2, the main steps of the algorithm are the following.

1. For each point $A_i \in \{A_1, A_2, \ldots, A_n\}$, map all points A_j to destinations $A_i + (i-j)\frac{2\pi}{n}$, for $j = 1, 2, \ldots, n$ (this implies that A_i is mapped to itself).
2. Select the point $A_i \in \{A_1, A_2, \ldots, A_n\}$ that optimizes the sum of moving arc distance

Thus, we have proved the following theorem.

Theorem 3 *There is an $O(n^2)$ algorithm that computes an optimal cost arrangement of the sensors on the perimeter of the unit disk where the sensors are allowed to move on the perimeter of the disk.*

4 Min-max problem

In this section, we study the min-max problem on a unit disk and a simple polygon, and provide efficient algorithmic solutions.

4.1 On the disk

Let λ_C^* be the optimal value of the min-max problem on a disk $\mathcal{C}$, i.e.,

$$\lambda_C^* = \min_{\sigma,\theta} M_n(\sigma, \theta).$$

It is easy to see that λ_C^* is no more than the diameter of the disk $\mathcal{C}$, i.e., $\lambda_C^* \leq 2$. In this section we propose a parametric-searching approach [13] to compute λ_C^*.

A non-negative value λ is *feasible* for the min-max problem if all the sensors can move from their initial positions to the perimeter of the disk such that the new positions form a regular n-gon and the maximum covered distance is no more than λ, otherwise λ is *infeasible*. Clearly, the min-max problem is to compute the minimum feasible value, which is equal to λ_C^*.

The remaining part of this section is organized as follows. We first show that a feasibility test of a given value λ ($0 \leq \lambda \leq 2$) can be performed in time $O(n^{3.5})$. Then we present a parametric-searching approach for the min-max problem, which runs in $O(n^{3.5} \log n)$ time.

4.1.1 An algorithm to check the feasibility test of λ

For each $i, 1 \leq i \leq n$, we construct a circle of radius λ centered at position A_i, denoted by $\mathcal{C}_i$. In Figure 5(b), for each $i, 1 \leq i \leq n$, we denote by x_i^l (resp. x_i^u) the shortest distance (resp. largest distance) between A_i and a point in $\mathcal{C}$. If $\mathcal{C}_i$ is contained in $\mathcal{C}$ for some i, i.e., $\lambda < x_i^l$, then λ is infeasible since sensor A_i cannot move to the perimeter of $\mathcal{C}$ within distance λ. We therefore assume that for each $i, 1 \leq i \leq n$, either $\mathcal{C}_i$ contains $\mathcal{C}$ (i.e., $\lambda > x_i^u$) or $\mathcal{C}_i$ intersects $\mathcal{C}$ (i.e., $x_i^l \leq \lambda \leq x_i^u$). Referring to Figure 5(a), for each $i, 1 \leq i \leq n$, we denote by Q_i the arc of $\mathcal{C}$ that lies in $\mathcal{C}_i$. Let $q_{i(1)}, q_{i(2)}$ be the angles of the two endpoints of arc Q_i in clockwise order, $i = 1, 2, \ldots, n$. We let $q_{i(1)} = 0$ and $q_{i(2)} = 2\pi$ if $\mathcal{C}_i$ contains $\mathcal{C}$.

The following property is important to our algorithm for the feasibility test of λ.

Lemma 4 *If $\lambda > 0$ is feasible for the min-max problem on the disk $\mathcal{C}$ then all the sensors can move from their initial positions to the perimeter of $\mathcal{C}$ such that the new positions form a regular n-gon, the maximum covered distance is no more than λ, and one corner point of the regular n-gon is an endpoint of arc Q_i for some $i, 1 \leq i \leq n$.*

Proof. Since λ is feasible, the destination points of $A_1, \ldots, A_n$, denoted by $A_1', \ldots, A_n'$ form a regular n-gon, and each point A_i' lies on the corresponding arc Q_i of the disk, which is cut by the λ-radius circle centered at A_i. Move the configuration of the destination points clockwise (or counter-clockwise) until one of the destination points reaches an endpoint of one of the arcs Q_i. In this new arrangement, the maximum moving distance is no more than λ, and these new destination points still form a regular n-gon. ■

The following algorithm is used to check the feasibility of λ.

Algorithm Check-Feasibility.

Step 1 Sort the angles of the endpoints of the arcs Q_i ($1 \leq i \leq n$) in clockwise order. Let $q_1', q_2', \ldots, q_{2n}'$ be the angles in increasing order. These angles partition the interval $[0, 2\pi)$ into at most $2n$ pairwise disjoint intervals, denoted by I_j ($1 \leq j \leq 2n$), i.e., $I_j = (q_j', q_{j+1}')$.

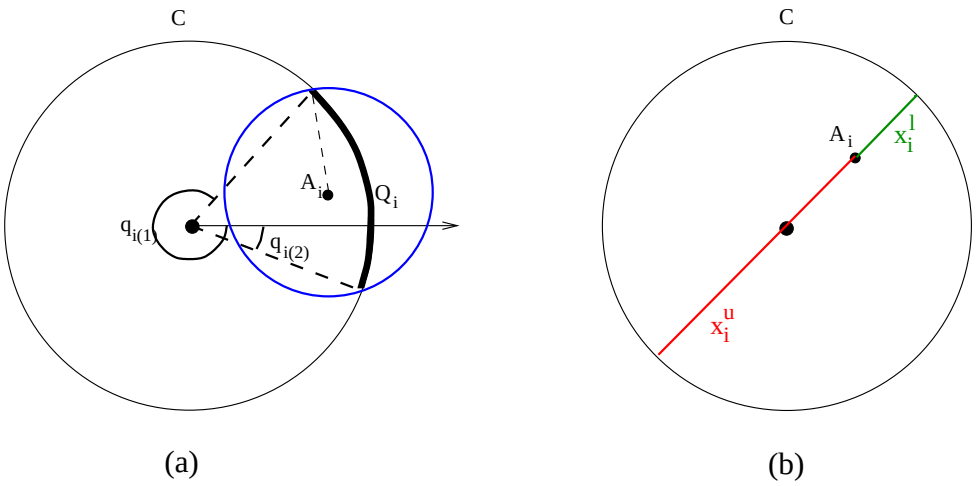

Figure 5: (a) The arc Q_i, and the angles $q_{i(1)}, q_{i(2)}$ of its endpoints; (b) x_i^l and x_i^u.

Step 2 For each interval I_j (resp. each angle q'_j), $1 \leq j \leq 2n$, determine the set of sensors, denoted by S_j (resp. S'_j), that lie within distance λ of the corresponding arc on $\mathcal{C}$ (resp. of point q'_j on $\mathcal{C}$).

Step 3 For each $1 \leq j \leq 2n$, do the following:

a Compute the angles of vertices of the regular n-gon, denoted by $B_1^j, B_2^j, \ldots, B_n^j$, where the angle of B_1^j is q'_j. The angles of vertices of this n-gon are: $q'_j, (q'_j + \frac{2\pi}{n}) \bmod 2\pi, \ldots, (q'_j + (n-1)\frac{2\pi}{n}) \bmod 2\pi$.

b For each B_k^j $(1 \leq k \leq n)$, determine the interval of $\{I_t, t = 1, \ldots, 2n\}$ that contains the angle of B_k^j or an angle $q'_s, 1 \leq s \leq 2n$ that is equal to the angle of B_k^j.

c Construct a bipartite graph H^j between the sensors $A_1, A_2, \ldots, A_n$ and the vertices $B_1^j, B_2^j, \ldots, B_n^j$: sensor A_i is linked to angle B_k^j if and only if A_i is in the set of sensors covering B_k^j within distance λ. Note that the set of sensors covering B_k^j within distance λ is evident from the combined results of Steps 2 and 3(b).

d Check if there exists a perfect matching in H^j. If so, terminate the process and return "Feasible".

Step 4 Return "Infeasible".

The sorting in the first step can be completed in $O(n \log n)$ time, and the computation of sets of sensors S_j and S'_j $(1 \leq j \leq 2n)$ in the second step can be completed in $O(n^2)$ time,

since $\sum_j |S_j| + |S'_j|$ could be $O(n^2)$. In the third step, the process might try all $O(n)$ regular n-gons. For each regular n-gon, it takes $O(n^2)$ to construct the corresponding bipartite graph and $O(n^{2.5})$ time to check if there exists a perfect matching (see [7]). Therefore, we have the following lemma.

Lemma 5 *One can determine whether a given positive value λ is feasible in the min-max problem in $O(n^{3.5})$ time.*

4.1.2 A parametric-searching approach

Our approach for the solution to the min-max problem is to run Algorithm *Check-Feasibility* parametrically, which has a single parameter λ, without specifying the value of λ^*_C a priori. For a fixed value of the parameter, the algorithm is executed in $O(n^{3.5})$ steps. Imagine that we start the algorithm without specifying a value of the parameter λ. The parameter is restricted to some interval, denoted by Λ, which is known to contain the optimal value λ^*_C. Initially, we start with the interval $\Lambda = [0, 2]$. As we proceed, at each step of the algorithm we update and shrink the interval Λ, ensuring that it includes the optimal value λ^*_C. The final interval contains λ^*_C and any value in it is feasible. Therefore, the minimum value of the final interval is the optimal value λ^*_C.

The whole approach for the min-max problem is described as follows. For each $i, 1 \le i \le n$, let $\mathcal{C}_i(\lambda)$ be the circle of radius λ centered at position A_i. Note that here λ is an unknown parameter. If $x^l_i \le \lambda \le x^u_i$, then we denote by $Q_i(\lambda)$ the arc of $\mathcal{C}$ that lies in $\mathcal{C}_i(\lambda)$ and let $q_{i(1)}(\lambda), q_{i(2)}(\lambda)$ be the angle functions of the two endpoints of arc $Q_i(\lambda)$ in clockwise order, $i = 1, 2, \ldots, n$. Note that, if $\lambda < x^l_i$ then $\mathcal{C}_i(\lambda)$ is completely contained in $\mathcal{C}$, and if $\lambda > x^u_i$, then $\mathcal{C}$ is completely contained in $\mathcal{C}_i(\lambda)$.

Algorithm Optimization.

Preprocessing Step Shrink the interval $\Lambda = [0, 2]$ to be $[\max\{x^l_i, 1 \le i \le n\}, 2]$. Among the set $\{x^u_i, 1 \le i \le n\}$, find the smallest feasible value and shrink Λ accordingly. If x^u_j is infeasible for some j, then $\mathcal{C}_j(\lambda^*_C)$ contains $\mathcal{C}$, since $\lambda^*_C > x^u_j$. Therefore, it is safe to remove sensor A_j without affecting the remaining computation. Also, if x^u_j is feasible, then $\mathcal{C}_j(\lambda^*_C)$ intersects with $\mathcal{C}$. These feasible/infeasible values can be identified by solving $O(\log n)$ feasibility tests.

In the following, we assume that none of the sensors is removed in the preprocessing step (to simplify the notations).

Step 1 with unknown λ^*_C Sort the angle functions $q_{i(1)}(\lambda), q_{i(2)}(\lambda)$ of the arcs $Q_i(\lambda), 1 \le i \le n$, in clockwise order, for an unknown value $\lambda = \lambda^*_C \in \Lambda$. Note that a comparison between any two angle functions when $\lambda = \lambda^*_C$ can be computed by performing at most two feasibility tests (see Figure 6 for reference). The sorting step with unknown λ^*_C can be performed by solving $O(\log^2 n)$ feasibility tests [13] if a parallel sorting network that runs $O(\log n)$ steps with n processors is used. However, it can be reduced to $O(\log n)$ feasibility tests if Cole's result is applied [4].

Let $q'_1(\lambda), q'_2(\lambda), \ldots, q'_{2n}(\lambda)$ be the sorted angle functions in clockwise order. Note that, in this ordering, two adjacent functions might be equal for some value λ in the most recent interval Λ. Now we have an interval Λ which contains λ^*_C, and for any value

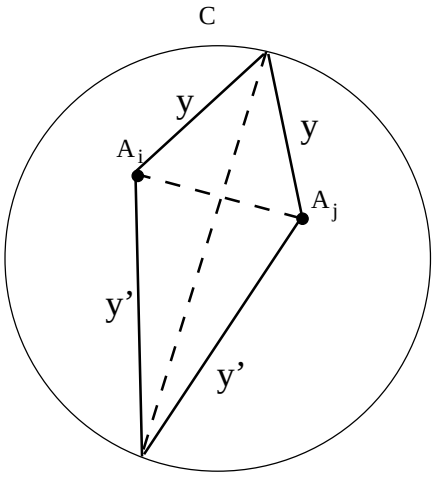

Figure 6: The two critical values y and y' in the comparison between angle functions of sensors A_i and A_j.

$\lambda \in \Lambda$, the ordering (just computed) of $\{q_{i(1)}(\lambda), q_{i(2)}(\lambda), 1 \leq i \leq n\}$ contains the ordering of $\{q_{i(1)}(\lambda^*_C), q_{i(2)}(\lambda^*_C), 1 \leq i \leq n\}$. Here we call an ordering O_1 containing another ordering O_2 if O_2 does not conflict with O_1. For example, the ordering $y_1 \leq y_2$ contains the ordering $y_1 < y_2$ and $y_1 = y_2$.

The angle functions, $q'_1(\lambda), \ldots, q'_{2n}(\lambda)$, partition the interval $[0, 2\pi)$ into at most $2n$ pairwise disjoint intervals, denoted by $I_j(\lambda)$, i.e., $I_j(\lambda) = (q'_j(\lambda), q'_{j+1}(\lambda))$, $0 \leq j \leq 2n$.

Step 2 with unknown $\lambda^*_C \in \Lambda$ The endpoints of each interval $I_j(\lambda)$ $(1 \leq j \leq 2n)$ are functions of $\lambda \in \Lambda$. But, we can compute the set of sensors, denoted by $S_j(\lambda)$, that lie within distance λ of the corresponding arc on $\mathcal{C}$ for each interval $I_j(\lambda)$. Similarly, we determine the set of sensors, denoted by $S'_j(\lambda)$, that covers the point on $\mathcal{C}$ with angle $q'_j(\lambda)$, $1 \leq j \leq 2n$. Note that sets $S_j(\lambda)$ and $S'_j(\lambda)$ are not functions of λ. They do not change for any value $\lambda \in \Lambda$.

In this step, no feasibility tests are needed, since the order of the endpoints of the arcs $Q_i(\lambda), 1 \leq i \leq n$, provides enough information to compute sets $S_j(\lambda)$ and $S'_j(\lambda)$, $1 \leq j \leq 2n$.

Step 3 with unknown $\lambda^*_C \in \Lambda$ For each $1 \leq j \leq 2n$, we compute the angles of vertices of a regular n-gon, denoted by $B^j_1(\lambda), B^j_2(\lambda), \ldots, B^j_n(\lambda)$, where the angle of $B^j_1(\lambda)$ is $q'_j(\lambda)$. The angles of vertices of this n-gon are: $q'_j(\lambda), (q'_j(\lambda) + \frac{2\pi}{n}) \bmod 2\pi, \ldots, (q'_j(\lambda) + (n-1)\frac{2\pi}{n}) \bmod 2\pi$.

a We now sort all the angle functions of $\{B^j_i(\lambda), 1 \leq i \leq n \text{ and } 1 \leq j \leq 2n\}$ for an unknown value $\lambda = \lambda^*_C \in \Lambda$. The set size is $O(n^2)$. Similar to the sorting step in Step 1 of Algorithm *Optimization*, the sorting of $O(n^2)$ angle functions for an unknown value λ^*_C can be completed by solving $O(\log n)$ feasibility tests. Let

$q''_1(\lambda), q''_2(\lambda), \ldots, q''_{2n^2}(\lambda)$ be the sorted angle functions of the vertices of $2n$ different regular n-gons in clockwise order. Now, we have a new interval Λ which contains λ^*_C and for any value $\lambda \in \Lambda$, the ordering (just computed) of angle functions of $\{B^j_i(\lambda), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$ contains the ordering of angles of $\{B^j_i(\lambda^*_C), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$.

b To obtain the exact ordering of angles of $\{B^j_i(\lambda^*_C), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$, we continue to investigate that: for each $k, 1 \le k \le 2n^2$, whether $q''_k(\lambda^*_C) = q''_{k+1}(\lambda^*_C)$ or $q''_k(\lambda^*_C) < q''_{k+1}(\lambda^*_C)$. We already know that $q''_k(\lambda^*_C) \le q''_{k+1}(\lambda^*_C)$.
For each adjacent pair, i.e., $q''_k(\lambda^*_C)$ and $q''_{k+1}(\lambda^*_C)$, $1 \le k \le 2n^2$ (Note that $q''_{2n^2+1}(\lambda^*_C) = q''_1(\lambda^*_C)$), we compute critical values determined by them. Clearly, there are only $O(n^2)$ such critical values. Let K be the set of these $O(n^2)$ critical values and y' be the smallest feasible value in K.

Return $\lambda^*_C = y'$.

We call an ordering an *exact* ordering if only $=$ and $<$ are used in the ordering. From the algorithms for feasibility tests and the optimization problem described above, an exact ordering of the angle functions of $\{B^j_i(\lambda), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$ determines the feasibility of a value λ, since an exact ordering determines the set of sensors covering each corner point in $\{B^j_i(\lambda), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$. Hence, we have the following facts.

- The optimal value λ^*_C is some critical value computed in the comparison between two adjacent angle functions $q_{i(s)}(\lambda) + \frac{2k_1\pi}{n}$ & $q_{j(t)}(\lambda) + \frac{2k_2\pi}{n}$, $1 \le i, j \le n$, $s, t = 1, 2$ and $0 \le k_1, k_2 \le n-1$.

- The minimum of λ-values, which produces an ordering of angle functions of $\{B^j_i(\lambda)\}$ that is exactly the same as the ordering of angles of $\{B^j_i(\lambda^*_C)\}$, is λ^*_C.

In Step 3(a), we sort all the angle functions of $\{B^j_i(\lambda), 1 \le i \le n \text{ and } 1 \le j \le 2n\}$ for an unknown value $\lambda = \lambda^*_C$. The computed ordering, denoted by O', might not be the exact ordering for $\lambda = \lambda^*_C$, denoted by O^*, but, O' contains O^*. In Step 3(b), we continue to find the exact ordering O^*, and said that the smallest feasible value y' in K is λ^*_C. The proof is described briefly as follows. Suppose that K does not contain λ^*_C. As we know, λ^*_C is a critical value between a pair of angle functions, say, $q''_k(\lambda)$ and $q''_l(\lambda)$ where $1 \le k < l \le 2n^2$ and $l \ne k+1$. So, we can see that $q''_k(\lambda^*_C) = q''_l(\lambda^*_C)$. Since $q''_k(\lambda^*_C) \le q''_{k+1}(\lambda^*_C) \le \cdots \le q''_l(\lambda^*_C)$, we have that $q''_k(\lambda^*_C) = q''_{k+1}(\lambda^*_C) = \cdots = q''_l(\lambda^*_C)$, which implies that λ^*_C is in K. This proves the correctness of the optimization algorithm.

From the above discussion, only $O(\log n)$ feasibility tests in total are needed to compute λ^*_C. Therefore, we have the following theorem.

Theorem 6 *The min-max problem on the disk can be solved in $O(n^{3.5} \log n)$ time.*

Note that our algorithm can be easily extended to the model in which all sensors are arbitrarily located on the plane (not restricted to the interior of the disk $\mathcal{C}$).

4.2 On a simple polygon

In this model we discuss the min-max problem on a simple polygon as defined in Section 2.2. The parametric-searching approach for a disk (described in Section 4.1) should work for the case of a polygon where the destination positions of all sensors lie on the perimeter of the polygon. The main difficulty here is that to check the feasibility of a positive value λ, there might be $O(m)$ isolated polygonal chains of ∂P within the disk $\mathcal{C}_i$ (of radius λ centered at position A_i) for each sensor A_i. In other words, for a given positive value of λ, each sensor will contribute $O(m)$ candidate sets of n destination positions on P instead of at most two candidate sets on a circle. Hence, one can determine whether a given positive value λ is feasible in the min-max problem on a simple polygon by solving $O(mn)$ matching problems of size n. Therefore, the feasibility test of the min-max problem on a simple polygon can be solved in $O(mn^{3.5})$ time.

Theorem 7 *The min-max problem on a simple polygon can be solved in $O(mn^{3.5} \log n)$ time where m is the size of the simple polygon.*

5 Approximation algorithms for the min-sum problem

In this section we discuss the min-sum problem on a disk and a simple polygon, and provide approximation solutions.

5.1 On the disk

Let $\lambda^*_{s,C}$ be the optimal value of the min-sum problem on the disk, i.e., $\lambda^*_{s,C} = \min_{\sigma,\theta} S_n(\sigma, \theta)$.

We present two approximation algorithms for the min-sum problem. One algorithm (the *first approach*) has an approximation ratio $\pi+1$ (Section 5.2). The other (the *second approach*) uses the first approach as a subroutine to obtain lower and upper bounds for $\lambda^*_{s,C}$ and has an approximation ratio $1 + \epsilon$, where ϵ is an arbitrary constant (Section 5.3).

More notations are introduced as follows. Let $\hat{d}_C(X, Y)$ denote the shortest arc distance between two points X and Y on the boundary of the disk $\mathcal{C}$ and let $\hat{\pi}_C(X, Y)$ denote the arc of length $\hat{d}_C(X, Y)$ between X and Y. Clearly, $\hat{d}_C(X, Y) \leq \pi$ for any two points X and Y on $\mathcal{C}$. For a point X on $\mathcal{C}$, we denote by $\hat{Q}_X(r)$ the arc consisting of all points Y on $\mathcal{C}$ such that $\hat{d}_C(X, Y) \leq r$.

For each $i = 1, 2, \ldots, n$, let ω_i be the smallest distance between A_i and the disk $\mathcal{C}$, and denote by B_i the point on $\mathcal{C}$ for which the distance $d(A_i, B_i) = \omega_i$. We note that for each $i = 1, 2, \ldots, n$, B_i is unique if A_i is not located at the center of $\mathcal{C}$. In the case when A_i is located at the center of $\mathcal{C}$, an arbitrary point on $\mathcal{C}$ is selected to be B_i. Let $\Omega = \sum_{i=1}^{n} \omega_i$. Hence, we have the following lemma.

Lemma 8 $\Omega \leq \lambda^*_{s,C}$.

5.2 The first approach

The first approach, called Algorithm 1, consists of three steps.

Algorithm 1

Step 1 For each sensor A_i, $i = 1, 2, \ldots, n$, compute B_i.

Step 2 Compute the optimal min-sum assignment for the set of n points $B_1, B_2, \ldots, B_n$, by using the algorithm for sensors on the perimeter of the disk described in Section 3.2. Let B'_i be the destination of B_i, $1 \leq i \leq n$.

Step 3 Move A_i to $B'_i, 1 \leq i \leq n$, and compute $S_n^1 = \sum_{i=1}^{n} d(A_i, B'_i)$.

In Section 3.2 we showed that Step 2 of Algorithm 1 can be implemented in $O(n^2)$ time. Thus the above algorithm can be solved in $O(n^2)$ time.

5.2.1 Approximation bound of Algorithm 1

In this section, we show that S_n^1 computed by the first approach is bounded by $(\pi+1) \times \lambda^*_{s,C}$.

Suppose that A'_i is the destination of sensor A_i, $i = 1, 2, \ldots, n$, in an optimal solution. Clearly, $A'_1, A'_2, \ldots, A'_n$ lie on $\mathcal{C}$ and form a regular n-gon. Also, $\sum_{i=1}^{n} \hat{d}_C(B_i, B'_i) \leq \sum_{i=1}^{n} \hat{d}_C(B_i, A'_i)$ since $\{B'_1, B'_2, \ldots, B'_n\}$ is an optimal solution for the one dimensional min-sum problem with the input $\{B_1, B_2, \ldots, B_n\}$. The following lemma is easy to show.

Lemma 9 *For any two points x, y on $\mathcal{C}$, $\hat{d}_C(x, y) \leq \frac{\pi}{2} \times d(x, y)$.*

In the following, we establish an upper bound on S_n^1.

$$
\begin{aligned}
S_n^1 &= \sum_{i=1}^{n} d(A_i, B'_i) \\
&\leq \sum_{i=1}^{n} [d(A_i, B_i) + \hat{d}_C(B_i, B'_i)] \text{ (Triangle Inequality)} \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \sum_{i=1}^{n} \hat{d}_C(B_i, A'_i) \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \frac{\pi}{2} \times \sum_{i=1}^{n} d(B_i, A'_i) \text{ (Lemma 9)} \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \frac{\pi}{2} \times \sum_{i=1}^{n} [d(B_i, A_i) + d(A_i, A'_i)] \text{ (Triangle Inequality)} \\
&\leq (\pi + 1) \times \lambda^*_{s,C} \text{ (Lemma 8).}
\end{aligned}
$$

From the above discussion, it follows that

Theorem 10 *Algorithm 1 can be implemented in $O(n^2)$ time and its approximation ratio is no more than $\pi + 1$.*

5.3 The second approach

The following lemma is crucial for the second approach (Algorithm 2).

Lemma 11 *In an optimal solution, there exists at least one sensor A_i $(1 \leq i \leq n)$, such that its destination A'_i on $\mathcal{C}$ is on the arc $\hat{Q}_{B_i}(\frac{\pi}{2} \times \frac{S_n^1}{n})$.*

Proof. It is clear that $S_n^1 \geq \lambda_{s,C}^*$. Let A_i' be the destination of sensor A_i in an optimal solution, $i = 1, 2, \ldots, n$. Then there is at least one sensor, say A_k $(1 \leq k \leq n)$, such that the distance $d(A_k, A_k')$ is no more than $\frac{S_n^1}{n}$.

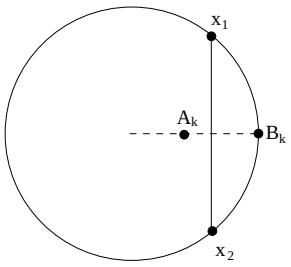

Figure 7: Lemma 11: $\hat{d}_C(x_1, B_k) = \hat{d}_C(x_2, B_k) = \frac{\pi}{2} \times \frac{S_n^1}{n} \Rightarrow d(x_1, x_2) \geq \frac{2S_n^1}{n}$ (Lemma 9).

By Lemma 9, all points on $\mathcal{C}$ at distance no more than $\frac{S_n^1}{n}$ from A_k lie on the arc $\hat{Q}_{B_k}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ (recall that B_k is the point on $\mathcal{C}$ closest to A_k), which completes the proof of Lemma 11. ■

We now describe the second algorithm.

Algorithm 2

Step 1 Use Algorithm 1 to compute S_n^1, as defined above.

Step 2 For each $i = 1, 2, \ldots, n$, find the arc $\hat{Q}_{B_i}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ and compute the set of points that partition the arc into $\lceil \frac{1}{\epsilon'} \rceil$ pieces of equal length where $\epsilon' = \frac{2\epsilon}{\pi(\pi+1)}$. Clearly, there are $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1)$ such points in total.

Step 3 For each point X, construct a regular n-gon P_X with one vertex located at X, and find the optimal assignment of the n sensors $A_1, A_2, \ldots, A_n$, to the vertices of P_X by solving a weighted bipartite matching problem. (The Hungarian method to solve the weighted matching problem in a complete bipartite graph of size n takes $O(n^3)$ time (see [10])).

Step 4 Among all $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1)$ regular n-gons thus constructed, find the one with the minimum cost (denoted by S_n^2), and output the arrangement of the n sensors to the vertices of the n-gon.

Remark. Step 1 and Step 2 of Algorithm 2 use Algorithm 1 as a subroutine to locate a destination point that lies within the arc $\hat{Q}_{B_i}(\frac{\pi}{2} \times \frac{S_n^1}{n})$, which is then partitioned into pieces.

5.3.1 Analysis of the second approach

First, it is evident that the running time of the second approach is determined by the time needed to solve $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1) \in O(\frac{n}{\epsilon})$ bipartite matching problems.

According to Lemma 11, there exists an optimal solution in which one of the vertices of the corresponding regular n-gon is located at a point on the arc $\hat{Q}_{B_k}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ for some k, $1 \leq k \leq n$. In Step 2, the arc $\hat{Q}_{B_k}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ is partitioned into $\lceil \frac{1}{\epsilon'} \rceil$ pieces, and therefore, the length of each piece is no more than $\frac{\pi S_n^1 \epsilon'}{n}$ (note that the length of $\hat{Q}_{B_k}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ is $\frac{\pi S_n^1}{n}$). Since all possible values of k are considered, the difference between S_n^2 (computed by the second approach) and $\lambda_{s,C}^*$ (the optimal cost) is no more than

$$n \times \frac{1}{2} \times \frac{\pi S_n^1 \epsilon'}{n} = \frac{\pi S_n^1 \epsilon'}{2} = \frac{S_n^1 \epsilon}{\pi + 1} \leq \epsilon \lambda_{s,C}^* \text{ (Theorem 10).}$$

Therefore, we have the following theorem.

Theorem 12 *The approximation ratio of Algorithm 2 is no more than $1 + \epsilon$ for a given constant ϵ, and the running time is $O(\frac{1}{\epsilon} n^4)$.*

5.4 On a simple polygon

Let $\lambda_{s,P}^*$ be the optimal value of the min-sum problem on a polygon P. In this subsection we present an approximation algorithm for the min-sum problem on P, which has an approximation ratio $1 + \epsilon$ where ϵ is an arbitrary constant.

Our algorithm for a simple polygon is very similar to the second approach for the disk. This is due to the fact that we use the Euclidean metric (not the geodesic metric) to measure the distance between two points in the plane. In our second approach for the disk, we use Algorithm 1 as a subroutine to obtain lower and upper bounds for $\lambda_{s,C}^*$. However, our approximation algorithm for a simple polygon uses the solution for the min-max problem on the polygon to obtain lower and upper bounds for $\lambda_{s,P}^*$. Let $\lambda_{m,P}^*$ be the optimal value of the min-max problem on P. It is easy to see that $\lambda_{m,P}^* \leq \lambda_{s,P}^* \leq n \times \lambda_{m,P}^*$.

Our algorithm for a simple polygon P is described below.

Step 1 Use the approach for the min-max problem on P to compute $\lambda_{m,P}^*$ as described above.

Step 2 For each i, j, where $1 \leq i \leq n$ and $0 \leq j < n$, find the sub-edge $e'_{i,j}$ of edge e_j that is within the circle of radius $\lambda_{m,P}^*$ centered at position A_i, and compute a set of points that partition the sub-edge into $\lceil \frac{n}{\epsilon} \rceil$ pieces of equal length. There are $mn \times (\lceil \frac{n}{\epsilon} \rceil + 1) \in O(\frac{mn^2}{\epsilon})$ such points in total.

Step 3 For each point X, construct a set of n positions on P such that one of them is located at X and the polygonal distance between any two consecutive positions is $L(P)/n$, and find the optimal assignment of the n sensors $A_1, A_2, \ldots, A_n$ to the set of n positions by using the algorithm [10].

Step 4 Among all $O(\frac{mn^2}{\epsilon})$ candidate sets of n positions thus constructed, find the one with the minimum cost.

It is evident that the running time of the above approach is determined by the time needed for solving $O(\frac{mn^2}{\epsilon})$ weighted bipartite matching problems.

The reason why the approximation ratio of the above approach is bounded by $1 + \epsilon$, is as follows. Since $\lambda_{s,P}^* \leq n \times \lambda_{m,P}^*$, there is at least one sensor whose moving distance to

its destination is no more than $\lambda^*_{m,P}$ in an optimal solution. Let A_i be one such sensor; its destination position lies on edge e_j in the optimal solution. In Step 2, the sub-edge $e'_{i,j}$ is partitioned into $\lceil \frac{n}{\epsilon} \rceil$ pieces, and therefore, the length of each piece is no more than $\frac{2\epsilon\lambda^*_{m,P}}{n}$. Since all possible values of i and j are considered, the difference between the value computed by the above approach and $\lambda^*_{s,P}$ (the optimal cost) is no more than

$$n \times \frac{1}{2} \times \frac{2\lambda^*_{m,P}}{n} = \epsilon\lambda^*_{m,P} \leq \epsilon\lambda^*_{s,P}.$$

Therefore, we have the following theorem.

Theorem 13 *The approximation ratio of the approach for a simple polygon is no more than* $1+\epsilon$ *for a given constant* ϵ*, and the running time is* $O(\frac{1}{\epsilon}mn^5)$.

5.5 Experimental results on the complexity of the min-sum problem

It is not known whether the min-sum problem can be solved optimally. Two related problems which could help clarify the issue are the following.

1. Given a counter-clockwise ordering of n sensors on the perimeter of the disk $\mathcal{C}$, solve the min-sum problem.

2. Sweep a regular n-gon along the perimeter of $\mathcal{C}$. The initial position of one corner point of the regular n-gon has angle of zero and the end position of that corner point has angle of π/n. Find the number of different counter-clockwise orderings of n sensors on the perimeter of $\mathcal{C}$, that is, the number of changes of their matchings to the corner points of the n-gon.

Table 1 shows our experimental results to resolve the second problem described above.

Table 1: Experimental result for the number of different orderings (over 20 test sets)

# of sensors (n)	# of regular n-gons (t)	average # of orderings
10	1,000	5.40
20	2,000	8.40
30	3,000	11.75
40	4,000	15.45
50	5,000	18.10
60	6,000	19.80
70	7,000	24.50
80	8,000	26.60
90	9,000	30.85
100	10,000	35.55

The first column in the table represents the number n of sensors contained in $\mathcal{C}$ (the n sensors are randomly generated), the second column represents the number t of regular n-gons being tested (i.e., the arc distance between two consecutive regular n-gons is $\frac{2\pi}{t*n}$),

and the third column represents the average number of different counter-clockwise orderings of the n sensors to the vertices of the t n-gons (for each value of n, 20 different sets of n sensors are generated). From this table, we can see that there are no more than n different counter-clockwise orderings in the experiment.

6 Conclusion and open problems

In this paper we gave an algorithm for solving the min-max problem and a PTAS (Polynomial Time Approximation Scheme) for the min-sum problem in both one and two dimensions. Although it is unknown whether the min-sum problem is NP-hard, we conjecture that it can be solved in polynomial time. In addition, several other variants of the problem on simple polygons and regions are of interest for further investigation, including k-barrier coverage, regions with holes, and various types of sensor placements and motions. Thus, in Subsection 2.2, in order to minimize the number of sensors used when scanning the perimeter, one should take into account sections already scanned. For example, this is the case if the polygon is a narrow rectangle of height less than the range of a sensor; this in itself is an interesting optimization problem which is worthy of further investigation. Also of interest is to refine the sensor motion model, the network model, and the communication model in order to enable effective intrusion detection and barrier coverage. For example, the communication model becomes crucial when assuming the sensors either do not have knowledge of the region or do not know their coordinates.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable feedback, insights and comments.

References

[1] N. Ahmed, S.S. Kanhere, and S. Jha. The holes problem in wireless sensor networks: a survey. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(2):4–18, 2005.

[2] 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.

[3] 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.

[4] R. Cole. Slowing down sorting networks to obtain faster sorting algorithms. *Journal of the ACM (JACM)*, 34(1):200–208, 1987.

[5] N. Heo and PK Varshney. A distributed self spreading algorithm for mobile wireless sensor networks. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 3, 2003.

[6] N. Heo and PK Varshney. Energy-efficient deployment of intelligent mobile sensor networks. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(1):78–92, 2005.

[7] J.E. Hopcroft and R.M. Karp. An $n^{2.5}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[8] S. S. Hu. 'Virtual Fence' along border to be delayed. *Washington Post*, Thursday, February 28, 2008.

[9] B. Jung and G.S. Sukhatme. Tracking targets using multiple robots: the effect of environment occlusion. *Autonomous Robots*, 13(3):191–205, 2002.

[10] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

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

[12] J. Liu, P. Cheung, F. Zhao, and L. Guibas. A dual-space approach to tracking and sensor management in wireless sensor networks. *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 131–139, 2002.

[13] N. Megiddo. Applying parallel computation algorithms in the dsign of serial algorithms. *Journal of the ACM (JACM)*, 30(4):852–865, 1983.

[14] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc. New York, NY, USA, 1987.

[15] A. Schrijver. Disjoint circuits of prescribed homotopies in a graph on a compact surface. *Journal of Combinatorial Theory Series B*, 51(1):127–159, 1991.

[16] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *The International Journal of Robotics Research*, 21(8):735, 2002.