Theoretical Computer Science 887 (2021) 1–10

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On synchronization and orientation in distributed barrier coverage with relocatable sensors ☆

Mohsen Eftekhari [a], Paola Flocchini [b,*], Lata Narayanan [c], Jaroslav Opatrny [c], Nicola Santoro [d]

[a] Google Inc., United States of America
[b] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada
[c] Dept. of Comp. Science and Soft. Eng., Concordia University, Montréal, Canada
[d] School of Computer Science, Carleton University, Ottawa, Canada

ARTICLE INFO

Article history:
Received 5 May 2018
Received in revised form 17 March 2020
Accepted 25 June 2021
Available online 1 July 2021
Communicated by L.A. Gasieniec

Keywords:
Distributed barrier coverage
Mobile sensors
Linear networks
Orientation
Asynchrony

ABSTRACT

Consider n identical relocatable *sensors*, with sensing range r and visibility range $2r$, initially placed at arbitrary positions on a line segment barrier; each sensor can detect the presence of an intruder in its sensing range, and is said to cover the portion of the barrier that intersects with its sensing range. Sensors operate in Look-Compute-Move cycles: in a cycle a sensor determines the positions of sensors in its visibility range, it computes its next position (within its visibility range), and then moves to the calculated position. The *barrier coverage* problem we consider is for the sensors to independently make decisions and movements so to reach final positions whereby they collectively cover the barrier; in particular we are interested in oblivious (or memoryless) sensors, and focus on the impact that the level of synchrony and the presence/absence of *orientation* (i.e., global notion of "left-right") have on the solvability of the problem.

It is known that without orientation, oblivious sensors can solve the problem if they are *fully synchronous* (i.e., they operate in synchronous rounds and are all active at every round). In this paper, we prove that orientation is critical to being able to solve the problem if we relax the assumption of full synchronization.

We first show that if sensors are unoriented, then barrier coverage is unsolvable even in the *semi-synchronous* setting. In contrast, if sensors agree on a global orientation, then we give an algorithm for barrier coverage, even in the completely *asynchronous* setting. Finally, we extend the existing result of Cohen and Peleg and show that *convergence* to barrier coverage by unoriented sensors in the semi-synchronous model is possible with bounded visibility range $2r + \rho$ (for arbitrarily small $\rho > 0$) and bounded mobility range r.

© 2021 Published by Elsevier B.V.

☆ A preliminary version of this paper has been presented at the 21st International Colloquium on Structural Information and Communication Complexity.
* Corresponding author.
E-mail address: paola.flocchini@uottawa.ca (P. Flocchini).

https://doi.org/10.1016/j.tcs.2021.06.038
0304-3975/© 2021 Published by Elsevier B.V.

1. Introduction

1.1. The problem

A wireless sensor network consists of several sensors, each equipped with a sensing module. Among the many applications of sensor networks (e.g., [15]), the establishment of *barrier coverage* has an important place, and it has been studied intensively in the literature; it guarantees that any intruder attempting to cross the perimeter of a region in the plane (e.g., crossing an international border) is detected by one or more of the sensors (e.g., see [1,2,5,6,11,16,18,19]). By protecting the access to the region, barrier coverage provides a less expensive alternative to a complete coverage of the region (e.g., [18]). A barrier of a two dimensional region can be modeled as a line segment of length $L \in \mathcal{Z}$, covering the interval $[0, L]$ on the x-axis; sensors are deployed along the barrier. Sensors are assumed to have an identical sensing range r. Any point on the line segment outside the sensing range r of any sensor would be a weak point on the barrier. Thus the barrier is *covered* if every point of the line segment is within the sensing range of at least one sensor. Clearly, at least $\bar{n} = \lceil \frac{L}{2r} \rceil$ sensors of range r are needed to achieve a coverage.

Barrier coverage, in the case of *static sensors*, can be achieved by careful (i.e., non ad hoc) deterministic deployment of $\bar{n}$ sensors, but this could be unfeasible when the access to the barrier is difficult. Alternatively, a large number $N \gg \bar{n}$ of sensors can be randomly deployed, but barrier coverage can only be probabilistically guaranteed [11–13].

In sensor networks composed of *relocatable sensors*, every sensor has a movement module that enables the sensor to move along the barrier. Hence, although initially sensors are located at arbitrary positions on the line without providing barrier coverage, they may move to new points on the line so that the entire barrier is covered (e.g., [3,5–7,17]). In this paper we study the problem of barrier coverage with relocatable sensors.

The *centralized* version of the problem, in which the initial locations of all sensors are known, has been studied and solutions proposed, focusing on minimizing some cost measures (e.g., traveled distances) [3,5,6,14]. In these centralized solutions, given the initial position of sensors, the algorithm determines the final positions that the sensors should occupy; notice that $\bar{n}$ sensors suffice for a centrally directed relocation of sensors. However, in the context of sensor networks deployed in an ad hoc manner, typically there is no central control or authority, and no global knowledge of the locations of the sensors is available. Indeed, the sensors might not even know the total number of sensors deployed, or the length of the barrier. Thus every sensor must make decisions on whether and where to move, based only on local information in an autonomous and decentralized way.

In order to develop a solution protocol for a *distributed* setting, it is first of all necessary to model such a setting. Following the approach used in the research on autonomous mobile robots (e.g., [10]), sensors are modeled as mobile computational entities. The entities are anonymous and identical, have no centralized coordination, have a sensing range as well as a visibility[1] range: their decisions are made solely based on their observations of their surroundings. Each entity alternates activity with inactivity. When becoming active, it executes a `Look-Compute-Move` operational cycle and then becomes inactive. In a cycle, an entity determines the positions of the other entities in its visibility range (`Look`); then it computes its own next position (`Compute`); and finally it moves to this new position (`Move`). In the cases of sensor networks, the visibility range v is limited [9]; we assume $v = 2r$, which is the minimum visibility radius necessary for sensors to determine local gaps in coverage. The movements of the sensors are said to be *bounded* if there is a maximum distance they can move in each cycle, and *rigid* if they are not interrupted (e.g., by an adversary).

Depending on the assumptions on the activation schedule and the duration of the cycles, three main settings are identified. In the *fully-synchronous* setting (FSYNC), all sensors are activated at each time step, and each cycle is executed simultaneously. The *semi-synchronous* setting (SSYNC) is like the fully synchronous one except that each activation might involve only a subset of the sensors; activations are fair: each sensor will be activated infinitely often. In the *asynchronous* setting (ASYNC), no assumption is made on timing of activation, other than fairness, nor on the duration of each computation and movement, other than it is finite.

The first *distributed* algorithmic investigation of the barrier coverage problem has been recently presented for the discrete line [7], solving the problem in the fully synchronous setting, FSYNC. Interestingly, it is shown that the sensors can be totally *oblivious*, that is, at the beginning of a cycle, a sensor does not (need to) have any recollection of previous operations and computations. Furthermore, the sensors are completely unoriented; they have no concept of left and right. Finally the algorithm terminates (i.e., there is no further movement of sensors) for any $n \geq \bar{n}$, hence even with the minimal number used by centralized solutions.

Notice that when $L/2r$ is an integer and $n = L/2r = \bar{n}$, the barrier coverage problem is equivalent to the *uniform deployment* problem (studied for lines and circles, see [4,8,9]) on a line segment, which requires the oblivious sensors to move to equidistant positions between the borders of the segment. This problem has been studied on a line [4] assuming that a sensor can always see the sensors that are closest to it, regardless of their distance, and it always reaches its destination, regardless of its distance; in other words, both visibility and movements are a priori unbounded. Under these assumptions, an SSYNC distributed protocol that *converges* with rigid movements to uniform covering (and thus to barrier coverage) was given in [4]. However, equidistant positions are not required for barrier coverage when $n > L/2r$.

[1] Combined with mobility, it provides stigmergic communication between sensors within range.

1.2. Main contributions

In this paper we first of all investigate under what conditions $\bar{n}$ oblivious sensors can actually achieve barrier coverage in the complex semi-synchronous and asynchronous settings, without requiring unbounded visibility or mobility range.

We prove that a crucial factor for solvability of the barrier coverage problem is whether the network is *oriented* or *unoriented*. In an oriented network each sensor has a notion of “left-right”, and this notion is *globally consistent*. In an unoriented network, sensors have no sense of “left-right” direction.

In particular, we prove that the problem is *unsolvable* by $\bar{n}$ oblivious sensors in SSYNC (and thus ASYNC) if the network is unoriented. The result holds even if all movements are rigid. On the other hand, we prove that, if the network is oriented, the problem is solvable even in ASYNC and even if movements are not rigid (i.e., they can be interrupted by an adversary). The proof is constructive: we present an ASYNC protocol that allows any $n \geq \bar{n}$ oblivious sensors to achieve barrier coverage within finite time and terminate, even if movements are non-rigid. In other words, we show that, with orientation, it is possible to achieve barrier coverage in a totally local and decentralized way, asynchronously, obliviously, and with movements interruptible by an adversary; furthermore, this is achievable with the same number of sensors of the optimal totally centralized solution with global knowledge of all parameters.

For an unoriented network we also show that allowing a slightly larger visibility range, (e.g., $v = 2r + \rho$ for an arbitrary small ρ), $\bar{n}$ oblivious sensors can converge with rigid movements to barrier coverage in SSYNC, extending the result of [4] to fixed limited visibility and bounded movements.

2. Model and notation

We model the barrier with a line segment of length $L \in Z$ covering the interval $[0, L]$ on the x-axis. A sensor network consists of a set of n sensors $\{s_1, s_2, \ldots, s_n\}$ located on the segment.

A sensor is modeled as a computational entity capable of moving along the segment; it is equipped with a sensing module and a visibility module. A sensor can sense an intruder if and only if it lies within the sensor’s sensing range; it can see another sensor if and only if it lies within the sensor’s visibility range. In this paper, we assume that all sensors have the same sensing range r and the same visibility range v.

Sensors are autonomous, anonymous and identical (i.e., without central authority, distinct markers or identifiers); they all execute the same algorithm. Sensors are said to be *oriented* if and only if all sensors agree on a global left and right directions; they are called *unoriented* if they do not have a sense of left and right.

Let x_i^t denote the position of ith sensor s_i from the left at time t. We assume that for every sensor $r \leq x_i^0 \leq L - r$, and that for $i \neq j$, we have $x_i^0 \neq x_j^0$. For convenience, we assume that $x_1^0 < x_2^0 \cdots < x_n^0$. We emphasize that while these names and positions of sensors facilitate our proofs, they are not known to any of the sensors. Sensors should not move outside of the barrier. Thus, in addition, we assume to have two special sensors s_0 and s_{n+1} that are used to mark the ends of the interval $[0, L]$. They are immobile, and are always located at $-r$ and $L + r$ and they do not require any sensing capabilities or visibility. However, the other sensors in the network cannot distinguish these special sensors from any other sensors. Thus the entire set of sensors is $S = \{s_0, s_1, \ldots, s_n, s_{n+1}\}$.

The sensors can be *active* or *inactive*. When *active*, a sensor performs a `Look-Compute-Move` cycle of operations: the sensor first observes the portion of the segment within its visibility range obtaining a snapshot of the positions of the sensors in its range at that time (`Look`); using the snapshot as an input, the sensor then executes the algorithm to determine a destination point (`Compute`); finally, the sensor moves towards the computed destination, if different from the current location (`Move`). After that, it becomes *inactive* and stays idle until the next activation. Sensors are *oblivious*: when a sensor becomes active, it does not remember any information from previous cycles.

A move is said to be *non-rigid* if it may stop before the sensor reaches its destination, e.g. because of limits to the sensor’s motion energy, or being stopped by an adversary before the sensor reaches its destination. The only constraint on the adversary is that, if it interrupts a movement of a sensor before reaching its destination, a sensor moves at least a minimum distance $\delta > 0$ (otherwise, no destination can ever be reached). If no such an adversary exists, the moves are said to be *rigid*.

Depending on the amount of synchronization existing among the cycles of the different sensors, three main sub-models are defined: fully synchronous, semi-synchronous, and asynchronous. In both the *fully-synchronous* (FSYNC) and the *semi-synchronous* (SSYNC) models there is a common clock, the sensors operate in synchronous rounds, and all sensors active in a round execute and terminate their cycle by the next round. The only difference is that in FSYNC all sensors are activated in every round, while in SSYNC a possibly different subset is activated in each round; the choice of the subset is adversarial but subject to the basic fairness assumption that every sensor will be activated infinitely often. In the *asynchronous* model (ASYNC), there is no global clock and there is no common notion of time; each sensor is activated infinitely often at arbitrary instants of time, and in any cycle the duration of each activity is finite but otherwise unpredictable.

A sensor with sensing range r *covers* the portion of the segment within its sensing range; therefore the *coverage length* of a sensor is $2r$. *Barrier coverage* is achieved if every point on the segment is covered by some sensor. An *overlap* is a maximal interval on $[0, L]$ such that every point in the interval is within the sensing range of more than one sensor. A *coverage gap* is any maximal interval of the segment where no point is within the sensing range of any sensor. We say that *ε-approximate barrier coverage* is achieved if the length of any coverage gap is at most ε.

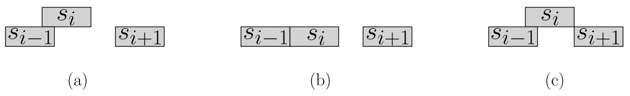

Fig. 1. The three types of sensors under consideration.

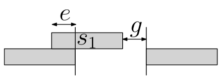

Fig. 2. Arrangement for proof of Lemma 3.1; $n = 1$.

The goal of an algorithm for barrier coverage is to move sensors to final positions so that the entire barrier is covered, i.e., there are no coverage gaps. Observe that if $2rn > L$, then the final consecutive positions are not necessarily equidistant.

We say an algorithm $\mathcal{A}$ for barrier coverage *terminates* on input S at time t if and only if when running $\mathcal{A}$ on S, no sensor in S moves at any time $t' \geq t$. We say that algorithm $\mathcal{A}$ *solves* the barrier coverage problem if there is a time t at which the algorithm terminates on any input S and barrier coverage is achieved. We say an algorithm $\mathcal{A}$ *converges* to barrier coverage on input S if and only if for any $\varepsilon > 0$ there is a time t such that at any time $t' \geq t$ the size of any coverage gap is at most ε. We say that algorithm $\mathcal{A}$ solves the ε*-approximate* barrier coverage problem for $\varepsilon > 0$ if and only if it converges on any input S.

Unless otherwise specified, we assume $v = 2r$, which is the minimum visibility radius necessary for sensors to determine local coverage gaps. More precisely, sensor s_i^t is able to see all other sensors located in $[x_i^t - 2r, x_i^t + 2r]$. For convenience, we say s_i^t sees s_j^t on its right if and only if $0 < x_j^t - x_i^t \leq 2r$ and s_i^t sees s_k^t on its left if and only if $0 < x_i^t - x_k^t \leq 2r$. Observe that a sensor is able to detect when its sensing area overlaps with another sensor's sensing area.

Note that in our figures, each sensor is represented by a rectangle which shows the interval that the sensor covers on the line barrier. Also for convenience, two sensors whose coverage lengths overlap are placed at different levels in the illustration; however in our assumptions, all sensors have circular sensing area and are initially placed on the barrier and can only move on the barrier.

3. Impossibility without orientation

In this section we consider the case where sensors are unoriented. We show that there is no algorithm solving the barrier coverage in the SSYNC model with $\bar{n}$ sensors; since an adversary in the ASYNC model has at least the power it has in the SSYNC model, obviously this impossibility result also holds for the ASYNC model.

We give an adversary argument, by creating input arrangements and activation schedules that force any algorithm in the SSYNC model to either not terminate, or terminate without coverage. This result is obtained even assuming that the movements are rigid; i.e., a sensor can always reach the destination it has computed.

We focus on three types of sensors (see Fig. 1): (a) sensors that have an overlap on one side, and a coverage gap (or a gap for short) on the other side, (b) sensors that are attached to the next sensor on one side and a gap on the other side and (c) sensors that have an overlap on one side and are attached to the next sensor on the other side. Any algorithm for barrier coverage must specify rules for movement in each of these situations. Note that, with $2r$ visibility range, sensors can only determine whether there exists a gap with a neighboring sensor but cannot determine anything about the length of such a gap. Thus, the magnitude of the movement of a sensor can only be a function of an overlap, if any, with a neighboring sensor, and cannot be a function of the length of an adjacent gap. We show that there exist arrangements and activation schedules for the sensors that defeat all possible combinations of these rules.

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

Lemma 3.1. *Consider an algorithm $\mathcal{A}$ for barrier coverage in the SSYNC model and assume that at time t we have sensor s_i with $x_i^t - x_{i-1}^t = 2r - e$ and $x_{i+1}^t - x_i^t = 2r + g$, with $e, g > 0$. If s_{i-1} and s_{i+1} are deactivated and only s_i is activated, there exists a time step $t' > t$ such that:*

(a) $x_i^{t'} = x_i^t + e$ *if* $g \geq e$ *and*
(b) $x_i^t + g \leq x_i^{t'} \leq x_i^t + e$ *if* $g < e$.

Proof. First we observe that $x_i^{t'} \geq x_i^t + min(g, e)$; i.e., the sensor s_i must eventually move at least distance $min(g, e)$ to the right. If not, the algorithm $\mathcal{A}$ would not terminate with barrier coverage on the arrangement shown in Fig. 2 for $n = 1$, since s_1 is the only sensor that can move in the arrangement.

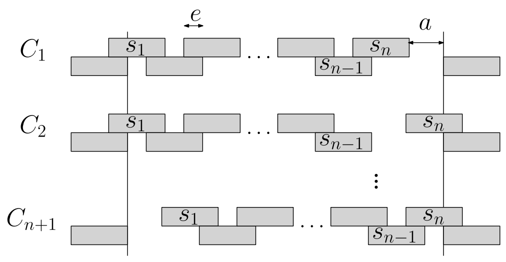

Fig. 3. Arrangement for proof of Lemma 3.1; $n = \lceil a/e \rceil$.

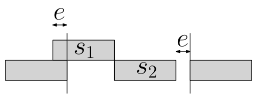

Fig. 4. Arrangement for proof of Lemma 3.3; $n = 2$.

Next we show that $x_i^{t'} \leq x_i^t + e$ for some $t' > t$. For the sake of contradiction, assume that there is a value of overlap e such that, according to $\mathcal{A}$, sensor s_i moves more than e; that is, s_i moves $e + a$ to the right, with $a > 0$. Then we can construct an activation schedule such that $\mathcal{A}$ never terminates on the input shown in Fig. 3. Choose $n = \lceil a/e \rceil$. A single sensor is activated in each step. Starting with configuration C_1, the sensors s_n to s_1 are activated in consecutive steps, yielding configurations $C_2, C_3, \ldots C_{n+1}$ in turn, and then the sequence of activations is reversed. It is easy to verify that at the end of the activation schedule, the initial arrangement C_1 is obtained again. The schedule can be repeated *ad infinitum*, forcing non-termination of the algorithm.

Summarizing, $x_i^t + e \geq x_i^{t'} \geq x_i^t + min(g, e)$ and the claim follows. □

Next we consider the behavior of a sensor s_i that is attached to its neighbor on its left, and has a gap on its right as in Fig. 1(b). We activate s_i and keep s_{i-1} and s_{i+1} deactivated. If s_i moves left, it creates an overlap with s_{i-1} and by Lemma 3.1(a), it will eventually move to the right to remove that overlap, and return to the same position. Alternatively, s_i may not move at all, or may move to the right. If it moves to the right, since it does not know the distance of the gap with s_{i+1} and has no overlap with s_{i-1}, it can only move a fixed constant distance, say b, calculated at the compute cycle. The lemma below is a consequence of the preceding discussion.

Lemma 3.2. *Let $\mathcal{A}$ be an algorithm for barrier coverage and s_i be a sensor with $x_i^t - x_{i-1}^t = 2r$ and $x_{i+1}^t - x_i^t > 2r$ at time t. If s_{i-1} and s_{i+1} are both kept deactivated and s_i is activated, there exists a time $t' > t$ such that $x_i^{t'} = x_i^t + h$ with $h \geq 0$.*

Finally, we consider the behavior of a sensor s_i that has an overlap e with s_{i-1} and is attached to sensor s_{i+1}, as shown in Fig. 1(c). As before, we activate only s_i and keep both s_{i-1} and s_{i+1} deactivated. If s_i moves left, it creates a gap with s_{i+1}. By Lemma 3.1(b), s_i must eventually move right, either returning to its initial position, or moving further right. If it moves right by more than the value of the overlap, then it creates a gap to its left, and once again by Lemma 3.1(b), it must move back left until the gap is removed. If for all values of the overlap, s_i makes a move to the right that does not eliminate the overlap, then we show below that the algorithm cannot achieve barrier coverage, leading to the conclusion that there must exist some value of overlap such that such a sensor will either not move, or move to exactly eliminate the overlap.

Lemma 3.3. *Consider an algorithm $\mathcal{A}$ for barrier coverage. There exists an overlap c with $0 < c < 2r$ such that for any sensor s_i with $x_i^t - x_{i-1}^t = 2r - c$ and $x_{i+1}^t - x_i^t) = 2r$, if s_i is the only one of $\{s_i, s_{i-1}, s_{i+1}\}$ to be activated, there exists a time step $t' > t$ such that either $x_i^{t'} = x_i^t + c$ (s_i moves right to exactly eliminate the overlap) or $x_i^{t'} = x_i^t$ (s_i returns to the same position).*

Proof. Assume the contrary. By the discussion preceding the lemma, we can conclude that for any overlap e, there exists a time step t' such that $x_i^{t'} = x_i^t + d$ with $0 < d < e$. Consider the arrangement of sensors shown in Fig. 4. We first activate s_1 until it moves distance d to the right. By assumption, there remains an overlap of $e - d$ between s_0 and s_1, and now there is an overlap of d between s_1 and s_2. We now keep s_1 deactivated, and activate s_2. Lemma 3.1 implies that sensor s_2 eventually moves exactly d to the right and eliminates the overlap completely. Observe that at this point, the arrangement repeats with only a different value of overlap. The new value of the overlap between s_0 and s_1 is strictly greater than zero, and the distance between s_1 and s_2 is exactly $2r$. Since this activation schedule can be repeated *ad infinitum*, algorithm $\mathcal{A}$ can never terminate with barrier coverage. □

We proceed to prove our main result:

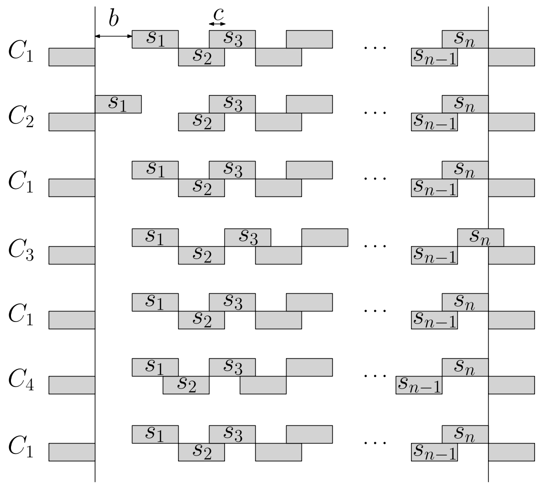

Fig. 5. Arrangement for proof of Theorem 3.1; $n = 1 + 2\lceil b/c \rceil$.

Theorem 3.1. *Let $s_1, s_2, \dots, s_n$ be n sensors with sensing range r initially placed at arbitrary positions on a line segment. If the sensors are unoriented and have visibility radius $2r$, there is no algorithm for barrier coverage in the* Ssync *model.*

Proof. Consider the arrangement of sensors shown in Fig. 5 with c chosen as in Lemma 3.3. If the value of h as specified in Lemma 3.2 is zero, then choose $b = c$, otherwise, choose $b = h$, and fix $n = 1 + 2\lceil b/c \rceil$. We create an activation schedule with three phases with a different set of sensors being activated in each phase, such that the sensors return to arrangement C_1 at the end of each phase. At each phase we only activate a subset of sensors and all other sensors are kept deactivated. We first activate only the sensor s_1. By Lemma 3.2, there is a future time step when either s_1 is in the same position (if $h = 0$), or it moves distance b to the left to yield arrangement C_2. In the second case, since sensors are unoriented, it will subsequently return to arrangement C_1. In the second phase, we activate only the sensors $\{s_3, s_5, \dots, s_n\}$. By Lemma 3.3, there is a future time when either these sensors return to arrangement C_1, or they have moved right by a distance c to reach arrangement C_3. In the second case, they will eventually return to arrangement C_1. In the third phase, we activate only the set of sensors $\{s_2, s_4, \dots, s_{n-1}\}$. Using the same logic, they will return to arrangement C_1, possibly via arrangement C_4. Observe that all sensors have been activated at least once during the schedule. By repeating the above schedule *ad infinitum*, we can force sensors to repeatedly return to the arrangement C_1, thus completing the proof. $\square$

4. Possibility with orientation

In this section, we present and analyze an algorithm, Oriented Sensors, for barrier coverage by any $n \geq \bar{n}$ oblivious oriented sensors in the Async model; that is, all sensors agree on left and right, but are completely asynchronous.

Algorithm 1: Oriented Sensors.

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

We proceed to prove the correctness of algorithm Oriented Sensors. A *collision* occurs if two distinct sensors move to exactly the same position. Since sensors are identical and anonymous, from the time a collision of two sensors happens, they cannot be distinguished and will behave exactly the same if they have the same activation schedule. Therefore a collision is fatal for a barrier coverage algorithm if $n = \bar{n}$, and must be avoided by the algorithm designer. This is precisely the reason that we restrict the distance of a move to the right to $r - \varepsilon$, while sensors move distance r when moving to the left. We show below that the algorithm above is *collision-free* and *order-preserving*.

Lemma 4.1. *Algorithm* Oriented Sensors *is a collision-free and order-preserving protocol.*

Proof. Consider a sensor s that is at position x and performs a `Look` at time t_1 and the corresponding `Move` to the *left* at time t_2. We claim that no sensor s' that is at a position $x' < x$ (to the left of s) at time t_1 can compute or perform a `Move` resulting in a collision or an order reversal with s at any time between t_1 and t_2. Since s computes a `Move` to the left at time t_1, it must be that $x' < x - 2r$, and furthermore, s will move to a position $\geq x - r$ at time t_2. Now, consider the *last* `Move` performed by s' at a time $\leq t_1$. Observe that the sensor s' must have been at position x' as a result of this `Move`. Consider the subsequent `Look` performed by s' at time t_3. If the `Move` computed as a result of this `Look` is a move to the right, the next position of s' is $\leq x' + r - \varepsilon < x - r$. Any subsequent `Look` performed by s' will compute a move to the right if and only if the position of s' is $\leq x - 2r$ and the computed destination must always be $< x - r$. Thus no collision or order reversal can result. Any moves to the left from the positions reachable by s' can clearly not cause collisions or order reversals.

Next we show that no sensor s' that is at a position $x' > x$ (to the right of s) at time t_1 can compute or perform a `Move` resulting in a collision or order reversal with s at any time between t_1 and t_2. Clearly, if $x' > x + r$, any move to the left can only bring it to a position $> x$. Suppose $x < x' \leq x + r$. If s' performs a `Look` after time t_1, then it can see s in its visibility range and therefore would not perform a `Move` to the left. So s' must have performed a `Look` at a time $t_3 < t_1$. For s' to have computed a move to the left, the position of s at time t_3 must have been $< x' - 2r$. As argued above, s cannot have subsequently arrived at position x at time t_1.

A similar argument shows that for a sensor s that is at position x and performs a `Look` at time t_1 and the corresponding `Move` to the *right* at time t_2, neither a sensor on its left nor a sensor on its right can compute a move resulting in a collision or order reversal with s. □

Next we show that there is a time after which no sensors will move left, and after this time, the sensors provide contiguous coverage of some part of the barrier including the sensor s_0.

Lemma 4.2. *For every sensor $s_i \in S - \{s_{n+1}\}$ there is a time t_i such that s_i never moves left at any time after t_i. Furthermore, there is no coverage gap between s_0 and s_i at any time after t_i.*

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

After time t_n, then, none of the sensors moves left, and furthermore there is no coverage gap between s_0 and s_n. The next two lemmas show that after this time, a sensor moves right under some circumstances, but can only move a finite number of times.

Lemma 4.3. *Assume s_i and s_{i+1} have an overlap of e at some time after t_n. Then for any j with $i + 1 \leq j \leq n$, if the sensors s_{i+1} to s_j are in attached position, and there is no overlap between s_j and s_{j+1}, then sensor s_j will eventually move at least $min(\delta, e)$ to the right.*

Proof. Let $t > t_n$ be a time when s_{i+1} performs a `Look` and s_i and s_{i+1} have an overlap of e. Clearly s_{i+1} will move right in the corresponding `Move`, creating an overlap between s_{i+1} and s_{i+2}. Inductively it can be seen that when s_{j-1} moves to the right, it creates an overlap with s_j, causing s_j to move at least $min(\delta, e)$ to the right. □

Lemma 4.4. *Every sensor makes a finite number of moves to the right after time t_n, which is the time when s_n never moves left anymore.*

Proof. We give an inductive proof. Clearly this is true for sensor s_0. Suppose sensor s_i has an overlap of e with sensor s_{i-1} at time t_n. Observe that s_{i-1} cannot move until and unless this overlap is removed. Since every time s_i moves to the right, it reduces this overlap by at least $min(e, \delta)$, it is clear that s_i can make at most $\lceil e/\delta \rceil$ moves to the right. If these moves remove the overlap, then s_i may move again only if s_{i-1} subsequently moves to the right and creates an overlap with s_i. Assuming inductively that s_{i-1} makes a finite number of moves to the right, we conclude that sensor s_i moves to the right a finite number of times. □

The above lemmas lead to the following theorem:

Theorem 4.1. *Let $s_1, s_2, \ldots, s_n$ be $n \geq \bar{n}$ sensors with sensing range r initially placed at arbitrary positions on a line segment. If the sensors have the same orientation and visibility radius of $2r$, Algorithm* ORIENTED SENSORS *always terminates with the barrier fully covered in the* ASYNC *model.*

Proof. Lemma 4.2 assures that after time t_n, no sensor moves left, and there is no coverage gap between sensors s_0 and s_n. It follows from Lemma 4.4 that there is a time, say $t' > t_n$, after which no sensor will move right. However, if there is a gap between s_n and s_{n+1} at time t', since there are enough sensors to cover the barrier, there must be an overlap between two sensors s_i and s_{i+1} for some $0 \leq i < n$. But Lemma 4.3 implies that the sensor s_n must eventually move to the right, a contradiction. It follows that after time t', there is no gap between s_n and s_{n+1} and therefore no gap between any sensors in S, that is, Algorithm ORIENTED SENSORS terminates with barrier coverage. □

5. On visibility and convergence

We have seen that, without orientation, barrier coverage with $\bar{n}$ sensors is impossible even in SSYNC (Theorem 3.1). Observe that the impossibility proof holds when the visibility range is precisely $2r$. So the question naturally arises of what happens in SSYNC if the visibility range is larger.

It is known that in SSYNC it is possible for $\bar{n}$ sensors to *converge* with rigid movements to equidistant positions *if* a sensor can always see the sensors that are closest to it, regardless of their distance (thus without a priori restrictions on the visibility range) and it can move to destination regardless of its distance (thus without a priori restrictions on the mobility range) [4]. In our setting these conditions do not hold. In this section, we show how that result can be extended to our setting. In fact, we prove that $\bar{n}$ oblivious sensors can converge with rigid movements to barrier coverage in SSYNC if $v = 2r + \rho$, where ρ is an arbitrarily small positive constant; furthermore they can do so with rigid movements of length at most r.

Consider Algorithm CONVERGENT COVERAGE shown below; it operates by first removing all visibility gaps within finite time, and then behaving as the algorithm of [4].

Algorithm 2: CONVERGENT COVERAGE.

```
Algorithm for sensor $s_i \in S$
if only one sensor $s_j \in \{s_{i+1}, s_{i-1}\}$ is visible to $s_i$ and $d = dist(s_i, s_j) < 2r$ then
  $s_i$ moves distance $\frac{2r-d}{2} + \frac{\rho}{2}$ away from $s_j$.
else
  if both $s_{i+1}, s_{i-1}$ are visible. then
    if $d_1 = dist(s_{i-1}, s_i) < d_2 = dist(s_{i+1}, s_i)$
    (resp. $d_1 = dist(s_{i+1}, s_i) < d_2 = dist(s_{i-1}, s_i)$) then
      $s_i$ moves $\frac{d_2-d_1}{2}$ toward $s_{i+1}$ (resp. toward $s_{i-1}$).
    end if
  end if
end if
```

Lemma 5.1. *If $s_j \in \{s_{i+1}, s_{i-1}\}$ is in the visibility range of s_i at time t, then for any time $t' > t$, sensor s_j remains in the visibility range of s_i.*

Proof. According to the algorithm, a movement is performed by s_i in a cycle only in two situations:

Case 1: Only one sensor s_j is visible to s_i and $d = dist(s_i, s_j) < 2r$. The case we need to consider is when also s_j is activated in this cycle and it sees only s_i. In this case, both sensors move at most $\frac{2r-d}{2} + \frac{\rho}{2}$ away from each other. After the movement we have that $dist(s_i, s_j)$ has become: $dist(s_i, s_j) = 2(\frac{2r-d}{2} + \frac{\rho}{2}) + d = 2r + \rho$. So, sensors s_i and s_j are still within visibility.

Case 2: Both s_{i+1} and s_{i-1} are visible to s_i. Let, without loss of generality, $d_1 = dist(s_j, s_i) < d_2 = dist(s_k, s_i)$ where $s_i, s_k \in \{s_{i-1}, s_{i+1}\}$. The case we need to consider is when s_j is also activated and it does not see the other neighboring sensor. In this case s_i moves at most $\frac{d_2-d_1}{2}$ toward s_k, and s_j moves at most $\frac{2r-d}{2} + \frac{\rho}{2}$ away from s_i. After the movement, we have that $dist(s_i, s_j)$ has increased as follows:

$$dist(s_i, s_j) = \frac{d_2 - d_1}{2} + d_1 + \frac{2r - d}{2} + \frac{\rho}{2} = \frac{d_2}{2} + r + \rho < 2r + \rho$$

So, sensors s_i and s_j are still within visibility. □

Lemma 5.2. *Within finite time there will be no visibility gaps.*

Proof. By Lemma 5.1, visibility is never lost once gained. Consider the visibility gaps. After each activation of a sensor next to a visibility gap, the size of that visibility gap is reduced by at least $\frac{\rho}{2}$. As a consequence, within a finite number of activations, all visibility gaps will be eliminated and all sensors will be within visibility to their neighbors. □

Lemma 5.3. *If at time t there are no visibility gaps, within finite time all coverage gaps will be of size at most ε, for any $\varepsilon > 0$.*

Proof. If there are no visibility gaps at time t then, by Lemma 5.1, for all $t' \geq t$ there will be no visibility gaps. Hence at all times $t' \geq t$ each sensor s_i when active sees its two neighbors s_{i-1} and s_{i+1}; furthermore, since the distance between two neighbors is at most $2r$, the computed destination of a sensor is at most at distance r. Notice that at this point in the algorithm, sensors move exactly as in Algorithm *Spread*, p. 76 of [4] for the one-dimensional local spreading of robots. Since the conditions for its the correct behavior, visibility of neighbors and reaching destination are met, the lemma follows. □

By Lemmas 5.2 and 5.3, and by the definition of approximate barrier coverage, the claimed result immediately follows:

Theorem 5.1. *Let $s_1, s_2, \ldots, s_n$ be n sensors with sensing range r initially placed at arbitrary positions on a line segment. If the sensors have no orientation and visibility radius $2r + \rho$, there is an algorithm for ε-approximate barrier coverage in* SSYNC *with rigid movements of length at most r.*

6. Conclusions

The results of this paper provide a first insight into the nature of the complexity and computability of distributed barrier coverage problems. Not surprisingly, it poses many new research questions. Here are some of them.

We have shown that barrier coverage is unsolvable in SSYNC with $\bar{n}$ unoriented sensors, but solvable in ASYNC with oriented sensors. Oriented sensors have a globally consistent sense of “left-right” while unoriented sensors have no sense of “left-right”. Hence the first immediate question is whether something weaker than global consistency would suffice. More precisely, if each sensor has a local orientation (i.e. a private sense of “left-right”) but there is no global consistency, is barrier coverage possible, at least in SSYNC? Is it impossible, at least in ASYNC?

Even in the presence of local orientation, solutions that work for unoriented sensors are desirable because they can tolerate the class of faults called *dynamic compasses*: a sensor is provided with a private sense of “left-right”, but this might change at each cycle (e.g., [20]). The open problem is to determine conditions which would make coverage possible under such conditions, at least in SSYNC. In particular, observing that the impossibility is established for $\bar{n}$ unoriented sensors, a relevant open question is what happens if $n > \bar{n}$ sensors are available? Would barrier coverage become possible in SSYNC?

For SSYNC we have shown that ε-approximate coverage is possible if $v > 2r$: is it possible to achieve the same result with $v = 2r$? In the case of unoriented sensors, no positive result exists in ASYNC. Is a higher visibility range sufficient for ε-approximate coverage in ASYNC?

Another interesting problem is whether the barrier coverage can be solved with unoriented sensors if the sensors are allowed to make random choices.

Declaration of competing interest

This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue.

The authors of this paper have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript

Acknowledgement

This research has been supported in part by the Natural Sciences and Engineering Research Council of Canada (N.S.E.R.C.) under the Discovery Grant program, and by Dr. Flocchini’s University Research Chair.

References

[1] P. Balister, B. Bollobas, A. Sarkar, S. Kumar, Reliable density estimates for coverage and connectivity in thin strips of finite length, in: Proceedings of 13th Annual International Conference on Mobile Computing and Networking (MobiCom), 2007, pp. 75–86.

[2] B. Bhattacharya, M. Burmester, Y. Hu, E. Kranakis, Q. Shi, A. Wiese, Optimal movement of mobile sensors for barrier coverage of a planar region, Theor. Comput. Sci. 410 (52) (2009) 5515–5528.

[3] D.Z. Chen, Y. Gu, J. Li, H. Wang, Algorithms on minimizing the maximum sensor movement for barrier coverage of a linear domain, Discrete Comput. Geom. 50 (2) (2013) 374–408.

[4] R. Cohen, D. Peleg, Local spreading algorithms for autonomous robot systems, Theor. Comput. Sci. 399 (2008) 71–82.

[5] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, M. Yazdani, On minimizing the maximum sensor movement for barrier coverage of a line segment, in: Proceedings of 8th International Conference on Ad-Hoc, Mobile and Wireless Networks (ADHOC-NOW), 2009, pp. 194–212.

[6] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, M. Yazdani, On minimizing the sum of sensor movements for barrier coverage of a line segment, in: Proceedings of 9th International Conference on Ad-Hoc, Mobile and Wireless Networks (ADHOC-NOW), 2010, pp. 29–42.
[7] M. Eftekhari, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, S. Shende, Distributed algorithms for barrier coverage using relocatable sensors, Distrib. Comput. 29 (5) (2016) 361–376.
[8] P. Flocchini, G. Prencipe, N. Santoro, Self-deployment of mobile sensors on a ring, Theor. Comput. Sci. 402 (1) (2008) 67–80.
[9] P. Flocchini, G. Prencipe, N. Santoro, Computing by Mobile Robotic Sensors, 2011, Chapter 21 of [15].
[10] P. Flocchini, G. Prencipe, N. Santoro, Distributed Computing by Oblivious Mobile Robots, Morgan & Claypool, 2012.
[11] S. Kumar, T.H. Lai, A. Arora, Barrier coverage with wireless sensors, Wirel. Netw. 13 (2007) 817–834.
[12] L. Li, B. Zhang, X. Shen, J. Zheng, Z. Yao, A study on the weak barrier coverage problem in wireless sensor networks, Comput. Netw. 55 (2011) 711–721.
[13] B. Liu, O. Dousse, J. Wang, A. Saipulla, Strong barrier coverage of wireless sensor networks, in: Proceedings of 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2008, pp. 411–419.
[14] M. Mehrandish, L. Narayanan, J. Opatrny, Minimizing the number of sensors moved on line barriers, in: Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), 2011, pp. 1464–1469.
[15] S. Nikoletseas, J. Rolim (Eds.), Theoretical Aspects of Distributed Computing in Sensor Networks, Springer, 2011.
[16] A. Saipulla, C. Westphal, B. Liu, J. Wang, Barrier coverage of line-based deployed wireless sensor networks, Ad Hoc Netw. 11 (4) (2013) 1381–1391.
[17] C. Shen, W. Cheng, X. Liao, S. Peng, Barrier coverage with mobile sensors, in: Proceedings of IEEE International Symposium on Parallel Architectures, Algorithms, and Networks (ISPAN), 2008, pp. 99–104.
[18] B. Wang, Coverage Control in Sensor Networks, Springer, 2010.
[19] F. Wu, Y. Gui, Z. Wang, X. Gao, G. Chen, A survey on barrier coverage with sensors, Front. Comput. Sci. 10 (6) (2016) 968–984.
[20] K. Yamamoto, T. Izumi, Y. Katayama, N. Inuzuka, K. Wada, The optimal tolerance of uniform observation error for mobile robot convergence, Theor. Comput. Sci. 444 (2012) 77–86.