

Distributed Freeze Tag: a Sustainable Solution to Discover and Wake-up a Robot Swarm

Cyril Gavaille, Nicolas Hanusse, Gabriel Le Boudier, Taïssir Marcé

Abstract

The Freeze Tag Problem consists in waking up a swarm of robots starting with one initially awake robot. Whereas there is a wide literature of the centralized setting, where the location of the robots is known in advance, we focus in the distributed version where the location of the robots \mathcal{P} are unknown, and where awake robots only detect other robots up to distance 1. Assuming that moving at distance δ takes a time δ , we show that waking up of the whole swarm takes $O(\rho + \ell^2 \log(\rho/\ell))$, where ρ stands for the largest distance from the initial robot to any point of \mathcal{P} , and the ℓ is the connectivity threshold of \mathcal{P} . Moreover, the result is complemented by a matching lower bound in both parameters ρ and ℓ . We also provide other distributed algorithms, complemented with lower bounds, whenever each robot has a bounded amount of energy.

1 Introduction

In order to save energy in distributed systems, the paradigm of sleeping models and algorithms has received a recent attention. Nodes or robots are, by default, inactive or on standby: the energy consumption is negligible and these periods can be used to harvest energy. A robot becomes active only if it is required.

The Freeze Tag Problem (FTP) consists in waking up a swarm of n inactive (or sleeping) robots as fast as possible assuming that one robot is initially active. To be woke up, a sleeping robot has to be reached by an awake robot, that are able to move in the plane. Once a robot becomes active, it can help wake up other robots.

FTP has been introduced in a centralized setting, where the n locations of the sleeping robots are known by the initial awake robot s . In this article, we propose a distributed version of the FTP: (1) the locations of the sleeping robots are not known in advance; (2) using a local snapshot, active robots only have a distance-1 visibility; and (3) robots need to be co-located to communicate. Note that due to the visibility constraint, it may be required to explore further than radius 1 to locate sleeping robots.

In order to get the most sustainable solution in a long-life perspective, we aim at minimizing the energy consumption. In particular, since moving is identified as an energy intensive task, the goal is to minimize *the makespan*, that is the time to wake up every robot, assuming unitary speed of the robots, i.e., moving at distance δ takes δ unit of time. It is also assumed that robots use discrete snapshots only, since continuous snapshots may be not energy friendly. We observe that since in our distributed model, robots do not know in general the locations of other robots, robot s has to move at least $\Omega(D^2)$ to discover and reach the closest other robot if it is located at distance D from s . Obviously, it can reach it with time $O(D^2)$ by following the trajectory of a spiral starting from s for instance.

1.1 State of the art

Any solution to the FTP can be seen as a rooted tree spanning the a set of $n + 1$ robot's positions, called the *wake-up tree*. The root node corresponds to the position of s , has one child, and the n nodes are other robots have at most two children each. Each edge has a *length*, representing the distance in the metric space between its endpoints. The makespan of the solution is nothing else than the (weighted) depth of the wake-up tree, and in particular a solution with optimal makespan has a wake-up tree with minimum depth.

Freeze Tag. Even for simple case, an optimal solution of the FTP can not be computed in polynomial time. Arkin *et al.* [ABF⁺06] showed that, even on star metrics, FTP is NP-hard. Moreover, they proved that getting an $5/3$ -approximation is NP-hard for general metrics on weighted graphs. However, in [ABG⁺03], the authors give a polynomial time algorithm to get an $O(1)$ for general graphs, assuming one sleeping robot per node.

In this paper, we focus on the *geometric* setting, where robot movements have no restrictions and where the position set \mathcal{P} lie on the Euclidean plane. Even in this setting, the problem remains NP-hard [AAJ17]. It has been shown by Yazdi *et al.* [YBMK15] that a wake-up tree of makespan of at most 10.07ρ can be (sequentially) computed in time $O(n)$, where ρ is the largest distance from s to any point of \mathcal{P} . The constant 10.1, aka the *wake-up constant* of the Euclidean plane, has been later on improved by Bonichon *et al.* [BCGH24] to 7.07. More generally, they proved that the wake-up constant for *any* norm is no more than 9.48, and that a corresponding wake-up tree can be computed in time $O(n)$. Very recently, the upper bound dropped to independently to 5.06 by [ABMS25] and to 4.63 [BGHO24]. It is known that $1 + 2\sqrt{2} \approx 3.83$ is a lower bound on the wake-up constant of the plane [BCGH24].

A first step toward the computation of a wake-up tree without a global knowledge of the robot's positions is the *on-line setting* [HNP06, BW20]. In

this case, each robot only appears at a specified time that is not known in advance. In [BW20], the authors propose a solution with a competitive ratio of $1 + \sqrt{2}$ w.r.t. to the optimal partial wake-up tree.

Collaborative Exploration. Obviously, any collaborative exploration problem requires to have a team of active robots. Conversely, the *distributed FTP* (dFTP for short), requires to explore some area to discover sleeping robots and thus is naturally connected to exploration of the plane with one or more robots. The survey of Das [Das19] contains many references such exploring problems in unknown graphs. In dFTP, we start with one active robot, and after few steps, we can have k active robots. So, the task of discovering new robots can indeed be seen as a collaborative exploration task. The question of improving exploration with the use of $k > 1$ robots is challenging and widely open. For instance, it has been shown by Fraigniaud *et al.* [FGKP06] that unweighted trees of diameter D , distributed exploration can be done in $O(D + n/\log k)$ unitary moves, even if robots are allowed to let some information at the nodes, whereas we could hope a speed-up of k with $O(D + n/k)$ unitary moves. However, if the underlying graph is a two dimensional sub-grid of n vertices, a grid with rectangular holes, Ortlof and Schindelhauer [OS12] show how to get an optimal speedup of factor k .

Discovering a robot at distance D with k co-located robots in the plane can be done within $\Theta(D + D^2/k)$ unitary moves per robot using either parallel spiral trajectories [FHG⁺16], or by partitionning into a square of width D into k rectangles of width D/k and height D . This problem, aka Treasure Hunt Problem or Cow-Path Problem, has been widely studied for $k = 1$ or with imprecise geometry [BDPP20]. Interestingly, the authors of [FKLS12] have showed that the knowledge of an approximation of k is required to get a time the bound $\Theta(D + D^2/k)$. The question of the knowledge of k arises whenever the k robots do not start the exploration together (as in the dFTP), or whenever the communication ability of the robots is limited.

Energy Consumption. Some recent works deal with the problem of distributed tasks with some energy constraints or minimizing the energy consumption. In the sleeping model [CGP20], nodes or robots are either sleeping or are active. If a robot is sleeping, its consumption is assumed to be negligible. The state of each robot in synchronized rounds is given by a centralized schedule. Distributed tasks have been considered in the sleeping model like coloring or MIS computations.

Energy Constrained Exploration Problems are perhaps more related to our problem. For instance, in the *Piece-Meal Graph Exploration*, robots have a given budget for the energy and needs to refuel at a home base before exploring unknown parts of the graph. Note that a solution of the treasure

hunt in a grid graph can be used for the plane. In [DKK06], the authors show that n -node and m -edge unweighted graphs of radius R can be explored by $k = 1$ agent with an energy budget $B = (1 + \alpha) \cdot R$ in $O(m + n/\alpha)$ unitary moves. For $k > 1$, [DKS06, DDU24] deal with the Energy Constrained Depth First Search while minimizing the number k of robots with an energy budget $O(R)$ per robot to explore a tree of radius R . Other distributed algorithms for energy constrained agents has been considered in [BCD⁺20]. They show how to provide a feasible movement schedule for mobile agents for the Delivery Problem, where each agent has limited energy which constrains the distance it can move. Hence multiple agents need to collaborate to move and deliver the package, each agent handing over the package to the next agent to carry it forward. However, the positions of the agents are assumed to be known and the computation is centralized.

However, all these results related to collaborative exploration and energy consumption are not directly related to our setting, and essentially because we are face to the fact that, by definition of the problem, the number of active robots collaborating keeps on evolving, from 1 to $n + 1$.

1.2 The Model

Computational Model. We consider a swarm of robots in the Euclidian plane. Robots are all initially asleep, except one which we call the *source* and denote s , initially located at position $p_0 = (0, 0)$. The set of all robots is denoted by $\mathcal{R} = \{s, r_1, \dots, r_n\}$, robots r_i being the initially asleep robots. We denote by p_i the initial position of robot r_i , and by \mathcal{P} the set of all initial positions of initially asleep robots, i.e., $\mathcal{P} = \{p_1, \dots, p_n\}$. Robots are endowed with a visible light indicating their status (sleeping or awake), which can be observed by any active robot close enough (in its distance-1 vicinity). Sleeping robots are computationally inactive. They can neither move, observe, nor do any type of computation. Awake robots are aware of the absolute coordinate system, a same global clock and are able to locate and distinguish sleeping and awake robots in their vicinity, by using a function *look*. They can also share variables of their memory with co-localated robots, and can operate computations based on the information they gathered previously. Finally, they can move in the plane, based on the computation they operated. Robots move at speed 1, which means it takes a time δ for a robot to move between any two points at Euclidean distance δ . Thus the behaviour of a robot can be described in the standard Look-Compute-Move Model (see [FPS19]). For synchronization purpose, robots can also wait for any duration of time at a fixed position. When an awake robot and a sleeping robot are co-located, the awake one can wake the other one up, and possibly share with it some information as previously said.

An algorithm \mathcal{A} aiming to solve the dFTP is executed in parallel by all

the awake robots. The execution of \mathcal{A} terminates when all active robots have terminated their computation and moves. The execution is valid if, when it terminates, all the initially asleep robots have been awakened. The makespan of an execution is the duration between the beginning of the algorithm and its termination, which basically counts the duration of the moving and waiting actions of robots.

Robots have a local unlimited memory. Typically, they can store the positions of some robots and their status (sleeping/awake) at the time they see them in their vicinity. Note that robots can give themselves a globally unique identifier as soon as they are awakened, by storing their initial position. We will also consider the variant of the model where the robots are not free to move as long as they want, but are rather limited by some *energy budget* B . In this variant, a robot can move for a total distance at most B .

Spread of \mathcal{P} . Our results are highly linked to the distribution of \mathcal{P} . Typically, if the distance between every pair of robots is much larger than 1, a robot may have the inaccurate belief that it is alone¹ which complicates a lot the resolution of the problem. To formally present our results as in Table 1, let us introduce some parameters related to $\mathcal{P} \subset \mathbb{R}^2$.

Given a real $\delta \geq 0$, and $\mathcal{X} \subset \mathbb{R}^2$, the δ -*disk graph* of \mathcal{X} is the edge-weighted geometric graph whose vertex set is \mathcal{X} , two points $u, v \in \mathcal{X}$ being connected by an edge if and only if u and v are at (Euclidean) distance at most δ , and the weight of the edge corresponds to the distance between their endpoints.

Let (\mathcal{P}, s) be an n -point set $\mathcal{P} \subset \mathbb{R}^2$ with a source $s \notin \mathcal{P}$. The *radius* of (\mathcal{P}, s) , denoted by ρ^* , is the largest distance from s to any point of \mathcal{P} . The *connectivity-threshold* of (\mathcal{P}, s) , denoted by ℓ^* , is the least radius δ such that the δ -disk graph of $\mathcal{P} \cup \{s\}$ is connected. Given $\ell > 0$, the ℓ -*eccentricity* of (\mathcal{P}, s) , denoted by ξ_ℓ , is the – finite or infinite – minimum weighted-depth of a spanning tree of the ℓ -disk graph of $\mathcal{P} \cup \{s\}$ rooted at s .

Problem Definition. Note that if $\rho^* \leq 1$, every robot is seen by the source s and can be waken up in time $O(1)$ with energy budget $O(1)$ by solving the centralized version in s , e.g., as done in [BCGH24]. We shall suppose that s starts with some information about the connectivity-threshold, the radius, and the number of asleep robots in \mathcal{P} that it is supposed to wake up. More precisely, a tuple of values (ℓ, ρ, n) is given to s at its start. Indeed, without any information, it is not difficult to see that s cannot terminate (and thus cannot solve the dFTP), being unable to distinguished (for instance) the case where $n = 0$ (s is alone) from $n > 0$, without moving for eternity. An algorithm with input (ℓ, ρ, n) and solving dFTP should terminate on any n -

¹The co-located robot that activated it excepted.

point set (\mathcal{P}, s) such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$. Also note that we always have $\ell^* \leq \rho^* \leq n\ell^*$, as proven in Proposition 1. And, so a tuple (ℓ, ρ, n) is said *admissible* if $\ell \leq \rho \leq n\ell$. Also, for the sake of simplicity, we suppose that parameters ℓ and ρ are positive integers. This hypothesis does not actually change the problem (in term of asymptotic complexity of the makespan), since (ℓ, ρ, n) is admissible if and only if $(\lceil \ell \rceil, \lceil \rho \rceil, n)$ is.

The dFTP is formally defined as follows:

Definition 1 (dFTP). *A distributed algorithm \mathcal{A} solves the dFTP if, for any admissible tuple (ℓ, ρ, n) , and for any n -point set \mathcal{P} with source s such that $\rho^* \leq \rho$ and $\ell^* \leq \ell$, the execution of \mathcal{A} on \mathcal{P} at source s , given (ℓ, ρ, n) , eventually wakes up all the robots and terminates. Moreover, it solves the dFTP with energy budget B if the previous holds, assuming that the total movement lengths of each robot does not exceed B .*

1.3 Contributions

Our contributions are summarized in Table 1. We present three algorithms, called $\mathcal{A}_{\text{Separator}}$, $\mathcal{A}_{\text{Grid}}$ and $\mathcal{A}_{\text{Wave}}$.

The first algorithm $\mathcal{A}_{\text{Separator}}$ solves dFTP, with no limits on the energy budget, and has makespan $O(\rho + \ell^2 \log(\rho/\ell))$. This result is complemented by a matching lower bound.

The two other algorithms consider the dFTP with energy budget B . We first show that no algorithm can solve the limited energy budget variant if $B < c\ell^2$, for some constant $c > 0$. Then, for $B \in \Theta(\ell^2)$, i.e., as little energy as possible to solve the dFTP, $\mathcal{A}_{\text{Grid}}$ achieves a makespan of $O(\xi_\ell \cdot \ell)$. Using slightly more energy, namely $B \in \Theta(\ell^2 \log \ell)$, $\mathcal{A}_{\text{Wave}}$ has a significantly lower makespan, which matches a second lower bound we introduce.

Energy	Algorithm	Makespan	Lower Bound
unconstrained	$\mathcal{A}_{\text{Separator}}$	$O(\rho + \ell^2 \log(\rho/\ell))$ - Th. 1	$\Omega(\rho + \ell^2 \log(\rho/\ell))$ - Th. 2
$< \pi(\ell^2 - 1)/2$	-	-	unfeasible - Th. 3
$\Theta(\ell^2)$	$\mathcal{A}_{\text{Grid}}$	$O(\xi_\ell \cdot \ell)$ - Th. 4	$\Omega(\xi_\ell + \ell^2 \log(\xi_\ell/\ell))$ - Th. 6
$\Theta(\ell^2 \log \ell)$	$\mathcal{A}_{\text{Wave}}$	$O(\xi_\ell + \ell^2 \log(\xi_\ell/\ell))$ - Th. 5	

Table 1: Complexity of the makespan for the dFTP given (ℓ, ρ, n) .

Roadmap. In Section 2 we present the main building blocks of our algorithm. These are high-level procedures we use to describe $\mathcal{A}_{\text{Separator}}$ in Section 3. In Section 4 $\mathcal{A}_{\text{Separator}}$ will also be used as a building block for algorithms with constrained energy $\mathcal{A}_{\text{Grid}}$ and $\mathcal{A}_{\text{Wave}}$. Due to space lim-

itations, the majority of the descriptions of our algorithms and proofs is provided in a separate appendix.

2 Building Blocks

The main parts of our algorithms are based on exploring regions, computing and realizing wake-up trees, and organizing teams of robots to explore regions in parallel. The recruitment of a team is based on a sampling of point sets. To avoid exploring large empty regions, we use geometric separators.

2.1 Exploration (EXPLORE)

One central task robots are led to realize is the exploration of a given region, in order to collect the positions of all the sleeping robots in that region. For the sake of simplicity, we only consider rectangular regions, whose orientation is parallel to the axis. Note that it can be used to explore any shape inscribed in a rectangle. We present in Section 6.1 a simple method for exploring a given rectangle with a single robot, using function `look`, which can be adapted to a team of robots. In this extension, every robot explores a sub-rectangle before moving to a meeting point where they can share their variables.

Lemma 1 (EXPLORE). *There exists a procedure EXPLORE such that, for any rectangle \mathcal{S} of dimensions $w \times h$, for any two positions $p, p' \in \mathcal{S}$, the execution of EXPLORE(\mathcal{S}, p') at time t , by a team of k robots $\mathcal{T} = \{r_1, \dots, r_k\}$ initially co-located at position p guarantees:*

- *it terminates at time t' with $(t' - t) \in O(wh/k + w + h)$; and*
- *at time t' , robots of \mathcal{T} have gathered the initial positions of all robots of \mathcal{S} that are asleep at t' .*

2.2 Realization of a Central Wake-up-Tree

In [YBMK15, BCGH24] the authors show that, knowing the initial positions of robots, it is possible to compute a wake-up tree in linear time, whose makespan is an approximation of the optimal. Yet, in the distributed setting, some specific problems may arise. Indeed, two awake robots r_i and r_j may compute independently two wake-up trees on different but not disjoint subsets X_i and X_j of \mathcal{P} . If p_k , the position of r_k , belongs to $X_i \cap X_j$, then r_i and r_j are said *in conflict*. Both need to use r_k for their wake-up trees and only the first robot to reach r_k is able to do it (since r_k will then move). The second one only find out that r_k has left its initial position p_k when itself or one of its descendent arrives close to p_k . In this situation the consistency of

the wake-up tree is broken, and a new wake-up tree has to be computed for the sub-tree rooted at r_k . Since this situation can be repeated an arbitrarily large number of times for a set $Y = X_i \cap X_j$, there is no evidence that r_j can wake up $X_j \setminus Y$ in a time proportional to the diameter of $X_j \setminus Y$. To deal with that issue, we ensure in our algorithms that wake-up trees are computed in separate regions of the plane, and that at most one robot computes a wake-up tree in a given region. This is formalized in Lemma 2, detailed in Section 6.2.

Lemma 2 (Distributed Makespan). *Given a square region \mathcal{S} of width R and a robot r positioned in the center of \mathcal{S} , knowing both \mathcal{S} and a set of sleeping robots \mathcal{R}_S whose initial positions are in \mathcal{S} , if no robots other than $\{r\} \cup \mathcal{R}_S$ takes action in \mathcal{S} , then r can wake-up all robots of \mathcal{R}_S in time $5R$.*

More generally, if robots do not know the positions of sleeping robots in \mathcal{S} , it is possible for a robot to discover them before computing a wake-up tree. This process will be used in \mathcal{A}_{Grid} , and is presented in the following Corollary, detailed in Section 6.3.

Corollary 1 (Explore and Wake up). *Given a square region \mathcal{S} of width R containing at least one awake robot, if no awake robot goes through the border \mathcal{S} , it is possible to wake-up all sleeping robots of \mathcal{S} in time $R^2 + (10 + \sqrt{2})R$.*

2.3 Geometric separators

One central tool we use is *geometric separators*. Given a square \mathcal{S} of width R centered at position p , the interior of \mathcal{S} , including \mathcal{S} , is noted \mathcal{S}^{in} and the remaining region of the plane is noted \mathcal{S}^{out} . Given a square \mathcal{S} with width $R > 2\ell$, we define the *separator of \mathcal{S}* , denoted $\text{sep}(\mathcal{S})$, as the region bounded by \mathcal{S} and a square of center p and width $R - 2\ell$. We immediately get:

Lemma 3. *Let \mathcal{S} be a square. Let ℓ be the connectivity threshold of (\mathcal{P}, s) . Any path in the ℓ -disk graph of \mathcal{P} linking robots $r \in \mathcal{S}^{in}$ and $r' \in \mathcal{S}^{out}$ contains at least one robot located in $\text{sep}(\mathcal{S})$.*

Corollary 2. *If $\mathcal{P} \cap \text{sep}(\mathcal{S}) = \emptyset$, either $\mathcal{P} \subset \mathcal{S}^{in}$ or $\mathcal{P} \subset \mathcal{S}^{out}$.*

2.4 Distributed ℓ -sampling

To begin, we point out that the time to wake up every robot in a square region of width R , $O(R)$ (Lemma 2), is often dominated by the time to discover sleeping robots in this region by a team of k robots, which is $O(R^2/k)$ (Lemma 1). However, knowing an upper bound ℓ on the connectivity threshold can help thanks to *ℓ -samplings*. More formally, *an ℓ -sampling of a region \mathcal{S} is a subset of positions $\mathcal{P}' \subseteq \mathcal{P} \cap \mathcal{S}$ that are pairwise at distance at least ℓ .*

We also say that \mathcal{S} is *covered* by \mathcal{P}' , if any robot within \mathcal{S} is at distance at most ℓ from a robot whose position is in \mathcal{P}' . Assuming that the ℓ -sampling is given and \mathcal{P} is covered by \mathcal{P}' , discovering the n robots can be done in parallel in time $O(\ell^2)$, gathering \mathcal{P}' at the source s in time R and waking-up every robot in time $O(R)$ using a centralized wake-up algorithm (Lemma 2). In total, it takes $O(R + \ell^2)$. The two main difficulties are how to efficiently compute in a distributive way an ℓ -sampling, and how large is an ℓ -sampling.

Lemma 4 (*ℓ -sampling cardinality*). *If \mathcal{P}' is an ℓ -sampling of a square region of width R , then $|\mathcal{P}'| \leq 16R^2/(\pi\ell^2)$.*

DFSAMPLING is a distributed algorithm computing an ℓ -sampling of a squared region \mathcal{S} . A single robot or a co-located team of robots start from a same position of $\mathcal{P} \cap \mathcal{S}$ and aims at finding an ℓ -sampling of size 4ℓ (See Figure 1b). This sampling represents positions of robots to be later recruited to become a new team. This sampling is discovered using a specific exploration algorithm. Since it may happen that exploring from one single position is not enough to reach a sample of size 4ℓ , the team may use several starting positions for the exploration task, which we call the *seeds* $\mathcal{X} \subset \mathcal{S}$. In Figure 1c, \mathcal{X} is the set of points within an ℓ -separator of a sub-square.

Roughly speaking, DFSAMPLING is based on a Depth-First Search in the 2ℓ -disk graph of $\mathcal{P} \cap \mathcal{S}$, starting by the position of seeds from \mathcal{X} . Whenever a point p is discovered, it is added to \mathcal{P}' only if it is at distance greater than ℓ from any other points already added to \mathcal{P}' . This is required to guarantee that \mathcal{P}' is indeed a ℓ -sampling. A detailed description is in Section 6.5 as well as the proof of Lemma 5.

Lemma 5 (DFSAMPLING). *In a region \mathcal{S} of width R containing a set of seeds $\mathcal{X} \subseteq \mathcal{P}$, a team \mathcal{T} of robots knowing the awake robots of \mathcal{S} can compute a set of positions \mathcal{P}' being an ℓ -sampling of \mathcal{P} . The computation is done using DFSAMPLING in time:*

$$\begin{aligned} &O(\ell^2 \log(|\mathcal{P}'|)) \text{ if } |\mathcal{T}| = 1 \text{ and } \mathcal{X} = \{p_s\} \text{ and } R \geq 2\rho^* \\ &O(R + \ell|\mathcal{P}'|) \text{ if } |\mathcal{T}| = \ell \text{ and } \mathcal{X} = \mathcal{P} \cap \text{sep}(\mathcal{S}) \end{aligned}$$

In both cases, either (1) $|\mathcal{P}'| = 4\ell$; or (2) $|\mathcal{P}'| < 4\ell$ and \mathcal{S} is covered by \mathcal{P}' .

3 Without energy constraint

Algorithm $\mathcal{A}_{\text{Separator}}$ is based on a divide-and-conquer strategy where robots are organized in teams. As shown by a matching lower bound, the algorithm has optimal makespan. We describe it with six phases: Initialization, Partition, Exploration, Recruitment, Reorganization and Termination.

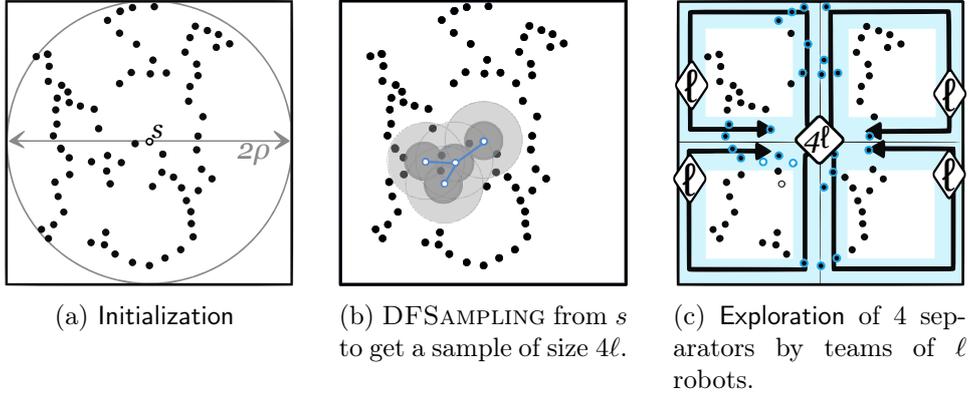


Figure 1: First phases of $\mathcal{A}_{\text{Separator}}$; \bullet sleeping robots; \circ awake robots; \bullet sleeping seeds; \circ awake seeds.

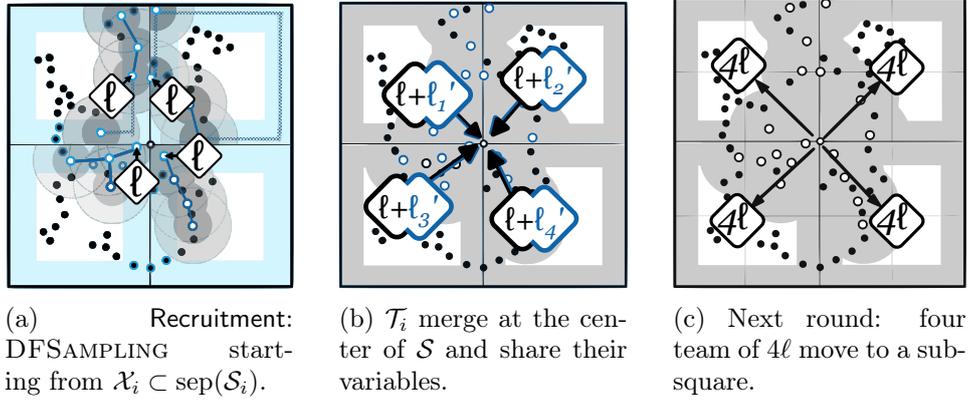


Figure 2: Recruitment and Reorganization phases of $\mathcal{A}_{\text{Separator}}$; \circ recruited robots; explored area is grayed.

Having first awoken 4ℓ robots in the square of center s and width 2ρ (Initialization and Recruitment Phase, presented in Figure 1a and 1b). we divide the square into 4 sub-squares, and send a team of ℓ robots in every sub-square (Partition Phase). During the Exploration Phase, each team collectively explores the separator of their sub-square and stores seeds in \mathcal{X} , that are the initial positions of robots (sleeping or already awake) belonging to the separator (Figure 1c). During the Recruitment Phase, each team wakes up new robots in its sub-square, such that these new robots *plus robots currently exploring* whose initial position is in that sub-square, are 4ℓ (Figure 2a). These 4ℓ robots are *recruited* to define a new team used in the next round.

Finally, during phase Reorganization, all 4 teams meet in the center of the square so robots can team up with robots initially located to the same

sub-square. Each team thus formed go into its corresponding sub-square, and repeat the process until all robots have been awakened (Figures 2b, 2c). **Recruitment Phase** relies on function `DFSAMPLING` presented in Section 2.4. Lemma 5 guarantee that if `DFSAMPLING` returns a set with less than 4ℓ positions, then every robot located in the sampled region has been discovered (but not necessarily awakened). This justifies the **Termination** phase where awaken robots wake up the remaining sleeping robots with a centralized algorithm. Figure 3 presents a more detailed description of $\mathcal{A}_{\text{Separator}}$.

Theorem 1. $\mathcal{A}_{\text{Separator}}$ solves the dFTP, given every admissible tuple (ℓ, ρ, n) , with a makespan $O(\rho + \ell^2 \log(\rho/\ell))$.

In Section 7 we prove Theorem 1 in two steps: Lemma 9 guarantees that $\mathcal{A}_{\text{Separator}}$ solves the dFTP, and Lemma 8 its makespan. Informally, we have $O(\log(\rho/\ell))$ rounds (Lemma 11) and Round $k \geq 1$, takes $O(\ell^2 + \rho/2^k)$ time units (Lemma 10).

We also provide a lower bound on the makespan of any algorithm solving the dFTP. This is done by building an n -point set \mathcal{P} depending on the considered algorithm. Details of the proof are given in Section 9.1

Theorem 2 (Lower Bound without energy constraint). *For every admissible tuple (ℓ, ρ, n) and algorithm \mathcal{A} solving the d-FPT, there exists an n -point set \mathcal{P} and a source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$ such that the makespan of the execution of \mathcal{A} with source s and inputs (ℓ, ρ, n) on \mathcal{P} is $\Omega(\rho + \ell^2 \log(\rho/\ell))$.*

4 With energy constraint

The construction used in the proof of Theorem 2 can be adapted to obtain Theorem 3. Details are given in Section 9.2.

Theorem 3 (Lower Bound on the energy budget). *For every admissible tuple (ℓ, ρ, n) and algorithm \mathcal{A} with energy budget $B < \pi(\ell^2 - 1)/2$, there exists an n -point set \mathcal{P} and a source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$ such that the execution of \mathcal{A} on \mathcal{P} does not wake-up any robot.*

We use a Breadth First Search based strategy to define $\mathcal{A}_{\text{Grid}}$, an algorithm that takes as input the parameter $\ell \geq \ell^*$. This algorithm is then combined with $\mathcal{A}_{\text{Separator}}$ to get $\mathcal{A}_{\text{Wave}}$, which provides a smaller makespan.

To sum up, $\mathcal{A}_{\text{Grid}}$ works as follows: the plane is partitioned into squares of width 2ℓ centered at positions $\{(2k\ell, 2k'\ell) \mid (k, k') \in \mathbb{Z}^2\}$. We wake-up the square \mathcal{S} containing s in time $t(\mathcal{S}) = R^2 + (10 + \sqrt{2})R$ (Corollary 1). Then every square containing a new awake robot try to wake up the 8 adjacent squares of a square ordered in a counter-clockwise order. A precise description of $\mathcal{A}_{\text{Grid}}$ is provided in Section 8.1.

1. **Round 0: Initialization and Recruitment**
 $\mathcal{S} \leftarrow$ square of width $R = 2\rho$ and centered at the source robot s
 $\mathcal{T} \leftarrow \{s\}$
 $\mathcal{X} \leftarrow \{p_s\}$
 \mathcal{T} recruits up to $4\ell - 1$ new robots using $\text{DFSAMPLING}(\mathcal{S}, \mathcal{X})$
Move \mathcal{T} to the center of \mathcal{S}
2. **Round $k \geq 1$ for a team \mathcal{T} in square \mathcal{S} :**
 - (i) **Termination**
If $|\mathcal{T}| < 4\ell$: do a centralized awakening of sleeping robots in \mathcal{S} and terminate.
 - (ii) **Partition**
Partition \mathcal{S} (resp. \mathcal{T}) into 4 sub-squares $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$ (resp. 4 teams $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4$ of ℓ robots each).
Each team \mathcal{T}_i performs in parallel:
 - (iii) **Exploration**
Using 4 times routine EXPLORE , collectively explore $\text{sep}(\mathcal{S}_i)$.
Save in \mathcal{X}_i the set of positions of $\text{sep}(\mathcal{S}_i)$ where a robot was found asleep, and those where a robot has already been awakened.
 - (iv) **Recruitment**
 \mathcal{T}_i recruits ℓ'_i robots by using $\text{DFSAMPLING}(\mathcal{S}_i, \mathcal{X}_i)$
Move \mathcal{T}_i to the center of \mathcal{S}_i .
 - (v) **Reorganization**
Wait until the four teams \mathcal{T}_i can merge and share their variables.
Reorganize robots in teams \mathcal{T}'_i of size ℓ'_i by square of origin \mathcal{S}_i .
For each team \mathcal{T}'_i , do in parallel:
 - Move \mathcal{T}'_i to the center of \mathcal{S}_i
 - Go to Round $k + 1$ with team \mathcal{T}'_i and square \mathcal{S}_i .

Figure 3: Description of $\mathcal{A}_{\text{Separator}}$

Theorem 4. $\mathcal{A}_{\text{Grid}}$ solves the $dFTP$, given every admissible tuple (ℓ, ρ, n) , with an energy budget of $O(\ell^2)$ and with a makespan of $O(\ell \cdot \xi_\ell)$.

The next algorithm, $\mathcal{A}_{\text{Wave}}$, is an adaptation of $\mathcal{A}_{\text{Grid}}$. Roughly speaking,

there are two changes: (1) the squares are now of width $8\ell^2 \log_2 \ell$; and (2) instead of using a simple process to explore and to wake up a square, we use $\mathcal{A}_{\text{Separator}}$ starting from a team of 4ℓ robots for Rounds $k \geq 1$ and from the source s for Round 0. A precise description of $\mathcal{A}_{\text{Wave}}$ is given in Section 8.2.

Theorem 5. $\mathcal{A}_{\text{Wave}}$ solves the dFTP, given every admissible tuple (ℓ, ρ, n) , with an energy budget of $O(\ell^2 \log \ell)$ and a makespan of $O(\xi_\ell + \ell^2 \log \xi_\ell / \ell)$.

We can find a lower bound of $\Omega(\xi)$ for a given range of values for ξ (See construction in Section 9.3):

Theorem 6 (Lower Bound for energy constrained). *For every admissible tuple (ℓ, ρ, n) , for every $B > \ell$, and for every $\xi \in [\rho/3, \min\{n\ell - \rho/3, \lfloor \rho^2 / (2(B+1)) + 1 \rfloor\}]$, there exists an n -points set \mathcal{P} and a source s of connectivity threshold ℓ , radius ρ , and ℓ -eccentricity $\xi_\ell = \xi$ such that the makespan of any algorithm \mathcal{A} solving the dFTP for (\mathcal{P}, s) given (ℓ, ρ, n) and energy budget B , is $\Omega(\xi + \ell^2 \log(\xi/\ell))$.*

5 Discussion

Although the input of our algorithms is (ℓ, ρ, n) , it turns out that only the knowledge of ℓ as an upper bound of ℓ^* is required. The number of robots to awake n is never used.

Only $\mathcal{A}_{\text{Separator}}$ requires an upper bound ρ on ρ^* . We can easily get a constant approximation of ρ^* as follows: (1) we build a team of 4ℓ robots using DFSAMPLING in time $O(\ell^2 \log \ell)$ and (2) we run explorations of the ℓ -separators squares of increasing width $\ell \cdot 2^i$ for $i = 1, 2, \dots, k$ until we have an empty separator. Then we take $\rho = \ell \cdot 2^k$ and we can prove that it is a 3-approximation of ρ^* . The overcost is $O(\ell^2 \log \ell + \sum_i^k \ell 2^i) = O(\ell^2 \log \ell + \ell 2^{k+1}) = O(\ell^2 \log \ell + \rho)$. Either step (1) does not provide 4ℓ robots meaning that \mathcal{P} is covered and that ρ^* is deduced from the discovered robots or (2) we have $\log \ell = O(\log(\rho^*/\ell))$. To conclude, the total overcost of computing a constant approximation of ρ^* from ℓ is of same order than $\mathcal{A}_{\text{Separator}}$.

We presented two algorithms that guarantee an optimal makespan under the hypothesis of knowing an upper bound on ℓ^* . $\mathcal{A}_{\text{Separator}}$ provides an optimal makespan in $\Theta(\rho^* + \ell^2 \log(\rho^*/\ell))$ without energy constraints, and $\mathcal{A}_{\text{Wave}}$ provides an optimal makespan of $\Theta(\xi_\ell + \ell^2 \log \xi_\ell / \ell)$ with an energy budget in $\Theta(\ell^2 \log \ell)$. Finally, $\mathcal{A}_{\text{Grid}}$ provides a less efficient makespan (apart under some specific relations between ℓ, ρ and ξ_ℓ), but requires an optimal energy budget of $\Theta(\ell^2)$.

Some questions remain open: can we get an optimal algorithm with only $\Theta(\ell^2)$ energy?

Moreover, our lower bound with energy constraints holds for a specific range of ξ , informally between ρ and $c \cdot (\rho/\ell)^2$ for some constant c , whereas

we know that ξ can approach $c \cdot \rho^2/\ell$: can we obtain a lower bound for a broader range of value of ξ ?

In our algorithms, robots has to communicate using rendez-vous. Is it possible to achieve efficient solution without face-to-face communication?

6 Building blocks - details

Main notations. Given $r \geq 0$ and $p \in \mathbb{R}^2$, we denote by $B_p(r)$ the disk of center p and radius r . Given a real $\delta \geq 0$, and $\mathcal{X} \subset \mathbb{R}^2$, the δ -disk graph of \mathcal{X} is the edge-weighted geometric graph whose vertex set is \mathcal{X} , two points $u, v \in \mathcal{X}$ being connected by an edge if and only if u and v are at (Euclidean) distance at most δ , and the weight of the edge corresponds to the distance between their endpoints.

We first emphasize a relationship between the different parameters of the point sets:

Proposition 1. *For every point set \mathcal{P} and source s , for any $\ell \geq \ell^*$:*

$$0 < \ell^* \leq \rho^* \leq \xi_\ell \leq n \cdot \ell^* .$$

Thus the input for our algorithm is a tuple (ℓ, ρ, n) such that $\ell \leq \rho \leq n\ell$.

Proof. The first three inequalities are straightforward using definition. Note $\ell = \ell^*$ and $\rho = \rho^*$. The last one is because: (1) each edge of the ℓ -disk graph of $\mathcal{P} \cup \{s\}$ has length at most ℓ ; and (2) a path from s to any point of \mathcal{P} in every tree spanning $n = |\mathcal{P}| + 1$ points contains at most $|\mathcal{P}|$ edges. \square

6.1 Explore

We now detail how a team of k robots can explore efficiently a square.

Lemma 1 (EXPLORE). *There exists a procedure EXPLORE such that, for any rectangle \mathcal{S} of dimensions $w \times h$, for any two positions $p, p' \in \mathcal{S}$, the execution of EXPLORE(\mathcal{S}, p') at time t , by a team of k robots $\mathcal{T} = \{r_1, \dots, r_k\}$ initially co-located at position p guarantees:*

- *it terminates at time t' with $(t' - t) \in O(wh/k + w + h)$; and*
- *at time t' , robots of \mathcal{T} have gathered the initial positions of all robots of \mathcal{S} that are asleep at t' .*

Proof. Let us start by describing a *Single exploration*, that is the path traveled by a single robot r to explore a rectangular region of width w and height h . Starting from a corner, keep on zigzagging rows by rows separated by $\sqrt{2}$

and every move of length $\sqrt{2}$, do a snapshot to discover up to radius 1, as shown in Figure 4a. Note that the algorithm may require an initial move and a final move, to go from p to the origin of the path, and to go from the endpoint of the path to p' , both moves being bounded by $w + h$. The sum of the length of vertical paths is at most h' , and the length of each horizontal path is at most w . Furthermore, the number of horizontal path is $\lceil h/\sqrt{2} \rceil$, which means that the sum of the length of the horizontal paths is at most $w \lceil h/\sqrt{2} \rceil \in O(w \times h + w)$. By summing all three parts, we end up with a complexity of `explore_single` in $O(w \times h + w + h)$. The validity of the procedure `explore_single` comes with the fact that any disk of radius 1 contains the square with identical center and width $\sqrt{2}$.

For a collaborative exploration with k robots, the procedure separates the targeted rectangle into k rectangles, each one being explored by a single robots as shown in Figure 4b. Sub-rectangles are of equal dimensions $w' = w$ and $h' = \frac{h}{k}$. The analysis conducted for the single exploration leads to a complexity of $O(\frac{w \times h}{k} + w + h)$ for this procedure. Since all robots can compute an upper bound on the time complexity for all robots, they can all reach the endpoint p' at the same moment t' and share the information they gathered in their own rectangles. \square

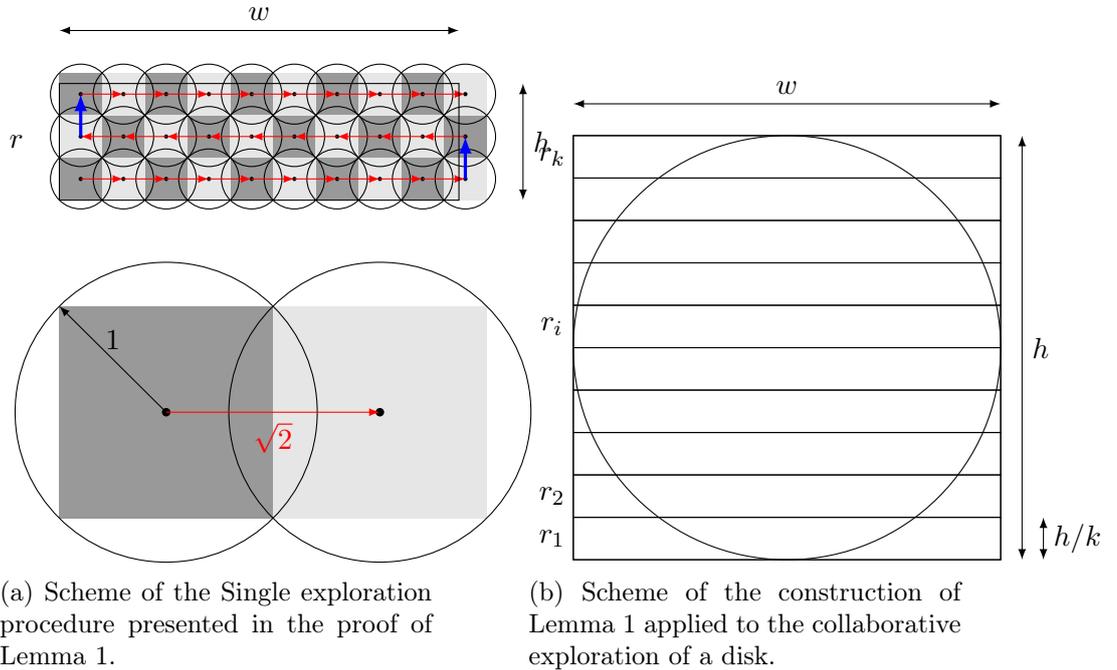


Figure 4: Depiction of the exploration process.

6.2 Realization of a wake-up tree.

In this section we explain how to apply some results about Centralized Freeze Tag Problem to awake robots in a distributed setting.

Once an awake robot r_0 has learned the positions of sleeping robots in region of the plane of radius R , it can compute an arbitrary wake-up tree \mathbb{W}_0 of weighted depth $O(R)$ [BCGH24]. This step is called the *centralized awakening* of the region by r_0 . It remains to actually wake up robots. r_0 has to awake its first child in the wake-up tree and transmit the rest of the tree so the newly awake robot can help. More generally, each new awake robot participate to the propagation of the wake-up tree. The procedure starts by having r_0 moving to its child r_1 in \mathbb{W}_r (Algorithm 1, line 6). When r_0 and r_1 are co-located, r_0 wakes up r_1 (line 7) and shares the wake-up tree \mathbb{W} (line 8). When a robot r_i wakes up a robot r_j , both use a simple procedure to update in parallel what remains of the wake-up tree:

- If the subtree of \mathbb{W}_i rooted in r_j is empty, ie r_j is a leaf, none of r_i or r_j has anything left to do (line 1 and 9).
- If r_j has a unique child in \mathbb{W}_i , r_j updates its local wake-up tree $\mathbb{W}_j \leftarrow \mathbb{W}_i \setminus \{r_i\}$ so \mathbb{W}_j is rooted in r_j (line 2) and r_i is done (line 10). r_j wakes up its unique child in \mathbb{W}_j and recursively propagate \mathbb{W}_j .
- If r_j has two children, \mathbb{W}_i is separated into two sub-trees: r_i keeps the right-hand sub-tree (line 11) while r_j keeps the left-hand sub-tree (line 3). Both robots update their memory accordingly and moves to the unique child of their respective wake-up trees to continue the propagation.

Algorithm 1: Propagate Wake-Up Tree - Code for a robot r

```

input:  $\mathbb{W}$ 
1 if  $|\mathbb{W}| = 0$  then stop;
2 else if  $|\mathbb{W}| = 1$  then  $\mathbb{W} \leftarrow \text{child}(\mathbb{W})$ ;
3 else if  $|\mathbb{W}| = 2$  then  $\mathbb{W} \leftarrow \text{child}_1(\mathbb{W})$ ;
4 while  $\mathbb{W}$  do
5    $\text{dest} \leftarrow \text{root}(\mathbb{W})$ ;
6    $\text{move}(\text{dest})$ ;
7    $\text{wake\_up}(\text{dest}, \mathbb{W})$ ;
8    $\text{exchange}(\mathbb{W})$ ;
9   if  $|\mathbb{W}| = 0$  then stop;
10  else if  $|\mathbb{W}| = 1$  then stop;
11  else if  $|\mathbb{W}| = 2$  then  $\mathbb{W} \leftarrow \text{child}_2(\mathbb{W})$  ;
12 end

```

Lemma 2 (Distributed Makespan). *Given a square region \mathcal{S} of width R and a robot r positioned in the center of \mathcal{S} , knowing both \mathcal{S} and a set of sleeping robots \mathcal{R}_S whose initial positions are in \mathcal{S} , if no robots other than $\{r\} \cup \mathcal{R}_S$ takes action in \mathcal{S} , then r can wake-up all robots of \mathcal{R}_S in time $5R$.*

Proof. In [BCGH24], authors show that if a set of sleeping robot is contained within a disk of radius R' around an awake robot, one can compute a wake-up tree with makespan $5\sqrt{2}R'$. The proof follow by considering the disk of center p_r and radius $\frac{R}{\sqrt{2}}$. \mathcal{S} is entirely contains in that disk. By assumption r known the positions of sleeping robots \mathcal{R}_S within \mathcal{S} . Therefore r can awake \mathcal{S} by computing an propagating a wake-up tree with makespan $\frac{R}{\sqrt{2}}\sqrt{2} = 5R$. The correctness is due to the assumption that no robots other than $r \cup \mathcal{R}_S$ awake robots in \mathcal{S} , which avoid any risk of conflicts. \square

6.3 Local Synchronization

Let us prove Corollary 1 whenever several awake robots located in the same square aim at waking up sleeping robots of the square. We assume that these awake robots start their action at the same time:

Corollary 1 (Explore and Wake up). *Given a square region \mathcal{S} of width R containing at least one awake robot, if no awake robot goes through the border \mathcal{S} , it is possible to wake-up all sleeping robots of \mathcal{S} in time $R^2 + (10 + \sqrt{2})R$.*

Proof. Every awake robots move to the lower-left corner of \mathcal{S} in at most $\sqrt{2}R$. One of them explores \mathcal{S} in time $R^2 + 5R$ (Lemma 1) and move to the center of \mathcal{S} . Among all awake robots of \mathcal{S} , only that one does take action. Lemma 2 apply: it can awake every sleeping robots in time $5R$. \square

6.4 ℓ -sampling and Eccentricity

As discussed in Section 2.4, computing an ℓ -sampling of a set of positions \mathcal{P} is an efficient way to discover \mathcal{P} , and thus to wake up robots. In this section we show some relationships between the values ℓ, ρ and ξ_ℓ .

Lemma 4 (ℓ -sampling cardinality). *If \mathcal{P}' is an ℓ -sampling of a square region of width R , then $|\mathcal{P}'| \leq 16R^2/(\pi\ell^2)$.*

Proof. For any $p_i, p_j \in \mathcal{P}'$, $B_{p_i}(\ell/2) \cap B_{p_j}(\ell/2) = \emptyset$. The area of each ball is $\pi\ell^2/4$. Although p_i 's are located within \mathcal{S} , it may happen that only a fraction of $B_{p_i}(\ell/2)$ is contained within \mathcal{S} . This fraction is at least $1/4$ whenever p_i is located at a corner of \mathcal{S} . Since $\bigcup_{i=1}^{|\mathcal{P}'|} (B_{p_i} \cap \mathcal{S}) \subset \mathcal{S}$, we have $|\mathcal{P}'|\pi\ell^2/16 \leq R^2$. \square

Lemma 6. *Let us consider a point set (\mathcal{P}, s) . For any $\ell \geq \ell^*$, we have $\xi_\ell \in [\rho^*, \frac{12\rho^{*2}}{\ell}]$, and for any position $p \in \mathcal{P}$, there exists a path from s to p in the ℓ -disk graph which is at most $1 + \frac{2\xi_\ell}{\ell}$ -hops long.*

Proof. The lower bound $\xi_\ell \geq \rho^*$ is straightforward.

Let $s, p_1, \dots, p_k = p$ be a shortest path from s to p in the ℓ -disk graph of (\mathcal{P}, s) , which is minimal in terms of hops. By definition, $\forall 1 \leq i \leq k - 2, |p_i p_{i+2}| > \ell$, otherwise p_i and p_{i+2} are connected in the ℓ -disk graph, and therefore there exists another shortest path, smaller by one hop. Therefore, $d_\ell(s, p)$ is at least ℓ times the number of hops from p_{2i} to p_{2i+2} : $d_\ell(s, p) \geq \ell \lfloor k/2 \rfloor \geq \ell \frac{k-1}{2}$. Since $d_\ell(s, p) \leq \xi_\ell$ we conclude $k \leq 1 + 2\xi_\ell/\ell$

Note also that we can apply Lemma 4 to the set of robots of even index in that path. Therefore we obtain $\lfloor k/2 \rfloor \leq \frac{16\rho^2}{\pi\ell^2}$ and thus $k \leq 1 + \frac{32\rho^2}{\pi\ell^2} < \frac{36\rho^2}{\pi\ell^2} < 12\rho^2/\ell^2$. Furthermore, we have $d_\ell(s, p) \leq k\ell$, which imply that $d_\ell(s, p) \leq 12\rho^{*2}/\ell$. Since this is true for any p , and $\xi_\ell = \max_{p \in \mathcal{P}} d_\ell(s, p)$ we deduce $\xi_\ell(G) \leq 12\rho^{*2}/\ell$. \square

6.5 Description and proof of Distributed ℓ -sampling

We describe more formally the algorithm DFSAMPLING. In a region \mathcal{S} a team \mathcal{T} compute an ℓ -sampling of \mathcal{S} , starting from a given set of seeds \mathcal{X} . If robots have been awakened previously in \mathcal{S} , we denote by A the set of their initial positions. A is assumed known by \mathcal{T} . The output is a set of points \mathcal{P}' that is a ℓ -sampling of \mathcal{S} . In practice, we use DFSAMPLING to recruit a team \mathcal{T}' of at most 4ℓ robots identified by their positions \mathcal{P}' .

Let us start by defining a function SORT(\mathcal{X}) to order positions of seeds $\mathcal{X} = \{X_i\}$ whenever $\mathcal{X} > 1$. Seeds are ordered with respect to a projection on the borders of \mathcal{S} in the clockwise order around the center of square \mathcal{S} . More precisely, a seed X_i is projected to the closest point of the border of \mathcal{S} breaking tie by choosing the first projected point in the clockwise order.

1. $\mathcal{P}' = \emptyset$. Positions $\mathcal{X} = [X_1, X_2, \dots, X_j]$ are ordered with SORT(\mathcal{X}). Set $i = 1$. Add X_i to \mathcal{P}' .
2. Team \mathcal{T} moves to X_i and performs a Depth First Search traversal of the 2ℓ -disk graph G induced by $\mathcal{P} \cap \mathcal{S}$. A neighbor p' of a current position p is either known if $p' \in A$ or is found using EXPLORE($B_p(2\ell), p$). The team explores as far as possible along each branch before backtracking and stop if $|\mathcal{P}'| = 4\ell$. A stack keeps track of the positions of \mathcal{P}' discovered so far along a specified branch which helps in backtracking of the graph. We have one condition to move to p' : *a position p' is added to \mathcal{P}' if its distance to any point of \mathcal{P}' is greater than ℓ .* If the

robot located at p' is sleeping, it is awakened by the team \mathcal{T} and added to \mathcal{T} .

3. If $|\mathcal{P}'| < 4\ell$ and $i < j$ then $i \leftarrow$ the index of the next seed of $X_{i'}$ such that $B_{X_{i'}}(\ell)$ is not covered and repeat Step 2;
4. return \mathcal{P}'

We are now ready to prove the corresponding lemma:

Lemma 5 (DFSAMPLING). *In a region \mathcal{S} of width R containing a set of seeds $\mathcal{X} \subseteq \mathcal{P}$, a team \mathcal{T} of robots knowing the awake robots of \mathcal{S} can compute a set of positions \mathcal{P}' being an ℓ -sampling of \mathcal{P} . The computation is done using DFSAMPLING in time:*

$$\begin{aligned} &O(\ell^2 \log(|\mathcal{P}'|)) \text{ if } |\mathcal{T}| = 1 \text{ and } \mathcal{X} = \{p_s\} \text{ and } R \geq 2\rho^* \\ &O(R + \ell|\mathcal{P}'|) \text{ if } |\mathcal{T}| = \ell \text{ and } \mathcal{X} = \mathcal{P} \cap \text{sep}(\mathcal{S}) \end{aligned}$$

In both cases, either (1) $|\mathcal{P}'| = 4\ell$; or (2) $|\mathcal{P}'| < 4\ell$ and \mathcal{S} is covered by \mathcal{P}' .

Proof. By construction, any p' added to \mathcal{P}' is at distance greater than ℓ from any other point of \mathcal{P}' . Inductively, any pair of positions in \mathcal{P}' has a pairwise distance greater than ℓ . Thus the output of DFSAMPLING is an ℓ -sampling of \mathcal{S} . Let us now analyze the time required to obtain that ℓ -sampling.

The Depth First Search traversal uses three operations: find neighbors, move to a neighbor and backtrack if no new neighbor is added.

In the following \mathcal{T} stands for the current team of robots. As soon as the robot team wakes up a sleeping robot, it is added to \mathcal{T} so the team size can increase. Let us start with seed X_1 . The time to find the list of potential neighbors in the 2ℓ -disk graph is the time to explore $B_p(2\ell)$. From Lemma 1, it takes a time in $O(\ell^2/|\mathcal{T}|)$. Thus exploring and moving to a neighbor takes $O(\ell^2/|\mathcal{T}| + \ell)$.

Let us first consider the case where the initial team $|\mathcal{T}| = 1$ and $\mathcal{X} = A = p_s$. Assume that at some point, k robots were awakened and added to \mathcal{T} . This happens as soon as there is a sleeping robot on a position p' that is added to \mathcal{P}' . The time to find and wake up these k robots is upper bounded by $\sum_{i=1}^k O(\ell^2/i) \in O(\ell^2 \log k)$.

Let us focus on the time dedicated to moves. For recruiting $k \geq 2$ robots, \mathcal{T} either have to move forward to a neighbor at a distance at most 2ℓ , or backtrack to branch from a neighbor of a previously added robot. The amount of moves required to backtrack is at most $2k\ell$ units since the tree corresponding to the Depth First Search traversal has k edges of length at most 2ℓ . In total, the total time of backtracking is less than $2k\ell \leq 8\ell^2 \in O(\ell^2)$ since $k \leq 4\ell$. In total, it takes $O(\ell^2 \log k)$ for this first case.

The second case is whenever we start with a team of size ℓ and a set of seeds $\mathcal{X} = \mathcal{P} \cap \text{sep}(\mathcal{S})$. The exploration is now done collaboratively with ℓ robots. By Lemma 1, for any position $p' \in \mathcal{P}'$ the exploration phase takes $O(\ell)$. As in the first case, the different moves take $O(k\ell)$. Thus the total time of exploration for the recruitment of k robots takes $O(k\ell)$.

The second change is that the Depth First Search traversal starting from seed X_1 can stop before reaching 4ℓ robots. In this case the search goes on starting from the next seed. We have to add the time corresponding to the moves from one seed to another. Note that \mathcal{T} starts a branch from at most 4ℓ positions $X_{i_1}, \dots, X_{i_{4\ell}}$ of \mathcal{X} . Assume that \mathcal{T} recruit k seeds among these 4ℓ positions. We rely on the fact that $\mathcal{X} \subseteq \text{sep}(\mathcal{S})$ and the ordering computed by $\text{SORT}(\mathcal{X})$ to bound the total duration of these intermediates moves.

Let Y_i be the projected point of X_i on the border of \mathcal{S} . By definition of the ordering $\text{SORT}(\mathcal{X})$, the distance between two consecutive seeds X_{i_j} and $X_{i_{j+1}}$ is at most $|X_{i_j}Y_{i_j}| + |Y_{i_j}, Y_{i_{j+1}}| + |X_{i_{j+1}}Y_{i_{j+1}}| \leq 2\ell + d_1(Y_{i_j}, Y_{i_{j+1}})$ where $d_1(p, p')$ is the distance in the L_1 -norm for $p, p' \in \mathbb{R}^2$. Thus the length of $\sum_{j=1}^{k-1} |X_{i_j}X_{i_{j+1}}| \leq \sum_{j=1}^{k-1} d_1(Y_{i_j}, Y_{i_{j+1}}) + 2\ell \leq 4R + 2k\ell$ since $\sum_{j=1}^{k-1} d_1(Y_{i_j}, Y_{i_{j+1}})$ is at most the perimeter of the square. To conclude, it takes $O(R + k\ell)$.

We now focus on the properties (1) and (2). Assume that DFSAMPLING ends. We show by contradiction that if $|\mathcal{P}'| < 4\ell$ then \mathcal{S} is covered by \mathcal{P}' : $\forall p \in \mathcal{P} \cap \mathcal{S}, \exists p' \in \mathcal{P}'$ such that $p \in B_{p'}(2\ell)$.

Assume that there exists a point $p_0 \in \mathcal{P} \cap \mathcal{S}$ that is not covered by \mathcal{P}' . Since the connectivity threshold is less than ℓ , the ℓ -disk graph G of \mathcal{P} is connected and any pair of points in \mathcal{P} are linked by a path in G . In particular, there must exist a seed $p_x \in \mathcal{X}$ such that there is a path $b = (p_0, p_1, \dots, p_x)$ in G entirely contains in \mathcal{S} : $\bigcup_{i=1}^x p_i \subset \mathcal{S}$. It is trivial if \mathcal{P} is fully contained in \mathcal{S} i.e., when $R \geq 2\rho^*$.

Else, if there is in \mathcal{P} at least one point q outside of \mathcal{S} and $\mathcal{X} = \mathcal{P} \cap \text{sep}(\mathcal{S})$. From Lemma 3 any path from $p_0 \in \mathcal{S}^{in}$ to $q \in \mathcal{S}^{out}$ contains at least one position $p_x \in \text{sep}(\mathcal{S}) \cap \mathcal{X}$. Consider a path b from p_0 to p_x . By assumption, p_0 is not covered by \mathcal{P}' . Let p_i be the first position on the path such that p_i is not covered by \mathcal{P}' and p_{i+1} is covered. By construction of \mathcal{P}' if p_{i+1} is covered there is a position $p'_{i+1} \in \mathcal{P}'$ contains the close neighborhood of p_{i+1} in G i.e., $|p_{i+1}p'_{i+1}| \leq \ell$ or $p_{i+1} = p'_{i+1}$. This implies that $B_{p'_{i+1}}(2\ell)$ covered by \mathcal{P}' . Since $|p_i p_{i+1}| \leq \ell$ and $|p_{i+1} p'_{i+1}| \leq \ell$ we get $|p_i p'_{i+1}| \leq 2\ell$ which conflicts with the assumption that p_i is not covered. \square

7 Proofs of $\mathcal{A}_{\text{Separator}}$

We first prove the following lemma, which will be useful in the proofs of the makespan of $\mathcal{A}_{\text{Separator}}$ and $\mathcal{A}_{\text{Wave}}$.

Lemma 7. *For any $r \geq \ell \geq 1$ we have: $O(r + \ell^2(\log \min\{\ell, r/\ell\} + \log \frac{r}{\ell^{3/2}})) \subseteq O(r + \ell^2 \log(r/\ell))$*

Proof. $r/\ell^{3/2} \leq r/\ell$ and we have two cases:

1. $\ell \geq r/\ell \geq r/\ell^{3/2}$ implying $\log(\min\{\ell, r/\ell\} + \log \frac{r}{\ell^{3/2}}) \leq 2 \log r/\ell$.
2. $1 \leq \ell \leq r/\ell \leq r/\ell^{1/2}$.

In this second case, $\log \ell + \log r/\ell = \log(1 + \ell) + \log(1 + r/\ell^{3/2}) \leq \log(1 + \ell + r/\ell^{3/2} + r/\ell^{1/2}) \leq \log(4r/\ell^{1/2})$. We also have $\log(r/\ell^{1/2}) = \log \ell^{1/2} + \log r/\ell \leq \log \ell + \log r/\ell \leq 2 \log r/\ell$. Thus $\log(\min\{\ell, r/\ell\} + \log \frac{r}{\ell^{3/2}}) \leq \log 4 + 2 \log(r/\ell) \in O(\log(r/\ell))$. \square

Makespan $\mathcal{A}_{\text{Separator}}$

Lemma 8 (Makespan of $\mathcal{A}_{\text{Separator}}$). *For every admissible tuple (ℓ, ρ, n) and for any n -point set \mathcal{P} and source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$, the execution of $\mathcal{A}_{\text{Separator}}$ with source s and inputs (ℓ, ρ, n) on \mathcal{P} terminates in time $O(\rho + \ell^2 \log(\rho/\ell))$*

Proof. Since $\mathcal{A}_{\text{Separator}}$ is decomposed into rounds, the proof is done by bounding the duration of a Round k , denoted t_k , and the number of rounds until $\mathcal{A}_{\text{Separator}}$ terminates.

The duration of Round 0 is the duration of recruiting a team \mathcal{T} of size up to 4ℓ within a square of width $2\rho^*$. The recruitment is done by computing a ℓ -sampling of the square with DFSAMPLING. From Lemma 4 there is at most $32\rho^{*2}/(\pi\ell^2)$ points in a ℓ -sampling of a square of width $2\rho^*$, therefore $|\mathcal{T}| = \min(4\ell, 32\rho^{*2}/(\pi\ell^2))$. Lemma 5 guarantee that the time of DFSAMPLING depends of the size of the output team. Hence the duration of Round 0 is:

$$t_0 \in O(\ell^2 \log(\min\{\ell, \rho^*/\ell\})) \subseteq O(\ell^2 \log(\min\{\ell, \rho/\ell\})). \quad (1)$$

For $k \geq 1$, we denote by $\mathcal{S}^{(k)}$ and $\mathcal{T}^{(k)}$ a square region and a team given as inputs at Round k . Let $R^{(k)}$ be the width of $\mathcal{S}^{(k)}$. Note that for a given $k > 1$, several squares are treated in parallel so neither $\mathcal{S}^{(k)}$ nor $\mathcal{T}^{(k)}$ are unique. Given k , $R^{(k)} = \frac{2\rho}{2^{k-1}}$ since at each round, the square $\mathcal{S}^{(k)}$ is partitioned into four squares implying that $R^{(k+1)}$ is the half of $R^{(k)}$, starting with $R^{(1)} = 2\rho$.

Round $k \geq 1$ with input $\mathcal{S}^{(k)}$ and $\mathcal{T}^{(k)}$ is a *terminating* Round if it begins with $|\mathcal{T}^{(k)}| < 4\ell$, otherwise it is a *partitioning* Round. A terminating round only takes time $O(R^{(k)}) \subseteq O(\rho)$ to move toward the center of the sub-square and wake-up remaining sleeping robots discovered in the DFSAMPLING execution.

Let us focus on partitioning rounds. We show in Lemma 10 that for $k \geq 1$, $t_k = O(R^{(k)} + \ell^2)$.

Let k_{max} be the maximum number of Rounds. From Lemma 11, $k_{max} = O(\log \frac{\rho}{\ell^{3/2}})$. The proof follows by summing the duration of Rounds from $k = 0$ to k_{max}

$$T_{\mathcal{A}_{Separator}} = \sum_{k=0}^{k_{max}} t_k = t_0 + \sum_{k=1}^{k_{max}} t_k \quad (2)$$

$$\in O(\ell^2 \log(\min\{\ell, \rho/\ell\})) + \sum_{k=1}^{k_{max}} O(R^{(k)} + \ell^2) \quad (3)$$

$$\in O(\ell^2 \log(\min\{\ell, \rho/\ell\})) + \sum_{k=1}^{k_{max}} O(\frac{2\rho}{2^{k-1}}) + k_{max} O(\ell^2) \quad (4)$$

$$\in O(\ell^2 \log(\min\{\ell, \rho/\ell\})) + O(\rho) + O(\log \frac{\rho}{\ell^{3/2}}) O(\ell^2) \quad (5)$$

$$\in O(\rho + \ell^2(\log \min\{\ell, \rho/\ell\} + \log \frac{\rho}{\ell^{3/2}})) \quad (6)$$

$$\in O(\rho + \ell^2 \log(\rho/\ell)) \quad (7)$$

The last line comes from Lemma 7. □

Lemma 9. *For every admissible tuple (ℓ, ρ, n) and for any n -point set \mathcal{P} and source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$, the execution of $\mathcal{A}_{Separator}$ with source s and inputs (ℓ, ρ, n) on \mathcal{P} eventually wakes up all robots.*

Proof. Consider an execution of $\mathcal{A}_{Separator}$ on \mathcal{P} . From Lemma 11 an execution terminates after a finite number of Rounds. We show that when the execution terminates, any robot $r \in \mathcal{R}$ initially located at $p \in \mathcal{P}$ is awakened. Let k be the largest k such that at Round k there is an input square \mathcal{S}^k containing p .

Either r is awakened and then $r \in \mathcal{T}^k$, or it is asleep. In this case, we show that r must be awakened at the end of Round k .

Firstly, Round k on \mathcal{S}^k and \mathcal{T}^k must be a terminating Round. If it is not the case, i.e., it is a partitioning Round, then for $\{i \in [1, 4] \mid \mathcal{T}'_i \neq \emptyset\}$ sub-square \mathcal{S}^k_i is an input of Round $k + 1$. Since robots are assigned to teams \mathcal{T}'_i based on their initial position, if \mathcal{T}'_i is empty then $\mathcal{P} \cap \text{sep}(\mathcal{S}^k_i) = \emptyset$, by Corollary 2, \mathcal{S}^k_i is empty and does not contain p . Hence if Round k is

a partitioning Round then at Round $k + 1$ there must be an input \mathcal{S}^{k+1} containing p , which breaks the assumption of k being maximal.

If Round k is terminating, by definition of $\mathcal{A}_{\text{Separator}}$, the input team \mathcal{T}^k contains less than 4ℓ robots. It implies that during the recruitment phase of Round $k - 1$, DFSAMPLING in \mathcal{S}^k has stopped before constituting a team of 4ℓ robots. Lemma 5 guarantee that \mathcal{S}^k is covered by \mathcal{T}^k i.e., each position of $\mathcal{P} \cap \mathcal{S}^k$ is known by \mathcal{T}^k at the beginning of Round k . Therefore r must be awakened when \mathcal{T}^k operates a centralized awakening of sleeping robots of \mathcal{S}^k . \square

Duration of Rounds

Lemma 10 (Duration of Rounds). *For $k \geq 1$, the duration of Round k is in $O(R^{(k)} + \ell^2)$, $R^{(k)}$ being the width of sub-squares.*

Proof. We show that:

- a terminating Round has a duration $O(R^{(k)})$
- a partitioning Round has a duration $O(R^{(k)} + \ell^2)$

Let us begin with the termination case, that is an input $\mathcal{S}^{(k)}$ and $|\mathcal{T}^{(k)}| < 4\ell$. The Round k consists of using a centralized algorithm to wake up sleeping robots in $\mathcal{S}^{(k)}$. Recall that $\mathcal{T}^{(k)}$ is recruited during the recruit phase of Round $k - 1$ by sampling $\mathcal{S}^{(k)}$. Since $|\mathcal{T}^{(k)}| < 4\ell$ Lemma 5 guarantees that $\mathcal{S}^{(k)}$ is covered by the subset of $\mathcal{T}^{(k-1)}$ sampling $\mathcal{S}^{(k)}$. At the end of Round $k - 1$ the knowledge of each subset of $\mathcal{T}^{(k-1)}$ is shared, thus sleeping robots of $\mathcal{P} \cap \mathcal{S}^{(k-1)}$ are known. The smallest disk containing $\mathcal{S}^{(k)}$ is of radius $(1/\sqrt{2})R^{(k)}$. By Lemma 2, one robot of $\mathcal{T}^{(k)}$ can wake up any set of robots sleeping in this disk with a makespan at most $O(R^{(k)})$.

Let us now consider a partitioning Round k with a non-terminating input $\mathcal{S}^{(k)}$ and $|\mathcal{T}^{(k)}| \geq 4\ell$. It consists of having four teams of ℓ robots exploring and recruiting within sub-squares of $\mathcal{S}^{(k)}$ of width $R^{(k+1)}$. The separator of a sub-square can be decomposed into 4 rectangles of dimension $\ell \times R^{(k+1)}$. By Corollary 1 a team of ℓ robots can explore each rectangle in $O(\frac{\ell \times R^{(k+1)}}{\ell} + \ell + R^{(k+1)}) = O(R^{(k+1)} + \ell)$. The recruitment is done by DFSAMPLING from a set of positions \mathcal{X} in $\text{sep}(\mathcal{S}^{(k)})$. From Lemma 5 the ℓ -sampling of $\mathcal{S}^{(k+1)}$ with a team of ℓ robot is done in time $O(R^{(k+1)} + \ell^2)$. Recall that $R^{(k+1)} = \frac{1}{2}R^{(k)}$, we get that the duration of a partitioning Round k is $O(R^{(k)} + \ell^2)$ \square

Number of Rounds

Lemma 11 (Number of Rounds). *The number of rounds of $\mathcal{A}_{\text{Separator}}$ is 1 for $\rho^* \leq \frac{\ell^{3/2}}{8}$ and $\log(1 + \sqrt{8/\pi\rho}/\ell^{3/2}) \in O(\log \frac{\rho}{\ell^{3/2}})$ otherwise. Both values are in $O(\log \frac{\rho}{\ell^{3/2}})$.*

Proof. We have two cases depending on ρ^* , being large or small with respect to ℓ . Set $R = 2\rho^*$.

If $R \leq \frac{\ell^{3/2}}{4}$, from Lemma 4, any ℓ -sampling has size at most $\frac{16R^2}{4\pi\ell^2} \leq 4\ell/\pi < 4\ell$. Algorithm $\mathcal{A}_{\text{Separator}}$ stops in Round 1.

From now, we assume that $R > \frac{\ell^{3/2}}{4}$. Let us compute the smallest k such that every execution of DFSAMPLING to $\mathcal{S}^{(k)}$ of width $R^{(k)}$ outputs a ℓ -sampling $\mathcal{T}^{(k)}$ strictly smaller to 4ℓ . In this case, $\mathcal{A}_{\text{Separator}}$ ends at the beginning of Round $k + 1$ during the termination phase.

From Lemma 4, we have

$$|\mathcal{T}^{(k)}| \leq \frac{16(R^{(k)})^2}{\pi\ell^2} \quad (8)$$

$$\text{Recall that } R^{(k)} = \frac{2\rho}{2^{k-1}} \quad (9)$$

$$\text{We get } R^{(k)} \leq \frac{\ell^{3/2}}{4} = \frac{256\rho^2}{2^{2k}}\pi\ell^2 < 4\ell \quad (10)$$

$$\text{if } 2^{2k} > \frac{8\rho^2}{\pi\ell^3} \quad (11)$$

$$k > \log(\sqrt{8/\pi\rho}/\ell^{3/2}). \quad (12)$$

□

8 Proofs of $\mathcal{A}_{\text{Grid}}$ and $\mathcal{A}_{\text{Wave}}$

8.1 Algorithm $\mathcal{A}_{\text{Grid}}$

We now describe in detail Algorithm $\mathcal{A}_{\text{Grid}}$:

Let $t(\ell)$ be an upper bound on the time required for the exploration and centralized awakening of a square region of width 2ℓ by one robot. By Corollary 1, $t(\ell) \in O(\ell^2)$.

- *Round 0 - Initialization:*
 $\mathcal{S} \leftarrow$ square of width $R = 2\ell$ and centered at the source robot s
 Explore and wake-up \mathcal{S}
- *Round $k \geq 1$ starting at time $t_k = t(\ell) + 8(k-1)(t(\ell) + \sqrt{2}R)$:*
 For every robot r awakened in Round $k-1$ do in parallel:
 $\mathcal{S} \leftarrow$ the square containing r
 For $i \in [1..8]$ do
 1. move to the lower-left corner of the i -th adjacent squares \mathcal{S}_i of \mathcal{S}
 2. Wait until time $t_k + (t(\ell) + \sqrt{2}R)i$

3. Explore and Wake-up \mathcal{S}_i

Let us remind the result on the makespan of \mathcal{A}_{Grid} :

Theorem 4. \mathcal{A}_{Grid} solves the *dFTP*, given every admissible tuple (ℓ, ρ, n) , with an energy budget of $O(\ell^2)$ and with a makespan of $O(\ell \cdot \xi_\ell)$.

Proof. The duration of Round 0 is $t(\ell) \in O(\ell^2)$. Then, in Round $k \geq 1$, we have to wake-up at most 8 squares in time $8t(\ell)$. Since every square are adjacent, it takes at most $\sqrt{2}R$ to go the next adjacent square to wake-up. In total, every round takes $O(\ell^2)$. Every robot awakened in Round k participates to Round $k + 1$ but stops do at the end of Round $k + 1$. Thus it only has to use $O(\ell^2)$ amount of energy.

Let us now upper bound the number of rounds. Given any robot r located at position p , let us consider a shortest path from s to r in the ℓ -disk graph of (\mathcal{P}, s) minimizing the number of hops and the sequence of squares $S = s_1, \dots, s_k$ crossed by this path. Since the path can cross several times a same square and every robot of a given square are awakened in the same round, the number of rounds required to wake up the sequence of squares is smaller or equal to the number of hops of the shortest path. From Lemma 6, we know that there exists a path of at most $2\xi_\ell/\ell$ hops implying the same upper bound for the number of rounds. Since every round takes a time $O(\ell^2)$, the makespan is $O(\ell \cdot \xi_\ell)$. Furthermore since every awakened robot in Round k only move in Round k and $k + 1$ and stop, robots only need $O(\ell^2)$ energy. \square

8.2 Algorithm \mathcal{A}_{Wave}

We now describe in detail Algorithm \mathcal{A}_{Wave} :

Let $t(R)$ be an upper bound on the time required for $\mathcal{A}_{Separator}$ to wake up all robots of a square of width R starting from a team of size 4ℓ . By Theorem 1, $t(R) \in \Theta(R + \ell^2 \log \ell)$.

- *Round 0 - Initialization:*
 $\ell \leftarrow \max\{\ell, 4\}$
 $\mathcal{S} \leftarrow$ square of width $R = 8\ell^2 \log_2 \ell$ and centered at the source robot s
Wake-up \mathcal{S} using $\mathcal{A}_{Separator}$
If there is no robot in $\text{sep}(\mathcal{S})$, \mathcal{A}_{Wave} stops
- *Round $k \geq 1$ starting at time $t_k = t(R) + 8(k - 1)(t(R) + \sqrt{2}R)$, for every r awakened in Round $k - 1$:*
 $\mathcal{S}(r) \leftarrow$ square containing r
Move r to the lower-left corner of $\mathcal{S}(r)$ to build a team \mathcal{T}_r
For every team \mathcal{T}_r such that $|\mathcal{T}_r| \geq 4\ell$ do in parallel:
For $i \in [1..8]$ do

1. move to the lower-left corner of the i -th adjacent squares \mathcal{S}_i of $\mathcal{S}(r)$
2. Wait until time $t_k + (t(R) + \sqrt{2}R)i$
3. Wake-up \mathcal{S}_i using $\mathcal{A}_{\text{Separator}}$ within \mathcal{S}_i with \mathcal{T}_r in Round k

Let us prove Theorem 5

Theorem 5. $\mathcal{A}_{\text{Wave}}$ solves the dFTP, given every admissible tuple (ℓ, ρ, n) , with an energy budget of $O(\ell^2 \log \ell)$ and a makespan of $O(\xi_\ell + \ell^2 \log \xi_\ell / \ell)$.

Proof. We first prove that $\mathcal{A}_{\text{Separator}}$ solves the dFTP with energy budget $O(\ell^2 \log \ell)$ and a makespan of $O(\xi_\ell + \ell^2 \log \ell)$, and explain in a second time how we obtain the announced makespan. Firstly, note that $t(R) \in \Theta(\ell^2 \log \ell)$.

To begin, every round takes a time at most $O(R) = O(\ell^2 \log \ell)$. The only robots that participates in Round k has been awakened only in Round $k + 1$, it means that the energy budget required per robot is $O(\ell^2 \log \ell)$.

At Round 0, $\mathcal{A}_{\text{Separator}}$ wakes up every robot of the initial square \mathcal{S} . If there is a robot within $\text{sep}(\mathcal{S})$, it means that \mathcal{S} contains at least $\lfloor \frac{R/2 - \ell}{\ell} \rfloor \geq \frac{R}{2\ell} - 2 = 4\ell \log_2 \ell - 2 \geq 4\ell$ robots since $\ell \geq 4$. Thus, \mathcal{S} has enough robots to apply $\mathcal{A}_{\text{Separator}}$ in every adjacent squares at Round 1, and every robot within the 8 adjacent squares are awakened during Round 1.

Let us now prove that all robots are awakened by the algorithm by providing an upper bound on the number of rounds to wake up a robot r located outside the 9 central squares. Let G be the ℓ -disk graph of \mathcal{P} and let $P = s, r_1, r_2, \dots, r$ be a shortest path in G from s to r , and let $s, r_1, r_2, \dots, r_{j'}$ be the maximal subpath of P of awake robots at the end of Round k . We now consider a robot r_j with $j < j'$ such that $R - 3\ell < d_\ell(r_j, r_{j'}) \leq R - 2\ell$. Such a robot exists since by definition of s , $d_\ell(s, r) > R$.

First, let us show that the subpath $P' = r_j, r_{j+1}, \dots, r_{j'}$ is contained within $\mathcal{S}(r_{j'})$ and at most 2 squares simultaneously adjacent to $\mathcal{S}(r_{j'})$ and $\mathcal{S}(r_{j'+1})$. Let \mathcal{S} be a square, adjacent to $\mathcal{S}(r_{j'})$ but not adjacent to $\mathcal{S}(r_{j'+1})$. The Euclidean distance between $r_{j'}$ and \mathcal{S} is greater than $R - \ell$ since $r_{j'}$ is at distance at most ℓ from $\mathcal{S}(r_{j'+1})$. Thus P' cannot cross \mathcal{S} . Moreover, it is impossible that P' crosses two adjacent squares of $\mathcal{S}(r_{j'})$, \mathcal{S} and \mathcal{S}' , such that \mathcal{S} and \mathcal{S}' are not adjacent because P' has a length strictly smaller than R . This guarantees the announced property on P' .

Secondly, let us show that $\mathcal{S}(r_{j'+1})$ is awakened at Round $k + 1$. To have this guarantee, we need to have at least 4ℓ awakened at Round k in an adjacent square. Since P' is contained within at most 3 adjacent squares of $\mathcal{S}(r_{j'+1})$, the most populated adjacent square contains at least $\lfloor \frac{R - 2\ell}{\ell} \rfloor / 3 = \lfloor \frac{R}{\ell} \rfloor / 3 - 2/3 \geq (R/\ell - 1)/3 - 2/3 = R/(3\ell) - 1$ awake robots. If $R = 8\ell^2 \log \ell$ and $\ell \geq 4$, $R/(3\ell) - 1 \geq 16\ell/3 - 1 \geq 4\ell$.

Now, take the maximal subpath P'' of P of length smaller than $R - 2\ell$ but starting from $r_{j'+1}$. We can show similarly as before that this path is either within 3 adjacent squares awakened in the worst case in Rounds $k + 1$, $k + 2$ and $k + 3$ or that this path ends in already awakened square. In any case, within 3 rounds, the length of the maximal subpath of P of awake robots at the end of Round $k + 3$ has increased of at least $R - 3\ell > 5\ell^2 \log \ell$ units. Thus the number of rounds to wake up the robots of highest eccentricity takes $O(\xi_\ell/\ell^2 \log \ell)$ rounds. The makespan is then bounded by $O(\xi_\ell)$.

Finally, let us be more precise on the bound on the makespan. We first consider the case where $\xi_\ell \leq \ell^{3/2}/16 \leq \ell^{3/2}/16$. We immediately have $\xi_\ell < R/2$ and therefore \mathcal{A}_{Wave} terminates at Round 0. But since $\xi_\ell \leq \ell^{3/2}/16$, we have by Proposition 1, $\rho^* \leq \ell^{3/2}/16$ and so, by Lemma 11, $\mathcal{A}_{Separator}$ terminates at Round 0. In that case, the makespan of \mathcal{A}_{Wave} is therefore $t_0 \in O(\ell^2 \log(\min\{\ell, \rho^*/\ell\}))$ as stated in Equation 1. Since by Proposition 1 and Lemma 6, $\rho^*/\ell \leq \xi_\ell/\ell \leq \rho^{*2}/(12\ell^2)$, we obtain an overall complexity of \mathcal{A}_{Wave} is in $O(\ell^2 \log(\min\{\ell, \xi_\ell/\ell\}))$.

Otherwise, we have $\xi_\ell \geq \ell^{3/2}/16$, which means that $\xi_\ell/\ell \geq \sqrt{\ell}/16$ and thus that $\min\{\sqrt{\ell}/16, \xi_\ell/\ell\} = \sqrt{\ell}/16$. Therefore, $O(\ell^2 \log \ell) = O(\ell^2 \log(\min\{\ell, \xi_\ell/\ell\}))$. This guarantee an overall complexity in $O(\xi_\ell + \ell^2 \log \min\{\ell, \xi_\ell/\ell\})$ for \mathcal{A}_{Wave} .

To summarize, we have an overall complexity in $O(\xi_\ell + \ell^2 \log(\min\{\ell, \xi_\ell/\ell\})) \subseteq O(\xi_\ell + \ell^2 \log(\min\{\ell, \xi_\ell/\ell\}) + \xi_\ell/\ell^{3/2})$. By Lemma 7, we conclude that \mathcal{A}_{Wave} has a makespan in $O(\xi_\ell + \ell^2 \log \xi_\ell/\ell)$. \square

9 Lower bounds - details

9.1 Without energy constraint

Let us recall and prove Theorem 2:

Theorem 2 (Lower Bound without energy constraint). *For every admissible tuple (ℓ, ρ, n) and algorithm \mathcal{A} solving the d -FPT, there exists an n -point set \mathcal{P} and a source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$ such that the makespan of the execution of \mathcal{A} with source s and inputs (ℓ, ρ, n) on \mathcal{P} is $\Omega(\rho + \ell^2 \log(\rho/\ell))$.*

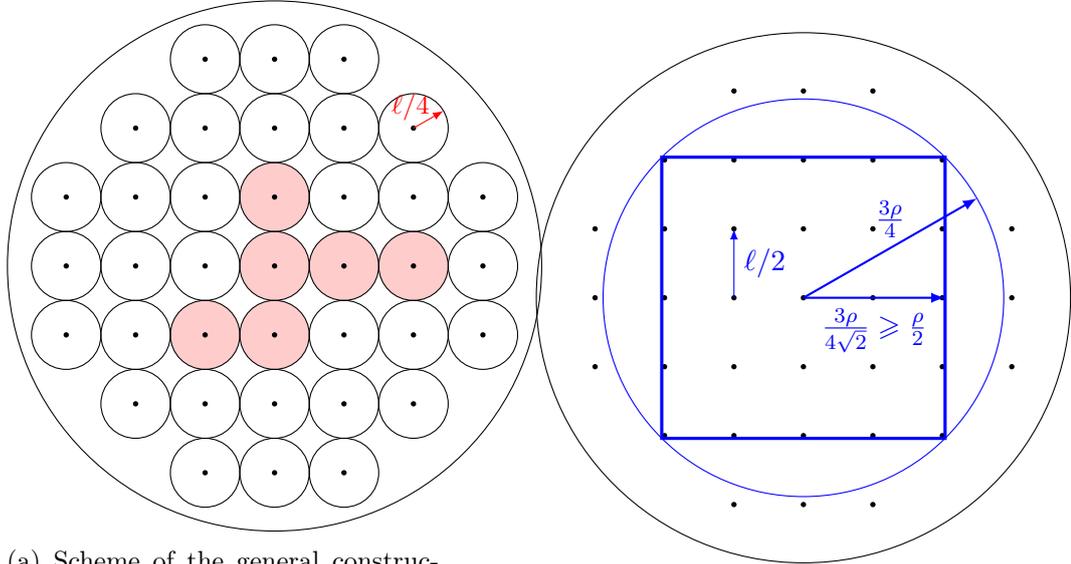
Let us consider an algorithm \mathcal{A} solving the dFTP, and an admissible tuple (ℓ, ρ, n) . To prove our lower bound, we first define some disjoint regions D_c of $B_{0,0}(\rho)$ as pictured in Figure 5a. Each region has area $\Theta(\ell^2)$, and we prove in Lemma 12 that there are $\Theta(\rho^2/\ell^2)$ such regions. The general idea is to place one sleeping robot in each region, depending on the behavior of \mathcal{A} , in a way that guarantees that awake robots have to visit the entire region before they find the sleeping robot in it. We prove in Lemma 13 that the set of points we propose is ℓ -connected, which makes the construction valid.

Centers and connectivity. We define $\mathcal{C} = \{(x, y) \in (\frac{\ell}{2}\mathbb{Z})^2 \mid \sqrt{x^2 + y^2} \leq \rho - \frac{\ell}{4}\}$ the vertices of the $\frac{\ell}{2}$ -grid, restricted to the disk of center $(0, 0)$ and radius $\rho - \frac{\ell}{4}$. We say that two elements (x, y) and (x', y') in \mathcal{C} are *adjacent* if $x' = x \wedge y' = y \pm \frac{\ell}{2}$ or $x' = x \pm \frac{\ell}{2} \wedge y' = y$. A subset $C \subseteq \mathcal{C}$ is *connected* if for any $c, c' \in C$, there exists a path of adjacent elements of C with extremities c and c' .

We also define $\mathcal{C}^* = \mathcal{C} \setminus \{(0, 0)\}$ and $m = \min(n, |\mathcal{C}^*|)$. Note that, since (ℓ, ρ, n) is admissible, and by Lemma 12, we have $m \geq \rho/\ell$. We are now going to prove a lower bound of $\Omega(\ell^2 \log m) \subseteq \Omega(\ell^2 \log \rho/\ell)$.

We denote by \mathcal{C}_m some subset of \mathcal{C}^* with m elements such that $\mathcal{C}_m \cup \{(0, 0)\}$ is connected, and contains at least $\{(0, \frac{\ell}{2}), (0, 2\frac{\ell}{2}), \dots, (0, \lfloor \rho/\ell \rfloor \frac{\ell}{2})\}$. Note that this is always feasible because, on the one hand, $m \geq \lfloor \rho/\ell \rfloor$, and on the other hand, $\lfloor \rho/\ell \rfloor \frac{\ell}{2} \leq \rho/2 \leq \rho - \frac{\ell}{4}$, so all these points are actually in \mathcal{C}^* .

Disks, ℓ -connectivity. For any $c = (x, y) \in \mathcal{C}_m$, let us define the disk of center c by $D_c = B_c(\ell/4)$, of area $\frac{\pi\ell^2}{16}$. Let us also define $\mathcal{D}_m = \cup_{c \in \mathcal{C}_m} D_c$ and $\mathcal{D}^* = \cup_{c \in \mathcal{C}^*} D_c$. Note that different disks in \mathcal{D}_m are pairwise disjoint (except one single point), that $\mathcal{D}_m \subset B_{(0,0)}(\rho)$, and that the area of \mathcal{D}_m is $|\mathcal{C}_m| \frac{\pi\ell^2}{16}$.



(a) Scheme of the general construction of Theorem 2, with set \mathcal{D}^* colored in red.

(b) Scheme of the proof of Lemma 12 with \mathcal{S} in blue.

Figure 5: Proof of the Lower Bound

Construction of the set $\mathcal{P}(\mathcal{A})$. Let us first suppose that $n \leq |\mathcal{C}^*|$. The construction of the set of initial positions $\mathcal{P}(\mathcal{A})$ depends on the considered algorithm \mathcal{A} . The process consists in placing exactly one robot per disk $D_c \in \mathcal{D}_m$, at position p_c . This construction guarantees that the instance (\mathcal{P}, s) is consistent with the tuple (ℓ, ρ, n) . In particular, the set is actually ℓ -connected according to Lemma 13 and because \mathcal{C}_m is connected. Given one disk $D_c \in \mathcal{D}_m$, the exact localisation p_c is defined as the last position of S_c to be explored by the previously awakened robots, under the execution of the algorithm. In other words, the algorithm must integrally explore S_c before it discovers the new robot in it. If $n > |\mathcal{C}^*|$, then the construction is similar for the first $|\mathcal{C}^*| - 1$ positions. The initial localization of the remaining robots can be in any arbitrary small disk of radius ε included in some area of \mathcal{D}_m that has not been discovered yet.

Proof of the lower bound $\Omega(\ell^2 \log m)$. We denote by $\mathcal{R}_{\mathcal{A}}(t)$ the set of awake robots at time t , and given the set of initial positions $\mathcal{P} = \mathcal{P}(\mathcal{A})$, we denote by $\mathcal{D}_{\mathcal{P}}(t) \subseteq \mathcal{D}_m$ the set of points of \mathcal{D}_m that have been discovered at time t or before. More formally, $\mathcal{D}_{\mathcal{P}}(t)$ is the set of points $p \in \mathcal{D}_m$ such that $\exists t' \leq t, \exists r \in \mathcal{R}_{\mathcal{A}}(t') : |p_r(t')p| \leq 1$.

We denote by $\|\mathcal{D}_{\mathcal{P}}(t)\|$ the area of $\mathcal{D}_{\mathcal{P}}(t)$. Finally, let us define, $\forall i \in [0, m], t_i = \inf\{t \geq 0 \mid \|\mathcal{D}_{\mathcal{P}}(t)\| \geq i\pi\ell^2/16\}$. By construction of $\mathcal{P}(\mathcal{A})$ we have $\forall i, \forall t < t_i, |\mathcal{R}_{\mathcal{A}}(t_i)| \leq i$. Since robots have a field of view of radius 1, they discover an area of amplitude 2 along an unitary move, which mean that an awake robot exploring during t' units of time can discover an area of at most $2t'$. Therefore we have $\forall i, \forall t \geq t_i, \|\mathcal{D}_{\mathcal{P}}(t)\| - \|\mathcal{D}_{\mathcal{P}}(t_i)\| \leq 2(t - t_i)|\mathcal{R}_{\mathcal{A}}(t)|$. Furthermore, $\|\mathcal{D}_{\mathcal{P}}(t_{i+1})\| - \|\mathcal{D}_{\mathcal{P}}(t_i)\| = \frac{\pi\ell^2}{16}$. By having t tend to t_{i+1} by inferior values, we obtain: $\forall i \geq 0, \|\mathcal{D}_{\mathcal{P}}(t_{i+1})\| - \|\mathcal{D}_{\mathcal{P}}(t_i)\| \leq 2(t_{i+1} - t_i) \times (i + 1)$.

Therefore by adding this telescopic sum, we obtain:

$$t_m \geq \frac{\pi\ell^2}{32} \sum_{i=1}^m i \geq \frac{\pi\ell^2}{32} \ln(m+1) \in \Omega(\ell^2 \log m)$$

Conclusion. Since $n \geq \frac{\rho}{\ell}$ and $|\mathcal{C}^*| \geq \rho^2/\ell^2$, and $m = \min(n, |\mathcal{C}^*|)$, we have obtained a lower bound of $\Omega(\ell^2 \log \frac{\rho}{\ell})$. Furthermore, by construction of \mathcal{C}^m , and by definition of \mathcal{P} , robots have explore all points of $D_{(0, \lfloor \rho/\ell \rfloor \frac{\ell}{2})}$, which means that there exists a path from $(0, 0)$ to a point at distance 1 of $(0, \lfloor \rho/\ell \rfloor \frac{\ell}{2})$. Such a path has a length at least $\lfloor \rho/\ell \rfloor \frac{\ell}{2} - 1 \geq \frac{\rho}{2\ell} \frac{\ell}{2} - 1 \geq \rho/4 - 1 \in \Omega(\rho)$. We therefore have shown that the makespan of \mathcal{A} on that set of point is $\Omega(\rho + \ell^2 \log \frac{\rho}{\ell})$.

Intermediate Lemmas.

Lemma 12. $|\mathcal{C}| \geq 1 + \rho^2/\ell^2$.

Proof. Let us consider \mathcal{S} the square included in the disk with center $(0,0)$ and radius $\frac{3\rho}{4}$, itself included in the disk with same center and with radius $\rho - \frac{\ell}{4}$, as pictured in Figure 5b. Let us count the number of points of \mathcal{C} included in \mathcal{S} . The semi-width of this square is $\frac{3\rho}{4\sqrt{2}} \geq \frac{\rho}{2}$. Therefore, the number of points of \mathcal{C} on one vertical line of \mathcal{S} is $1 + 2 \lfloor \frac{\rho/2}{\ell/2} \rfloor = 1 + 2 \lfloor \frac{\rho}{\ell} \rfloor$. Since $\rho \geq \ell$, we have $\lfloor \frac{\rho}{\ell} \rfloor \geq \frac{1}{2} \frac{\rho}{\ell}$, and thus the number of points of \mathcal{C} in \mathcal{S} is at least $(1 + \frac{\rho}{\ell})^2 \geq 1 + \rho^2/\ell^2$. \square

Lemma 13. *For any adjacent points $c, c' \in \mathcal{C}_m$, for any two points $(p, p') \in D_c \times D_{c'}$, we have $|pp'| \leq \ell$.*

Proof. Both D_c and $D_{c'}$ are included in the disk of center $\frac{c+c'}{2}$ and radius $\frac{\ell}{2}$. \square

9.2 About energy constraint

Let us recall and prove Theorem 3:

Theorem 3 (Lower Bound on the energy budget). *For every admissible tuple (ℓ, ρ, n) and algorithm \mathcal{A} with energy budget $B < \pi(\ell^2 - 1)/2$, there exists an n -point set \mathcal{P} and a source s such that $\ell^* \leq \ell$ and $\rho^* \leq \rho$ such that the execution of \mathcal{A} on \mathcal{P} does not wake-up any robot.*

Proof. We construct the n -point set in a similar way as in the proof of Theorem 2. Similarly as before, let us first present the situation where $n = 1$. The position of robot r_1 in v depends of the behavior of algorithm \mathcal{A} . More precisely, it is the last position of $B_{(0,0)}(\ell)$ to be discovered by s . Therefore, s can wake up robot r_1 only after it has discovered the entire disk $B_{(0,0)}(\ell)$, with area $\pi\ell^2$. At $t = 0$, the discovered area is π , and corresponds to the disk of radius 1 around s . During a move of amplitude δ , robot s discover a new area of at most 2δ , which mean that to discover the entire disk, s has to move with a total amplitude of at least $\frac{\pi\ell^2 - \pi}{2}$, which concludes the proof. If $n > 1$, then we can either place all the sleeping robots at the exact same position, either consider a small enough disk with radius ε which is not discovered by s before it has consumed its entire energy. \square

9.3 With energy constraint

Let us recall Theorem 6:

Theorem 6 (Lower Bound for energy constrained). *For every admissible tuple (ℓ, ρ, n) , for every $B > \ell$, and for every $\xi \in [\rho, \min\{n\ell -$*

$\rho/3, \lfloor \rho^2/(2(B+1)) + 1 \rfloor \}$, there exists an n -points set \mathcal{P} and a source s of connectivity threshold ℓ , radius ρ , and ℓ -eccentricity $\xi_\ell = \xi$ such that the makespan of any algorithm \mathcal{A} solving the dFTP for (\mathcal{P}, s) given (ℓ, ρ, n) and energy budget B , is $\Omega(\xi + \ell^2 \log(\xi/\ell))$.

We first show how to define a n -point set \mathcal{P} and a source s with a prescribed eccentricity ξ such that $\ell^* \leq \ell$, $\rho^* \leq \rho$ and $\xi_\ell = \xi$. We prove in Lemma 14 that the construction is valid. We then prove Theorem 6 using this construction and Theorem 2.

Let us briefly explain how we construct (\mathcal{P}, s) . The source s has position $p_s = (0, 0)$. The main idea is to define the positions of points \mathcal{P} along a rectilinear path Π , that is a path that consists only of horizontal and vertical segments, defined such that any point on a horizontal segment is space out of at least $B + 1$ from any other horizontal segment. Positions of \mathcal{P} are spread over Π such that the ℓ -disk graph of $\mathcal{P} \cap p_s$ is indeed connected.

Our construction of Π is such its length determines the ℓ -eccentricity of (\mathcal{P}, s) . The length of Π should match the prescribed ξ , which yields some constraints over the values of ξ , depending on the values of ρ, n and B .

Path definition The rectilinear path Π is defined by vertical segments of length $V = B + 1$ and horizontal segments of length $H = \rho/\sqrt{2}$. Let $J = \lfloor \xi/(H + V) \rfloor$ be the number of vertical segments. We define Π by the points $u_0, v_0, v_1, u_1, u_2, v_2, \dots, u_{J-1}, v_{J-1}, v_J, u_J$. There is a vertical segment between $[v_j v_{j+1}]$ or $[u_j u_{j+1}]$ and a horizontal segment between $[u_j v_j]$. The position of points are more formally defined by the following coordinates.

$$\forall j \in [0, J] : \begin{cases} u_j = (0, j(B+1)) \\ v_j = (\frac{\rho}{\sqrt{2}}, j(B+1)) \end{cases}$$

For $j \in [0, J[$ the j -th *section* of Π is a sequence of a horizontal segment $[u_j v_j]$ and a vertical segment, either $[v_j v_{j+1}]$ for even j or $[u_j u_{j+1}]$ if j is odd.

If no point of Π is at distance ρ from p_s then we need to define an additional segment line between $v_0 = (\rho/\sqrt{2}, 0)$ and a point $w_0 = (\rho, 0)$ to spread out some points of \mathcal{P} on $[v_0 w_0]$ until $\rho^* = \rho$. Conversely, note that Π should fit in a square S_ρ of width $\rho/\sqrt{2}$ whose bottom left corner is p_s . Thus the sum of the length of vertical segments JV can not exceed $\rho/\sqrt{2}$.

$$\frac{\xi}{H + V} \leq \frac{H}{V} \tag{13}$$

$$\xi \leq \frac{H^2 + V}{V} \tag{14}$$

$$\xi \leq \frac{\rho^2}{2(B+1)} + 1 \tag{15}$$

Placements of \mathcal{P} . Let us set the first position p_t such that the sub-path of Π between p_s and p_t has length exactly ξ . This point lies on the the J -th section of Π . Other points of \mathcal{P} are positioned on the subpath from p_s to p_t and on $[p_s w_0]$ if needed. The ℓ -eccentricity of the resulting (\mathcal{P}, s) must be ξ . This requires to carefully assign positions around a corner $\widehat{u_j v_j v_{j+1}}$ or $\widehat{v_j u_j u_{j+1}}$ in order to avoid any shortcut in the ℓ -disk graph between horizontal and vertical segment.

We start by placing a subset $\mathcal{P}_1 \subset \mathcal{P}/\{p_t\}$ of size $2(J-1)$ on the extremities of each segment except for u_0 where p_s lies and v_J (*resp.* u_J) if J is even (*resp.* odd). This ensure that no points are positioned beyond p_t . Remaining positions are placed such that (1) there is no point at distance strictly less than ℓ from a point of \mathcal{P}_1 and (2) the resulting (\mathcal{P}, s) has connectivity threshold ℓ .

Note that this requirement about the connectivity threshold constrains the eccentricity of (\mathcal{P}, s) as the number n may not be always large enough to cover Π . There must be at least ξ/ℓ positions of \mathcal{P} on Π and $\rho(1 - 1/\sqrt{2}) \leq \rho/3\ell$ positions on $[v_0 w_0]$. Therefore the number of positions n is such that $n \geq \xi/\ell + \rho/3\ell$. This is true whenever $\xi \leq n\ell - \rho/3$.

Lemma 14. *The construction of (\mathcal{P}, s) such that $\rho^* = \rho$, $\ell^* = \ell$ and $\xi_\ell = \xi$.*

Proof. Firstly we show $\rho^* = \rho$. Since the path Π is entirely contained in the square S_ρ , the farthest point from p_s to a point on Π is the up right corner of S_ρ . Its distance from p_s is equal to the diagonal of S_ρ which is exactly ρ . If no points of Π does not reach that corner, then we added the segment $[v_0 w_0]$ which ensures that there is at least one point at distance ρ from p_s .

We show that $\xi_\ell = \xi$. By Theorem 3, $B > \ell$, so the eccentricity of the ℓ -disk graph of \mathcal{P} is the maximum between the length of Π and the length of $[p_s, w_0]$ which is ρ . Since by assumption $\xi \geq \rho$, we get that the eccentricity of the ℓ -disk graph of \mathcal{P} is the length of Π . Let us compute the length of Π . By definition it is

$$J(H + V) = \left\lfloor \frac{\xi}{H + V} \right\rfloor (H + V) = \xi \quad (16)$$

By construction we directly have $\ell^* = \ell$. □

Let us prove Theorem 6.

Proof. Let us start by showing that the makespan of the instance (\mathcal{P}, s) constructed above is $\Omega(\xi)$. Any wake-up strategy can not be quicker than the eccentricity of s in the B -disk graph of (\mathcal{P}, s) . In the B -disk graph any path from p_s to p_t goes entirely through section. This is because for $j \in [0, J[$

any point of $[u_j v_j]$ is at distance $B + 1$ from any point of $[u_{j+1}, v_{j+1}]$, so the path goes through the points of the vertical segment $[u_j, u_{j+1}]$ or $[v_j, v_{j+1}]$ to reach $[u_{j+1}, v_{j+1}]$.

Such a path has a length longer than the sum of $|u_j v_j|$, that is $JH = \left\lfloor \frac{\xi}{H+V} \right\rfloor H$.

We have several cases depending the value of J :

- If $J = 0$, the ℓ -eccentricity of (\mathcal{P}, s) is $[p_s, w_0] = \rho$ and we have $\rho \leq \xi \leq H + V \leq 2\rho/\sqrt{2}$ therefore $\xi_\ell = \rho = \Theta(\xi)$
- If $J = 1$, then $\rho \leq \xi \leq 2(H + V) \leq 2\sqrt{2}\rho$ and $\xi_\ell = \Theta(\xi)$
- If $J \geq 2$, then $\xi \geq 2(H + V)$

$$\begin{aligned} \left\lfloor \frac{\xi}{H+V} \right\rfloor H &\geq \frac{\xi H}{H+V} - H \\ &\geq \frac{H(\xi - H - V)}{H+V} \geq \frac{\xi}{4} \end{aligned}$$

We conclude the proof by recalling that the lower bound with unconstrained energy naturally apply to dFTP with constrained energy (See Theorem 2), and by noticing that $\rho/\ell \leq \xi_\ell/\ell \leq 12\rho^2/\ell^2$. Therefore the makespan of \mathcal{A} is $\Theta(\xi_\ell + \ell^2 \log(\xi_\ell/\ell))$. \square

References

- [AAJ17] Zachary Abel, Hugo A. Akitaya, and Yu Jingjin. Freeze tag awakening in 2D is NP-hard. In *27th Annual Fall Workshop on Computational Geometry (FWCG)*, November 2017.
- [ABF⁺06] Esther M. Arkin, Michael A. Bender, Sándor P. Fekete, Joseph S.B. Mitchell, and Martin Skutella. The freeze-tag problem: How to wake up a swarm of robots. *Algorithmica*, 46:193–221, 2006.
- [ABG⁺03] Esther M. Arkin, Michael A. Bender, Dongdong Ge, Simai He, and Joseph S.B. Mitchell. Improved approximation algorithms for the freeze-tag problem. In *15th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 295–303. ACM Press, June 2003.
- [ABMS25] Sharareh Alipour, Kajal Baghestani, Mahdis Mirzaei, and Soroush Sahraei. Geometric freeze-tag problem. Technical Report 2412.19706v2 [cs.DC], arXiv, January 2025. To appear in AAMAS’25.

- [BCD⁺20] Andreas Bärtzchi, Jérémie Chalopin, Shantanu Das, Yann Disser, Barbara Geissmann, Daniel Graf, Arnaud Labourel, and Matúš Mihalák. Collaborative delivery with energy-constrained mobile robots. *Theoretical Computer Science*, 810:2–14, 2020. Special issue on Structural Information and Communication Complexity.
- [BCGH24] Nicolas Bonichon, Arnaud Casteigts, Cyril Gavoille, and Nicolas Hanusse. Freeze-tag in L_1 has wake-up time five with linear complexity. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing (DISC)*, volume 319 of *LIPICs*, pages 9:1–9:16, October 2024.
- [BDPP20] Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc, and Franck Petit. Deterministic treasure hunt in the plane with angular hints. *Algorithmica*, 82(11):3250–3281, 2020.
- [BGHO24] Nicolas Bonichon, Cyril Gavoille, Nicolas Hanusse, and Saeed Odak. Euclidean freeze-tag problem on plane. In *36th Canadian Conference on Computational Geometry (CCCG)*, pages 199–205, July 2024.
- [BW20] Josh Brunner and Julian Wellman. An optimal algorithm for online freeze-tag. In *10th International Conference Fun with Algorithms (FUN)*, volume 157 of *LIPICs*, pages 8:1–11, September 2020.
- [CGP20] Soumyottam Chatterjee, Robert Gmyr, and Gopal Pandurangan. Sleeping is efficient: Mis in $o(1)$ -rounds node-averaged awake complexity. In *Proceedings of the 39th Symposium on Principles of Distributed Computing, PODC '20*, page 99–108, New York, NY, USA, 2020. Association for Computing Machinery.
- [Das19] Shantanu Das. Graph explorations with mobile agents. In Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors, *Distributed Computing by Mobile Entities*, volume 11340 of *Lecture Notes in Computer Science*, chapter 16, pages 403–422. Springer, Cham, 2019.
- [DDU24] Shantanu Das, Dariusz Dereniowski, and Przemyslaw Uznanski. Energy constrained depth first search. *Algorithmica*, 86(12):3759–3782, 2024.
- [DKK06] Christian A. Duncan, Stephen G. Kobourov, and V. S. Anil Kumar. Optimal constrained graph exploration. *ACM Trans. Algorithms*, 2(3):380–402, July 2006.

- [DKS06] Mirosław Dynia, Mirosław Korzeniowski, and Christian Schindelhauer. Power-aware collective tree exploration. In Werner Grass, Bernhard Sick, and Klaus Waldschmidt, editors, *Architecture of Computing Systems - ARCS 2006, 19th International Conference, Frankfurt/Main, Germany, March 13-16, 2006, Proceedings*, volume 3894 of *Lecture Notes in Computer Science*, pages 341–351. Springer, 2006.
- [FGKP06] Pierre Fraigniaud, Leszek Gąsieniec, Dariusz R. Kowalski, and Andrzej Pelc. Collective tree exploration. *Networks*, 48(3):166–177, 2006.
- [FHG⁺16] G. Matthew Fricke, Joshua P. Hecker, Antonio D. Griego, Linh T. Tran, and Melanie E. Moses. A distributed deterministic spiral search algorithm for swarms. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016, Daejeon, South Korea, October 9-14, 2016*, pages 4430–4436. IEEE, 2016.
- [FKLS12] Ofer Feinerman, Amos Korman, Zvi Lotker, and Jean-Sébastien Sereni. Collaborative search on the plane without communication. In Darek Kowalski and Alessandro Panconesi, editors, *ACM Symposium on Principles of Distributed Computing, PODC '12, Funchal, Madeira, Portugal, July 16-18, 2012*, pages 77–86. ACM, 2012.
- [FPS19] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors. *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*. Springer, 2019.
- [HNP06] Mikael Hammar, Bengt J. Nilsson, and Mia Persson. The online freeze-tag problem. In *7th Latin American Symposium on Theoretical Informatics (LATIN)*, volume 3887 of *Lecture Notes in Computer Science*, pages 569–579. Springer, March 2006.
- [OS12] Christian Ortolf and Christian Schindelhauer. Online multi-robot exploration of grid graphs with rectangular obstacles. In *ACM Symposium on Parallelism in Algorithms and Architectures*, 2012.
- [YBMK15] Ehsan Najafi Yazdia, Alireza Bagheri, Zahra Moezkarimia, and Hamidreza Keshavarz. An $O(1)$ -approximation algorithm for the 2-dimensional geometric freeze-tag problem. *Information Processing Letters*, 115(6-8):618–622, June 2015.