

A unified approach for different tasks on rings in robot-based computing systems

Gianlorenzo D’Angelo^{*†}, Gabriele Di Stefano[‡], Alfredo Navarra^{*}, Nicolas Nisse[†] and Karol Suchan[§]

^{*}Dipartimento di Matematica e Informatica Università degli Studi di Perugia, Italy.

Email: alfredo.navarra@unipg.it, gianlorenzo.dangelo@dm.i.unipg.it

[†]COATI Project, INRIA/I3S(CNRS/UNSA), France. Email: nicolas.nisse@inria.fr

[‡]Dipartimento di Ingegneria e Scienze dell’Informazione e Matematica Università degli Studi dell’Aquila, Italy.

Email: gabriele.distefano@univaq.it

[§]Facultad de Ingeniería y Ciencias Universidad Adolfo Ibáñez, Chile. Email: karol.suchan@uai.cl

Abstract—A set of autonomous robots have to collaborate in order to accomplish a common task in a ring-topology where neither nodes nor edges are labeled. We present a unified approach to solve three important problems: the exclusive perpetual exploration, the exclusive perpetual search and the gathering problems. In the first problem, each robot aims at visiting each node infinitely often; in perpetual graph searching, the team of robots aims at clearing the whole network infinitely often; and in the gathering problem, all robots must eventually occupy the same node. We investigate these tasks in the Look-Compute-Move distributed computing model where the robots cannot communicate but can perceive the positions of other robots. Each robot is equipped with visibility sensors and motion actuators, and it operates in asynchronous cycles. In each cycle, a robot takes a snapshot of the current global configuration (Look), then, based on the perceived configuration, takes a decision to stay idle or to move to one of its adjacent nodes (Compute), and in the latter case it eventually moves to this neighbor (Move). Moreover, robots are endowed with very weak capabilities. Namely, they are anonymous, oblivious, uniform (execute the same algorithm) and have no common sense of orientation. In this setting, we devise algorithms that, starting from an exclusive rigid (i.e. aperiodic and asymmetric) configuration, solve the three above problems in anonymous ring-topologies.

I. INTRODUCTION

In the field of robot-based computing systems, we consider $k \geq 1$ robots placed on the nodes of an input graph. Robots are equipped with visibility sensors and motion actuators, and operate in *Look-Compute-Move* cycles in order to achieve a common task (see [1]).

The Look-Compute-Move model considers that in each cycle a robot takes a snapshot of the current global configuration (Look), then, based on the perceived configuration, takes a decision to stay idle or to move to one of its adjacent nodes (Compute), and in the latter case it moves to this neighbor (Move). Cycles are performed asynchronously, i.e., the time between Look, Compute, and Move operations is finite but unbounded, and it is decided by the adversary for each robot. Hence, robots that cannot communicate may move based on

outdated perceptions. From the practical viewpoint, the Look-Compute-Move model faithfully describes the behavior of some real robots.

In the continuous plane, this model is referred in the literature also as the *CORDA* model [2]. The inaccuracy of the sensors used by robots to scan the surrounding environment motivates its discretization. Moreover, robots can model software agents moving on a computer network.

Various problems have been studied in this setting such as *pattern formation* (in Euclidean metric spaces), *graph exploration with stop*, *exclusive perpetual exploration*, *exclusive perpetual graph searching* and *gathering*. Recently, several algorithms have been proposed to solve these problems in particular topologies such as lines, rings, trees and grids. Here, we propose a unified approach to solve the last three problems in rings. The relevance of the ring topology is motivated by its completely symmetrical structure. It means that algorithms for rings are more difficult to devise as they cannot exploit any topological structure, as all nodes look the same. In fact, our algorithms are only based on robots’ disposal and not on topology.

We consider a minimalist variant of the Look-Compute-Move model which has very weak hypothesis. Neither nodes nor edges of the graph are labeled and no local memory is available on nodes. Robots are *anonymous*, *uniform* (i.e. they all execute the same algorithm), *oblivious* (memoryless) and have no common sense of orientation. Guided by physical constraints, the robots may also satisfy the *exclusivity property*, according to which at most a node can be occupied by at most one robot [3]. In contrast to the *CORDA* model in the continuous plane, we assume that moves are instantaneous, and hence any robot performing a Look operation sees all other robots at nodes and not on edges. Note that, in a discrete asynchronous environment this does not constitute a limitation to the model. In fact, an algorithm cannot take advantages from seeing robots on the edges as the adversary can decide to perform the Look operations only when the robots are on the nodes. On the other hand, if an algorithm takes advantage from the assumption that the robots always occupy nodes, the same algorithm can be applied by adding the rule that if a robot sees another robot on an edge, it just don’t move (i.e. it

This work has been partially supported by Fondazione CARISPAQ (Italy) within project “ARISE” (Arising Robust Internetworked System for Emergency contexts). N. Nisse and K. Suchan are partially supported by Project ECOS-SUD Chile (Action ECOS C12E03) and the Inria Associated Team AIDyNet.

waits until all the robots occupy only nodes). In the following, we denote such model as the *discrete CORDA model*.

The discrete CORDA model received a lot of attention in the recent years. Most of the proposed algorithms consider that the starting configuration is *exclusive*, i.e., any node is occupied by at most one robot, and *rigid*, i.e., asymmetric and aperiodic. In the following, we review the literature concerning on graph topologies focusing on rings. For the literature about the three problems under study in different settings, the interest reader can refer to [4–10].

Related work. In the problem of graph exploration with stop [11–13], it is required that each node (or each edge) of the input graph is visited a finite number of times by at least one robot and, eventually, all the robots have to stop. Whereas, the exclusive perpetual graph exploration [3, 14–16] requires that each robot visits each node of the graph infinitely many times. Moreover, it adds the exclusivity constraint. In [15], first results on n -node rings are given. In detail, the paper gives algorithms for $k = 3$ and $n \geq 10$, for $k = n - 5$ (if $n \bmod k \neq 0$), and shows that the problem is infeasible for $k = 3$ and $n \leq 9$, and for some symmetric configurations where $k \geq n - 4$.

Graph searching has been widely studied in centralized [8] and distributed setting (e.g., [9]). The aim is to make the robots clear all the edges of a contaminated graph. An edge is cleared if it is traversed by a robot or if both its ends are occupied. However, a clear edge can be recontaminated if there is a path without robots from a contaminated edge to it. The study of graph searching in the discrete CORDA model when the exclusivity property must be always satisfied is introduced in [17] where a characterization of the perpetual exclusive graph searching on tree topologies is given. As far as we know, no results have been proposed in ring topologies for the perpetual exclusive graph searching problem in the discrete CORDA model.

The gathering problem consists in moving all the robots in the same node and remain there. In [18], a full characterization of the gathering on grid topologies without any multiplicity detection is given. On rings, it has been proven that the gathering is unsolvable if the robots are not empowered by the so-called *multiplicity detection* capability [19], either in its *global* or *local* version. In the former type, a robot is able to perceive whether any node of the graph is occupied by a single robot or more than one (i.e., a *multiplicity* occurs). In the latter type, a robot is able to perceive the multiplicity only if it is part of it. Using the global multiplicity detection capability, in [19], some impossibility results have been proven. Then, several algorithms have been proposed for different kinds of exclusive initial configurations in [19–21]. These papers left open some cases which have been closed in [22] where a unified strategy for all the gatherable configurations has been provided. With local multiplicity detection capability, an algorithm starting from rigid configurations where the number of robots k is strictly smaller than $\lfloor \frac{n}{2} \rfloor$ has been designed in [23]. In [24], the case where k is odd and strictly smaller than $n - 3$ has been solved. In [25], the authors provide an algorithm for the case

where n is odd, k is even, and $10 \leq k \leq n - 5$. Papers [24] and [25] do not assume that the initial configuration is rigid. The remaining cases with local multiplicity detection are left open and a the design of a unified algorithm for all the cases is still not known.

Contribution. In this work, we provide a unified approach for solving different tasks in the discrete CORDA model on ring topologies. Namely, we present an algorithm that, starting from any rigid configuration, solves: the exclusive perpetual exploration, the exclusive perpetual search, and the gathering with local multiplicity detection capability. Our main algorithms consist of two phases. The first phase is common to all problems and allows $k > 2$ robots to achieve a particular rigid exclusive configuration, denoted below by \mathcal{C}^* , in an n -node ring, $k < n - 2$. The second phase depends on the task. On the one hand, we design an algorithm that, starting from configuration \mathcal{C}^* , solves the gathering problem with local multiplicity detection for any team of k robots in n -node rings, $2 < k < n - 2$ (note that, if $n = 2$ or $k \geq n - 2$, no rigid configuration exists). On the other hand, we present an algorithm that, starting from configuration \mathcal{C}^* , solves both the perpetual exclusive graph exploration and the perpetual exclusive search problems, for any team of k robots in n -node rings, $n \geq 10$, $5 \leq k < n - 3$ (but for $k = 5$ and $n = 10$). Moreover, we design a specific algorithm that, starting from any rigid configuration, solves the perpetual exclusive graph searching problem using $n - 3$ robots in any n -node ring, $n \geq 10$. Finally, we provide some impossibility results for the perpetual exclusive graph searching problem, showing that for $2 < n \leq 9$ and $k < n$, or $k \in \{1, 2, 3, n - 2, n - 1\}$ and $n > 4$, the problem cannot be solved in a n -node ring with k robots. All together, we obtain an almost full characterization of exclusive perpetual graph searching in rings, leaving only open the cases ($k = 4, n > 9$) and ($k = 5, n = 10$). Moreover, for the exclusive perpetual exploration and for the gathering problem, besides being a unified approach, we solve some open cases.

Outline. In the next section we define the notation used in the paper and describe the discrete CORDA model. In Section III, we propose an algorithm to achieve the special configuration \mathcal{C}^* . Perpetual exclusive graph searching is formally defined and studied in Section IV. We note that the algorithms given in this section also solve the perpetual exclusive exploration problem. The gathering problem is considered in Section V. We then conclude with some possible future research directions. Due to space constraints, some of the proofs of the lemmata and theorems and the pseudo-codes of each algorithm are given in the full paper [26].

II. MODEL AND NOTATIONS

We consider a team of $k \geq 1$ robots spread in an n -node ring, $n \geq 3$. The ring is *anonymous*, that is its nodes and edges are undistinguishable, and no orientation is provided.

A *configuration* consists of the set of nodes that are occupied by a robot. Note that, it does not take into account the number of robots in each node. A configuration is said

exclusive if each node is occupied by at most one robot. For $2 \leq k < n - 2$, we denote by \mathcal{C}^* the configuration that consists of $k - 1$ consecutive occupied nodes, one empty node, one occupied node, and remaining ≥ 2 consecutive empty nodes. An *interval* in a configuration is an inclusion-maximal (possibly empty) subset of consecutive empty nodes, i.e., a subpath of empty nodes that stands between two occupied nodes. For instance, in \mathcal{C}^* , there are $k - 2$ intervals of length 0, one interval of length 1 and one interval of length $n - k - 1 > 1$.

In a configuration \mathcal{C} , a *view* at some occupied node $r \in \mathcal{C}$ is a sequence of integers $W(r) = (q_0, q_1, \dots, q_j)$, $j < k$, that represents the sequence of the lengths of the intervals met when traversing the ring in one direction (clockwise or anti-clockwise) starting from r . Abusing the notation, for any $i \leq j$, we refer to q_i as the corresponding interval. Note that, if \mathcal{C} is exclusive, then $j = k - 1$ and $\sum_{0 \leq i < j} q_i = n - k$. Note also that, a node r may have 2 distinct views, depending on the direction. Unless differently specified, we refer to $W(r) = (q_0, q_1, \dots, q_j)$ as the view at r that is minimum in the lexicographical order.

Let $W(\mathcal{C})$ be the set of the at most $2k$ views (at most two views per occupied node) in the configuration \mathcal{C} . The *supermin configuration view* $W_{min}^{\mathcal{C}}$ of the configuration \mathcal{C} is the minimal view in $W(\mathcal{C})$ in the lexicographical order. Note that, in $W_{min}^{\mathcal{C}}$, no interval has length strictly smaller than q_0 , and, moreover, if $k < n$, then $q_{k-1} > 0$. For instance, $W_{min}^{\mathcal{C}^*} = (q_0, \dots, q_{k-2}, q_{k-1})$ with $q_0 = \dots = q_{k-3} = 0$, $q_{k-2} = 1$ and $q_{k-1} = n - k - 1$.

For any view $W = (q_0, q_1, \dots, q_j)$ in a configuration \mathcal{C} , we set $\overline{W} = (q_0, q_j, q_{j-1}, \dots, q_1)$, and $W_i = (q_i, q_{(i+1) \bmod (j+1)}, \dots, q_{(i+j) \bmod (j+1)})$ denotes the view obtained by reading W starting from q_i as first interval. Note that $W(\mathcal{C}) = \{W_i, \overline{W}_i, \mid 0 \leq i \leq j\}$. Let $I_{\mathcal{C}}$ be the set of intervals q_i such that W_i or \overline{W}_i are equal to $W_{min}^{\mathcal{C}}$. The intervals in $I_{\mathcal{C}}$ are the *supermins* of \mathcal{C} . E.g., $|I_{\mathcal{C}^*}| = 1$.

An exclusive configuration is called *symmetric* if the ring admits a geometrical *axis of symmetry*, dividing the ring into two specular halves. An exclusive configuration is called *periodic* if it is invariable under non-trivial (i.e., non-complete) rotations. A configuration which is aperiodic and asymmetric is called *rigid*.

We now give some useful properties that are proved in [22]. In particular, Lemma 1 is used to detect possible symmetry or periodicity of a configuration.

Property 1 ([22]): Given a view W of a configuration \mathcal{C} , (i) there exists $0 < i \leq j$ such that $W = W_i$ iff \mathcal{C} is periodic; (ii) there exists $0 \leq i \leq j$ such that $W = \overline{W}_i$ iff \mathcal{C} is symmetric; (iii) \mathcal{C} is aperiodic and symmetric iff there exists one unique axis of symmetry.

It follows that if a configuration is rigid, then each occupied node has a view which is different from any other occupied node.

Lemma 1 ([22]): Given a configuration \mathcal{C} , (i) $|I_{\mathcal{C}}| = 1$ iff \mathcal{C} is either rigid or it admits only one axis of symmetry passing through the supermin; (ii) $|I_{\mathcal{C}}| = 2$ iff \mathcal{C} is either aperiodic and symmetric with the axis not passing through any supermin or

it is periodic with period $\frac{n}{2}$; (iii) $|I_{\mathcal{C}}| > 2$ iff \mathcal{C} is periodic, with period at most $\frac{n}{3}$.

We consider a discrete variant of the CORDA model introduced in [2] where the robots have no explicit way of communicate to each other (e.g., they cannot exchange messages). However, they are endowed with visibility sensors allowing each robot to perceive their own position in the graph and the positions of all the other robots.

The robots proceed by cycles of three phases *Look-Compute-Move*. In the Look-phase, a robot at some node r accesses a *snapshot* of the network that consists of the view $W(r)$. In the Compute-phase, the robot decides its action based on the information it received during the Look-phase. Finally, during the Move-phase, the robot executes its action, i.e., it moves to a neighboring node or stays idle. The environment is fully asynchronous which, in particular, means that the Compute-phase may be executed based on an outdated view of the network.

We consider a minimalist variant of the model, where the robots have very weak abilities. Robots are anonymous, i.e., they do not have identifiers, and are *uniform*, i.e., they all run the same algorithm. Robots are *oblivious* (memoryless). The robots have no *sense of direction*, i.e., they do not agree on a common orientation of the ring. Unless differently specified, two or more robots cannot occupy the same node (*exclusivity property*). When the exclusivity property is not imposed (e.g. for solving the gathering problem), the robots have the so called *local multiplicity detection* capability that is, a robot is able to detect whether the node where it resides is occupied by more than one robot or only by itself, but it is not able to detect the exact number of robots occupying the node. Note that this is the weakest assumption that has to be made to solve the gathering since it has been shown that the gathering is impossible if no multiplicity detection capability is allowed [19].

In contrast to the CORDA model in the continuous plane, we assume that moves are instantaneous, and hence any robot performing a Look operation sees all other robots at nodes and not on edges. We remark that, in a discrete asynchronous environment this does not constitute a limitation to the model. We call such model the *discrete CORDA model*.

Our goal is to investigate the feasibility of several collaborative tasks with these weak hypothesis. We assume that the starting configuration is rigid and exclusive.

III. REACHING CONFIGURATION \mathcal{C}^*

In this section, we propose an algorithm, called ALIGN, in the discrete CORDA model that allows to reach configuration \mathcal{C}^* starting from any exclusive rigid configuration. Algorithm ALIGN will be used in next sections to achieve the configurations suitable for the graph exploration, searching, or gathering problems. We first describe the algorithm allowing to reach configuration \mathcal{C}^* . In the second subsection, we prove its correctness.

A. Algorithm ALIGN

The assumption of initial rigidity and exclusivity ensures that one single robot moves at a time. The moves performed aim to reduce the unique supermin of a rigid configuration in a way that the obtained configuration is again rigid and exclusive, until configuration C^* is achieved.

By rigidity and exclusivity, the starting configuration has a unique supermin interval and each node has a unique supermin configuration view. Therefore, the snapshots provided to the robot allow to agree on a common view (the unique minimum one) where each robot can identify its position. This ensures that a single robot will move and that the next configuration is still exclusive. Given a configuration \mathcal{C} , four rules, called $\text{REDUCTION}_i(\mathcal{C})$, $i \in \{-1, 0, 1, 2\}$, are defined below where, for each rule, a single robot is asked to move to an empty node. $\text{REDUCTION}_0(\mathcal{C})$ is executed only if the supermin has length at least one. If the supermin has null length, $\text{REDUCTION}_1(\mathcal{C})$ is executed if the corresponding move does not create any symmetry. Otherwise, $\text{REDUCTION}_2(\mathcal{C})$ is executed if it does not create any symmetry, and $\text{REDUCTION}_{-1}(\mathcal{C})$ is executed otherwise. We prove that, starting from any rigid configuration, the move resulting from this algorithm achieves a new rigid configuration. The only exception is configuration C^s such that $W_{min}^{C^s} = (0, 1, 1, 2)$. In fact, from such a configuration, any single move would generate either a symmetric configuration or configuration C^s itself. In this case, we first perform $\text{REDUCTION}_1(C^s)$, obtaining the symmetric configuration \mathcal{C}' such that $W_{min}^{\mathcal{C}'} = (0, 0, 2, 2)$, then we perform $\text{REDUCTION}_1(\mathcal{C}')$ which leads to C^* . In any case, in the entire algorithm, only one robot is allowed to move at one time. Moreover, we prove that $\text{REDUCTION}_i(\mathcal{C})$, $i \in \{0, 1, 2\}$ strictly decreases the supermin. Finally, from some configuration \mathcal{C} , applying $\text{REDUCTION}_{-1}(\mathcal{C})$ may lead to a configuration \mathcal{C}' with a greater supermin configuration view. However, we prove that, in this case, the next move will reach a new configuration whose supermin configuration view is strictly smaller than the one of \mathcal{C} . Since, clearly, C^* is the rigid configuration with smallest supermin configuration view, this will prove that executing Algorithm ALIGN eventually achieves C^* .

We now formally define the four rules mentioned above. Let \mathcal{C} be any exclusive rigid configuration and let $W_{min}^{\mathcal{C}} = (q_0, q_1, \dots, q_{k-1})$ be its unique supermin configuration view. Let ℓ_1 be the smallest integer such that $q_{\ell_1} > 0$ and let ℓ_2 be the second smallest integer such that $q_{\ell_2} > 0$. That is, if $0 < \ell_1$ and $\ell_1 + 1 < \ell_2$, $W_{min}^{\mathcal{C}} = (0, \dots, 0, q_{\ell_1}, 0, \dots, 0, q_{\ell_2}, q_{\ell_2+1}, \dots, q_{k-1})$. Let a, b, c and d be the nodes between the intervals q_0 and q_{k-1} , q_{ℓ_1} and q_{ℓ_1+1} , q_{ℓ_2} and q_{ℓ_2+1} , and q_{k-2} and q_{k-1} respectively.

- **REDUCTION₀(\mathcal{C}):** The robot at a moves to its neighbor in the interval $q_0 > 0$. Then, the new configuration is $(q_0 - 1, q_1, \dots, q_{k-2}, q_{k-1} + 1)$;
- **REDUCTION₁(\mathcal{C}):** The robot at b moves to its neighbor in the interval $q_{\ell_1} > 0$. Then, the new configuration is $(q_0, q_1, \dots, q_{\ell_1-1}, q_{\ell_1} - 1, q_{\ell_1+1} + 1, \dots, q_{k-1})$;
- **REDUCTION₂(\mathcal{C}):** The robot at c moves to its neighbor

in the interval $q_{\ell_2} > 0$. Then, the new configuration is $(q_0, q_1, \dots, q_{\ell_2-1}, q_{\ell_2} - 1, q_{\ell_2+1} + 1, \dots, q_{k-1})$;

- **REDUCTION₋₁(\mathcal{C}):** The robot at d moves to its neighbor in the interval $q_{k-1} > 0$. Then, the new configuration is $(q_0, q_1, \dots, q_{k-2} + 1, q_{k-1} - 1)$.

It is clear from the definition of the rules that, from an exclusive rigid configuration, only one robot can execute a move and that the reached configuration is still exclusive. Note that, in the case that the configuration is C^s (i.e. $W_{min}^{C^s} = (0, 1, 1, 2)$), any REDUCTION move creates a symmetric configuration. In this case, we perform REDUCTION₁ which produces the symmetric configuration \mathcal{C} such that $W_{min}^{\mathcal{C}} = (0, 0, 2, 2)$. After this, REDUCTION₁ is again performed and it leads to C^* (i.e. $W_{min}^{C^*} = (0, 0, 1, 3)$). As \mathcal{C} is symmetric, the supermin configuration view can be obtained by reading the ring in both possible direction (i.e. $W_{min}^{\mathcal{C}} = (W_{min}^{\mathcal{C}})$). However robot b is unequivocally identified as the single robot on the axis of symmetry and REDUCTION₁ corresponds to moving b in an arbitrary direction. In any case C^* is achieved. In next subsection, we formally prove that C^* is eventually achieved and that, except for the case of C^s , the obtained intermediate configurations are always rigid.

B. Correctness

We consider a rigid exclusive configuration \mathcal{C} with unique (by Lemma 1) supermin configuration view $W_{min}^{\mathcal{C}} = (q_0, q_1, \dots, q_{k-1})$. We prove that, when one of the four rules is applied by Algorithm ALIGN, the resulting configuration \mathcal{C}' is still rigid. Moreover, in the case of the first three rules, the supermin configuration view of \mathcal{C}' is strictly smaller than $W_{min}^{\mathcal{C}}$. In the case of REDUCTION₋₁, we must consider the next move to strictly reduce the supermin configuration view.

Since $W_{min}^{\mathcal{C}} = (q_0, q_1, \dots, q_{k-1})$ is the supermin configuration view, no interval has length smaller than q_0 and $q_1 \leq q_{k-1}$. Therefore, if $q_0 > 0$ and REDUCTION₀ is applied, the view $(q_0 - 1, q_1, \dots, q_{k-2}, q_{k-1} + 1)$ is clearly the unique supermin configuration view of the resulting configuration \mathcal{C}' . By Lemma 1, we obtain:

Lemma 2 ([22]): The configuration \mathcal{C}' obtained by applying REDUCTION₀ in the rigid exclusive configuration \mathcal{C} with $q_0 > 0$ is rigid. Moreover, $W_{min}^{\mathcal{C}} > W_{min}^{\mathcal{C}'}$ (in lexicographical order).

Algorithm ALIGN performs REDUCTION₀ until it reaches a rigid exclusive configuration \mathcal{C} with supermin configuration view $W_{min}^{\mathcal{C}} = (0, q_1, \dots, q_{k-1})$ (i.e., $q_0 = 0$). In this case, REDUCTION₀ cannot be applied as otherwise there would be a collision. Therefore REDUCTION₁, REDUCTION₂ or REDUCTION₋₁ are applied depending on the configuration \mathcal{C} . In particular, REDUCTION₁ is applied if it does not create any symmetry. If $q_0 = 0$, by performing REDUCTION₁ we cannot obtain a symmetry except for some particular configurations given in the next lemma.

Lemma 3: Let \mathcal{C} be a rigid exclusive configuration with supermin configuration view $W_{min}^{\mathcal{C}} = (q_0, q_1, \dots, q_{k-1})$, $q_0 = 0$ and $\ell_1 > 0$ be the smallest integer such that $q_{\ell_1} > 0$. Then, the configuration \mathcal{C}' resulting from the application of

REDUCTION₁ is aperiodic. Moreover, C' is symmetric if and only if the following conditions hold:

$$q_i = 0, \text{ for each } i = 0, 1, \dots, \ell_1 - 1; \quad (1)$$

$$q_{\ell_1} = 1; \quad (2)$$

$$q_{\ell_1+1} + 1 = q_{k-1}; \quad (3)$$

$$\text{the sequence } q_{\ell_1+2}, q_{\ell_1+3}, \dots, q_{k-2} \text{ is symmetric.} \quad (4)$$

Proof: By rigidity of C , only one robot can perform REDUCTION₁ and then C' is well defined and admits a view $W = (q'_0, q'_1, \dots, q'_{k-1}) = (q_0, q_1, \dots, q_{\ell_1} - 1, q_{\ell_1+1} + 1, \dots, q_{k-1})$.

If C' is periodic, by Property 1, there must exist $j > 0$ such that $(q'_{j \bmod k}, q'_{(j+1) \bmod k}, \dots, q'_{(j+\ell_1) \bmod k}) = (q_0, q_1, \dots, q_{\ell_1} - 1) = (0, \dots, 0, q_{\ell_1} - 1)$. Note that, as $q'_{\ell_1+1} > 0$, then $j > \ell_1 + 1$. Hence, in that case, the view $(q_j, \dots, q_{k-1}, q_0, \dots, q_{j-1})$ is a view of C strictly smaller than W_{min}^C , a contradiction. Therefore, C' is aperiodic.

If equations 1–4 hold, then C' has a view $W = (0, \dots, 0, q_{\ell_1+1} + 1, q_{\ell_1+2}, q_{\ell_1+3}, \dots, q_{k-2}, q_{\ell_1+1} + 1)$ which is symmetric with the axis of symmetry passing through the middle of the sequences $q_0, q_1, \dots, q_{\ell_1} - 1$ and $q_{\ell_1+2}, q_{\ell_1+3}, \dots, q_{k-2}$.

We now show the only if statement. Note that Condition 1 is always satisfied by the hypothesis that $q_0 = 0$ and the definition of ℓ_1 . Let us assume that C' is symmetric and let $W = (q_0, q_1, \dots, q_{\ell_1-1}, q_{\ell_1} - 1, q_{\ell_1+1} + 1, \dots, q_{k-1}) = (0, 0, \dots, 0, q_{\ell_1} - 1, q_{\ell_1+1} + 1, \dots, q_{k-1})$.

For the sake of contradiction, let us assume that $q_{\ell_1} > 1$. Then, because $q_{\ell_1} \leq q_{k-1}$ and $q_{\ell_1} - 1 > 0$, it is easy to check that W is the supermin configuration view of C' , and $W < W_{min}^C$. Hence, q_0 must be the unique supermin of C' since otherwise, a supermin interval different from q_0 would have been a supermin interval in C , contradicting the fact that W_{min}^C is the supermin minimum view of C . By Lemma 1, since $|I_{C'}| = 1$ and C' is symmetric, the (unique) axis of symmetry of W passes through the edge corresponding to q_0 . However, since $q_{\ell_1} - 1 < q_{k-1}$, C' is not symmetric, a contradiction. It follows that $q_{\ell_1} = 1$. In this case, the first ℓ_1 elements of W are 0 and, as before, this sequence is unique and the possible axis of symmetry of C' passes through the middle of such unique sequence. This implies that C' is symmetric only if $q_{\ell_1+1} + 1 = q_{k-1}$ and that the sequence $q_{\ell_1+2}, q_{\ell_1+3}, \dots, q_{k-2}$ is symmetric. ■

It follows that if W_{min}^C does not satisfy Conditions 1–4, the application of REDUCTION₁ results in a rigid configuration. Otherwise, if applying REDUCTION₂ does not create any symmetry, it is applied. The following Lemma 4 shows that actually, when Conditions 1–4 hold, REDUCTION₂ can create symmetries only for some specific configurations.

For the next lemmata, we need further notation. A *pattern* is the set of possible configurations admitting a view that fulfills some rules defined by a string of integer numbers and the following symbols. Let x be an integer number: x^* denotes the repetition of x zero or more times; x^+ denotes the repetition of x one or more times; $x^{\{n\}}$ denotes the repetition

of x exactly n times. Given a configuration C we say that C belongs to a pattern P if it has a view W that matches the rules of the pattern. We denote it by $W \in P$. As an example, the configuration C with a view $(0, 0, 0, 1, \dots, 1, 2, 2, \dots, 2)$ belongs to $(0^{\{3\}}, 1^*, 2^+)$.

Lemma 4: Let C be a rigid exclusive configuration with supermin configuration view $W_{min}^C = (q_0, q_1, \dots, q_{k-1})$, with $3 \leq k < n - 2$, $q_0 = 0$, ℓ_1 be the smallest integer such that $q_{\ell_1} > 0$ and Conditions 1–4 hold. Then, the configuration C' resulting from the application of REDUCTION₂ is aperiodic. Moreover, C' is symmetric if and only if one of the following conditions hold:

$$W_{min}^C \in (0, 1, 1^+, 2); \quad (5)$$

$$W_{min}^C \in (0^{\{\ell_1\}}, 1, \{0^{\{\ell_1-1\}}, 1\}^+, 0^{\{\ell_1-2\}}, 1). \quad (6)$$

It follows that we can use REDUCTION₂ in all the configurations which satisfy Conditions 1–4 but not Conditions 5–6. The next lemma shows that in the remaining cases we can use REDUCTION₋₁, the resulting configuration being rigid.

Lemma 5: Let C be a rigid exclusive configuration with supermin configuration view W_{min}^C . If either $W_{min}^C \in (0, 1, 1, 1^+, 2)$ or $W_{min}^C \in (0^{\{\ell_1\}}, 1, \{0^{\{\ell_1-1\}}, 1\}^+, 0^{\{\ell_1-2\}}, 1)$, then, the configuration C' resulting from the application of REDUCTION₋₁ is rigid.

By the above lemma, it follows that if we apply REDUCTION₋₁ to a supermin configuration view W_{min}^C fulfilling Condition 5 or 6, the only case in which the obtained configuration can be symmetric is when $W_{min}^C = (0, 1, 1, 2)$. The correctness of ALIGN then follows from next theorem.

Theorem 1: Let $3 \leq k < n - 2$ robots standing in an n -node ring and forming a rigid exclusive configuration, Algorithm ALIGN eventually terminates achieving configuration C^* and all intermediate configurations obtained are exclusive and either rigid or such that the supermin view is $(0, 0, 2, 2)$.

Proof: As ALIGN starts from a rigid exclusive configuration, by Lemma 1, there exists a unique supermin in the initial configuration. Hence exactly one robot moves at one time. Moreover, all the performed movements reduce an interval which is strictly greater than 0 and hence the obtained configuration is exclusive.

First, we assume that the initial configuration is not C^s .

In a current rigid exclusive configuration C with unique supermin configuration view $W_{min}^C = (q_0, q_1, \dots, q_{k-1})$, we prove that the next move is unique and result in a rigid exclusive configuration.

If $q_0 > 0$, the algorithm performs REDUCTION₀. This involves a unique robot and the resulting configuration is rigid by Lemma 2.

If $q_0 = 0$, a unique robot executes REDUCTION₁ if the resulting configuration is rigid. Otherwise, by Lemma 3, W_{min}^C satisfies Conditions 1–4. In that case, a unique robot executes REDUCTION₂ if the resulting configuration is rigid. Otherwise, by Lemma 4, $W_{min}^C \in (0, 1, 1^+, 2)$ or $W_{min}^C \in (0^{\{\ell_1\}}, 1, \{0^{\{\ell_1-1\}}, 1\}^+, 0^{\{\ell_1-2\}}, 1)$. In this case, a unique robot executes REDUCTION₋₁. By Lemma 5, as the initial

configuration is different from \mathcal{C}^s , this results in a rigid configuration.

Since configuration \mathcal{C}^* is the configuration with the smallest supermin configuration view, it only remains to show that each movement reduces the supermin. Hence, in the following, we show that each movement (or each two movements) of ALIGN reduces the supermin.

Let us denote by $W = (q'_0, q'_1, \dots, q'_{k-1})$ the view of the configuration \mathcal{C}' obtained after the movement. W is the view of \mathcal{C}' at the same node and in the same direction as $W_{min}^{\mathcal{C}}$. Let $W_{min}^{\mathcal{C}'}$ be the supermin configuration view of \mathcal{C}' . If the movement is REDUCTION₀, then $q'_0 = q_0 - 1$ and hence $W_{min}^{\mathcal{C}'} \leq W < W_{min}^{\mathcal{C}}$. If the movement is REDUCTION _{i} , $i \in \{1, 2\}$ then $W = (q_0, q_1, \dots, q_{\ell_i} - 1, q_{\ell_i+1} + 1, \dots, q_{k-1}) < W_{min}^{\mathcal{C}}$ and therefore $W_{min}^{\mathcal{C}'} \leq W < W_{min}^{\mathcal{C}}$. If the movement is REDUCTION₋₁ it follows that $W_{min}^{\mathcal{C}'} \in (0, 1, 1, 1^+, 2)$ or $W_{min}^{\mathcal{C}'} \in (0^{\{\ell_1\}}, 1, \{0^{\{\ell_1-1\}}, 1\}^+, 0^{\{\ell_1-2\}}, 1)$. In the latter case, $W \in (0^{\{\ell_1+1\}}, 1, \{0^{\{\ell_1-1\}}, 1\}^+, 0^{\{\ell_1-3\}}, 1)$ and hence $W_{min}^{\mathcal{C}'} \leq W < W_{min}^{\mathcal{C}}$. In the former case, $W \in (0, 1, 1, 1^*, 2, 1)$ and hence $W > W_{min}^{\mathcal{C}}$. However, \mathcal{C}' is rigid and does not satisfy Conditions 1–4 and hence the movement performed in \mathcal{C}' is REDUCTION₁. Therefore, the configuration \mathcal{C}'' obtained after performing REDUCTION₁ on \mathcal{C}' is $W'' \in (0, 0, 2, 1^*, 2, 1)$. Therefore, $W'' < W_{min}^{\mathcal{C}}$.

Let us now assume that the initial configuration is \mathcal{C}^s . Note that, this is the only initial configuration with $k = 4$ and $n = 8$ which is rigid and different from \mathcal{C}^* . From \mathcal{C}^s , REDUCTION₁ is performed and the symmetric configuration \mathcal{C} such that $W_{min}^{\mathcal{C}} = (0, 0, 2, 2)$ is achieved. The next movement performed is again REDUCTION₁ which leads to \mathcal{C}^* (i.e. $W_{min}^{\mathcal{C}^*} = (0, 0, 1, 3)$) independently from the supermin view. In fact, even if configuration \mathcal{C} is symmetric, robot b is unequivocally identified as the single robot on the axis of symmetry and REDUCTION₁ corresponds to moving b in an arbitrary direction. In any case \mathcal{C}^* is achieved. ■

IV. CLEARING A RING

In this section, we study the exclusive perpetual graph searching problem of an n -node ring ($n \geq 3$) by a team of $1 \leq k \leq n$ robots in the discrete CORDA model, starting from any rigid exclusive configuration. In the case, $5 \leq k < n - 3$ and $n \geq 10$ (or $n > 10$ if $k = 5$), we propose an algorithm that makes use of Algorithm ALIGN presented in previous section. We then propose a specific algorithm for the case $k = n - 3$ and $n \geq 10$. On the other hand, we show that for $k \in \{1, 2, 3, n - 2, n - 1\}$ and $n > 3$, or for $2 < n \leq 9$ and $k < n$, there is no algorithm that solves the problem, even if the initial configuration is given. The cases $k = 4$ and ($k = 5, n = 10$) are left as open problems.

A. Perpetual exclusive graph searching

Given a n -node graph G where all edges are *contaminated*, the graph searching problem consists in coordinating a team of robots to eventually *clear* all edges. The robots occupy the nodes of G and a robot can move along an edge from its current position to a neighboring node. An edge is *cleared*

by a robot when it traverses it or if both its ends are simultaneously occupied by some robots. However, a clear edge is instantaneously *recontaminated* if there is a path from one of its end to the end of a contaminated edge and no node of this path is occupied by some robot. This variant of graph searching is classically referred as *mixed graph searching* [27]. Motivated by physical constraints and following [17], we moreover impose the *exclusivity constraint*, i.e., a node can be occupied by at most one robot.

A *search strategy* using $1 \leq k \leq n$ robots consists of a set of k nodes, the *initial positions*, and a sequence of moves of the robots, sliding the robots along the edges to empty neighbors, that eventually clear all edges. For instance, there is no search strategy that clears a n -node ring using one robot. On the other hand, a possible strategy using two robots is the following: first place two robots at adjacent nodes u and v , then slide the robot at u along the empty nodes of the ring until it reaches the other neighbor w of v .

In this section, we consider the graph searching problem in n -node rings in the discrete CORDA model. More precisely, we aim at designing algorithms that allow robots to clear a n -node ring starting from any rigid exclusive configuration. As our algorithms ensure that all met configurations are rigid and exclusive, and as the robots are oblivious of the cleared edges, the resulting strategies clear the ring *perpetually*, i.e., the whole ring is cleared infinitely often. Moreover, we study the exclusive perpetual exploration. Perpetual graph searching and perpetual exploration are not equivalent. For instance, one robot always moving clockwise will perpetually explore a ring without clearing it. On the other hand, the above search strategy using two robots perpetually clears a ring (one robot is at v and the other one alternate its move from u to w and then from w to u) but does not perpetually explore it since the robot at v never moves. The algorithms we propose in the sequel both perpetually explore and clear the rings.

B. Impossibility results

In this section, we show that for $k \in \{1, 2, 3, n - 2, n - 1\}$ or for $n \leq 9$, no algorithm in the discrete CORDA model allows to clear an n -node graph using k robots. For these results we do not assume that the initial configurations are rigid, that is the impossibility results hold on a stronger model. We start with a simple result.

Lemma 6: For any $n > 2$ and for any exclusive configuration \mathcal{C} , there is no algorithm that solves the exclusive perpetual graph searching problem in a n -node ring using $n - 1$ robots starting from \mathcal{C} .

Let us consider the case of two robots in a ring with at least three nodes. Two nodes u and v of an n -node ring are called *diametral* if either n is even and there are two shortest paths between u and v ; or n is odd and the length of the two paths from u to v differ by one. We say that two robots occupy a diametral configuration if they occupy two diametral nodes.

We show that any algorithm for perpetual searching with two robots needs to reach a configuration where the two robots occupy two diametral nodes. Then, we show that when the two

robots reach occupy two diametral nodes they cannot break the symmetry and hence they cannot search the ring. The next theorem follows.

Theorem 2: For any $n > 2$ and for any initial configuration \mathcal{C} , there is no algorithm that solves the exclusive perpetual graph searching problem in a n -node ring using $k \leq 2$ robots starting from \mathcal{C} .

Let us now consider the case of three robots in a ring with at least four nodes. For ease of presentation, we give identifiers to the robots. Of course, the robots are anonymous in the sense that they are not aware of these identifiers and that no algorithm for searching the ring can make use of them. However, the adversarial scheduler will use them. Hence, let us call the three robots by A, B and C . At any step s , we denote by $dist_s(X, Y)$ the distance (i.e., the number of consecutive edges) between the nodes occupied by robots X and Y at this step (if no ambiguity, the subscript will be omitted). Let \mathcal{C}_c be the configuration where the 3 robots occupy three consecutive nodes. Given any algorithm \mathcal{A} for perpetually clearing a ring with 3 robots, we say that a configuration \mathcal{C} is *bad* if, in this configuration, $dist(A, B) \leq dist(B, C)$ and there exists a robot such that, if this robot executes \mathcal{A} in configuration \mathcal{C} , then the configuration reached after its move is such that $dist(A, B) > dist(B, C)$.

In what follows, we show that any algorithm for perpetually clearing a ring with 3 robots must always avoid the configuration \mathcal{C}_c . Then, we show that such an algorithm cannot avoid to reach a bad configuration. Finally, we show that from any bad configuration, it is possible to schedule the three robots such that either they reach the configuration \mathcal{C}_c , or (1) each robot is scheduled at least once; and (2) this reaches a configuration such that $dist(A, B) \leq dist(B, C)$ and B has been adjacent to C in the meantime; and (3) if the new configuration is not \mathcal{C}_c , then from this new configuration, \mathcal{A} will reach another bad configuration before B is adjacent to C .

Since any algorithm for perpetually clear the ring must ensure that B is infinitely many times adjacent to C , this proves that such an algorithm cannot exist.

Theorem 3: For any $n > 3$ and for any initial configuration \mathcal{C} , there is no algorithm that solves the exclusive perpetual graph searching problem in a n -node ring using 3 robots starting from \mathcal{C} .

By using similar argument as Theorem 2 the next theorem can be shown.

Theorem 4: For any $n > 2$ and for any exclusive initial configuration \mathcal{C} , there is no algorithm that solves the exclusive perpetual graph searching problem in a n -node ring using $n-2$ robots starting from \mathcal{C} .

The next theorem is proven by an exhaustive study of the possible configurations.

Theorem 5: For any $2 \leq k < n \leq 9$ and for any initial configuration \mathcal{C} , there is no algorithm that solves the exclusive perpetual graph searching problem in a n -node using k robots starting from \mathcal{C} .

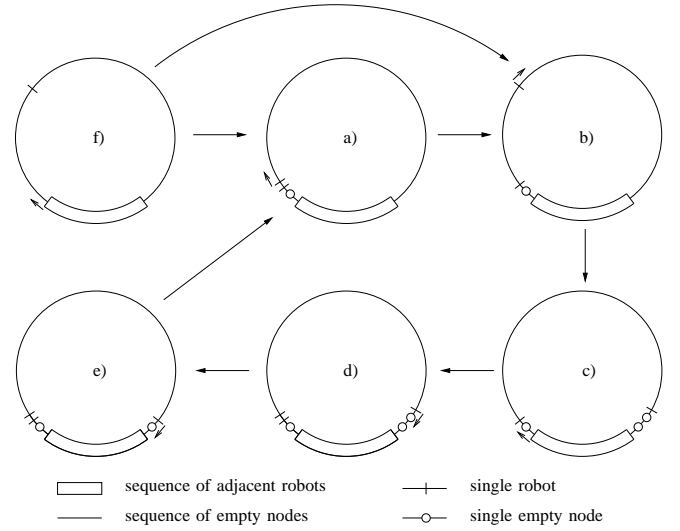


Fig. 1. Second phase of the RING SEARCH algorithm. The arrows close to the robots indicate the robot that is moving and its direction.

C. Algorithm RING SEARCH

In this section, we give an algorithm, called Algorithm RING SEARCH, to search a ring of $n \geq 10$ nodes with $5 \leq k < n-3$ robots (except for $n = 10$ and $k = 5$) starting from any rigid configuration.

Algorithm RING SEARCH works in two phases. In the first phase, Algorithm ALIGN is executed until one configuration in the set of configurations \mathcal{A} (described below and that contains \mathcal{C}^*) is reached. Then, the robots execute the algorithm illustrated in Fig. 1. The assumption of initial rigidity ensures that, in the entire algorithm, only one robot is allowed to move at one time. Moreover, the set of configurations in the two phases are disjoint and hence the robots can always distinguish which phase is performing.

We denote as \mathcal{A} the set of the following configurations:

- \mathcal{A} -a: Those with $k-2$ adjacent robots and two adjacent robots separated by one empty node from the first $k-2$ (Fig. 1a).
- \mathcal{A} -b: Those with $k-2$ adjacent robots, one robot separated by an empty node from the first $k-2$, and another robot not adjacent to any other one (Fig. 1b).
- \mathcal{A} -c: Those with $k-2$ adjacent robots, one robot separated by one empty node from the first $k-2$, and another robot separated by two empty nodes from the first $k-2$ on the other side of the first robot. (Fig. 1c).
- \mathcal{A} -d: Those with $k-3$ adjacent robots, two adjacent robots separated by one empty node from the first $k-3$ on one side, and another robot separated by two empty nodes from the first $k-3$ on the other side (Fig. 1d).
- \mathcal{A} -e: Those with $k-3$ adjacent robots, two adjacent robots separated by one empty node from the first $k-3$ on one side, and another robot separated by one empty node from the first $k-3$ on the other side (Fig. 1e).
- \mathcal{A} -f: Asymmetric configurations with $k-1$ adjacent robots and one single robot (Fig. 1f).

Note that the configuration \mathcal{C}^* belongs to the set \mathcal{A} -f.

The algorithm perpetually cycles among configurations \mathcal{A} -a — \mathcal{A} -e as depicted in Fig. 1. The next theorem shows that it perpetually clears the ring.

Theorem 6: Starting from any exclusive and rigid configuration, Algorithm RING SEARCH solves both the perpetual exclusive graph searching and exploration problems using k robots in any n -node ring, $n \geq 10$ and $5 \leq k < n - 3$ (but for $n = 10$ and $k = 5$).

Proof: By Theorem 1, Algorithm ALIGN eventually achieves configuration $\mathcal{C}^* \in \mathcal{A}$ -f. If the configuration is in \mathcal{A} -f, let us denote as r the single robot and by r' the robot on the border of the sequence of $k - 1$ robots which is the closest to r . Note that, as the initial configuration is assumed to be rigid, then we can always distinguish robot r' . The algorithm moves r' towards the only direction allowed. The obtained configuration is either \mathcal{A} -a or \mathcal{A} -b.

In the following, we show that if a configuration is in any of the configurations in \mathcal{A} , the algorithm perpetually cycles among them in the sequence (\mathcal{A} -a, \mathcal{A} -b, \mathcal{A} -c, \mathcal{A} -d, \mathcal{A} -e). Hence the algorithm never goes back to a configuration in \mathcal{A} -f and without loss of generality, we can assume that the first configuration is of type \mathcal{A} -a.

In this case, we call S the sequence of $k - 2$ adjacent robots and r and r' the robot at distance 3 and 2 from S , respectively. The algorithm identifies robot r in a configuration in \mathcal{A} -a. The view read by r is $(q_0, q_1, \dots, q_{k-1}) = (0, 1, 0, 0, \dots, 0, q_{k-1})$, where $q_{k-1} > 2$. In a configuration of type \mathcal{A} -a, the edges which are searched are the internal edges of S and the edge between r and r' . The algorithm first searches the edges in the sequence of empty nodes. To this aim, it moves robot r towards the direction opposite to r' . Note that as the configuration is rigid, only r can read such a configuration.

The configuration obtained is of type \mathcal{A} -b where the distance between the two single robots is 2. In particular, robot r can read the configuration in two directions, depending on the size of its adjacent intervals.

In this way, r searched the edge where it passed through. The algorithm keeps on moving r in the same direction until it reaches a configuration of type \mathcal{A} -c, in this way the configuration is still \mathcal{A} -b and all the edges between r and r' where r passed through are searched. Note that, the robots are always able to identify the correct direction thanks to the position of robot r' . More precisely, robot r can read the configuration in two directions, depending on the size of its adjacent intervals. If it reads in clockwise order with respect to Fig. 1, then the configuration read is $(q_0, q_1, \dots, q_{k-1}) = (q_0, 0, 0, \dots, 0, 1, q_{k-1})$, where $q_0 > 2$ and $q_{k-1} > 0$ and r has to move towards q_0 . Otherwise, the configuration read is $(q_0, q_1, \dots, q_{k-1}) = (q_0, 1, 0, 0, \dots, 0, q_{k-1})$, where $q_0 > 0$ and $q_{k-1} > 2$ and r has to move towards q_{k-1} .

When the configuration is of type \mathcal{A} -c, the only edges which are not searched are the two edges between r' and S and the three edges between r and S . Let r'' be the robot on the border of S which is the closest to r' . If the configuration is of type \mathcal{A} -c, the algorithm moves robot r'' towards r' , searching the

two edges between r' and $S \setminus \{r''\}$. The obtained configuration is of type \mathcal{A} -d, where the only edges which are not searched are those between r and S . Therefore, the algorithm moves r towards S obtaining first a configuration of type \mathcal{A} -e and again a configuration of type \mathcal{A} -a. Note that, in all the above configuration, it is always possible to distinguish robots r , r' and r'' . Moreover, the direction of the movements can be identified by the robots thanks to the position of robots r , r' and r'' themselves. Summarizing, the algorithm perpetually cycles among configurations \mathcal{A} -a — \mathcal{A} -e. ■

D. Clearing an n -node ring using $n - 3$ robots

In this section, we propose a specific algorithm to clear any n -node ring, $n \geq 10$ using $n - 3$ robots. Together with the previous algorithm and the impossibility results, this closes all the cases, but for $n = 10$ and $k = 5$.

In any exclusive configuration with $k = n - 3$ robots, all the nodes of the rings but three are occupied. In other words, the ring is made of at most three sequences of adjacent occupied nodes. We denote by A , B and C the number of nodes in such three sequences. If two empty nodes are adjacent, the corresponding sequence between them has size 0. Note that, as the configuration is rigid, such three sequences are all different and then, we can assume w.l.o.g. that $A < B < C$. In the following, we denote a configuration as (A, B, C) . We call *final* configurations the three configurations: $(0, 2, k - 2)$, $(0, 3, k - 3)$, and $(1, 2, k - 3)$. Note that, since $k = n - 3 \geq 7$, the final configurations are well defined and distinguishable, that is B is always strictly smaller than C . Our algorithm is denoted as NMINUSTHREE. and it works in two phases: In the first phase, it creates a final configuration and in the second one it performs the perpetual searching.

The first phase is performed if the configuration is not final and it is accomplished by performing the following rules in the priority given by the following ordering.

- R1.1:** If $A > 0$, move towards C the robot on the border of A which is closer to C ;
- R1.2:** If $B = 1$, move towards B the robot on the border of C which is closer to B ;
- R1.3:** If $B > 3$, move towards C the robot on the border of B which is closer to C .

Rule **R1.1** is executed for A steps until $A = 0$. Afterwards, either Rule **R1.2** or Rule **R1.3** is executed. If $A = 0$ and $B = 1$, then $C = k - 1$. It follows that, after one step of Rule **R1.2**, the final configuration $(0, 2, k - 2)$ is achieved. If $A = 0$ and $B > 3$, then the configuration is $(0, B, k - B)$ and the final configuration $(0, 3, k - 3)$ is achieved after $B - 3$ steps or Rule **R1.3**. If $A = 0$ and either $B = 2$ or $B = 3$, the configuration is final. The following lemma follows.

Lemma 7: The first phase of the algorithm eventually achieves a final configuration if $n \geq 10$ and $k = n - 3$. The second phase of the algorithm performs the searching. It starts from any final configuration and performs the following rules.

- R2.1:** If $(A, B, C) = (0, 2, k - 2)$, move towards B the robot on the border of C which is closer to B ;

- R2.2:** If $(A, B, C) = (0, 3, k - 3)$, move towards A the robot on the border of B which is closer to A ;
- R2.3:** If $(A, B, C) = (1, 2, k - 3)$, move the robot of A towards C .

The next theorem states the correctness of the algorithm.

Theorem 7: Starting from any exclusive and rigid configuration, Algorithm NMINUSTHREE solves both the perpetual exclusive graph searching and exploration problems using $n-3$ robots in any n -node ring, $n \geq 10$.

V. GATHERING IN A RING

In this section, we devise a strategy to accomplish the *gathering* task on a ring under the discrete CORDA model. The problem requires the robots to reach a common node and remain in there. Hence, more than one robot must be allowed to occupy a node, i.e. a *multiplicity* occurs. We assume that the robots have the *local* multiplicity detection capability. This is necessary as proven in [19].

In accordance to the other tasks previously shown, we make use of procedure ALIGN in order to achieve configuration C^* starting from any (exclusive) rigid configuration on rings of $n > k + 2$ nodes and $k > 2$ robots. In fact, any configuration with $n = 2$, $n = k + 1$, or $n = k + 2$ nodes is symmetric. Hence, the next algorithm provides a full characterization of rigid configurations where the gathering can be accomplished. Before providing the algorithm, we need some more notation.

A configuration is said to be of type C^* if it is composed by an ordered sequence of $j - 2$ intervals of length 0, one interval of length 1 and one interval of length $n - j - 1$, with $3 \leq j \leq k$. Consequently, also the nodes of the ring can be considered ordered according to the intervals' order. Hence, the first two nodes of the sequence will constitute interval $q_0 = 0$ in the current configuration. Clearly, C^* is a C^* -type configuration.

Rule CONTRACTION allows to move any robot occupying the first node of a C^* -type configuration towards the second one. Possibly, such nodes can be occupied by many robots.

From C^* -type configurations, the algorithm simply applies CONTRACTION until only two nodes are occupied. Note that, at each intermediate step, the current configuration is always a C^* -type, and the algorithm allows to move the robot(s) from the first node of the current interval q_0 towards the second one. Eventually, the number of intervals of length 0 is reduced by one. This is repeated until only two nodes at distance 2 remain occupied. Note that, in such a configuration, $k - 1$ robots are gathered on the same node and the other occupied node contains a single robot. From this configuration the robots can distinguish which is the node occupied by a single robot by using the local multiplicity detection. Therefore, only the single robot is allowed to move towards the other occupied node until joining it, while robots composing the multiplicity do not move. We can now state the next theorem.

Theorem 8: There exists an algorithm performing the gathering of $k > 2$ robots on rings of $n > k + 2$ nodes when the initial configuration is exclusive and rigid, and the robots are empowered with the local multiplicity detection.

VI. CONCLUSION

In this work, we provided a unified strategy for solving three tasks in the discrete CORDA model on ring topologies when the initial configuration is rigid. Namely we solved the exclusive perpetual search, the exclusive perpetual exploration and the gathering with local multiplicity detection capability. Moreover, the given algorithms solve some open problems and the impossibility results provided for the exclusive perpetual search problem fully characterize any initial configuration. Our work opens two main research direction: use the ALIGN algorithm to solve other problems in rigid configurations and devise similar algorithms to handle symmetric or periodic configurations.

BIBLIOGRAPHY

- [1] P. Flocchini, G. Prencipe, and N. Santoro, *Distributed Computing by oblivious mobile robots*. Morgan and Claypool, 2012.
- [2] G. Prencipe, "Impossibility of gathering by a set of autonomous mobile robots," *Theor. Comput. Sci.*, vol. 384, pp. 222–231, 2007.
- [3] R. Baldoni, F. Bonnet, A. Milani, and M. Raynal, "Anonymous graph exploration without collision by mobile robots," *Inf. Process. Lett.*, vol. 109, no. 2, pp. 98–103, 2008.
- [4] C. Ambühl, L. Gąsieniec, A. Pelc, T. Radzik, and X. Zhang, "Tree exploration with logarithmic memory," *ACM Trans. Algorithms*, vol. 7, no. 2, pp. 17:1–17:21, 2011.
- [5] E. Bampas, J. Czyzowicz, L. Gąsieniec, D. Ilcinkas, and A. Labourel, "Almost optimal asynchronous rendezvous in infinite multidimensional grids," in *24th Int. Symp. on Distributed Computing (DISC)*, ser. LNCS, vol. 6343, 2010, pp. 297–311.
- [6] R. Cohen, P. Fraigniaud, D. Ilcinkas, A. Korman, and D. Peleg, "Label-guided graph exploration by a finite automaton," *ACM Trans. Algorithms*, vol. 4, no. 4, pp. 42:1–42:18, Aug. 2008.
- [7] J. Czyzowicz, L. Gąsieniec, and A. Pelc, "Gathering few fat mobile robots in the plane," *Theoretical Computer Science*, vol. 410, no. 6–7, pp. 481 – 499, 2009.
- [8] F. V. Fomin and D. M. Thilikos, "An annotated bibliography on guaranteed graph searching," *Theor. Comput. Sci.*, vol. 399, no. 3, pp. 236–245, 2008.
- [9] D. Ilcinkas, N. Nisse, and D. Soguet, "The cost of monotonicity in distributed graph searching," *Distributed Computing*, vol. 22, no. 2, pp. 117–127, 2009.
- [10] A. Pelc, "Deterministic rendezvous in networks: A comprehensive survey," *Networks*, vol. 59, no. 3, pp. 331–347, 2012.
- [11] S. Devismes, F. Petit, and S. Tixeuil, "Optimal probabilistic ring exploration by semi-synchronous oblivious robots," in *16th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, ser. LNCS, vol. 5869, 2009, pp. 195–208.

- [12] P. Flocchini, D. Ilcinkas, A. Pelc, and N. Santoro, "Computing without communicating: Ring exploration by asynchronous oblivious robots," *Algorithmica*, to appear.
- [13] —, "Remembering without memory: Tree exploration by asynchronous oblivious robots," *Theor. Comput. Sci.*, vol. 411, no. 14-15, pp. 1583–1598, 2010.
- [14] R. Baldoni, F. Bonnet, A. Milani, and M. Raynal, "On the solvability of anonymous partial grids exploration by mobile robots," in *12th Int. Conf. On Principles Of Distributed Systems (OPODIS)*, ser. LNCS, vol. 5401. Springer-Verlag, 2008, pp. 428–445.
- [15] L. Blin, A. Milani, M. Potop-Butucaru, and S. Tixeuil, "Exclusive perpetual ring exploration without chirality," in *24th Int. Symp. on Distributed Computing (DISC)*, ser. LNCS, vol. 6343, 2010, pp. 312–327.
- [16] F. Bonnet, A. Milani, M. Potop-Butucaru, and S. Tixeuil, "Asynchronous exclusive perpetual grid exploration without sense of direction," in *15th Int. Conf. On Principles Of Distributed Systems (OPODIS)*, ser. LNCS, vol. 7109. Springer, 2011, pp. 251–265.
- [17] L. Blin, J. Burman, and N. Nisse, "Distributed exclusive and perpetual tree searching," in *26th Int. Symp. on Distributed Computing (DISC)*, ser. LNCS, vol. 7611, 2012, pp. 403–404.
- [18] G. D'Angelo, G. Di Stefano, R. Klasing, and A. Navarra, "Gathering of robots on anonymous grids without multiplicity detection," in *19th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, ser. LNCS, vol. 7355, 2012, pp. 327–338.
- [19] R. Klasing, E. Markou, and A. Pelc, "Gathering asynchronous oblivious mobile robots in a ring," *Theor. Comput. Sci.*, vol. 390, pp. 27–39, 2008.
- [20] G. D'Angelo, G. Di Stefano, and A. Navarra, "Gathering of six robots on anonymous symmetric rings," in *18th Int. Coll. on Structural Inf. and Com. Complexity (SIROCCO)*, ser. LNCS, vol. 6796, 2011, pp. 174–185.
- [21] R. Klasing, A. Kosowski, and A. Navarra, "Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring," *Theor. Comput. Sci.*, vol. 411, pp. 3235–3246, 2010.
- [22] G. D'Angelo, G. Di Stefano, and A. Navarra, "How to gather asynchronous oblivious robots on anonymous rings," in *26th Int. Symp. on Distributed Computing (DISC)*, ser. LNCS, vol. 7611, 2012, pp. 330–344.
- [23] T. Izumi, T. Izumi, S. Kamei, and F. Ooshita, "Mobile robots gathering algorithm with local weak multiplicity in rings," in *17th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, ser. LNCS, vol. 6058, 2010, pp. 101–113.
- [24] S. Kamei, A. Lamani, F. Ooshita, and S. Tixeuil, "Asynchronous mobile robot gathering from symmetric configurations," in *18th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, ser. LNCS, vol. 6796, 2011, pp. 150–161.
- [25] —, "Gathering an even number of robots in an odd ring without global multiplicity detection," in *37th Int. Symp. on Math. Foundations of Computer Science (MFCS)*, ser. LNCS, vol. 7464, 2012, pp. 542–553.
- [26] G. D'Angelo, G. Di Stefano, A. Navarra, N. Nisse, and K. Suchan, "A unified approach for different tasks on rings in robot-based computing systems," INRIA, Rapport de recherche RR-8013, 2012.
- [27] D. Bienstock and P. D. Seymour, "Monotonicity in graph searching," *J. Algorithms*, vol. 12, no. 2, pp. 239–245, 1991.