arXiv:2507.14723v1 [cs.DC] 19 Jul 2025

Simulating Chirality: Solving Distance-k-Dispersion on an 1-Interval Connected Ring

Brati Mondal
Department of Mathematics, Jadavpur University, Kolkata-700032, India

Pritam Goswami
Department of Computer Science, Sister Nivedita University, Kolkata-700156, India

Buddhadeb Sau
Department of Mathematics, Jadavpur University, Kolkata-700032, India

Abstract

We study the *Distance-k-Dispersion* (D-k-D) problem for synchronous mobile agents in a 1-interval connected ring network having n nodes and with l agents, where $3 \leq l \leq \lfloor \frac{n}{k} \rfloor$, without the assumption of *chirality* (a common sense of direction for the agents). This generalizes the classical dispersion problem by requiring that agents maintain a minimum distance of k hops from each other, with the special case $k = 1$ corresponding to standard dispersion.

The contribution in this work is threefold. Our first contribution is a novel method that enables agents to *simulate chirality* using only local information and bounded memory. This technique demonstrates that chirality is not a fundamental requirement for coordination in this model.

Building on this, our second contribution partially resolves an open question posed by Agarwalla et al. (ICDCN 2018), who considered the same model (1-interval connected rings, synchronous agents, no chirality). We prove that D-k-D—and thus dispersion is solvable from *arbitrary initial configurations* under these assumptions (excluding vertex permutation dynamism) for any size of the ring network which was earlier limited to only odd-sized ring or a ring of size four.

Finally, we present an algorithm for D-k-D in this setting that works in $O(ln)$ rounds, completing the constructive side of our result.

Altogether, our findings significantly extend the theoretical understanding of mobile agent coordination in dynamic networks and clarify the role of chirality in distributed computation.

2012 ACM Subject Classification Theory of computation → Distributed algorithms; Theory of computation → Design and analysis of algorithms

Keywords and phrases mobile agents, dispersion, D-k-D, dynamic ring, chirality

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding *Brati Mondal*: (Optional) author-specific funding acknowledgements
Pritam Goswami: [funding]
Buddhadeb Sau: [funding]

1 Introduction

In recent years, mobile agents in distributed networks have emerged as a significant research area for solving fundamental coordination problems. These agents are generally simple, autonomous, and low-cost mobile units that can collaboratively solve tasks such as *gathering*, *exploration*, *flocking*, and *dispersion*.

This paper focuses on the *dispersion* problem, first introduced in [2]. In this problem, l mobile agents are deployed on a graph with n nodes, and the goal of the problem is to make the agents reposition themselves autonomously so that each node contains at most $\lceil \frac{l}{n} \rceil$ agents. Note that, when $l \geq n$, dispersion ensures that each node holds at most one agent. Dispersion is closely related to other well-known problems such as *scattering* [3], *exploration* [6], *self-deployment* [7, 24], and *load balancing* [23], and has practical applications

© Author: Please provide a copyright holder;
licensed under Creative Commons License CC-BY 4.0
42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:31
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

in many real world scenarios such as relocating self-driving vehicles to charging stations, deploying robots in hazardous environments, and more.

We study a generalization of this problem known as *Distance-k-Dispersion* (D-k-D), introduced in [22]. The goal of this problem is to reposition agents such that the distance between any two agents on the graph is at least k. By definition, this problem inherits all practical applications of dispersion, while also broadening its scope. For instance, if each agent has a sensing radius of k hops, solving D-k-D helps maximize sensing coverage across the network by minimizing the overlapping covering area.

In this work, we focus on solving the D-k-D problem on a dynamic ring network, specifically under the 1-interval connectivity model (i.e., at any particular round at most one edge of the ring might stay missing), with $l \geq 3$ agents starting from any arbitrary initial deployment, and without assuming chirality (i.e., no common agreement on clockwise or counterclockwise direction). It is assumed that the agents are capable of local communication (i.e., agents can communicate only when they are on the same node) and have bounded memory.

1.1 Literature, Background, and Motivation

Over the last decade, the dispersion problem has been studied extensively across a range of models and network topologies [1, 2, 4, 5, 8, 9, 11, 10, 12, 13, 15, 14, 16, 17, 18, 19, 20, 21, 25, 26], owing to its practical relevance. Many real-world scenarios involving l agents and n spatially distributed resources can be modeled as a dispersion problem to minimize coordination time and system cost. For example, smart electric vehicles coordinating to reach available charging stations can be modeled as a dispersion task—driving to a station may take minutes, but recharging takes hours, so efficient coordination is crucial.

The Distance-k Dispersion problem generalizes dispersion and has its own set of real-world applications. For instance, in environments where each agent can sense or affect a region within k hops, D-k-D helps ensure maximum effective coverage by maintaining sufficient separation between agents.

In the context of ring networks, Agarwalla et al. [1] studied dispersion in dynamic rings under two key types of network dynamism:

- *1-interval connectivity*, where in any round, at most one edge (chosen arbitrarily by the adversary) may be missing.
- *Vertex permutation*, where the adversary can reorder nodes arbitrarily in each round.

Assuming local communication (i.e., agents can communicate only when co-located), the authors proposed several algorithms under different assumptions:

1. With chirality and full visibility, they presented *VP-1-Interval Chain* to solve dispersion.
2. In achiral settings where agents are co-located with full visibility, first they used *No-Chiral-Preprocess* to establish chirality, followed by *VP-1-Interval Chain* to achieve dispersion. This approach requires agents to store the smallest ID, thus requiring memory.
3. When agents are scattered, they proposed *Achiral-Odd-VP-1-Interval-Chain* and *Achiral-Even4-VP-1-Interval-Chain* to solve dispersion, but only for rings with an odd number of nodes or exactly four nodes, respectively. The latter also requires memory.

These results naturally raise the following question:

Is chirality necessary to solve dispersion on a 1-interval connected ring with $n = 2n'$ nodes for $n'(> 2) \in \mathbb{N}$, under the same model with memory as in [1]?

Our primary goal in this paper is to address this question by exploring the achiral setting in greater depth under the same assumptions as in [1] (excluding vertex permutation), while allowing agents to have memory. This investigation reveals important insights into the relative power of the model with and without chirality. Building on these findings, we further study the D-k-D problem in the same model, partially resolving the open question posed in [1] about solving dispersion from arbitrary configurations on even-sized 1-interval connected rings (excluding the vertex permutation dynamism). A summary of our results is presented in the following subsection.

1.2 Our Contribution

In this work, we study the problem of D-k-D on a 1-interval connected ring under the same bounded memory model described in [2]. Here, we consider the 1-interval connectivity dynamism only, while excluding the vertex permutation. This is because, in vertex permutation dynamism, D-k-D is not at all solvable if $k \geq 2$, as the adversary can always swap vertices to make the distance between two agents less than k. While our original objective was to solve the D-k-D problem from any arbitrary configuration under the bounded-memory model described in [1], excluding vertex permutation dynamism (hereafter referred to as $\mathcal{M}$), our investigation led to a deeper and unexpected insight:

The model $\mathcal{M}$ without chirality is computationally equivalent to $\mathcal{M}$ with chirality.

To formally establish this equivalence, we introduce a novel algorithm, ACHIRAL-2-CHIRAL, which enables a team of $l \geq 3$ agents, starting without any common sense of direction, to collectively establish chirality. The algorithm completes its execution within $O(ln)$ synchronous rounds, where l is the number of agents and n is the number of nodes of the ring. This result is significant, as it shows that agents can simulate the benefits of chirality using only local communication, full visibility, and bounded memory ($O(\log n)$ specifically), even in dynamic settings.

Building on this, we then address the original D-k-D problem. Once chirality is established, agents execute our second algorithm, DISPERSED to achieve a dispersed configuration and then the third algorithm, DISPERSED-TO-$k-$DISPERSED, which solves the D-k-D problem in dynamic rings and guarantees termination within $O(ln)$ rounds, where n is the size of the ring.

Together, these results not only settle the open question posed in [1] about solving dispersion (equivalent to solving D-k-D where $k = 1$) in even-sized dynamic rings without chirality, under bounded memory and full visibility partially (only for 1-interval connectivity dynamism), but also introduce a powerful framework for transforming achiral systems into chiral ones within the same model, without any additional assumptions. The following Table 1 provides a comparative overview of this work with previous literature.

2 Preliminaries

In this section, we discuss the preliminaries, including the model assumed for agents, the scheduler, and other relevant components, along with some necessary definitions, that will be used throughout the paper. In the following subsection, we first describe the agent model and the ring structure. In the subsequent sections, several necessary terms and the problem under consideration are formally defined.

2.1 Model and Problem Definition

Let $\mathcal{R} = (V_{\mathcal{R}}, E_{\mathcal{R}})$ be a connected network where $|v_{\mathcal{R}}| = n$ and $\forall\ v \in V_{\mathcal{R}}, deg(v) = 2$. We call $\mathcal{R}$ a ring network with n vertices. Let $A = \{r_1, r_2, \cdots r_l\}$ be a set of l agents arbitrarily

Aspect	**Agawalla et al. [1] (ICDCN, 2018)**	**This work**
Problem	**Dispersion** $(k=1)$	Distance-k-Dispersion for any k such that $\lfloor \frac{n}{l} \rfloor \geq k \geq 1$
Initial Chirality Agreement	No	No
Achieves Chirality Agreement in between	No	Yes
Dynamism of the ring	1-interval connectivity + vertex permutation	Only 1-interval connectivity; vertex permutation makes the problem impossible if $k \geq 2$
Initial Configuration	Any arbitrary configuration	Any arbitrary configuration
Size of ring (n)	n is either odd or four	n can be any natural number
Time Complexity	$O(n)$	$O(ln)$, where l is the number of agents

Table 1 Comparison with Agawalla et al. (ICDCN 2018)

deployed on the nodes of $\mathcal{R}$, where $l \geq 3$. Each of the agents has the same computational capabilities with a bounded persistent memory of $O(\log n)$ bits. Each of the agents has a unique ID in the range of $[1, n^c]$ for some constant c. We denote the ID of an agent r_i as $ID(r_i)$ and the $x-$th bit of the ID of r_i from right as $ID_x(r_i)$. The number of bits used to store ID for each agent is B. Thus $B \approx O(\log n)$. The agents are autonomous (i.e., no central control), identical (i.e., physically indistinguishable), and homogeneous (i.e., execute the same algorithm). One or more agents can occupy the same node at a particular time. If at a particular routine, more than one agent is at a node then we call that node a *multiplicity* at that time. Otherwise, if a node has only one agent at any particular round, that node is called *singleton* at that time. The agents do not have any chirality agreement but their clockwise direction is consistent throughout the execution.

The agents operate in a synchronous setting, where time is divided into rounds of equal duration. Each round consists of the following four stages:

1. **Edge Removal (by the adversary):** At the beginning of each round, the adversary may arbitrarily remove one edge from the ring $\mathcal{R}$, or leave the ring unchanged.
2. **Look:** Each agent, with *full visibility*, captures a snapshot of the entire ring, identifying singleton nodes, multiplicity nodes, unoccupied nodes, and the location of the missing edge, if any. It also accesses its memory and that of any co-located agents (Known as *Local Communication*). This enables the agent to determine the exact number of agents at its current node—an ability known as *local strong multiplicity detection.* For other nodes, however, the agent can only distinguish between singleton and multiplicity without knowing the exact agent count, referred to as *global weak multiplicity detection.*
3. **Compute:** Using the information collected during the Look phase, each agent runs the given algorithm to determine its next action.
4. **Move:** Based on the output of the Compute phase, each agent updates its memory and moves to an adjacent node, as specified by the algorithm. The move is assumed to be *instantaneous*, i.e., at any particular round, during the Look stage all agents are seen to be on nodes of the ring.

Dynamism of the ring $\mathcal{R}$: The ring $\mathcal{R}$ is an 1-interval connected ring. In a 1-interval connected ring at most one edge of $\mathcal{R}$ might stay missing in any particular round. We now proceed to define the problem formally in the subsection.

▶ **Definition 1** (**Problem Definition**: Distance-k-Dispersion)**.** *Let l agents ($3 \leq l \leq \lfloor \frac{n}{k} \rfloor$) reside on an 1-interval connected ring $\mathcal{R}$ with n nodes. Initially, the agents are placed arbitrarily on the nodes of the ring. The problem Distance-k-Dispersion (D-k-D) asks to reposition the agents to achieve a configuration that satisfies the following:*
(i) The occupied node contains exactly one agent.
(ii) The distance between any two occupied nodes is at least k.
(iii) There exists at least one pair of nodes whose distance is exactly k.

We need to define some necessary terms as preliminaries. The definitions are in the following subsection. Some diagrams (Figure 1 and Figure 2) are given for better visualization of the definitions.

2.2 Definitions and Preliminaries

▶ **Definition 2** (Configuration at time t)**.** *Configuration at time t or, $\mathcal{C}(t)$ is the 3-tuple $(\mathcal{R}, e_t, f_t)$ where $\mathcal{R}$ is the ring network on which agents are deployed, $e_t \in \{\varnothing\} \cup E_{\mathcal{R}}$ such that at t, edge e_t is missing in $\mathcal{R}$, and $f_t : V_{\mathcal{R}} \to \mathbb{N}$ be a function such that $f_t(v)$ denotes the number of agents on v at time t.*

In a configuration $\mathcal{C}(t)$, we call a vertex $v \in V_{\mathcal{R}}$ occupied if $f_t(v) \neq 0$. An occupied vertex v in a configuration $\mathcal{C}(t)$, is called *singleton* if $f_t(v) = 1$. v is called *multiplicity* if $f_t(v) > 1$. Also, in a configuration $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$, the vertex of agent r is denoted as $v_t(r)$. If the time t is clear from context, we use the symbol $v(r)$. If $f_t(v) \leq 1$ for all $v \in V_{\mathcal{R}}$ we call the configuration $\mathcal{C}(t)$ *dispersed*.

Henceforth, CW and CCW denote the clockwise and counterclockwise direction respectively. Let u and v be two vertices of a ring $\mathcal{R}$ and $\mathcal{D} \in \{CW, CCW\}$ be a direction. An *arc from u to v in direction $\mathcal{D}$, denoted by $(u,v)_{\mathcal{D}}$*, is the subgraph induced by the set of all vertices in between u and v starting from u in the direction $\mathcal{D}$ including u and v. We define the *distance from u to v in direction $\mathcal{D}$* as the length of the arc $(u,v)_{\mathcal{D}}$ and denote it as $d_{\mathcal{D}}(u,v)$.

▶ **Definition 3** (Adjacent occupied node pair)**.** *Let $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ be a configuration at some time t. Two nodes, u, v in $\mathcal{R}$ are said to form adjacent occupied node pairs in $\mathcal{C}(t)$ if there exists a direction $\mathcal{D} \in \{CW, CCW\}$ such that both of the following conditions hold in $\mathcal{C}(t)$.*

1. *$f_t(u), f_t(v) \neq 0$*
2. *$\forall\ w \in (u,v)_{\mathcal{D}}$ where w is not u or v, $f_t(w) = 0$*

▶ **Definition 4** (Chain)**.** *Let $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ be a configuration at time t. Let $(u,v)_{\mathcal{D}}$ be an arc for some $u, v \in V_{\mathcal{R}}$ and $\mathcal{D} \in \{CW, CCW\}$. The arc $(u,v)_{\mathcal{D}}$ is called a chain if all the following conditions hold in $\mathcal{C}(t)$.*

1. *$f_t(x) \neq 0\ \forall x \in (u,v)_{\mathcal{D}}$*
2. *If x_u and x_v be the adjacent nodes of u and v respectively such that $x_u, x_v \notin (u,v)_{\mathcal{D}}$ (if exists), then $f_t(x_u) = f_t(x_v) = 0$.*

We denote this chain as $C_{\mathcal{D}}(u,v)$

We call $(u,v)_{\mathcal{D}}$ *the arc of the chain $C_{\mathcal{D}}(u,v)$*. A chain $C_{\mathcal{D}}(u,v)$ is called an *$i-$chain* if $d_{\mathcal{D}}(u,v) = i$. Two chains $C_{\mathcal{D}}(u,v)$ and $C_{\mathcal{D}}(x,y)$ are called *adjacent* if at least one of the two arcs, $(v,x)_{\mathcal{D}}$ or, $(y,u)_{\mathcal{D}}$ has no occupied node.

In a configuration $\mathcal{C}(t)$, let r be an agent in a 0-chain for some t. If $f_t(v(r)) = 1$, then we call the chain a *singleton 0-chain*. Similarly if agents r_1 and r_2 are in same 1-chain in $\mathcal{C}(t)$, such that $v(r_1) \neq v(r_2)$ and $f_t(v(r_1)) = f_t(v(r_2)) = 1$ then, we call such 1-chains *Singleton 1-chain*. Let in an configuration $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$, $C_{\mathcal{D}}(u, v)$ be a chain for some $u, v \in V_{\mathcal{R}}$ and $\mathcal{D} \in \{CW, CCW\}$. Let u' and v' be nodes adjacent to u and v respectively such that $u', v' \notin (u, v)_{\mathcal{D}}$. We call the arc $(u', v')_{\mathcal{D}}$ the *extended arc of chain* $C_{\mathcal{D}}(u, v)$.

▶ **Definition 5** (Symmetric Configuration). *Let $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ be a configuration at time t. If $e_t \neq \varnothing$ then we call $\mathcal{C}(t)$ symmetric if the following condition is true in $\mathcal{C}(t)$*

- *$\exists$ an $i-$chain $C_{\mathcal{D}}(u, v)$ such that i is odd and $e_t = \overline{u'v'}$, $u', v' \in (u, v)_{\mathcal{D}}$ and $d_{\mathcal{D}}(u, u') = d_{\mathcal{D}}(v', v)$ where $d_{\mathcal{D}}(u, u') < d_{\mathcal{D}}(u, v')$*

▶ **Definition 6** (Asymmetric Configuration). *Let $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ be a configuration at time t. If $e_t \neq \varnothing$ then we call $\mathcal{C}(t)$ asymmetric if the following condition is true in $\mathcal{C}(t)$*

- *$\exists$ an $i-$chain $C_{\mathcal{D}}(u, v)$ whose extended arc be $(u_1, v_1)_{\mathcal{D}}$. Let for $e_t = \overline{u'v'}$, $u', v' \in (u_1, v_1)_{\mathcal{D}}$ and $d_{\mathcal{D}}(u_1, u') \neq d_{\mathcal{D}}(v', v_1)$ where $d_{\mathcal{D}}(u, u') < d_{\mathcal{D}}(u, v')$*

▶ **Definition 7** (visible direction of a chain). *Let $C_{\mathcal{D}}(u, v)$ be an $i-$chain in a configuration $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ where $i \geq 1$. We say that the chain $C_{\mathcal{D}}(u, v)$ has a visible direction if for exactly one of u and v, say w, $f_t(w) = 1$ and for the vertex, say w' other than w among u and v, $f_t(w') > 1$.*

If for a chain $C_{\mathcal{D}}(u, v)$ with visible direction in configuration $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$, $f_t(u) = 1$ then we say the direction of the chain is $\mathcal{D}$ otherwise if $f_t(u) > 1$ then the direction is $\mathcal{D}'$, where $\mathcal{D}'$ is the reverse direction of $\mathcal{D}$. The class of all $i-$chains where $i \geq 2$ and all visibly directed 1-chains is called the *feasible class* and is denoted as $\mathcal{FC}$. On the other hand, the class of 0-chains and 1-chains without visible direction is called the *non-feasible class* or $\mathcal{NFC}$.

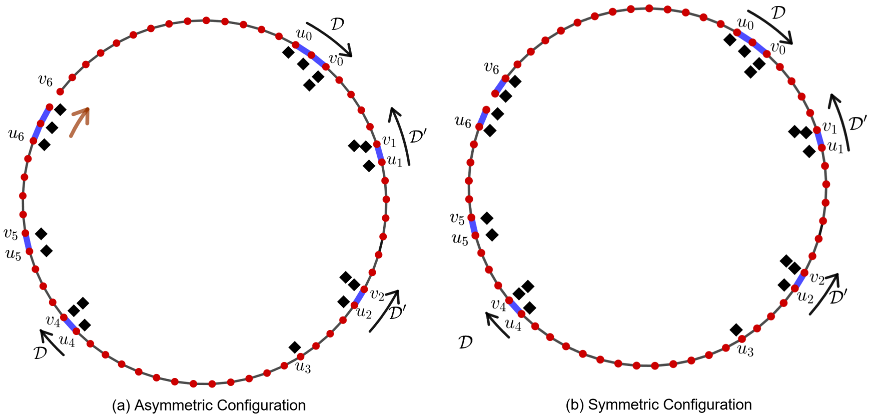

Figure 1 Here in this diagram arc of each chain are marked in blue. The visibly directed chains are marked with their corresponding direction. In each configuration the number of visibly directed chains in $\mathcal{D}$ and $\mathcal{D}'$ are same hence the configurations are not global. In the asymmetric configuration in left, the missing edge defines a global direction but in the symmetric configuration the missing edge is not defining any global direction.

▶ **Definition 8** (Global Configuration). *Let in a configuration $\mathcal{C}(t)$, there are p chains with visible direction such that among these p chains p_1 has a direction $\mathcal{D}$ for some $\mathcal{D} \in \{CW, CCW\}$ and rest $p_2 = p - p_1$ chains has direction $\mathcal{D}'$ where $\mathcal{D}'$ is the reverse direction of $\mathcal{D}$. If $p_1 \neq p_2$, then the configuration is referred to as a global configuration.*

In a global configuration, the agents without chirality can agree on a global direction. If $p_1 > p_2$ then the agents can agree on direction $\mathcal{D}$ as the global direction. Otherwise, if $p_2 > p_1$, then the agents agree on $\mathcal{D}'$ as the global direction.

▶ **Definition 9** (k-Link). *Let $\mathcal{C}(t) = (\mathcal{R}, e_t, f_t)$ be a dispersed configuration at time t. The agents $r_0, r_1, r_2, \ldots, r_p$ are said to form a k-Link, in clockwise (CW) direction, if for any $i \in \{0, 1, 2, \ldots, p-1\}$ $v(r_{i+1})$ is adjacent occupied node of $v(r_i)$ in direction CW and the distance $d_{CW}(v(r_i), v(r_{i+1})) < k$. We denote this k-Link by $(r_0, r_1, \ldots, r_p)_{CW}$.*

▶ **Definition 10** (Sub-k-Link of a k-Link). *Let there be two k-Links denoted as $KS_1 = (b_0, b_1, \ldots b_q)_{CW}$ and $KS_2 = (a_0, a_1, \ldots a_p)_{CW}$, we call KS_1 a sub-k-Link of KS_2 if both the following conditions hold.*

1. *$b_0 = a_i$ for some $i \in \{0, 1, \ldots, p\}$ and $b_q = a_j$ for some $j \in \{0, 1, \ldots p\}$ where $i \leq j$*
2. *$b_x = a_{i+x}$ for any $0 \leq x \leq p - i$*

We call a k-Link $KS = (a_0, a_1, \ldots a_p)_{CW}$ *Maximal* if KS is not a sub-k-Link of any other k-Link. From this point onward, whenever we mention k-Link, we will mean maximal k-Link unless otherwise specified. For a k-Link $KS = (r_0, r_1, \ldots, r_p)_{CW}$, the node $v(r_0)$ is called the *tail* and the node $v(r_p)$ is called the *head* of the k-Link KS and are denoted as $H(KS)$ and $T(KS)$ respectively. By $|KS|$ we define the number of agents in the k-Link

▶ **Definition 11** (Movable k-Link). *Let KS be a k-Link in some configuration $\mathcal{C}(t)$ for some round t. KS is called a movable k-Link if $d_{CW}(H(KS), T(KS')) > k$, where KS' is the next k-Link in the clockwise direction of KS.*

Let $KS = KS_0$ be a movable k-Link. The *Clockwise Nominee Set* and *Counter clockwise Nominee Set* of KS denoted by $\mathcal{NS}_{CW}(KS)$ and $\mathcal{NS}_{CCW}(KS)$ respectively are sets of agents defined as follows.

- If $|KS| \geq 2$, then $\mathcal{NS}_{CW}(KS) = \{r \in A : r = H(KS)\}$ and $\mathcal{NS}_{CCW}(KS) = A \setminus \mathcal{NS}_{CW}(KS)$
- Let us consider the case where $|KS| = 1$. Let all k-Links starting from $KS = KS_0$ in CCW direction be $KS_0, KS_1, \cdots, KS_\alpha$. Let KS_x be the first in the above order such that $|KS_x| \geq 2$. Now if $\nexists\, y \in \{1, 2, \cdots x\}$ such that KS_y is movable k-Link then $\mathcal{NS}_{CW}(KS) = \{r \in A : r = H(KS_j), 0 \leq j \leq x\}$ and $\mathcal{NS}_{CCW}(KS) = A \setminus \mathcal{NS}_{CW}(KS)$. Otherwise, let us consider z (where $y \leq z \leq x$) be such that KS_z be the last movable k-Link in the above mentioned order $\mathcal{NS}_{CW}(KS) = \{r \in A : r = H(KS_j), 0 \leq j \leq y-1\}$. Then, $\mathcal{NS}_{CCW}(KS) = A \setminus \{r \in A : r = H(KS_j), z \leq j \leq x\}$.

▶ **Definition 12** (Elected agent set in $\mathcal{C}(t)$). *Let $KS_0, KS_1, \cdots, KS_\alpha$ be all k-Links in $\mathcal{C}(t)$. Let $I = \{1, 2, \cdots \alpha\}$ be the index set. Now let us consider the set $S(t) = \{KS_i : KS_i$ is movable k-Link and $0 \leq i \leq \alpha\}$. Let $I' \subseteq I$ such that if $i \in I'$ then, $KS_i \in S(t)$. We define the elected agent set in $\mathcal{C}(t)$, (denoted as $\mathcal{EAS}(t)$) to be $\mathcal{EAS}(t) = \bigcup_{i \in I'} \mathcal{NS}_{CW}(KS_i)$*

3 Description of Algorithm D-k-D_Scattered_Dynamic_Ring

In this section, we describe the proposed algorithm D-k-D_SCATTERED_DYNAMIC_RING (Algorithm 1) that solves the Distance-k-Dispersion problem defined in Definition 1.

Initially, the agents do not have chirality. The algorithm works in three phases. In the first phase, the agents execute the algorithm ACHIRAL-2-CHIRAL until chirality is achieved. It is guaranteed that, within at most $O(ln)$ rounds, all agents will agree on chirality. This

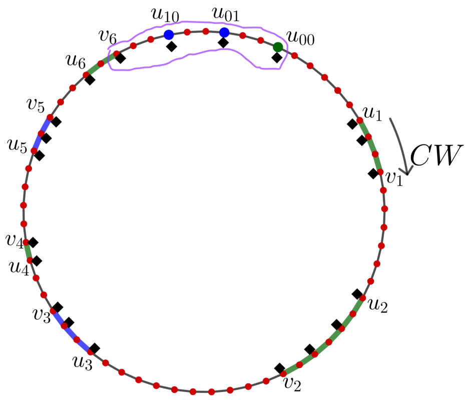

Figure 2 Assuming $k = 3$, we have the movable k-Links which are marked with green arcs and the k-Links with blue arcs are non-movable k-Links. Let KS be the k-Link with arc $(u_{00}, u_{00})_{CW}$, then $\mathcal{NS}_{CW}(KS)$ = agents in arc $(v_6, u_{00})_{CW}$ enclosed in the purple loop and $\mathcal{NS}_{CCW}(KS)$ = rest of the agents

novel algorithm can serve as a technique to simulate chirality in systems lacking an inherent sense of direction, under the assumptions of the considered model. After chirality is achieved, agents then execute Algorithm 11 (DISPERSED) to achieve dispersion in the second phase. The agents execute this algorithm until the dispersion is achieved. Note that, at this point, the open problem regarding solving dispersion in any 1-interval connected ring from any arbitrary configuration is solved. To solve D-k-D from this dispersed configuration, the agents execute Algorithm 12 (DISPERSED TO $k-$DISPERSED) until the problem is solved. The pseudocode of D-k-D_SCATTERED_DYNAMIC_RING is as follows.

Algorithm 1 D-k-D_SCATTERED_DYNAMIC RING(r,$\mathcal{C}(t_c)$)

```
if r has no global direction then
    execute ACHIRAL-2-CHIRAL
else
    if in $\mathcal{C}(t_c)$, $\exists$ multiplicity node then
        execute DISPERSED($r$,$\mathcal{C}(t_c)$)
    else
        execute DISPERSED TO $k$-DISPERSED($r$,$\mathcal{C}(t_c)$)
```

In the following three subsections, we describe these three phases described above.

3.1 Algorithm ACHIRAL-2-CHIRAL

In this subsection, we describe the high-level idea of the algorithm ACHIRAL-2-CHIRAL (Algorithm 2). The main aim of this algorithm is to ensure chirality agreement for the agents within at most $O(ln)$ rounds, where $l(\geq 3)$ is the number of agents and n is the number of nodes in the ring. This novel algorithm not only helps to solve D-k-D, but also works as an independent technique to achieve chirality from any achiral initial configuration under the concerned model. Thus, this algorithm proves that the concerned model $\mathcal{M}$ without chirality is computationally equivalent to $\mathcal{M}$ with chirality.

There are three stages of execution of this algorithm. In the first stage, the goal of this

Algorithm 2 Achiral-2-Chiral

```
if $\mathcal{C}(t_c)$ is asymmetric then
    align the direction of asymmetry with clockwise direction;
else
    if $\mathcal{C}(t_c)$ is global then
        align the clockwise direction with global direction and terminate;
    else
        if state= Init then
            execute Guider($r$, $\mathcal{C}(t_c)$);
        else if state= Oscillate then
            execute GetDirected($r$, $\mathcal{C}(t_c)$);
        else if state=PreMerge-I then
            execute PreProcess($r$, $\mathcal{C}(t_c)$);
        else if state=Merge-I then
            execute Merge-I($r$,$\mathcal{C}(t_c)$);
        else if state=Merge-II then
            execute Merge-II($r$,$\mathcal{C}(t_c)$);
        else if state= Roundabout then
            execute RoundTheRing-0($r$,$\mathcal{C}(t_c)$);
        else if state=PreRoundabout then
            execute RoundTheRing-I($r$,$\mathcal{C}(t_c)$);
        else if state=0-Merge-1 then
            execute Merge-0To1($r$,$\mathcal{C}(t_c)$)
```

algorithm is to ensure that the configuration becomes free of chains from the class $\mathcal{NFC}$. It is because the chains in the class $\mathcal{NFC}$ can't be made visibly directed, which is required to achieve a global direction agreement in the proposed algorithm. Next, the second and third stage repeats alternately until the chirality is achieved. In the second stage, all visibly undirected chains become visibly directed. So, after the completion of the second stage, all chains in the configuration become visibly directed. In this situation, if chirality is still not achieved, it means that the number of visibly directed chains in the clockwise direction is equal to the number in the counterclockwise direction. To resolve this, the agents proceed to the third stage, which ensures a reduction in the number of chains in the configuration. However, during this process, the newly formed chains may become visibly undirected. As a result, agents repeatedly alternate between Stage 2 and Stage 3 until the worst-case scenario, where there is exactly one visibly directed chain. In such a configuration, all agents agree on the direction of this chain as their global clockwise direction.

At any particular round, the agents can be in any one from the following set of states $\{\texttt{Init}, \texttt{Roundabout}, \texttt{PreRoundabout}, \texttt{PreMerge-I}, \texttt{Merge-I}, \texttt{Merge-II}, \texttt{Oscillate}, \texttt{0-Merge-1}\}$. In the initial configuration, all agents are in state `Init`. If in any particular round, the configuration is global or asymmetric, then the agents can easily achieve chirality as described in the preliminaries. Otherwise, according to the algorithm, in any particular round, an agent executes one of the eight procedures depending upon the state it is in at that round. In the pseudocode of the algorithm Achiral-2-Chiral, we mention which procedure agents execute in a particular round under a specific state. The pseudocodes of the Procedures (excluding Procedure Guider()) are not included in this part due to page limitations and can be found in the appendix.

3.1.1 Overview of the algorithm

Here we describe a high level idea of the algorithm Achiral-2-Chiral. In $\mathcal{C}(0)$ all l agents are placed arbitrarily in an 1-interval connected ring of size n. The agents do not have chirality and has a memory of $O(\log n)$. Initially the agents are in state `Init`. During state `Init`, an agent execute Procedure 3 (procedure Guider). This procedure guides the agents to different states depending on different conditions. The pseudo code is given below.

Procedure 3 Guider(r,$\mathcal{C}(t_c)$)

```
if $\mathcal{C}(t_c))$ has visibly directed chains then
    if all chains are in $\mathcal{FC}$ then
        if all chains are visibly directed then
            change state to Merge-II;
        else
            Set $round = 0$, $osc\_wait = 0$ and $ret = 0$;
            change state to Oscillate;
    else
        set $round = 0$;
        change state to PreMerge-I;
else
    if $\exists$ $i-chain$ where $i \leq 1$ then
        set $round = 0$;
        change state to PreMerge-I;
    else
        Set $round = 0$, $osc\_wait = 0$ and $ret = 0$;
        change state to Oscillate;
```

The main target of the algorithm is to create visibly directed chains. If visibly directed chains in direction $\mathcal{D}$ is greater than visibly directed chains in direction $\mathcal{D}'$ then all agents align their clockwise direction with $\mathcal{D}$.

Now, it is easy to make an visibly undirected i-chain visibly directed if $i \geq 2$ (by performing Procedure 9 (Procedure GetDirected()) which will be described in details later in this section). So the primary target of the algorithm is to eliminate all 0-chains and visibly undirected 1-chain by merging them with other chains. We describe the elimination of 0-chain and visibly undirected 1-chains as follows.

Let the configuration is neither global nor asymmetric and contains 0-chains or visibly undirected 1-chains, for this the agents changes state to `PreMerge-I` from `Init`(See Procedure 3). In this state agents execute Procedure 4 (Procedure PreProcess).

After the first two round of executing this procedure it is ensured that there does not exists any 0-chain with multiplicity and visibly undirected 1- chain where both the occupied nodes of it are multiplicities. In this scenario there can be four cases.

Case-I: All chains are singleton 0-chains. In this case agents changes state to `Roundabout`.

Case-II: All chains are singleton 1-chains and all occupied nodes are singleton. In this case agents changes state to `PreRoundabout`

Procedure 4 PREPROCESS(r, $\mathcal{C}(t_c)$)

```
if $round = 0$ then
    $round++$;
    if $r$ is an agent in a 0-chain then
        if $f_{t_c}(v(r)) > 1$ and $ID(r)$ is least on $v(r)$ then
            move in clockwise direction;
else if $round = 1$ then
    $round++$;
    if $r$ is in a 1-chain $\wedge$ both vertex of the chain are multiplicity then
        if $ID(r)$ is lowest in $v(r)$ then
            move outward of the chain;
else
    set $round = 0$;
    if In $\mathcal{C}(t_c)$ all chains are either Singleton 0-chains or, Singleton 1-chain then
        if In $\mathcal{C}(t_c)$ all chains are Singleton 0-chain then
            change state to Roundabout;
        else if In $\mathcal{C}(t_c)$ all chains are Singleton 1-chain then
            set $move = 0$;
            change state to PreRoundabout;
        else
            change state to 0-Merge-1;
    else
        change state to Merge-I;
```

Case-III: There exists 0-chain and 1-chains and all chains are either 0-chain or 1-chain such that each occupied nodes are singleton. In this case agents changes state to `0-Merge-1`
Case-IV: There exists an $i-$chain $i \geq 2$ or visibly directed 1-chain. In this case agents changes state to `Merge-I`.

Procedure 5 ROUNDTHERING-0(r,$\mathcal{C}(t_c)$)

```
if $round < n$ then
    $round++$;
    if $r$ is not a part of an 1-chain $\vee$ $f_{t_c}(v(r)) = 1$ then
        move clockwise;
else
    if all chains are Singleton 0-chain then
        align clockwise direction with global clockwise direction and terminate;
    else
        change state to Init;
```

In state `Roundabout` agents execute Procedure 5 (Procedure ROUNDTHERING-0) for n many rounds. To be specific agents move in their clockwise direction for n many rounds unless they merge with another 0-chain. If after n rounds all are still 0-chains then they have common clockwise sense so chirality is achieved else they move to a case similar to case II, III or, IV.

Procedure 6 RoundTheRing-I(r,$\mathcal{C}(t_c)$)

```
if $round < B$ then
    $round++$;
    if $\mathcal{C}(t_c)$ is symmetric then
        if $r$ is a part of the 1-chain with $e_t$ then
            change state to Init;
            move outwards ;
    else
        if $r$ is not part of a 0-chain such that $f_{t_c}(v(r)) > 1$ then
            if $ID_{round+1}(r) = 1$ then
                move inwards of the chain;
else if $round = B$ then
    $round++$;
    // In this point, all chains are 0- chains with 2 agents on each
       occupied node
    if $ID(r)$ is lowest in $v(r)$ then
        set $move = 1$;
        move in its clockwise direction;
else if $round = B + 1$ then
    $round++$;
    // In this point, all chains are 1-chains with all occupied nodes
       singleton and one agent has $move = 1$ the other has $move = 0$ in
       each 1-chain
    if $move = 0$ then
        align clockwise direction to the direction it sees the other agent in its chain;
        // At this point all agents in same chain agree in one
           particular direction
else if $B + 1 < round \leq B + 2n + 1$ then
    $round++$;
    if $\mathcal{C}(t_c)$ is symmetric and $e_t$ is in the chain of $r$ then
        wait;
    else
        if $r$ is not a part of an $i-$chain where $i \geq 2$ then
            move clockwise;
else
    if In $\mathcal{C}(t_c)$ all chains are Singleton $1-$chain then
        align clockwise direction with global clockwise direction and terminate;
    else
        change state to Init;
```

In state PreRoundabout, agents execute Procedure 6 (Procedure RoundTheRing-I) for $2n + B + 2$ rounds, where B is the bit-length of IDs. In the first $B + 2$ rounds, if no symmetric configuration arises, agents in the same 1-chain agree on a clockwise direction as follows:

In round t ($1 \leq t \leq B$), each agent checks the t-th bit of its ID (from the right). If it's 1, it moves inward within its chain; otherwise, it stays. If symmetry arises during this phase due to a missing edge $e_{t_c} = \overline{uv}$ in an 1-chain, agents on u and v move away, resulting in Case-III or Case-IV. On the other hand, if symmetry does not occur until the $B-$th round, all 1-chains reduce to 0-chains of two agents due to their unique Ids. If no symmetry occurs by round B, then in round $B+1$, the agent with the lowest ID in each 0- chain moves clockwise. In round $B + 2$, the other agent observes this movement and adopts the same direction, ensuring agreement within each chain. For the next $2n$ rounds, agents follow their clockwise direction unless their chain includes the missing edge or has merged into a larger chain. note that if after $2n$ rounds all of them are still singleton 1-chain then they agree on chirality as all of them has same clockwise direction. If one pair of chains has different clockwise sense then within $2n$ rounds, at least one chain moves n times, confirming formation of a larger i-chain ($i \geq 2$). In this case agents again change state to Init leading to Case-IV.

Procedure 7 Merge-0To1(r,$\mathcal{C}(t_c)$)

```
if 0-chain exists ∧ no chains are in $\mathcal{FC}$ then
    if $r$ is part of 0-chain and at least one adjacent chain is 1- chain then
        if exactly one nearest adjacent 1-chain then
            move in the direction of the 1-chain;
        else
            move clockwise;
else
    change state to Init;
```

In state 0-Merge-1, the agents execute Procedure 7 (Procedure Merge-0To1). This runs until the configuration contain at least one chain in $\mathcal{FC}$ or, the configuration contains no 0-chain. In this procedure an agent in a 0-chain moves in the direction of the nearest 1-chain if the 1-chain is adjacent to it. In case of tie (i.e agent in the 0-chain sees adjacent 1-chains in the same distance in both clockwise and counter clockwise direction) the agent moves in clockwise direction. This ensures after a finite time the configuration contains no 0-chains and there must exist at least one $i-$chain where $i \geq 2$ (Case-IV).

In state Merge-I, agents execute Procedure 8 (Procedure Merge-I) until all 0-chains and visibly undirected 1-chains (i.e., all chains in $\mathcal{NFC}$) are eliminated. Note that for all occupied nodes in all 0-chains and visibly undirected 1-chains in state Merge-I are singleton. Since at least one $i-$chain ($i \geq 2$) exists, there exists at least one 0-chain or undirected 1-chain which is adjacent to such a chain or a visibly directed 1-chain. If a 0-chain is adjacent to exactly one such chain in direction $\mathcal{D}$, its agent moves in direction $\mathcal{D}$; if adjacent on both sides, it moves clockwise. This way the 0-chains will get merged in $i-$chains where $i \geq 1$ and makes a larger chain.

Now, if a singleton 1-chain is adjacent to an $i-$chain ($i \geq 2$) or a visibly directed 1-chain, its agents move outward. This move ensures that at least one moving agent among these two either gets merged to a larger chain or makes a singleton 0-chain which will be further merged into a larger chain (using Procedure 8). This process ensures that, in finite time, all

Procedure 8 MERGE-I(r,$\mathcal{C}(t_c)$)

```
if ∃ chain in $\mathcal{NFC}$ then
    if $r$ is in a 0-chain ∧ $f_{t_c}(v(r)) = 1$ then
        if (exactly one of adjacent chain of $r$ is in $\mathcal{FC}$ in direction $\mathcal{D}$ then
            move in the direction $\mathcal{D}$
        else if Both adjacent chains of $r$ are in $\mathcal{FC}$ then
            move clockwise;
    else if $r$ is in a Singleton 1-chain $C_{\mathcal{D}}(u,v)$ then
        if both adjacent chains of the chain of $C_{\mathcal{D}}(u,v)$ are in $\mathcal{FC}$ then
            move outwards of the chain;
        else if Exactly one adjacent chain $C_{\mathcal{D}}(u',v')$ of $C_{\mathcal{D}}(u,v)$ in $\mathcal{FC}$ then
            if $r$ is at $v$ then
                move in direction $\mathcal{D}$;
else
    change state to Init;
```

0-chains and undirected 1-chains are eliminated.

At this point all agents are in state `Init` and there are no 0-chains or visibly undirected 1-chains. Now if the configuration at this time is not asymmetric, the agents check if the configuration is global or not. If the configuration is not global then number of visibly directed chains having one particular direction must be same with visibly directed chains in the reverse direction (note that the count can be 0 too). In this scenario if there are visibly undirected chains then agents changes state to `Oscillate` and otherwise if all chains are visibly directed they changes state to `Merge-II`.

In state `Oscillate`, the agents execute Procedure 9 (Procedure GETDIRECTED()). The main aim of this procedure is to make all chains visibly directed. This procedure is executed by each agent for $2B+2$ rounds. In the first round (i.e., $round = 0$) , the agents basically ensures that the terminal nodes of a visibly undirected chain becomes singleton. This is done by the agents located at a terminal node of a visibly undirected chain. If the node is not singleton then all agents except the agent having the lowest ID at that node move inwards the chain. For the next $2B$ rounds, in each odd round $2x+1$ ($x < B$), singleton agents at terminal nodes check the $(x+1)$-th bit of their ID. If it is 1, they move inward; otherwise, they stay. In each even round $2x$ ($x \le B$), agents that moved in the previous round check if the chain is now visibly directed. If so, they wait; otherwise, they move outward. Note that, after $2B+1$ execution of procedure GETDIRECTED(), all chains becomes visibly directed due to the fact that there must exist one $x < B$ such that $ID_{x+1}(r_1) \neq ID_{x+1}(r_2)$, for any two agents r_1 and r_2 at the singleton terminal nodes of the same chain. So, at the $2B+2$-th round all chains become visibly directed and all agents changes state to `Init`.

Note that as described earlier, if the configuration is not global still now, then the agents change their state to `Merge-II`. At state `Merge-II`, the agents execute the Procedure 10 (Procedure MERGE-II()). The main aim of this procedure is to ensure that number of chains decrease. Note that when all chains have visible direction and the configuration is not global then there must exist two adjacent chains whose directions are facing each other. Using this fact, in this procedure, two adjacent chains facing each other must move towards each other until they become a single chain. Note that if one such chain contain a missing edge in a symmetric configuration at some round then in that round no agent of that chain moves.

Procedure 9 GetDirected(r, $\mathcal{C}(t_c)$)

```
if $round = 0$ then
    $round{+}{+}$;
    if $v(r)$ is at terminal of an $i-$chain without visible direction and $i \geq 2$ then
        if $f_{t_c}(v(r)) > 1$ and $r$ is not the lowest ID agent on $v(r)$ then
            move inwards of the chain;
else if $1 \leq round \leq 2B$ then
    // $B$ is the ID length uniform for each agent as homogeneous
    $round{+}{+}$;
    if $v(r)$ is terminal node of an $i-$chain without visible direction where $i \geq 2$ then
        if $round - 1 = 2x + 1$ for some $x < B$ then
            if $osc\_wait = 0$ then
                if $ID_{x+1}(r)$=1 then
                    // $ID_x(r)$ is the $x-$th bit of the ID of $r$ from right
                    $ret = 1$;
                    move inwards of the chain;
        else
            if the chain containing $v(r)$ is visibly directed then
                $osc\_wait = 1$;
            else
                if $ret = 1$ then
                    $ret = 0$;
                    move outward of the chain;
else if $round > 2B$ then
    change state to Init
```

Since in each round one of the two adjacent chain can move they will eventually merge together. Note that this procedure is executed until there is no more adjacent visibly directed chains with their direction towards each other. At this stage the agents change state to `Init` again.

Due to merging, the number of chains decreases. However, merging two visibly directed chains results in a longer visibly undirected chain. If the configuration is still not global, agents are guided back to state `Oscillate` via Guider(), and re-execute GetDirected() to ensure visible direction for all chains. This process repeats until, in the worst case, only one visibly directed chain remains, making the configuration global. This way the agents can achieve chirality.

3.1.2 Correctness and Analysis of Algorithm Achiral-2-Chiral

Before proving the correctness of Algorithm Achiral-2-Chiral, let us first define the necessary notations for the proofs. We define $\mathcal{Z}_0(t)$ and $\mathcal{Z}_1(t)$ to be the number of 0-chains and visibly undirected 1-chains in $\mathcal{C}(t)$ respectively. We define $\Phi(t)$ as $\Phi(t) = \mathcal{Z}_0(t) + \mathcal{Z}_1(t)$. We denote the set of all chains in $\mathcal{C}(t)$ that are in $\mathcal{NFC}$ as $S_\Phi(t)$. and the set of rest of the chains in $\mathcal{C}(t)$ that are in $\mathcal{FC}$ as $S'_\Phi(t)$

Procedure 10 MERGE-II(r,$\mathcal{C}(t_c)$)

```
if ∃ at least a pair of adjacent visibly directed chains such that directions of them are towards each other then
    if $r$ is a part of such a chain and $e_{t_c}$ is not a part of chain of $r$ then
        move towards the direction of such adjacent chain;
else
    change state to Init;
```

▶ **Lemma 13.** *Let $\mathcal{C}(t_0)$ be a configuration where all agents are in state* `Merge-I` *and $\Phi(t_0) > 0$. Then, there exists a chain $C_{\mathcal{D}}(u, v) \in \mathcal{NFC}$ and a round $t_f > t_0$ such that in configuration $\mathcal{C}(t_f)$, all agents that were part of the chain $C_{\mathcal{D}}(u, v)$ have merged and become part of chains in the class $\mathcal{FC}$.*

Proof. Since in $\mathcal{C}(t_0)$, the agents are in state `Merge-I`, there must exist at least one chain from the class $\mathcal{FC}$. Also, it is given that at least one chain from class $\mathcal{NFC}$ exists in $\mathcal{C}(t_0)$. Then there must exists an adjacent pair of chains $C_{\mathcal{D}}(u_1, v_1) \in \mathcal{NFC}$ and $C_{\mathcal{D}}(u_2, v_2) \in \mathcal{FC}$. First note that, $C_{\mathcal{D}}(u_1, v_1)$ is either a singleton 0-chain or a singleton 1-chain as agents can have state `Merge-I` only after they execute the first two rounds of procedure 4, which ensures the claim. Now there can be two cases, either $C_{\mathcal{D}}(u_1, v_1)$ is singleton 0-chain or singleton 1-chain.

Case-I: In this case, let us consider $C_{\mathcal{D}}(u_1, v_1)$ is a 0-chain (in this case $u_1 = v_1$). Let $C_{\mathcal{D}}(u_2, v_2) \in \mathcal{FC}$ is adjacent to $C_{\mathcal{D}}(u_1, u_1)$. Let another adjacent chain of $C_{\mathcal{D}}(u_1, v_1)$ be $C_{\mathcal{D}'}(u_3, v_3)$ (if exists). Let us first consider the case where $C_{\mathcal{D}'}(u_3, v_3) \notin \mathcal{FC}$. In this case the agent on u_1 in $\mathcal{C}(t_0)$ moves in direction $\mathcal{D}$ (following Procedure 8) until it reaches the adjacent node of u_2, say u_2' that is in $(u_1, u_2)_{\mathcal{D}}$ at a round, say $t_1 > t_0$. Note that, the chain $C_{\mathcal{D}}(u_2, v_2)$ in $\mathcal{C}(t_0)$ changes to a 2-chain $C_{\mathcal{D}}(u_2', v_2)$ in $\mathcal{C}(t_1)$. So, for this case taking $t_f = t_1$ and $C_{\mathcal{D}}(u, v) = C_{\mathcal{D}}(u_1, v_1)$ suffices our purpose. Now, for the case where $C_{\mathcal{D}'}(u_3, v_3) \in \mathcal{FC}$, without loss of generality let us assume the agent in u_1 moves towards $C_{\mathcal{D}}(u_2, v_2)$. Now similarly the agent merges with $C_{\mathcal{D}}(u_2, v_2)$ at some time $t_2 > t_0$ creating the new 2-chain $C_{\mathcal{D}}(u_2', v_2)$ in $\mathcal{C}(t_2)$. For this case, assuming $t_f = t_2$ and $C_{\mathcal{D}}(u, v) = C_{\mathcal{D}}(u_1, v_1)$, suffices our purpose. Note, here the adversary can not stop the movements of the agent in u_1 without the agents agreeing on chirality. Because, stopping the agent to move from a singleton 0-chain by removing an edge would make the configuration asymmetric, which ensures that the agents will get an agreement on chirality.

Case-II: In this case, let us consider $C_{\mathcal{D}}(u_1, v_1)$ is a singleton 1-chain (in this case $d_{\mathcal{D}}(u_1, v_1) = 1$). Let $C_{\mathcal{D}}(u_2, v_2) \in \mathcal{FC}$ be adjacent to $C_{\mathcal{D}}(u_1, v_1)$. Let r_1 and r_2 be two agents at u_1 and v_1 respectively in $\mathcal{C}(t_0)$. Let $C_{\mathcal{D}'}(u_1', v_1')$ be another adjacent chain of $C_{\mathcal{D}}(u_1, v_1)$ in $\mathcal{C}(t_0)$. Now there can be two subcases.

1. $C_{\mathcal{D}'}(u_1', v_1') \notin \mathcal{FC}$ in $\mathcal{C}(t_0)$: Let $C_{\mathcal{D}'}(u_3, v_3)$ be the last chain from the class $\mathcal{NFC}$ in direction $\mathcal{D}'$ from chain $C_{\mathcal{D}}(u_1, v_1S)$ and $C_{\mathcal{D}'}(u_4, v_4) \in \mathcal{FC}$ be adjacent to $C_{\mathcal{D}'}(u_3, v_3)$ in direction $\mathcal{D}'$. If we can prove that all agents of at least one chain among $C_{\mathcal{D}}(u_1, v_1)$ and $C_{\mathcal{D}'}(u_3, v_3)$ in $\mathcal{C}(t_0)$, merges with chains among $C_{\mathcal{D}}(u_2, v_2)$ and $C_{\mathcal{D}'}(u_4, v_4)$ then we are done with this case. For the sake of contradiction, let us assume there exists agents, say r_1', and r_2' in chains $C_{\mathcal{D}}(u_1, v_1)$ and $C_{\mathcal{D}'}(u_3, v_3)$ respectively (in $\mathcal{C}(t_0)$) that do not merge with any of $C_{\mathcal{D}}(u_2, v_2)$ or $C_{\mathcal{D}'}(u_4, v_4)$. Note that r_1' and r_2', can not be at v_1 and v_3 in $\mathcal{C}(t_0)$ respectively. This is because, otherwise according to Procedure 8 agent r_1' and r_2' first move in direction $\mathcal{D}$ and $\mathcal{D}'$ respectively and forms singleton 0-chains adjacent to $C_{\mathcal{D}}(u_2, v_2)$ or, $C_{\mathcal{D}'}(u_4, v_4)$

respectively in $\mathcal{C}(t_0+1)$ (if not already merged with $C_{\mathcal{D}}(u_2,v_2)$ or $C_{\mathcal{D}'}(u_4,v_4)$). Again, these agents execute Procedure 8 and r_1' starts moving in direction $\mathcal{D}$ and r_2' in direction $\mathcal{D}'$. Without loss of generality let us assume that r_1' be the first agent that reaches u_2' before r_2' reaches u_4' where, u_2' is the vertex adjacent to u_2 in the arc $(v_1,u_2)_{\mathcal{D}}$ and u_4' is adjacent to u_4 in $(v_3,u_4)_{\mathcal{D}'}$. This implies r_1' merges with $C_{\mathcal{D}}(u_2,v_2)$ and becomes a part of the chain in $\mathcal{FC}$, which contradicts our assumption. So, now let r_1' and r_2' is at u_1 and u_3. As described in the procedure 8, there must exists a time $t_3>t_0$ when both agents those were on v_1 and v_3 merges with the chains $C_{\mathcal{D}}(u_2,v_2)$ and $C_{\mathcal{D}'}(u_4,v_4)$ respectively. So, in $\mathcal{C}(t_3)$, r_1' forms a singleton 0-chain that is adjacent to the chain $C_{\mathcal{D}}(u2',v_2)$ and r_2' also forms a singleton 0-chain that is adjacent to the chain $C_{\mathcal{D}'}(u_4',v_4)$ where, u_2' and u_4' are adjacent vertices of u_2 and u_4 respectively which are in the arcs $(v_1,u_2)_{\mathcal{D}}$ and $(v_3,u_4)_{\mathcal{D}'}$ respectively. Let without loss of generality, $d_{\mathcal{D}}(u_1,u_2')\le d_{\mathcal{D}'}(u_3,u_4')$. Then in $\mathcal{C}(t_3+d_{\mathcal{D}}(u_1,u_2')-1)$, r_1' reaches u_2'', where u_2'' is adjacent to u_2' in direction $\mathcal{D}'$ from u_2'. This implies r_1' merges with the chain $C_{\mathcal{D}}(u_2,v_2)$ and forms $C_{\mathcal{D}}(u_2'',v_2)\in\mathcal{FC}$. This contradicts our assumption. And so, the claim is true.

2. $C_{\mathcal{D}'}(u_1',v_1')\in\mathcal{FC}$ in $\mathcal{C}(t_0)$: First we consider the distances $d_{\mathcal{D}}(v_1,u_2)$ and $d_{\mathcal{D}'}(u_1,u_1')$. First let us assume that $d_{\mathcal{D}}(v_1,u_2),d_{\mathcal{D}'}(u_1,u_1')\ge 3$. In this case, while executing Procedure 8, r_1 and r_2 first move away from each other and create two 0-chains. After that r_1 moves towards u_1' in direction $\mathcal{D}'$ an r_2 moves in direction $\mathcal{D}$ until they reach u_2' and $(u_1')'$ respectively where u_2' is adjacent to u_2 in direction $\mathcal{D}'$ and $(u_1')'$ is adjacent to u_1' in direction $\mathcal{D}$. This way both agents of chain $C_{\mathcal{D}}(u_1,v_1)$ from configuration $\mathcal{C}(t_0)$ merges with chains among $C_{\mathcal{D}}(u_2,v_2)$ and $C_{\mathcal{D}'}(u_1.v_1')$ in $\mathcal{C}(t_1)$ where $t_1=t_0+\max\{d_{\mathcal{D}}(v_1,u_2'),d_{\mathcal{D}'}(u_1,(u_1')')\}$. Now let exactly one among $d_{\mathcal{D}}(v_1,u_2)$ and $d_{\mathcal{D}'}(u_1,u_1')$ be greater than 3 and the other is exactly 2. Without loss of generality, let $d_{\mathcal{D}}(v_1,u_2)\ge 3$ and $d_{\mathcal{D}'}(u_1,u_1')=2$. In this case, as described earlier in $\mathcal{C}(t_0)$ both r_1 and r_2 move in the opposite of each other. This move ensures that r_1 merges with $C_{\mathcal{D}'}(u_1',v_1')$ after moving in $\mathcal{C}(t_0)$ and forms a new chain $C_{\mathcal{D}'}((u_1')',v_1')\in\mathcal{FC}$ in $\mathcal{C}(t_0+1)$ and r_2 in $\mathcal{C}(t_0+1)$ creates a 0- chain which is adjacent to $C_{\mathcal{D}}(u_2,v_2)\in\mathcal{FC}$ in direction $\mathcal{D}$ and also to $C_{\mathcal{D}'}((u_1')',v_1')$ in direction $\mathcal{D}'$. Let without loss of generality according to Procedure 8, r_2 moves in Direction $\mathcal{D}$ in $\mathcal{C}(t_0+1)$. Now in $\mathcal{C}(t_0+d_{\mathcal{D}}(v_1,u_2))$, r_2 also merges with the chain $C_{\mathcal{D}}(u_2,v_2)$ from the configuration $\mathcal{C}(t_0)$ creating a new chain $C_{\mathcal{D}}(u_2',v_2)\in\mathcal{FC}$, where u_2' is adjacent to u_2 in direction $\mathcal{D}'$. So for these cases also all agents of chain $C_{\mathcal{D}}(u_1,v_1)$ from configuration $\mathcal{C}(t_0)$ merge and form chains in class $\mathcal{FC}$. Now if both $d_{\mathcal{D}}(v_1,u_2)=d_{\mathcal{D}'}(u_1,u_1')=2$, then in $\mathcal{C}(t_0+1)$, r_1 merges and creates chain $C_{\mathcal{D}'}((u_1')',v_1')\in\mathcal{FC}$ and r_2 merges and creates chain $C_{\mathcal{D}'}(u_2',v_2)\in\mathcal{FC}$. Note that, the adversary can not stop any move by any agent when all of them are in state `Merge-I` by removing edges without agents achieving chirality agreement. This is because, to do that adversary has to make the configuration asymmetric, and as a result of that, agents will achieve chirality.

So, for any configuration $\mathcal{C}(t_0)$ for any t_0, if agents are in state `Merge-I` and $\Phi(t_0)>0$ then we can see that there exists one chain from the class $\mathcal{NFC}$ in $\mathcal{C}(t_0)$ and $t_f>t_0$ such that all agents of that chain merges with other chains to form a chain in $\mathcal{FC}$ in $\mathcal{C}(t_f)$. Also, $t_f-t_0\approx O(n)$ for all of these cases. ◀

▶ **Corollary 14.** *Let in $\mathcal{C}(t_0)$ all agents are in state `Merge-I` and $\Phi(t_0)>0$. Then $\exists\ t_f>0$ such that in $\mathcal{C}(t_f)$, $\Phi(t_f)=0$ and agents are in state `Init`.*

Proof. Since agents are in state `Merge-I` and $\Phi(t_0)>0$, by Lemma 13, there exists a round, say t_1 within $O(n)$ rounds from t_0 such that $\Phi(t_1)\le\Phi(t_0)-1$. Now if $\Phi(t_1)=0$, we are done. Otherwise, in $\mathcal{C}(t_1)$, there must exist at least one chain in $\mathcal{NFC}$, and, so agents are still in state `Merge-I`. Now using this same argument recursively we can conclude that within $O(ln)$ (l being the number of agents) rounds from t_0, there exists a round t'_f such that $\Phi(t_f)$ becomes 0 and agents are still at state `Merge-I` in $\mathcal{C}(t'_f)$. Now, in $\mathcal{C}(t'_f + 1)$, since there are no chains in $\mathcal{NFC}$, agents change state to `Init`. So, taking $T_f = T'_f + 1$ is sufficient for the proof. ◀

▶ **Lemma 15.** *Let $\mathcal{C}(t_0)$ be a configuration in which all agents are in state `0-Merge-1`. Then, within finite rounds, there will be a time t_f such that exactly one of the following occurs.*

1. *all agents achieve chirality at some round t such that $t_0 \leq t \leq t_f$.*
2. *In $\mathcal{C}(t_f)$, $\Phi(t_f) = 0$ and all agents are in state `Init`.*

Proof. In $\mathcal{C}(t_0)$, all agents are in state `0-Merge-1`. Now, if $\mathcal{C}(t_0)$ does not contain any singleton 0-chain or, has at least one chain in $\mathcal{FC}$, then all agents change their state first to `Init` . Let at $\mathcal{C}(t_0 + 1)$, all chains be in $\mathcal{FC}$. For this case, taking $T_f = t_0 + 1$ suffices. On the other hand, if at $\mathcal{C}(t_0 + 1)$ there is at least one chain in $\mathcal{NFC}$, then within constant round the agents changes state to `Merge-I`and by Corollary 14, there exists a time $t_f \leq t_0 + O(ln)$ such that in $\mathcal{C}(t_f)$, either chirality is achieved, or $\Phi(t_f) = 0$ where all agents are in state `Init`, and we are done.
So, let us now assume that $\mathcal{C}(t_0)$ contains at least one 0-chain and no chains are in $\mathcal{FC}$. Also, as described in Lemma 13, an agent changes state to `0-Merge-1` at the third round of execution of Procedure 4. So, all chains of $\mathcal{C}(t_0)$ that are in $\mathcal{NFC}$ must be singleton chains. At this point, as the agents are in state `0-Merge-1`, they begin executing Procedure 7 (Procedure MERGE-0TO1()). Note that, in $\mathcal{C}(t_0)$ there must exists two adjacent chains one singleton 0-chain $Ch_1 = C_{\mathcal{D}}(u_1, u_1)$ and another singleton 1-chain $Ch_2 = C_{\mathcal{D}}(u_2, v_2)$. According to the procedure 7, agent in Ch_1, say r, moves in direction $\mathcal{D}$, towards Ch_2 until it reaches u'_2 at time say t_1 for some $t_1 > t_0$, where u'_2 is adjacent to u_2 in direction $\mathcal{D}$. Note that between t_0 and t_1, if the adversary tries to stop any move by some agent, the configuration becomes Asymmetric and agents agree on a chirality, and then we are done. So, let us assume chirality can never be achieved in finite time. In this case, within finite time (i.e., in $O(n)$ rounds), the agent r merges with Ch_2 from the configuration $\mathcal{C}(t_0)$ and together they form a new chain $C_{\mathcal{D}}(u'_2, v_2) \in \mathcal{FC}$. Now, as argued initially in this proof, within $O(1)$ rounds after t_1, there exists a time t'_1 when the agents change state to either `Merge-I`, or `Merge-II`, or `Oscillate` . In $\mathcal{C}(t'_1)$ agents were in state `Init` with $\Phi(t'_1) = 0$ if agents change state to `Merge-II`, or, `Oscillate`, at round t'_1. So, taking $t'_1 = t_f$ is sufficient for this case. Now in $\mathcal{C}(t'_1)$ if agents change state to `Merge-I`, then from Corollary 14 , within further $O(ln)$ rounds there exists a round t_2, such that either agents achieve chirality at round t_2 or, agents will have $\Phi(t_2) = 0$ and state `Init` in $\mathcal{C}(t_2)$. So, taking $t_f = t_2$ suffices for this case.

◀

▶ **Lemma 16.** *Let $\mathcal{C}(t_0)$ be the first configuration where all agents are in state `PreRoundabout` for some $t_0 > 0$. Then, exactly one of the following statements is true.*

1. *there is a $t_1 < t_0 + 2n + B + 1$ such that in $\mathcal{C}(t_1)$ all agents change state to `Init`, then in further constant rounds to one of the following states `0-Merge-1` or `Merge-I`, or `Oscillate`*
2. *in $\mathcal{C}(t_0 + 2n + B + 1)$, all agents achieve chirality.*
3. *in $\mathcal{C}(t_0 + 2n + B + 1)$, there exists a chain from the class $\mathcal{FC}$.*

Proof. Since the agents change state to `PreRoundabout`, in $\mathcal{C}(t_0)$, all chains are singleton 1-chains. Then the agents start to execute Procedure 6 for $2n + B + 2$ rounds. We first claim the following.

▷ Claim 17. Let all agents at $\mathcal{C}(t_0)$ be in state `PreRoundabout` for the first time. Then, exactly one of the following is true.

1. $\exists\ t_1 \leq t_0 + B$ such that in $\mathcal{C}(t_1)$ all agents changes state to `Init` and then in further $O(1)$ rounds change state to exactly one of `0-Merge-1`, `Merge-I` or, `Oscillate`.
2. in $\mathcal{C}(t_0 + B + 1)$ all chains are singleton 1-chain, and for any chain, both agents have the same chirality.

Proof of Claim: Note that during the first B execution of Procedure 6, in round $t_0 + x$ $(0 < x < B - 1)$, each agent checks the $x + 1$-th bit of their ID and if it is 1 it moves in the direction of the other agent in the chain otherwise it stays at the same node. Note that, if at some round $t_0 + x$ both have their $x + 1$-th bit 1, they cross each other and the 1-chain remains a 1-chain. This is also true if both of them have their $x + 1$th bit as 0. Note that, during this move, the adversary can interrupt the movements by removing one edge of a chain making the configuration symmetric. In this case, the agents, say r_1 and r_2, in the chain with a missing edge move outwards on that round, and all agents change state to `Init`. Note that since $l \geq 3$ in $\mathcal{C}(t_0)$, there are at least two chains. Now, suppose that after this move, agents r_1 and r_2 each create a singleton 0-chain. In this case, the agents currently in state `Init`, eventually transition to state `0-Merge-1` in 3 more rounds. On the other hand, if either r_1 or r_2 merges with another chain during the outward move and forms an i-chain with $i \geq 2$, then the agents transition to state `Merge-I` within 3 more rounds, provided that there are still chains remaining in the set $\mathcal{NFC}$. If no such chains remain, the agent eventually transitions to the state `Oscillate`in the next round. So, if we assume that, to interrupt the agent's move, the adversary removes an edge to make the configuration symmetric, then this leads to the first point of claim 1.

So, let us now assume that the adversary never removes any edge to interrupt the agent's move that would make the configuration symmetric. For this case, let us choose any singleton 1-chain arbitrarily. Let r_1 and r_2 be the agents on this chain. Let the ID of r_1 and r_2 first differs at the y-th bit from the right. So at round $t_0 + y - 1$, one of r_1 and r_2 moves towards the other, and the other remains. So, they form a 0-chain with two agents. Now, since all agents have different IDs and all of them differ at some bit $y \leq B$, at $\mathcal{C}(t_0 + B)$ all chains are 0-chains with two agents. Now, let us consider one such arbitrary 0-chain with two agents r_1 and r_2. At round $t_0 + B$, the lowest ID agent of them (W.L.O.G say r_1) moves in its clockwise direction, making each chain a singleton 1-chain again. Now, at round $t_0 + B + 1$ when r_2 sees that r_1 is one hop away in a particular direction, r_2 also agrees on the same clockwise direction as r_1. We call a singleton 1-chain *locally chiral*, if both the agents on the chain have the same clockwise direction. So, we proved that, if the adversary does not interrupt any move, all chains will be locally chiaral in $\mathcal{C}(t_0 + B + 1)$. So, we prove that point 2 of the claim is true if point 1 is not true. ◀

So, if not already changed to one of the following states i.e., `0-Merge-1`, `Merge-I`, `Oscillate`, from round $t_0 + B + 1$, all agents start to move in clockwise direction for $2n$ consecutive rounds if they are still part of 1-chains and no missing edge in their chain. We claim that if all agents do not have the same clockwise direction, then in this $2n$ round, there must exist one pair of chains that get merged and form a chain in $\mathcal{FC}$. Note that, since $l \geq 3$, there must be at least two chains in $\mathcal{C}(t_0)$, which assures the existence of at least two singleton 1-chains. Let us assume all agents do not have the same direction agreement in $\mathcal{C}(t_0 + B + 1)$. This implies there are two locally directed adjacent singleton 1-chains, say Ch_1 and Ch_2, whose clockwise direction faces each other. Now, the adversary can stop them from merging if both of these chains are forced not to move for at least n rounds each, as the distance between them can be at most n. But, if Ch_1 is stopped for n rounds, Ch_2 moves for those

n rounds, and vice versa. So, Ch_1 and Ch_2 merge and form one chain in $\mathcal{FC}$. So, it is evident that, if all agents do not have the same clockwise direction in $\mathcal{C}(t_0 + B + 1)$, then in $\mathcal{C}(t_0 + 2n + B + 1)$ at least one chain from the class $\mathcal{FC}$ exists. Otherwise, if all agents have the same direction in $\mathcal{C}(t_0 + B + 1)$ then the adversary can create both the following configurations as $\mathcal{C}(t_0 + 2n + B + 1)$. Either, all chains are still singleton 1-chain or, there exists at least one chain from $\mathcal{FC}$. So, if in $\mathcal{C}(t_0 + 2n + B + 1)$ all chains are still singleton 1-chain, we can conclude that agents have achieved chirality or, in $\mathcal{C}(t_0 + 2n + B + 1)$, there must exists a chain in $\mathcal{FC}$ and agents change state to `Init` in the next round (According to Procedure 6). This concludes the proof. ◀

▶ **Corollary 18.** *Let $\mathcal{C}(t_0)$ be the first configuration where all agents are in state `PreRoundabout`. Then in $\mathcal{C}(t_0)$, $\Phi(t_0) = \frac{l}{2}$ and within $O(ln)$ rounds, there exists a round t_f such that exactly one of the following is true.*

1. *all agents achieves chirality in $\mathcal{C}(t_f)$*
2. *In $\mathcal{C}(t_f)$, $\Phi(t_f) = 0$ and all agents are in state `Init`.*

Proof. Let $\mathcal{C}(t_0)$ be the first configuration when all agents are in state `PreRoundabout`. This implies that, in $\mathcal{C}(t_0)$, all chains are singleton 1-chains. Thus, each chain is from the class $\mathcal{NFC}$ and contains exactly two agents. So $\Phi(t_0) = \frac{l}{2}$. Now, from Lemma 16, one of the following three scenarios can occur.

1. there is a $t_1 < t_0 + 2n + B + 1$ such that in $\mathcal{C}(t_1)$ all agents change state to `Init` and in further constant rounds to one of the following states `0-Merge-1` or `Merge-I`, or `Oscillate`
2. in $\mathcal{C}(t_0 + 2n + B + 1)$, all agents achieve chirality.
3. in $\mathcal{C}(t_0 + 2n + B + 1)$, there exists a chain from the class $\mathcal{FC}$ and agents change state to `Init`.

Now, for case 2, we are already done. For Case 1, at $\mathcal{C}(t_1)$ agents are in state `Init`, $t_1 < t_0 + 2n + B + 1$. Now, if in $\mathcal{C}(t_1)$ there are no chains in $\mathcal{NFC}$. in this case agents change state to `Oscillate`in round t_1 from `Init`. So, taking $t_f = t_1$, we are done for this case. So let there be chains from class $\mathcal{NFC}$ in $\mathcal{C}(t_1)$. So, within $O(1)$ further rounds, there is a round t_2 such that at t_2, the agents change state to either `0-Merge-1` or `Merge-I`. Now if at $\mathcal{C}(t_2)$ agents are at state `Merge-I`, then by Corollary 14 within $O(ln)$ further rounds there exists a round t_f such that either chirality is achieved in $\mathcal{C}(t_f)$, or $\Phi(t_f) = 0$ with each agents being in state `Init`. Similarly, we can say the same for the case where in $\mathcal{C}(t_2)$ all agents change to state `0-Merge-1` from Lemma 15.

For Case 3, at $\mathcal{C}(t_0 + 2n + B + 1)$, there exists a chain from the class $\mathcal{FC}$. This implies that, in the next configurations all agents will have state `Init`(according to Procedure 6) and in constant further rounds there is a round t_3 such that in $\mathcal{C}(t_3)$ agents either change to state `Merge-I`, or `Oscillate` or `Merge-II`.

Now, if agents change state to either `Oscillate`, or `Merge-II` at t_3 then in $\mathcal{C}(t_3)$, $\Phi(t_3) = 0$ and all agents are in state `Init`. So, taking $t_f = t_3$ is sufficient for this case.

Now for the case where in t_3 agents change state to `Merge-I`, by Corollary 14, within further $O(ln)$ rounds there exists a round t_4 such that either agents achieve chirality at t_4 or, in $\mathcal{C}(t_4)$, $\Phi(t_4) = 0$ and all agents are in state `Init`. Now, taking $t_f = t_4$ suffices for this case. ◀

▶ **Lemma 19.** *Let $\mathcal{C}(t_0)$ be the first configuration when all agents are in state `Roundabout`, for some $t_0 > 0$. Then, exactly one of the following statements is true.*

1. *$\exists\ t_1 \leq t_0 + n$ such that at $\mathcal{C}(t_1)$ all agents achieve chirality.*

2. $\exists\ t_1 \approx t_0 + n + o(1)$ *such that at* $\mathcal{C}(t_1)$*, all agents change state to exactly one of the following.* `PreRoundabout`*,* `Merge-I`*,* `Oscillate`*, or,* `Merge-II`

Proof. Let all agents at $\mathcal{C}(t_0)$ be in state `Roundabout` for the first time. Then, all chains in $\mathcal{C}(t_0)$ are singleton 0-chains. Thus $\Phi(t_0) = l$. In this case, the agents start executing Procedure 5 for n consecutive rounds. In this procedure, an agent moves along its clockwise direction for n consecutive rounds unless within these n rounds it forms either a 0-chain with multiplicity, or an $i-$chain, where $i \geq 1$. Note that, during this move, the adversary can not stop the movement of an agent by removing an edge and without making the configuration asymmetric. And, agents can achieve chirality from an asymmetric configuration So, if the adversary stops movement by any agent at a round $t_1 < t_0 + n$, agents can achieve chirality in configuration $\mathcal{C}(t_1)$. Now, let the adversary not try to interrupt any agent by removing any edge. In this case, if all agents in $\mathcal{C}(t_0)$ have the same clockwise direction. Then, at $\mathcal{C}(t_0 + n)$, all agents will remain part of a singleton 0-chain. So, if this is the case, then at $\mathcal{C}(t_0 + n)$ agents can agree upon the chirality. Now, if at $\mathcal{C}(t_0)$, agents do not have the same clockwise direction, then there exist two adjacent agents and they move towards each other. Now, they either form a 0-chain with multiplicity or one 1-chain. Now, since $l \geq 3$, other agents may also merge in this chain and form a 0-chain with multiplicity or, visibly directed 1-chain, or, visibly undirected 1-chain, or, $i-$chain where $i \geq 2$. Note that, no singleton 0-chain would remain, as it would imply that a singleton 0-chain from $\mathcal{C}(t_0)$ completed n clockwise moves without meeting any other, which is not possible as agents have different clockwise directions. For this case, at $\mathcal{C}(t_0 + n)$ agents change state to `Init`.

Now, in the next round (i.e., $t_0 + n + 1$), if all chains are visibly directed, then either the configuration is global (in that case chirality is achieved), or they change state to `Merge-II`.

Now, if all chains in round $t_0 + n + 1$ are not visible directed, but all chains are from the class $\mathcal{FC}$, then either the configuration is global for which all agents achieve chirality or, they change state to `Oscillate`. For the above two cases, taking $t_1 = t_0 + n + 1$ is sufficient for the proof.

On the other hand, if at $t_0 + n + 1$, all chains are not from class $\mathcal{FC}$, then either they achieve chirality (if the configuration is global or asymmetric) or, they change to state `PreMerge-I`. If agents change state to `PreMerge-I`, then agents execute Procedure 4 for 3 consecutive rounds. In the first two rounds, all 0-chains with multiplicity become either a visibly directed 1-chain or, singleton 1-chain, and all 1-chains with multiplicity on both vertices become $i-$chain where $i \geq 2$. So, at the third round of executing Procedure 4 (i.e., at round $t_0 + n + 4$), if all chains are visibly directed then either chirality is achieved, or agents change state to `Merge-II`. On the other hand, if all chains are not visibly directed, but all are from $\mathcal{FC}$, then agents either achieve chirality (if the configuration is global), or they change state to `Oscillate`. Now there is further one case that can occur at round $t_0 + n + 4$, that is at this round, there are chains from the class $\mathcal{NFC}$, in this case, either agents achieve chirality (if configuration is global or, asymmetric)or, agents changes to states either `PreRoundabout` or, `Merge-I`. (state `0-Merge-1` does not occur as there can not be any 0-chain at $t_0 + n + 1$ as described above). So, for the above cases, taking $t_1 = t_0 + n + 4$ is sufficient. ◀

▶ **Corollary 20.** *Let* $\mathcal{C}(t_0)$ *be the first configuration when all agents are in state* `Roundabout`*, for some* $t_0 > 0$*. Then,* $\Phi(t_0) = l$ *and there exists a round* $t_f \leq t_0 + O(ln)$ *such that in* $\mathcal{C}(t_f)$ *exactly one of the following is true*

1. *all agents achieve chirality.*
2. $\Phi(t_f) = 0$ *and all agents are in state* `Init`*.*

Proof. From Lemma 19 it is clear that at $\mathcal{C}(t_0)$, $\Phi(t_0) = l$. Also from the same lemma we can conclude that $\exists\ t_1 < t_0 + O(n)$, such that at t_1 either chirality is achieved or, at t_1 agents are in one of the following states `PreRoundabout`, `Merge-I`, `Merge-II`, or, `Oscillate`.

If at t_1 agents change state to `Oscillate`, or `Merge-II`then we take $t_f = t_1$. In this case, $\Phi(t_f) = 0$. And, in $\mathcal{C}(t_f)$, all agents are in state `Init`. On the other hand, if at t_1 all agents are in state either `PreRoundabout`, or `Merge-I`, then by Corollary 14, and Corollary 18, within further $O(ln)$ rounds there will be a round t_2 such that at t_2 either agents achieve chirality or, in $\mathcal{C}(t_2)$, $\Phi(t_2) = 0$ with all agents in state `Init`. So, for this case we take $t_f = t_2$, which proves the result. ◀

▶ **Theorem 21.** *For any 1-interval connected ring with $l \geq 3$ agents executing* ACHIRAL-2-CHIRAL*, there exists a round $t_f < O(ln)$ such that in $\mathcal{C}(t_f)$, either chirality is achieved, or, $\Phi(t_f) = 0$ where all agents are in state* `Init`*.*

Proof. Let $\mathcal{C}(0)$ be any initial configuration with number of agents $l \geq 3$. Now in $\mathcal{C}(0)$ if all chains are in $\mathcal{FC}$ then $t_f = 0$, and we are done. So, let there exist chains from the class $\mathcal{NFC}$ in $\mathcal{C}(0)$. So, when agents execute Algorithm 2, they change state to state `PreMerge-I`. In state `PreMerge-I`, agents execute Procedure 4. The agents execute this process for three consecutive rounds. In the first two rounds, if the configuration is not global or asymmetric, then all 0-chains with multiplicity become either a visibly directed 1-chain or a singleton 1-chain. And all 1-chain, with multiplicity on both vertices, becomes $i-$chain, where $i \geq 2$. Otherwise, chirality will be achieved, and we are done. So, let the configuration not be global or asymmetric in the first two rounds of executing Procedure 4. Thus, at the beginning of the third round of executing Procedure 4, say t_1, there remains no 0-chain with multiplicity or, 1-chain that has multiplicity on both the vertices. So, in $\mathcal{C}(t_1)$ the chains can be either a singleton 0-chain, a singleton 1-chain, or any other chains from the class $\mathcal{FC}$. Now $\mathcal{C}(t_1)$ can be exactly one of the following configurations,

- All chains are singleton 0-chain in $\mathcal{C}(t_1)$. For this case, agents change state to state `Roundabout`.
- All chains are singleton 1-chain in $\mathcal{C}(t_1)$. For this state, agents change state to `PreRoundabout`.
- $\exists$ a singleton 0-chain and a singleton 1-chain and $\nexists$ any chain from $\mathcal{FC}$. For this case, agents change state to `0-Merge-1`.
- $\exists$ a chain from $\mathcal{FC}$. For this case, agents change state to `Merge-I`.

So, at $t_1 + 1$, agents can be at any one of the following states, `Roundabout`, `PreRoundabout`, `0-Merge-1`, `Merge-I`. Now, by Corollary 14, Lemma 15, Corollary 18 and Corollary 20, Within further $O(ln)$ rounds from t_1, there will be a round t_f such that at t_f chirality is achieved by all the agents or, $\Phi(t_f) = 0$. ◀

▶ **Theorem 22.** *Let $\mathcal{C}(t_0)$ be a configuration for some $t_0 \geq 0$ such that in $\mathcal{C}(t_0)$, all agents are in state* `Init` *and $\Phi(t_0) = 0$. Let $\mathcal{C}(t_0)$ has p chains such that not all of them are visibly directed. Then, $\exists\ t_f \geq t_0$ such that in $\mathcal{C}(t_f)$ the agents either achieve chirality or, in $\mathcal{C}(t_f)$, all of the p chains become visibly directed and in state* `Init`*.*

Proof. Let $\mathcal{C}(t_0)$ be a configuration where all agents are in state `Init` and $\Phi(t_0) = 0$. The existence of such a round is ensured by Theorem 21. Now if the configuration is global or, asymmetric agents achieve chirality at t_0. So, taking $t_f = t_0$ would suffice. Now let $\mathcal{C}(t_0)$ be neither global nor asymmetric. Now, since there are chains, not visibly directed, the agents change state to `Oscillate` at round t_0. Now, from round $t_0 + 1$ to round $t_0 + 2B + 2$, all agents execute Procedure 9. Let $C_D(u, v)$ be a visibly undirected chain in $\mathcal{C}(t_0 + 1)$ and at least one of u and v is of multiplicity. Note that $C_D(u, v) \in \mathcal{FC}$. So, by definition,

$d_{\mathcal{D}}(u,v) \geq 2$. Now, in the first execution of procedure 9, only the lowest ID agents stay at u and v. All other agents of u and v (if they exist) move inside the chain, so that both u and v become singletons. Let r_u be the agent at u and r_v be the agent at v in $\mathcal{C}(t_0+2)$. Now for the next $2B$ rounds starting from t_0+2, for round t_0+2x+2 $(0 \leq x < B)$, both r_u and r_v check their $(x+1)$-th bit from right and if the bit is 1, they move inwards the chain. Otherwise, they do not move. Now, for each round t_0+2x+3 $(0 \leq x < B)$, if any agent among r_u and r_v moved in round t_0+2x+2, it moves back. Note that, since the chains are i-chains with $i \geq 2$, halting the agents at nodes u and v, to move inward and then outward, effectively removing an edge, results in an asymmetric configuration, which ensures that the agents achieve chirality. So, let us assume the agents do not interrupt r_u and r_v's movement. Also note that the movement by r_u and r_v ensures that no chain breaks into multiple chains or no two or more chains merge during this procedure. So the total number of chains remains the same in each configuration from round t_0 to t_0+2B+2. In that case, since $ID(r_u) \neq ID(r_v)$, there exists $y \leq B$ such that the y-th bit from the right is the first bit that is different for r_u and r_v. So, at round t_0+2y, only one of r_u and r_v moves inward the chain. So, in $\mathcal{C}(t_0+2y+1)$ the chain $C_{\mathcal{D}}(u,v)$ becomes visibly directed. From t_0+2y+1 to t_0+2B+1, agents in the visibly directed chain do not do anything. The above argument is true for any visibly undirected chain in $\mathcal{C}(t_0+1)$. So at $\mathcal{C}(t_0+2B+2)$ all chains become visibly directed. Further, all agents change state to `Init`. This concludes the proof. ◀

▶ **Theorem 23.** *Let $\mathcal{C}(t_0)$ be a configuration where all agents are in the state `Init` and all chains are visibly directed. If the configation $\mathcal{C}(t_0)$ contains $p > 1$ such chains, then within $O(n)$ rounds, there exists a round t_f, at which the configuration $\mathcal{C}(t_f)$ satisfies one of the following:*

- *all agents agree on a common chirality, or*
- *all agents remain in the state `Init`, all chains are visibly directed, and the number of chains has reduced to $p' < p$.*

Proof. Let $\mathcal{C}(t_0)$ be a configuration in which all agents are in state `Init` and all p chains are visibly directed. If the configuration is global or asymmetric, then we are done, as chirality will be achieved by the agents. So, let the configuration $\mathcal{C}(t_0)$ be neither global nor asymmetric. So all agents change their state to `Merge-II`. This implies that among p chains there exists at least a pair of adjacent chains whose direction is towards each other. Let us arbitrarily choose one such pair, and name them $Ch_1 = C_{\mathcal{D}}(u_1,v_1)$ and $Ch_2 = C_{\mathcal{D}}(v_2,u_2)$. During the state `Merge-II`, agents execute Procedure 10. In this procedure, the chains Ch_1 and Ch_2 move towards each other by moving according to the direction of the chains if there is no missing edge in of Ch_1 or Ch_2. Without loss of generality, let Ch_1 has a missing edge. If the configuration is asymmetric due to the missing edge, then the agents achieve chirality, and we are done. On the other hand, if the configuration is symmetric, then agents in Ch_1 do not move, but agents in Ch_2 move towards Ch_1. So, the distance between Ch_1 and Ch_2 decreases at least by one and at most by two in each execution of Procedure 10. So, there exists a time t_1 such that both Ch_1 and Ch_2 merges and creates a new chain $Ch_3 \in \mathcal{FC}$ in the configuration $\mathcal{C}(t_1)$ in expense of two chains(as the chain length will be greater or equals to 2). Note that, Ch_3 will not be visibly directed as both the terminal nodes are singletons also as the initial distance between Ch_1 and Ch_2 can be at most $O(n)$, to $t_1 - t_0 \approx O(n)$. This will be true for all such pairs of visibly directed chains directed towards each other in $\mathcal{C}(t_0)$. Let there be q such pairs. So, until all q pairs are merged, the agents in the already merged chains do nothing. Now, there will be a time $t_2 \geq t_1$ such that at t_2 there are $p' = p - q$ chains where $q > 0$ and all chains are in $\mathcal{FC}$ but not all chains are visibly directed.

Note that, for the same reason stated above, $t_2 - t_0 \approx O(n)$. In $\mathcal{C}(t_2)$, all agents change state from `Merge-II` to `Init`. Then at $t_2 + 1$, all agents are in state `Init`and $\Phi(t_2 + 1) = 0$. Then by Theorem 22, within $O(B) = O(\log n)$ rounds from t_2 there exists a round $t_3 \geq t_2$ such that either agents achieve chirality in $\mathcal{C}(t_3)$ or, in $\mathcal{C}(t_3)$ all $p' = p - q < 0$ chains becomes visibly directed along with all agents being in state `Init`. For this case, taking $t_f = t_3$ proves the result. ◀

▶ **Theorem 24.** *In any* 1*-interval connected ring with* n *nodes and* l *achiral agents* ($n \geq l \geq 3$)*, where each agent has* $O(\log n)$ *memory and the agents are initially placed arbitrarily on the nodes, chirality can be achieved in* $O(ln)$ *rounds by executing the* ACHIRAL-2-CHIRAL *algorithm.*

Proof. By Theorem 21, we can conclude that for any arbitrary initial configuration $\mathcal{C}(0)$, there exists a round t_1 within $O(ln)$ rounds from the initial configuration, such that if chirality is not achieve till t_1 then in $\mathcal{C}(t_1)$ all agents are in state `Init` and $\Phi(t_1) = 0$. Now, in $\mathcal{C}(t_1)$ if all chains are not visibly directed then by Theorem 22 we can conclude that within $O(B) = O(\log n)$ more rounds there exists a round t_2 such that if chirality is not achieved till t_2, all chains become visibly directed chains with all agents being in state `Init`, in $\mathcal{C}(t_2)$. Now, let at $\mathcal{C}(t_2)$, there are p chains. If $p = 1$, then the direction of this chain is the agreed direction for all the agents. On the other hand, let $p > 1$. For this case, by Theorem 23, within further $O(n)$ rounds, there exists a round t_3, such that if chirality is not achieved by t_3, in $\mathcal{C}(t_3)$, all agents are in state `Init`, all chains visibly directed and number of chains, say $p' < p$. If, $p' \neq 1$, and chirality is not achieved, then then we can recursively use Theorem 23 until some time $t_4 > t_3$ such that if until t_4 chirality is still not achieved then at $\mathcal{C}(t_4)$, all agents are in state `Init` there is exactly one visibly directed chain. In this case, agents can agree on chirality using the direction of the chain. Note that in the worst case, there can be $O(l)$ chains in t_2 and each $O(n)$ further rounds, the number of chains reduces by one. This way, we can conclude that $t_4 - t_2 \approx O(ln)$. So, if we take $t_f = t_4$, we can conclude that, from any initial configuration within $O(ln)$ rounds, the agents will agree on a chirality. ◀

3.2 Algorithm DISPERSED

In this subsection, we describe the Algorithm DISPERSED. After chirality is achieved by the agents and if the configuration is not dispersed, the agents first execute Algorithm DISPERSED to achieve dispersion before achieving distance-k-dispersion. Note that, achieving dispersion partially solves the open problem regarding dispersion from any arbitrary configuration for any 1-interval connected ring proposed in [1]. Let us have a high-level idea of the algorithm.

3.2.1 Overview of DISPERED

Now, we provide a brief overview of the subroutine DISPERSED(), which is designed to transform the multiplicity nodes into singleton nodes to achieve dispersion. Let a configuration have a multiplicity node in some chain $C_{CW}(T, H)$, where T and H are the first node and last node, respectively, called *tail* and *head*, of the chain in clockwise direction. First, we try to convert the multiplicity node, M say, that is nearest to the tail of the chain.

Algorithm 11 DISPERSED(r,$\mathcal{C}(t_c)$)

```
$C_{CW}(T,H)$ = chain in which $r$ is located;
if $C_{CW}(T,H)$ contains a multiplicity then
    $M$ = multiplicity node in $C_{CW}(T,H)$ nearest to $T$.;
    $T'$ = Node adjacent to $T$ in $CCW$ direction;
    $H'$ = Node adjacent to $H$ in $CW$ direction;
    if $e_t \notin (T',H')_{CW} \vee e_t \in (T',M)_{CW}$ then
        if $r$ is at $M$ then
            if $r$ is not least ID agent at $M$ then
                move in clockwise direction.
        else if $r \in C_{CW}(M,H) \wedge r \notin M$ then
            move in clockwise direction.
    else if $e_t \in (M,H')_{CW}$ then
        if $r$ is at $M$ then
            if $r$ is not theleast ID agent at $M$ then
                move in counterclockwise direction.
        else if $r \in C_{CW}(T,M) \wedge r \notin M$ then
            move in counterclockwise direction.
```

Let the chain be partitioned into two sub-chains, $C_{CW}(T,M)$ and $C_{CW}(M,H)$ in clockwise direction with respect to the multiplicity M. Then, in this scenario, three cases may occur according to the position of the missing edge e_t (if it exists).

Case-I: There is no missing edge in the arc $(T',H')_{CW}$, where T' is the adjacent node of T in counterclockwise direction and H' is the the adjacent node of H in clockwise direction.

Case-II: The missing edge is in the arc $(T',M)_{CW}$.

Case-III: The missing edge is in the arc $(M,H')_{CW}$.

For, first and second case, i.e., when there is there is no missing edge in arc $(T',H')_{CW}$ or, the missing edge is in the chain $(T',M)_{CW}$, then the agent with least ID on the multiplicity M stay at its node and the other agents of the sub-chain $C_{CW}(M,H)$, except the least ID agent at M, move in clockwise direction. This way, M becomes a singleton node. Now, for the third case, i.e., when the missing edge is in the arc $(M,H')_{CW}$, the agent with least ID on the multiplicity M stay at its node and the other agents of the sub-chain $C_{CW}(T,M)$, except the least ID agent at the multiplicity M, move in counterclockwise direction. In each round during the execution of algorithm DISPERSED, the number of agents on a multiplicity node decreases. Now, since there can be at most $O(l)$ multiplicity points, within $O(l)$ rounds, the configuration becomes a dispersed configuration.

3.2.2 Correctness and Analysis of Algorithm DISPERSED

Let us first discuss some necessary notation for the correctness proof. We define the number of unoccupied nodes in a configuration $\mathcal{C}(t)$ with $\psi(t)$ for some time t. Note that if at some round t, the configuration $\mathcal{C}(t)$ is dispersed, then $\psi(t) = n - l$. Otherwise $\psi(t) > n - l$. To prove the correctness of the algorithm DISPERSED, we prove the following lemma first.

▶ **Lemma 25.** *Let $\mathcal{C}(t)$ be a configuration where $\psi(t) > n - l$. Then $\psi(t+1) < \psi(t)$ if agents execute* DISPERSE *in round t.*

Proof. Since $\psi(t) > n - l$, there must exist a chain with multiplicity. Let $Ch_1 = C_{CW}(T, H)$ be any such chain. Let T' and H' be the nodes adjacent to T and Hin the direction CCW and CW respectively. Note that T' and H' are both unoccupied nodes in $\mathcal{C}(t)$. Let M be the multiplicity node in Ch_1 that is nearest to T. Then, there can be three cases depending on the position of the missing edge e_t (if it exists). These three cases are,

1. $e_t \notin (T', H')_{CW}$
2. $e_t \in (T', M)_{CW}$
3. $e_t \in (M, H')_{CW}$

For cases 1 and 2, all agents of Ch_1 in the arc $(M, H)_{CW}$ except the lowest ID agent at M move clockwise according to algorithm Dispersed. Note that due to this H' along with becomes occupied in $\mathcal{C}(t+1)$. Also, while executing algorithm Dispersed, no occupied nodes become unoccupied. So for these cases $\psi(t+1) < \psi(t)$. Now for case 3, all agents in arc $(T, M)_{CW}$ move counterclockwise except the lowest ID agent at M according to algorithm Dispersed. This ensures that T' becomes occupied in $\mathcal{C}(t+1)$. Also, as no occupied nodes in $\mathcal{C}(t)$ become unoccupied in $\mathcal{C}(t+1)$, we can say $\psi(t+1) < \psi(t)$ in this case too. ◀

Now, using the above lemma, we prove the following theorem that justifies the correctness of algorithm Dispersed.

▶ **Theorem 26.** *If at round t, the configuration $\mathcal{C}(t)$ is not dispersed but agents have chirality agreement, then a dispersed configuration $\mathcal{C}(t_f)$ will be reached within $O(l)$ rounds from t.*

Proof. Since the configuration $\mathcal{C}(t)$ is not dispersed, there must exist a multiplicity node. Thus $\psi(t) > n - l$. Also, the agents have chirality agreement. Thus, at t agents execute Dispersed. Now by Lemma 25,in $\mathcal{C}(t+1)$, $\psi(t+1)$ is at least one less than $\psi(t)$ (i.e., $\psi(t+1) \leq \psi(t) - 1$). Now, if at $\mathcal{C}(t+1)$, $\psi(t+1)$ is still strictly greater than $n - l$, using the same argument, we can say there exists a round t_f such that $\psi(t_f) = n - l$. So, $\mathcal{C}(t_f)$ becomes a dispersed configuration. Note that in $\mathcal{C}(t)$ there can be at most $n - 1$ unoccupied nodes. So $t_f - t$ is asymptotically equal to $O(l)$. ◀

3.3 Algorithm Dispersed To k-Dispersed

When all agents achieve dispersion and if $k > 1$, then agents execute the algorithm Dispersed To $k-$Dispersed to achieve Distance-k- Dispersion. Following this, a brief overview of the algorithm is discussed.

3.3.1 Overview of the Algorithm

The agents execute the algorithm Dispersed-To-$k-$Dispersed until the configuration is a distance-$k-$dispersed configuration. If the configuration is not distance-k-dispersed, there must exist a k-Link with more than one agent. In each round, a selected set of agents moves either clockwise or counterclockwise so that the distance between at least one pair of agents placed in adjacent occupied nodes in the same k-Link increases and no pair of agents placed at adjacent occupied nodes in the same k-Link decreases. Further, it also ensures that agents placed at adjacent occupied nodes, which are already at a distance more than or equal to k, never decrease beyond k.

The priority is given to clockwise movement first in any particular round, say t. For clockwise movement, the selected set is $\mathcal{EAS}(t)$. Agents in $\mathcal{EAS}(t)$ only move clockwise in round t, if the missing edge e_t is not adjacent to $v(r)$ for any $r \in \mathcal{EAS}(t)$ in the clockwise direction. Otherwise, let $r' \in \mathcal{EAS}(t)$ be the agent such that the incident edge of $v(r')$ in

Algorithm 12 DISPERSED-TO-k-DISPERSED(r,$\mathcal{C}(t_c)$)

```
if $\mathcal{C}(t_c)$ is not distance-$k$-dispersed then
    if $e_{t_c}$ is not incident to $v(r')$ in the clockwise direction for some $r' \in \mathcal{EAS}(t_c)$ then
        if $r \in \mathcal{EAS}(t_c)$ then
            move clockwise;
    else
        $r'$ = the agent in $\mathcal{EAS}(t_c)$ such that $e_{t_c}$ is incident to $v(r')$in the clockwise direction;
        $KS$ = the $k$-Link such that $r' \in \mathcal{NS}_{CW}(KS)$ ;
        $KS'$ = the first movable $k$-Link in the clockwise direction starting from $KS$.
        if $r \in \mathcal{NS}_{CCW}(KS')$ then
            move counter clockwise;
else
    terminate;
```

the clockwise direction is the missing edge. Let r' be the part of the k-Link KS and KS' be the first movable k-Link in the clockwise direction starting from KS. Then the set of agents in the set of $\mathcal{NS}_{CCW}(KS')$ move counterclockwise. We now provide a brief overview of the correctness of the algorithm, ensuring that within a bounded time, any dispersed configuration becomes distance-k-dispersed.

3.3.2 Correctness and Analysis of DISPERSED-TO-$k-$DISPERSED

▶ **Lemma 27.** *If a configuration $\mathcal{C}(t)$ is not a distance-k-dispersed configuration for some round t, then there exists a movable k-Link in $\mathcal{C}(t)$.*

Proof. Let $KS_1, KS_2, \cdots, KS_\alpha$ be all the k-Links in $\mathcal{C}(t)$ and $|KS_i| = l_i$ for all $i \in \{1, 2, \cdots \alpha\}= I$. Let $I' \subseteq I$ such that $|KS_i| \geq 2, \forall i \in I'$ and $|KS_j| = 1, \forall j \in I \setminus I'$. Let $Len(KS_i) = d_{CW}(T(KS_i), H(KS_i))$ for all $i \in I$. Then $Len(KS_i) < (l_i - 1)k$ where $i \in I'$ and otherwise $Len(KS_j) = 0$ (i.e., when $j \in I \setminus I'$).

If possible, let all KS_i be non-movable k-Links. Then the exact next k nodes in the clockwise direction from the head of any KS_i are unoccupied. Let, for the k-Link KS_i, the arc with the next k unoccupied nodes is denoted as U_i. We denote $Len(U_i) = k$ for all $i \in I$. So for every $i \in I'$, $Len(KS_i) + Len(U_i) < l_i k$ and for every $j \in I \setminus I'$, $Len(KS_i) + Len(U_i) = k$.

Now, $n = \sum_{i=1}^{\alpha} Len(KS_i) + Len(U_i) = \sum_{i \in I'}[Len(KS_i) + Len(U_i)] + \sum_{i \notin I'}[Len(KS_i) + Len(U_i)] < \sum_{i \in I'}(l_i k) + xk$ (as I' is not an empty set and assuming $I \setminus I'$ has x elements without loss of generality). Now, from the above, we can say $n < (l - x)k + xk = lk$. But this is a contradiction, as according to the assumption $n \geq lk$. Hence, among the α k-Links, at least one movable k-Link exists. ◀

▶ **Lemma 28.** *Let r be an agent such that r' is the next agent of r in clockwise direction in $\mathcal{C}(t)$ for some round t. Let $d_{CW}(r, r') > k$ in $\mathcal{C}(t)$, then $\forall t_1 > t$, $d_{CW}(r, r') \geq k$ in $\mathcal{C}(t_1)$.*

Proof. Let $\exists\ t' > t$ such that in $\mathcal{C}(t')$, $d_{cw}(r, r') < k$. This implies there exists t'' such that $t > t'' > t'$ and in $\mathcal{C}(t'')$, $d_{CW}(r, r')$ decreases from k. This is possible only if, exactly one of r or r' stays still while the other moves in round t''. If r moves in $\mathcal{C}(t'')$. Then, it must have moved clockwise to reduce the distance from k further. This implies $r \in \mathcal{EAS}(t'') \implies r \in \mathcal{NS}_{CW}(KS)$, for some movable k-Link KS. This implies $r = H(KS)$. Thus $d(r, r') > k$ in $\mathcal{C}(t'')$(by definition of a movable k-Link) which leads to contradiction.

So, now let r' move and r stays still in t''. Then, in $\mathcal{C}(t'')$, r' must have moved counterclockwise. This implies $\exists$ k-Link KS_1 such that $e_{t''}$ is incident to $v(r_1)$ in the clockwise direction where $r_1 = H(KS_1)$. Let KS'_1 be the first movable k-Link in the clockwise direction starting from KS_1. This implies, $r' \in \mathcal{NS}_{CCW}(KS'_1)$ and $r \notin \mathcal{NS}_{CCW}(KS'_1)$. In fact $r = H(KS'_1)$ in $\mathcal{C}(t'')$. Thus $d_{CCW}(r, r') > k$ in $\mathcal{C}(t'')$ which is a contradiction to the assumption that $d_{CCW}(r, r') = k$ in $\mathcal{C}(t'')$. Thus, for all t_1 t, $d_{CCW}(r, r') \geq k$ in $\mathcal{C}(t_1)$. ◀

▶ **Lemma 29.** *Let for some round t, $d_{CW}(r, r') = k' < k$ where $v(r)$ and $v(r')$ be two adjacent occupied nodes in $\mathcal{C}(t)$. Then $\forall t' > t$, $d_{CW}(r, r') \geq k'$ in $\mathcal{C}(t')$.*

Proof. If possible let $\exists$ $t_1 > t$, such that at t_1, $d_{CW}(r, r')$ decreases from k'. This implies exactly one of r or, r' moves in round t_1. First, let r move while r' stays still. Then r must have to move clockwise to decrease the distance. Which implies $r \in \mathcal{EAS}(t_1)$ and hence $r = H(KS)$ for some movable k-Link KS. Thus in $\mathcal{C}(t_1)$, $k' = d_{CW}(r, r') > k$ which is a contradiction as $k' < k$.

So, now let us consider that r stays still whereas r' moves counterclockwise in $\mathcal{C}(t_1)$. using similar argument as in second part of Lemma 28 we can say that there is a movable k-Link KS', such that $r = H(KS')$ in $\mathcal{C}(t_1)$, and thus $k' = d_{CW}(r, r') > k$ in $\mathcal{C}(t_1)$ which is again a contradiction. Thus for any $t' > t$, $d_{CW}(r, r') \geq k'$ in $\mathcal{C}(t')$. ◀

▶ **Lemma 30.** *Let $\mathcal{C}(t)$ be a configuration that is not distance-k-dispersed. Then there exists a pair of agents r and r' located on adjacent occupied nodes $v(r)$ and $v(r')$ such that:*

1. *r and r' belong to the same k-Link in $\mathcal{C}(t)$, and*
2. *$d_{CW}(r, r')$ in $\mathcal{C}(t+1) > d_{CW}(r, r')$ in $\mathcal{C}(t)$.*

Proof. Since the configuration $\mathcal{C}(t)$ is not distance-k-dispersed, there exists a k-Link such that the number of agents on that k-Link is at least two. Also, by Lemma 27, in $\mathcal{C}(t)$ there is at least one movable k-Link. Let KS_0 be one such movable k-Link. Let starting from KS_0, all the k-Links in counter clockwise direction be $KS_1, KS_2, \cdots, KS_\alpha$. Let $0 \leq x \leq \alpha$ be such that KS_x is the first k-Link with number of agents at least two in counterclockwise direction starting from KS_0 (including it too). Let $r_x = H(KS_x)$ and r'_x be the next agent of r_x in the same k-Link in counterclockwise direction. Note that by definition $r_x \in \mathcal{NS}_{CW}(KS_0) \subseteq \mathcal{EAS}(t)$ but r'_x is not.

Now let the missing edge e_t be not incident to $v(r_y)$ in the clockwise direction for some $r_y \in \mathcal{EAS}(t)$. In this case, r_x moves clockwise and r'_x does not move in $\mathcal{C}(t)$. Thus, taking $r = r'_x$ and $r' = r_x$ is sufficient for this case.

Now let e_t be incident to $v(r_y)$ in the clockwise direction for some $r_y \in \mathcal{EAS}(t)$. Let r_y is part of the k-Link KS_z for some $0 \leq z \leq \alpha$. Let KS'_z be the first movable k-Link in the clockwise direction starting from KS_z. Let us consider the set $M = A \setminus \mathcal{NS}_{CCW}(KS'_z)$. Now, let r_z be the last agent in M starting from $H(KS'_z)$ in the counterclockwise direction. Then, $r_z = H(KS_\beta)$ for some k-Link, with at least two agents. Now let r'_z be the next agent in the counterclockwise direction of r_z. Now r_z and r'_z are in same k-Link by definition of $\mathcal{NS}_{CCW}(KS'_z)$ and also in $\mathcal{C}(t)$ r'_z moves counter clockwise and r_z stays still. Thus $d_{CW}(r'_z, r_z)$ increases in $\mathcal{C}(t+1)$ from $\mathcal{C}(t)$. So, taking $r = r'_z$ and $r' = r_z$ is sufficient for the proof. Note that for each case there exists a pair r and r' in the same k-Link such that $v(r')$ is the adjacent occupied node of $v(r)$ in clockwise direction and $d_{CW}(r, r')$ increases in $\mathcal{C}(t+1)$ ◀

Note that in Lemma 28 we proved that any pair of agents on adjacent occupied nodes separated by a distance greater than k can never be in the same k-Link further. So, the number of k-Links never decreases. Also in Lemma 29, we proved that for any two agents at adjacent occupied nodes in the same k-Link, their smallest distance never decrease. In the worst case, let all agents be in the same k-Link in the initial configuration. So, to achieve Distance-k-Dispersion, the smallest distance between the agents for each of $O(l)$ pairs of agents in adjacent occupied nodes, has to be increased at least k times. For $O(l)$ pairs, the total increase in distance should be $O(lk)$ in the worst case. Now, by Lemma 30, in each round, there exists a pair of agents in adjacent occupied nodes with distance strictly less than k, such that their distance increases. So, in $O(lk)$ rounds of executing Dispersed-To-$k-$Dispersed, the Distance-k-Dispersion will be solved. From this discussion, we have the following theorem.

▶ **Theorem 31.** *Let $\mathcal{C}(t)$ be any dispersed configuration in which all l agents possess chirality at some round t. Then, by executing the algorithm Dispersed-To-k-Dispersed, the agents can achieve Distance-k-Dispersion within $O(lk)$ rounds.*

4 Conclusion

In this work, we studied the Distance-k-Dispersion (D-k-D) problem for synchronous mobile agents in 1-interval connected rings without assuming chirality. We introduced a new technique that enables agents to simulate chirality using only local communication, bounded memory, vision, and global weak multiplicity detection and thereby showing that chirality is not a fundamental requirement for coordination in this model. With this insight, we solved D-k-D—and hence the standard dispersion problem—from arbitrary initial configurations, addressing an open question posed by Agarwalla et al. (ICDCN 2018) for even-sized rings. Finally, we provided the first finite-time algorithm that solves D-k-D in $O(ln)$ rounds, where l is the number of agents and n is the ring size.

An interesting direction for future work is to explore whether similar techniques for simulating chirality can be applied in other dynamic graph topologies or under more adversarial conditions, such as vertex permutation dynamism or weaker visibility models. Additionally, the overall time complexity of $O(ln)$ is solely due to the chirality simulation step; the subsequent algorithms for dispersion and distance-k-dispersion are time-optimal. This raises a natural question: What is the lower bound on the time required to simulate chirality in this model? It would be really interesting to find out the answer to this question.

References

1 Ankush Agarwalla, John Augustine, William K. Moses Jr., Sankar Madhav K., and Arvind Krishna Sridhar. Deterministic dispersion of mobile robots in dynamic rings. In Paolo Bellavista and Vijay K. Garg, editors, *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018, Varanasi, India, January 4-7, 2018*, pages 19:1–19:4. ACM, 2018. doi:10.1145/3154273.3154294.

2 John Augustine and William K. Moses Jr. Dispersion of mobile robots: A study of memory-time trade-offs. In Paolo Bellavista and Vijay K. Garg, editors, *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018, Varanasi, India, January 4-7, 2018*, pages 1:1–1:10. ACM, 2018. doi:10.1145/3154273.3154293.

3 Lali Barriere, Paola Flocchini, Eduardo Mesa-Barrameda, and Nicola Santoro. Uniform scattering of autonomous mobile robots in a grid. In *2009 IEEE International Symposium on Parallel & Distributed Processing*, pages 1–8, 2009. doi:10.1109/IPDPS.2009.5160871.

4 Prabhat Kumar Chand, Manish Kumar, Anisur Rahaman Molla, and Sumathi Sivasubramaniam. Fault-tolerant dispersion of mobile robots. In Amitabha Bagchi and Rahul Muthu, editors, *Algorithms and Discrete Applied Mathematics - 9th International Conference, CALDAM 2023, Gandhinagar, India, February 9-11, 2023, Proceedings*, volume 13947 of *Lecture Notes in Computer Science*, pages 28–40. Springer, 2023. doi:10.1007/978-3-031-25211-2_3.

5 Archak Das, Kaustav Bose, and Buddhadeb Sau. Memory optimal dispersion by anonymous mobile robots. In Apurva Mudgal and C. R. Subramanian, editors, *Algorithms and Discrete Applied Mathematics - 7th International Conference, CALDAM 2021, Rupnagar, India, February 11-13, 2021, Proceedings*, volume 12601 of *Lecture Notes in Computer Science*, pages 426–439. Springer, 2021. doi:10.1007/978-3-030-67899-9_34.

6 G.A. Di Luna, S. Dobrev, P. Flocchini, and N. Santoro. Live exploration of dynamic rings. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 570–579, 2016. doi:10.1109/ICDCS.2016.59.

7 Yotam Elor and Alfred M. Bruckstein. Uniform multi-agent deployment on a ring. *Theoretical Computer Science*, 412(8):783–795, 2011. doi:10.1016/j.tcs.2010.11.023.

8 Barun Gorain, Tanvir Kaur, and Kaushik Mondal. Distance-2-dispersion with termination by a strong team. In Subrahmanyam Kalyanasundaram and Anil Maheshwari, editors, *Algorithms and Discrete Applied Mathematics - 10th International Conference, CALDAM 2024, Bhilai, India, February 15-17, 2024, Proceedings*, volume 14508 of *Lecture Notes in Computer Science*, pages 44–58. Springer, 2024. doi:10.1007/978-3-031-52213-0_4.

9 Barun Gorain, Partha Sarathi Mandal, Kaushik Mondal, and Supantha Pandit. Collaborative dispersion by silent robots. *J. Parallel Distributed Comput.*, 188:104852, 2024. doi:10.1016/J.JPDC.2024.104852.

10 Tanvir Kaur and Kaushik Mondal. Distance-2-dispersion: Dispersion with further constraints. In David Mohaisen and Thomas Wies, editors, *Networked Systems - 11th International Conference, NETYS 2023, Benguerir, Morocco, May 22-24, 2023, Proceedings*, volume 14067 of *Lecture Notes in Computer Science*, pages 157–173. Springer, 2023. doi:10.1007/978-3-031-37765-5_12.

11 Ajay D. Kshemkalyani and Faizan Ali. Efficient dispersion of mobile robots on graphs. In R. C. Hansdah, Dilip Krishnaswamy, and Nitin H. Vaidya, editors, *Proceedings of the 20th International Conference on Distributed Computing and Networking, ICDCN 2019, Bangalore, India, January 04-07, 2019*, pages 218–227. ACM, 2019. doi:10.1145/3288599.3288610.

12 Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Fast dispersion of mobile robots on arbitrary graphs. In Falko Dressler and Christian Scheideler, editors, *Algorithms for Sensor Systems - 15th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers*, volume 11931 of *Lecture Notes in Computer Science*, pages 23–40. Springer, 2019. doi:10.1007/978-3-030-34405-4_2.

13 Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Dispersion of mobile robots in the global communication model. In Nandini Mukherjee and Sriram V. Pemmaraju, editors, *ICDCN 2020: 21st International Conference on Distributed Computing and Networking, Kolkata, India, January 4-7, 2020*, pages 12:1–12:10. ACM, 2020. doi:10.1145/3369740.3369775.

14 Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Dispersion of mobile robots on grids. In M. Sohel Rahman, Kunihiko Sadakane, and Wing-Kin Sung, editors, *WALCOM: Algorithms and Computation - 14th International Conference, WALCOM 2020, Singapore, March 31 - April 2, 2020, Proceedings*, volume 12049 of *Lecture Notes in Computer Science*, pages 183–197. Springer, 2020. doi:10.1007/978-3-030-39881-1_16.

15 Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Efficient dispersion of mobile robots on dynamic graphs. In *40th IEEE International Conference on Distributed Computing Systems, ICDCS 2020, Singapore, November 29 - December 1, 2020*, pages 732–742. IEEE, 2020. doi:10.1109/ICDCS47774.2020.00100.

16 Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Dispersion of mobile robots using global communication. *J. Parallel Distributed Comput.*, 161:100–117, 2022. doi:10.1016/J.JPDC.2021.11.007.

17 Ajay D. Kshemkalyani and Gokarna Sharma. Near-optimal dispersion on arbitrary anonymous graphs. In Quentin Bramas, Vincent Gramoli, and Alessia Milani, editors, *25th International Conference on Principles of Distributed Systems, OPODIS 2021, December 13-15, 2021, Strasbourg, France*, volume 217 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.OPODIS.2021.8.

18 Anisur Rahaman Molla and William K. Moses Jr. Dispersion of mobile robots: The power of randomness. In T. V. Gopal and Junzo Watada, editors, *Theory and Applications of Models of Computation - 15th Annual Conference, TAMC 2019, Kitakyushu, Japan, April 13-16, 2019, Proceedings*, volume 11436 of *Lecture Notes in Computer Science*, pages 481–500. Springer, 2019. doi:10.1007/978-3-030-14812-6_30.

19 Anisur Rahaman Molla, Kaushik Mondal, and William K. Moses Jr. Efficient dispersion on an anonymous ring in the presence of weak byzantine robots. In Cristina Maria Pinotti, Alfredo Navarra, and Amitabha Bagchi, editors, *Algorithms for Sensor Systems - 16th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2020, Pisa, Italy, September 9-10, 2020, Revised Selected Papers*, volume 12503 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2020. doi:10.1007/978-3-030-62401-9_11.

20 Anisur Rahaman Molla, Kaushik Mondal, and William K. Moses Jr. Byzantine dispersion on graphs. In *35th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2021, Portland, OR, USA, May 17-21, 2021*, pages 942–951. IEEE, 2021. doi:10.1109/IPDPS49936.2021.00103.

21 Anisur Rahaman Molla, Kaushik Mondal, and William K. Moses Jr. Optimal dispersion on an anonymous ring in the presence of weak byzantine robots. *Theor. Comput. Sci.*, 887:111–121, 2021. doi:10.1016/J.TCS.2021.07.008.

22 Brati Mondal, Pritam Goswami, and Buddhadeb Sau. Time optimal distance-k-dispersion on dynamic ring. In Amos Korman, Sandip Chakraborty, Sathya Peri, Chiara Boldrini, and Peter Robinson, editors, *Proceedings of the 26th International Conference on Distributed Computing and Networking, ICDCN 2025, Hyderabad, India, January 4-7, 2025*, pages 259–263. ACM, 2025. doi:10.1145/3700838.3700843.

23 Thomas Sauerwald and He Sun. Tight bounds for randomized load balancing on arbitrary network topologies. In *Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, FOCS '12, page 341–350, USA, 2012. IEEE Computer Society. doi:10.1109/FOCS.2012.86.

24 Masahiro Shibata, Yuichi Sudo, Junya Nakamura, and Yonghwan Kim. Almost uniform deployment of mobile agents in dynamic rings. *Information and Computation*, 289:104949, 2022. doi:10.1016/j.ic.2022.104949.

25 Takahiro Shintaku, Yuichi Sudo, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. Efficient dispersion of mobile agents without global knowledge. In Stéphane Devismes and Neeraj Mittal, editors, *Stabilization, Safety, and Security of Distributed Systems*, pages 280–294, Cham, 2020. Springer International Publishing.

26 Yuichi Sudo, Masahiro Shibata, Junya Nakamura, Yonghwan Kim, and Toshimitsu Masuzawa. Near-Linear Time Dispersion of Mobile Agents. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing (DISC 2024)*, volume 319 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:22, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DISC.2024.38.