The Multi-Agent Rotor-Router on the Ring: A Deterministic Alternative to Parallel Random Walks

Ralf Klasing, Adrian Kosowski, Dominik Pajak, Thomas Sauerwald

▶ **To cite this version:**

Ralf Klasing, Adrian Kosowski, Dominik Pajak, Thomas Sauerwald. The Multi-Agent Rotor-Router on the Ring: A Deterministic Alternative to Parallel Random Walks. PODC 2013 - ACM Symposium on Principles of Distributed Computing, Jul 2013, Montreal, Canada. pp.365-374, 10.1145/2484239.2484260 . hal-00735113v1

HAL Id: hal-00735113

https://inria.hal.science/hal-00735113v1

Submitted on 26 Sep 2012 (v1), last revised 29 Sep 2014 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

The Multi-Agent Rotor-Router on the Ring: A Deterministic Alternative to Parallel Random Walks

Ralf Klasing* Adrian Kosowski† Dominik Pająk† Thomas Sauerwald ‡

Abstract

The *rotor-router mechanism* was introduced as a deterministic alternative to the random walk in undirected graphs. In this model, an agent is initially placed at one of the nodes of the graph. Each node maintains a cyclic ordering of its outgoing arcs, and during successive visits of the agent, propagates it along arcs chosen according to this ordering in round-robin fashion. The behavior of the rotor-router is fully deterministic but its performance characteristics (cover time, return time) closely resemble the expected values of the corresponding parameters of the random walk.

In this work we study the scenario in which multiple, indistinguishable agents are deployed in parallel in the nodes of the graph, and move around the graph in synchronous rounds, interacting with a single rotor-router system. This setting was introduced by Yanovski *et al.* (2003). We propose new techniques which allow us to compare this model theoretically with the intensively studied scenario of parallel independent random walks in a graph. Our main results concern the n-node ring, and suggest a strong similarity between the performance characteristics of both models.

We show that the rotor-router with k agents achieves on the ring a cover time of between $\Theta(n^2/k^2)$ and $\Theta(n^2/\log k)$, depending on the initial locations of the agents, and both these bounds are tight. The corresponding expected values of cover time for k random walks are proved to be $\Theta(n^2/(k^2/\log^2 k))$ and $\Theta(n^2/\log k)$. We then show that after the rotor-router system has stabilized, all nodes of the ring are always visited every $\Theta(n/k)$ steps, regardless of initialization. This corresponds asymptotically to the expected time between visits in the case of k random walks. All our results hold up to a polynomially large number of agents ($1 \leq k < n^{1/11}$).

1 Introduction

The study of deterministic exploration strategies in agent-based models of computation is largely inspired by considerations of random walk processes. For an undirected graph $G = (V, E)$, exploration with the random walk has many advantageous properties: the expected arrival time of the agent at the last unvisited node of the graph, known as the *cover time* $C(G)$, can in general be bounded as, e.g., $C(G) \in O(D|E| \log |V|)$, where D is the diameter of the graph. The random walk also has the property that in the limit it visits all of the edges of the graph with the same frequency, on average, traversing each once every $|E|$ rounds. The rotor-router model, introduced

*LaBRI, CNRS — Université de Bordeaux — Inria, 33400 Talence, France. E-mail: ralf.klasing@labri.fr

†Inria Bordeaux Sud-Ouest, 33400 Talence, France.
E-mails: {adrian.kosowski,dominik.pajak}@inria.fr

‡MPI für Informatik, 66123 Saarbrücken, Germany

by Priezzhev *et al.* [15] and further popularised by James Propp, provides a mechanism for the environment to control the movement of the agent deterministically, whilst retaining similar properties of exploration as the random walk.

In the rotor-router model, the agent has no operational memory and the whole routing mechanism is provided within the environment. The edges outgoing from each node v are arranged in a fixed cyclic order known as a *port ordering*, which does not change during the exploration. Each node v maintains a *pointer* which indicates the edge to be traversed by the agent during its next visit to v. If the agent has not visited node v yet, then the pointer points to an arbitrary edge adjacent to v. The next time when the agent enters node v, it is directed along the edge indicated by the pointer, which is then advanced to the next edge in the cyclic order of the edges adjacent to v.

The behavior of the rotor-router for a single agent is well understood. Yanovski *et al.* [19] showed that, regardless of the initialization of the system, the agent stabilizes to a traversal of a directed Eulerian cycle (containing all of the edges of the graph) within $2D|E|$ steps. A complementary lower bound was provided by Bampas *et al.* [4], who showed that for any graph there exists an initialization of the system for which covering all the nodes of the graph and entering the Eulerian cycle takes $\Theta(D|E|)$ steps. Despite seemingly similar general-case bounds on the cover time for the random walk and the rotor-router, there exist graphs for which these times differ. For example, for the two-dimensional square grid the rotor-router covers all nodes in $\Theta(|V|^{3/2})$ rounds in the worst case, while the cover time of the random walk is $\Theta(|V| \log^2 |V|)$.

Our work deals with the problem of exploring a graph with the *multi-agent rotor-router*, i.e., a rotor-router system in which more than one agent are deployed in the same environment. Due to the interaction of the agents, which move the same set of pointers at nodes, this can be seen as an example of a deterministic interacting particle system. We compare our results with the so-called *parallel random walk*, achieved by deploying independent agents performing random walks in a graph independently and without any form of coordination. Recent work on the area of parallel random walks [3, 11, 10, 17] contains a characterization of the improvement of the cover time due to the deployment of k independent random walkers with respect to the case with a single walker. It is shown in these works that the achieved speed-up depends on different parameters, such as the mixing time [11] and edge expansion [17] of the graph. The speed-up may sometimes be as low as $\Theta(\log k)$ [3], and sometimes as high as exponential in terms of k [10]. For many classes of graphs the speed-up is linear in terms of k (especially when k is small, $k \in O(\log n)$).

1.1 Our results and organization of the paper

In this work, we perform a comparative case study of two seemingly different scenarios: deterministic exploration with interacting particles in the rotor-router model vs. randomized exploration with non-interacting particles in the random walk, showing certain similarities between them.

We focus on two parameters of exploration. The first is the *cover time*, understood as the time before each node of the graph is covered by at least one agent. The second is the *return time*, i.e., the longest time during which some node remains unvisited in the limit, disregarding the initialization phase of the rotor-router. (Note that the rotor-router, as a deterministic finite-state system, has to stabilize to a cyclic traversal of some set of configurations on the graph.) We present our results taking into account different initial locations of the set of agents.

In Section 2, we formally describe the model of the rotor-router system, and introduce the techniques used in the analysis of the multi-agent rotor-router system. The basic tool is applicable

Model	*Cover time*		*Return time*
	for worst placement	*for best placement*	
k-agent rotor-router	$\Theta(n^2/\log k)$ Thm 3.2, 3.4	$\Theta(n^2/k^2)$ Thm 3.5, 3.7	$\Theta(n/k)$ Thm 4.1
k random walks (expectations)	$\Theta(n^2/\log k)$ [3]	$\Theta\left(n^2 \Big/ \frac{k^2}{\log^2 k}\right)$ Thm 3.11	$\Theta(n/k)$ e.g. [1]

Table 1: The cover time of the multi-agent roter-router on the ring compared to multiple random walks (for $k < n^{1/11}$).

to general graphs and gives us an algorithmic perspective for analysis of the rotor-router through *delayed deployments* (Subsection 2.1), allowing the occasional stopping of some of the agents without affecting asymptotic cover time. For the specific case of the rotor-router on the ring (cycle), we describe states in the evolution of the system in which particular agents cover nearly disjoint, dynamically changing parts of the graph, known as *agent domains* (Subsection 2.2). We also introduce a *continuous time approximation* of the evolution of the system on the ring (Subsection 2.3), which allows us to postulate an asymptotic description of the behavior of the agents on the ring. Formal proofs of correctness are obtained through an analysis of the motion of agents within their domains in delayed deployments of the rotor-router.

Our main results for the case when the explored graph is a ring are presented in Section 3 (cf. Table 1 for an overview). We show that for a k-agent rotor-router system, the cover time is between $\Theta(n^2/k^2)$ and $\Theta(n^2/\log k)$, depending on the initial placement of the agents in the rotor-router. The first bound is achieved, in particular, for agents distributed uniformly on the ring, while the latter for agents initially located on the same node of the ring. The return times for the ring of the k-agent rotor-router is determined in Section 4 as $\Theta(n/k)$.

We remark that for a single agent, the rotor-router on the ring deterministically achieves a cover time of $\Theta(n^2)$, which matches that of the random walk. As the number of agents k increases, the speed-up of the rotor-router with respect to a single-agent system is seen from our results as between $\Theta(\log k)$ and $\Theta(k^2)$, depending on the initialization. These results are comparable with the corresponding speed-up of the random walk, which is between $\Theta(\log k)$ and $\Theta(k^2/\log^2 k)$. The speed-up in terms of return time is k-fold, in both cases.

1.2 Related work

Deterministic graph exploration. Deterministic approaches which provide guarantees on worst-case cover time even on unknown anonymous graphs are a tempting alternative to random walks. However, their implementation proves complicated when considering anonymous networks, in which the agent is not helped by the environment, and when located at a node, it has to decide on its next move based only on its local state memory, the local port ordering at the node, and the port by which it entered the current node. It is a well-established result that no memory-less agent can explore all graphs deterministically; this impossibility result has also been extended to a finite team of memory-less agents with extended capabilities in the so-called JAG (jumping automata on graphs) model. Moreover, it has been shown [12] that an agent must be equipped with at least $|V|$ states (i.e., $\Omega(\log |V|)$ bits of memory) to be able to explore all graphs with $|V|$ nodes. On the positive side, unknown anonymous graphs can be deterministically explored by following so called universal traversal/exploration sequences. These exist for any number of nodes, and have

polynomial length [2]. Cover time obtained using such an approach is, however, usually by a factor of about $|V|^2$ greater than the (expected) cover time of a corresponding random walk. It has only been shown very recently in the seminal result [16] that such universal exploration sequences can be constructed and followed using very small memory, and consequently, deterministic graph exploration can be performed by an agent with only $O(\log n)$ bits of state memory. However, the exploration time achieved by such a procedure may potentially be extremely long, expressed by a polynomial with a high exponent.

By extending the capabilities of the agent and allowing it to interact with the environment, it is possible to decrease the time of deterministic exploration without requiring the agent to use more memory. Numerous models have been proposed which rely either on the existence of informative labeling schemes in the network, or on the capability of the agent to leave pebbles on nodes, move tokens, or write to so called "white-board" memory on nodes. The reader is referred to [14] for an extensive survey.

An important line of research is devoted to *equitable strategies*, in which the environment attempts to mimic the fairness properties of the random walk with respect to the use of edges. Two such strategies, in which the agent is always directed to the least often used, or the longest unused, from among the edges adjacent to the current node were studied in [7]. When considering fairness of traversal of arcs of the graph (i.e., taking into account the direction of traversal), the strategy which directs the agent along the outgoing edge which has not been used for the longest time is precisely equivalent to the rotor-router model.

The rotor-router model. Studies of the rotor-router started with works of Wagner *et al.* [18] who showed that in this model, starting from an arbitrary configuration (arbitrary cyclic orders of edges, arbitrary initial values of the port pointers and an arbitrary starting node) the agent covers all edges of the graph within $O(|V||E|)$ steps. Bhatt *et al.* [6] showed later that within $O(|V||E|)$ steps the agent not only covers all edges but *enters (establishes) an Eulerian cycle*. More precisely, after the initial *stabilisation period* of $O(|V||E|)$ steps, the agent keeps repeating the same Eulerian cycle of the directed symmetric version $\vec{G}$ of graph $\vec{G}$ (see the model description for a definition). Subsequently, Yanovski *et al.* [19] and Bampas *et al.* [4] showed that the Eulerian cycle is in the worst case entered within $\Theta(D|E|)$ steps in a graph of diameter D. Considerations of specific graph classes were performed in [13]. Robustness properties of the rotor-router were further studied in [5], who considered the time required for the rotor-router to stabilize to a (new) Eulerian cycle after an edge is added or removed from the graph. Regarding the terminology, we note that the rotor-router model has also been referred to as the *Propp machine* [4] or *Edge Ant Walk algorithm* [18, 19], and has also been described in [6] in terms of traversing a maze and marking edges with pebbles.

In the context of graph exploration, before this work, the only study of the multi-agent rotor-router was performed by Yanovski *et al.* [19], who showed that adding a new agent to the system cannot slow down exploration, and provided some experimental evidence showing a nearly-linear speed-up of cover time with respect to the number of agents in practical scenarios. They also show that the multi-agent rotor-router eventually visits all edges of the graph a similar number of times. Beyond this, a characterization of the behavior of the k-agent rotor-router in general graphs remains an open question.

A variant of the multi-agent rotor-router mechanism has been extensively studied in a different setting, in the context of balancing the workload in a network. The single agent is replaced with a number of agents, referred to as *tokens*. Cooper and Spencer [8] study d-dimensional grid graphs and show a constant bound on the difference between the number of tokens at a given node v

in the rotor-router model and the expected number of tokens at v in the random-walk model. Subsequently Doerr and Friedrich [9] analyse in more detail the distribution of tokens in the rotor-router mechanism on the 2-dimensional grid.

1.3 Model definition

Let $G = (V, E)$ be an undirected connected graph with n nodes, m edges and diameter D. We denote the neighborhood of a node $v \in V$ by $\Gamma(v)$. The directed graph $\vec{G} = (V, \vec{E})$ is the directed symmetric version of G, where the set of arcs $\vec{E} = \{(v, u), (u, v) : \{v, u\} \in E\}$.

We consider the rotor-router model (on graph G) with $k \geq 1$ indistinguishable agents, which run in rounds, synchronized by a global clock. In each round, each agent moves in discrete steps from node to node along the arcs of graph $\vec{G}$. A *configuration* at the current step is a triple $((\rho_v)_{v\in V}, (\pi_v)_{v\in V}, \{r_1, \ldots, r_k\})$, where ρ_v is a cyclic order of the arcs (in graph $\vec{G}$) outgoing from node v, π_v is an arc outgoing from node v, which is referred to as *the (current) port pointer at node* v, and $\{r_1, \ldots, r_k\}$ is the (multi-)set of nodes currently containing an agent. For each node $v \in V$, the cyclic order ρ_v of the arcs outgoing from v is fixed at the beginning of exploration and does not change in any way from step to step (unless an edge is dynamically added or deleted as discussed in the previous section). For an arc (v, u), let $next(v, u)$ denote the arc next after arc (v, u) in the cyclic order ρ_v.

The exploration starts from some initial configuration and then keeps running in all future rounds, without ever terminating. During the current round, first each agent i is moved from node r_i traversing the arc π_{r_i}, and then the port pointer π_{r_i} at node r_i is advanced to the next arc outgoing from r_i (that is, π_{r_i} becomes $next(\pi_{r_i})$). This is performed sequentially for all k agents. Note that the order in which agents are released within the same round is irrelevant from the perspective of the system, since agents are indistinguishable. For example, if a node v contained two agents at the start of a round, then it will send one of the agents along the arc π_v, and the other along the arc $(v, next(\pi_v))$. In some considerations, we will also assign explicit labels $\{0, 1, \ldots, \deg(v) - 1\}$ to the ports adjacent to v, in such a way that initially $\pi_v = 0$, and $next(v, i) = (v, (i+1) \mod \deg v)$. Then, at the completion of any round, the total number of traversals of agents along an arc (v, u) is equal to $\left\lceil \frac{e_v - port_v(u)}{\deg(v)} \right\rceil$, where e_v is the total number of times agents exited node v until the completion of the round and $port_u(v)$ denotes the label of the port leading from v to u.

In all our considerations, we will assume that the initialization of ports and pointers in the system is performed by an adversary. In particular, when studying a best-case scenario of initial agent locations, we assume that the ports and pointers have been set by the adversary so as to maximize the studied parameter (e.g., cover time). For the case of the ring, there exists only one cyclic permutation of the two neighbors of each node, hence only the initial pointer arrangement (and not the configuration of ports) is relevant.

2 Techniques for the Multi-agent Rotor-Router

2.1 Delayed deployments

In our work we will consider both the unmodified k-agent rotor-router system $R[k]$ and its *delayed deployments*, in which some agents may be stopped at a node, skipping their move for some number of rounds. A delayed deployment D of k agents is formally defined as a function $D : V \times \mathbb{N} \to \mathbb{N}$, where $D(v, t) \geq 0$ represents the number of agents which are stopped in vertex v in round t

of the execution of the system. (The rotor-router system $R[k]$ corresponds to the deployment $R[k](v,t) = 0$, for all v and t). Delayed deployments may be conveniently viewed as algorithmic procedures for delaying agents, and are introduced for purposes of analysis, only.

We will say that a node is *visited* by an agent in round t if the agent is located at this node at the start of round $t+1$. Let $n_v^D(t)$ denote the total number of visits of agents to node v during the interval of rounds $[1,t]$ for agents following some (possibly delayed) deployment D, and let $C(D)$ be the cover time of this deployment. The notation $n_v^D(0)$ refers to the number of agents at a node directly after initialization (at the start of round 1).

We start by showing that by delaying more agents in a deployment, one cannot increase the number of visits to nodes at any time. We assume that all considered deployments start from the same (arbitrarily chosen) initial configuration.

Lemma 2.1. *Let D_1 and D_2 be two delayed deployments of the k-agent rotor-router system, such that for all vertices $v \in V$ and rounds t, $D_1(v,t) \geq D_2(v,t)$. Then, for all vertices $v \in V$ and rounds t, we have $n_v^{D_1}(t) \leq n_v^{D_2}(t)$.*

Proof. For $t = 0$, the claim holds, since by definition:

$$n_v^{D_1}(0) = n_v^{D_2}(0) = n_v^{R[k]}(0), \quad \text{for all } v \in V. \tag{1}$$

Now, denote by $e_v^{D_1}(t)$ and $e_v^{D_2}(t)$ the total number of traversals of arcs outgoing from v during the interval of rounds $[1,t]$ for executions D_1 and D_2, respectively. For an arbitrary agent, the difference between the number of times the agent leaves v in rounds $[1,t+1]$ and the number of times it enters node v in rounds $[0,t]$ is equal to either -1 or 0, depending on whether the agent is delayed at v in round $t+1$ or not. Summing over all agents, we obtain:

$$e_v^{D_i}(t+1) = n_v^{D_i}(t) - D_i(v,t+1), \quad i \in \{1,2\} \tag{2}$$

The rest of the proof proceeds by induction on time t. Suppose that for some $t > 1$, $n_v^{D_1}(t-1) \leq n_v^{D_2}(t-1)$ holds for all $v \in V$. Then, we have from (2):

$$e_v^{D_1}(t) + D_1(v,t) \leq e_v^{D_2}(t) + D_2(v,t)$$

and since $D_1(v,t) \geq D_2(v,t)$:

$$e_v^{D_1}(t) \leq e_v^{D_2}(t), \quad \text{for all } v \in V. \tag{3}$$

Now, fix an arbitrary node u and observe that the number of visits to node u within the interval $[1,t+1]$ is equal to the sum of the number of agents placed at u in round 1, and the number of times an agent exited one of its neighbors $v \in \Gamma(u)$ along an arc (v,u) in rounds $[1,t]$:

$$n_u^{D_i}(t+1) = n_u^{D_i}(0) + \sum_{v \in \Gamma(u)} \left\lceil \frac{e_v^{D_i}(t) - port_v(u)}{\deg(v)} \right\rceil, \tag{4}$$

where we took into account that agents leaving a node v exit along the ports adjacent to v in round-robin fashion. Combining expressions (1), (2), and (4) we obtain $n_u^{D_1}(t+1) \leq n_u^{D_2}(t+1)$. Since $u \in V$ was arbitrarily chosen, the inductive claim follows. □

We remark that the above lemma immediately implies that $n_v^{R[k-1]}(t) \leq n_v^{R[k]}(t)$, since the $(k-1)$-agent rotor-router $R[k-1]$ is equivalent to a deployment of the k-agent rotor-router with one agent permanently stopped. (This observation is due to [19].)

Lemma 2.2. *Let D be a delayed deployment of the k-agent rotor-router system. Let T be any fixed time round, and let τ be the number of rounds in the interval $[1,T]$ such that all the agents are active in D, i.e., $\tau = |\{t \in [1,T] : \forall_{v \in V}\ D(v,t) = 0\}|$. Then, for all vertices v, we have: $n_v^{R[k]}(\tau) \leq n_v^D(T) \leq n_v^{R[k]}(T)$.*

Proof. The right inequality follows directly from Lemma 2.1. To prove the left inequality, we rewrite for round $t \geq 1$ the sets of recurrence equations (2) and (4) on the number of visits and exits to each node v for deployment D:

$$\begin{cases} e_v^D(t) = n_v^D(t-1) - D(v,t), \\ n_v^D(t) = n_v^D(0) + \sum_{w \in \Gamma(v)} \left\lceil \frac{e_w^D(t-1) - port_w(v)}{\deg(w)} \right\rceil, \\ n_v^D(0) = n_v^{R[k]}(0), \quad e_v^D(0) = 0. \end{cases}$$

Consider a function $f : [1,\tau] \to [1,T]$, with $f(i)$ being the i-th time round in which all agents are active in delayed deployment D. Denote by $F \subseteq [1,T]$ the image of f. Taking into account that $D(v,t) = 0$ for all $t \in F$ and that the counters e_v^D and n_v^D are always non-decreasing in time, we obtain the following set of inequalities by restricting evolution to moments of time $t = f(i)$, with $i \in [1,\tau]$:

$$\begin{cases} e_v^D(f(i)) = n_v^D(f(i)-1) - D(v,f(i)) \geq n_v^D(f(i-1)) - 0 = n_v^D(f(i-1)), \\ n_v^D(f(i)) = n_v^D(0) + \sum_{w \in \Gamma(v)} \left\lceil \frac{e_w^D(f(i)-1) - port_w(v)}{\deg(w)} \right\rceil \geq n_v^D(f(0)) + \sum_{w \in \Gamma(v)} \left\lceil \frac{e_w^D(f(i-1)) - port_w(v)}{\deg(w)} \right\rceil, \\ n_v^D(f(0)) = n_v^{R[k]}(f(0)), \quad e_v^D(f(0)) = 0, \end{cases}$$

where we put $f(0) = 0$ for convenience of notation. By comparing the above with the corresponding equations for the undelayed rotor-router $R[k]$, written for round $i \in [1,\tau]$:

$$\begin{cases} e_v^{R[k]}(i) = n_v^{R[k]}(i-1), \\ n_v^{R[k]}(i) = n_v^{R[k]}(0) + \sum_{w \in \Gamma(v)} \left\lceil \frac{e_w^{R[k]}(i-1) - port_w(v)}{\deg(w)} \right\rceil, \\ e_v^{R[k]}(0) = 0. \end{cases}$$

it follows by induction that $n_v^D(f(i)) \geq n_v^{R[k]}(i)$. Putting $i = \tau$, we obtain the sought inequality $n_v^D(T) \geq n_v^{R[k]}(\tau)$. □

Observe that by the above lemma, we have that if node v is visited for the first time after T rounds in a delayed deployment D, i.e., $n_v^D(T) = 0$ and $n_v^D(T+1) = 1$, then $n_v^{R[k]}(\tau) = 0$ and $n_v^{R[k]}(T+1) \geq 1$. From this, we directly obtain the key lemma for the approach we use to analysing the cover time of k-rotor-router systems in this paper.

Lemma 2.3 (the slow-down lemma). *Let $R[k]$ be a k-rotor router system with an arbitrarily chosen initialization, and let D be any delayed deployment of $R[k]$. Suppose that deployment D covers all the vertices of the graph after $T = C(D)$ rounds, and in at least τ of these rounds, all agents were active in D. Then, the cover time $C(R[k])$ of the system can be bounded by:*

$$\tau \leq C(R[k]) \leq T.$$

□

If the deployment D is defined so that agents in D are delayed in at most a constant proportion of the first $C(D)$ rounds, then the above inequalities lead to an asymptotic bound on the value of the undelayed rotor-router, $C(R[k]) = \Theta(C(D))$. This is the case, e.g., in the proof of Theorem 3.2.

2.2 Agent domains on the ring

For a given (possibly delayed) deployment of the k-rotor-router system such that no two agents ever occupy the same node at the same time, and a fixed round t, we consider the partition of the node set into so called *domains*. We set $V(t) = V_0(t) \cup V_1(t) \cup \ldots \cup V_k(t)$, where $V_0(t)$ denotes the set of nodes which have not yet been visited until round t, and $V_i(t)$, $1 \leq i \leq k$, is the set of all nodes such that the i-th agent was the last agent visiting the node until round t, inclusive. When the considered graph is a ring, we have the following simple characterization of the structure of the domains of particular agents in deployments in which agents never meet. We state the following simple properties without proof.

Lemma 2.4. *Consider a deployment in which no two agents ever meet at a node, and let $v_i(t) \in V_i(t)$ be the location of the i-th agent at a given round t, $1 \leq i \leq k$. The following properties hold:*

- *$V_i(t)$ induces a sub-path of the ring.*
- *The pointers of all nodes $u \in V_i(t)$ point away from $v_i(t)$, i.e., not along the arc on the path leading from u to $v_i(t)$ in $V_i(t)$. In particular, if $v_i(t)$ is an end-point of the path induced by $V_i(t)$, then all the pointers of $V_i(t) \setminus \{v_i(t)\}$ point in the same direction.*
- *In each round, $V_i(t)$ loses or gains at most one node at each end of the path. In particular, $|V_i(t+1) \oplus V_i(t)| \leq 2$.* □

The following three lemmas provide a partial characterization of the changes of size of domains during the runtime of a deployment. We first define the borders and interiors of the domains. Suppose that at some time t the domain of each agent is of size at least 3. For time t, we define the *border* between these two domains as the two adjacent nodes from these two domains. The *interior* of a domain at this time t is defined as the domain without its border nodes. For $t' > t$, borders and interiors are defined recursively based on borders from previous step and positions of agents. We will denote the interior of the domain of agent i in any step $t' \geq t$ by $I_i(t')$. Let j be an agent with a domain neighboring with that of i. If in step t', $I_i(t'-1) \cap V_j(t') \neq \emptyset$ (i.e., if j captured some node belonging to the interior of the domain of agent i in step t') then the border between the domains of i and j moves by two nodes towards the domain of agent i. Thus, the interior of the domain of agent j grows by two nodes. Note that in order to move the border, agent j has to make two visits to the border, in between which there is no visit to this border by agent i.

We will say that agent a makes a full cycle starting at some node v when it visits every node of its domain and returns to v. Note that in order to make a full cycle agent a has to visit every node of its domain twice, except extremal nodes from both sides of the domain.

Let a,b,c any three agents with consecutive domains. Let t be the time step, when borders were defined. We first show that if the interiors of the domains are sufficiently large at some point, then the will never decrease below a threshold size of $11k$.

Lemma 2.5. *If $k \geq 5$ and initially the interior of every domain has size at least $22k$, then for any $t' \geq t$:*

(1) if $|I_a(t')| - 7 > |I_b(t')|$ and $|I_b(t')| \leq 2|I_c(t')|$, then in step $t'+1$ the border between a and b will not move towards b,

(2) the size of the interior of any domain is at least $11k$.

Proof. We will prove this lemma by induction on time. First take time t. In time $t+1$ no border can move, because t is time of initialization of borders and agent has to visit the border twice to move it. We will prove pair of implications to prove the lemma. Fix any $t' > t$.

We want to prove the first implication. Assume that condition (1) and (2) are true for $t, t+1, \ldots, t'$ and initially every interior has size at least $22k$. We want to prove that condition (2) is true for $t'+1$. We can denote by $i_{min}(t^*)$ the minimum size of interior in time t^*. We want to show, that $i_{min}(t'+1) \geq 11k$. We define the following function $j_a(t') = \min\{|I_a(t')|, 22k\}$. Take any step $t^* \in [t, t']$. From the assumption $i_{min}(t^*) \geq 11k$ thus if for some agents a, b with adjacent domains $|j_a(t^*) - j_b(t^*)| \geq 8$, then the border between a and b cannot move in any direction in step t^*+1. If $|j_a(t^*) - j_b(t^*)| = 7$, then the border still can move thus the difference can increase up to 11. Since initially for every a, $|I_a(t)| \geq 22k$, then $j_a(t) = 22k$. Consider the configuration that yields the minimum possible value of function j_a for some agent a. Let $a_1, a_2, \ldots a_k$ be a sequence of consecutive agents. Then the configuration is $j_{a_1}(t^*) = 22k, j_{a_2}(t^*) = 22k-11, \ldots j_{a_k}(t^*) = 11k+11$. It is true for any $t^* \in [t, t'+1]$. Thus the minimum size of interior of any domain in step $t'+1$ is at least $11k$.

We will prove the second implication. Take any time step $t' \geq t$. We want to prove, that condition (2) for $t, t+1, \ldots, t'$ implies condition (1) for t'. Assume, by contradiction, that agent a moves the border between a and b in step $t'+1$. So, in step t', agent a is located at the extremal point of the border of domains a and b, having completed a cyclic exploration of its domain. Let us denote the mentioned extremal point of border by v_b. Let $t^* < t'$ be time step, when a previously visited v_b. If such t^* does not exist, then either a is making first cycle after domains were defined or node v_b was not visited by a in previous cycle in both cases it cannot move the border in step $t'+1$. Note, that since in $t'+1$ agent a moves the border then b did not visit v_b in time interval $[t^*+1, t']$. It is however possible, that in time t^* both a and b were located in v_b.

Note, that $t' - t^* - 1 \leq 2|I_b(t^*)| + 4$ because b cannot reach the border between a and b in time interval $[t^*+1, t']$. In the worst case b can be at node v_b at time t^* it has to traverse its whole interior, then it may move its other border and then traverse its whole interior again. Thus in time at most $2|I_b(t^*)| + 5$ it would reach v_b.

On the other hand a finished its cycle thus $t' - t^* \geq 2|I_a(t')| + 4$. Cycle of agent a consisted both nodes from border with b, interior of a then at least one node of other border, whole interior again and again both nodes from border with b. We denote $i_c = \min_{s \in [t^*, t']}\{|I_c(s)|\}$ which is the minimum size of interior of c in time interval $[t^*, t']$. We will consider two cases. First assume, that $i_c > |I_b(t^*)|/4$, thus $t' - t^* \leq 2|I_b(t^*)| + 5 \leq 8i_c + 5$ and c can make at most 4 complete cycles of its domain. Thus b can lose at most 8 nodes to c in time interval $[t^*, t']$. Since $|I_b(t^*)| - 8 \leq |I_b(t')|$, then $t' - t^* \leq 2|I_b(t')| + 18 < 2|I_a(t')| + 4$ which leads to a contradiction.

Now consider case, when $i_c \leq |I_b(t^*)|/4$. To increase size of domain from i_c to $i_c + 2$ agent c has to make at least a full cycle of his domain and visit border twice thus $2i_c + 4$ steps are needed. Similarly to increase from i_c to $i_c + 2\alpha$, we need at least $\alpha(2i_c + 2\alpha)$. Thus to increase from i_c to $|I_c(t')|$ at least $(i_c + |I_c(t')|)\left\lfloor \frac{|I_c(t')| - i_c}{2} \right\rfloor$ steps are needed. Thus $t' - t^* \geq (i_c + |I_c(t')|)\frac{|I_c(t')| - i_c - 2}{2}$. On the other hand since $t' - t^* \leq 2|I_b(t^*)| + 5$. We have

$$2|I_b(t^*)| + 5 \geq (i_c + |I_c(t')|)\frac{|I_c(t')| - i_c - 2}{2} \geq (2i_c + |I_b(t')|)\frac{|I_b(t')| - 2i_c - 4}{8} \geq$$

$$\geq |I_b(t')|\frac{|I_b(t')| - 2i_c - 4}{8} \geq \frac{|I_b(t')|^2}{8} - \frac{|I_b(t^*)||I_b(t')| + 8|I_b(t')|}{16}.$$

Since a made one cycle in time $[t^*, t']$ then all nodes that b lost during this interval were taken by

c. Thus c had to make at least $(i_c+2)(|I_b(t^*)|-|I_b(t')|)$ steps.

$$2|I_b(t^*)|+5 \geq t'-t^* \geq (i_c+2)(|I_b(t^*)|-|I_b(t')|) \geq 22(|I_b(t^*)|-|I_b(t')|),$$

$$24|I_b(t')|+5 \geq 20|I_b(t^*)| \geq \frac{20|I_b(t')|^2-10|I_b(t')||I_b(t^*)|-80|I_b(t')|}{16}-100.$$

This also leads to contradiction since $|I_b(t')| \geq 11k \geq 55$ and $|I_b(t^*)| \geq 11k \geq 55$.

□

We now show two auxiliary lemmas which allow us to conclude that the sizes of all domains will eventually even out in time.

Lemma 2.6. *If $k \geq 5$ and initially the interior of every domain has size at least $22k$ and $|I_a(t')| > 1.4|I_b(t')|$, then in step $t'+1$ the border between a and b will not move towards b.*

Proof. Assume by contradiction, that a moves the borders in step $t'+1$. Let t^* be the last time step, when b visited its other border. Using the same argument as in the proof of Lemma 2.5, we have $2|I_a(t')|+4 \leq t'-t^* \leq 2|I_b(t^*)|+5$. It is possible, that $|I_b(t^*)| > |I_b(t')|$ if b lost some nodes during time interval $[t^*,t']$. But this number of nodes is limited since the size of every domain is at least $11k$ (from Lemma 2.5). Thus station b loses at most 2 nodes once every $22k$ time steps. Thus during $2|I_b(t^*)|+5$ time steps, b lost at most $\frac{2|I_b(t^*)|+5}{11k}+2$ nodes. Thus

$$|I_b(t')| \geq |I_b(t^*)|\left(1-\frac{2}{11k}\right)-\frac{5}{11k}-2,$$

$$|I_b(t^*)| \leq \left(I_b(t')+\frac{5}{11k}+2\right)\left(1+\frac{2}{11k-2}\right).$$

Since $k \geq 4$, and $|I_b(t')| \geq 11k \geq 22$, then $|I_b(t^*)| \leq 1.22|I_b(t')|$. Thus

$$t'-t^* \leq 2|I_b(t^*)|+5 < 2.44|I_b(t')|+5 < 2.67|I_b(t')| < 2|I_a(t')|+4 \leq t'-t^*.$$

And we obtain a contradiction. □

Lemma 2.7. *If $|I_a(t^*)|-4 > |I_b(t^*)|$ for a sufficiently large number of consecutive time steps, then the border between a and b will move towards a.*

Proof. Any full cycle of agent a takes at least $2|I_a(t^*)|$. Any full cycle of b takes at most $2|I_b(t^*)|+6$ (visiting the whole interior twice and both borders) time steps. Thus if $2|I_a(t^*)| > 2|I_b(t^*)|+8$, then the cycle of b is shorter by at least two steps. Thus after a sufficiently large number of time steps b will visit the border twice in some time interval $[t_1,t_2]$ and a will not visit the border in this time interval. Thus b will move the border towards a and gain two nodes. □

From our considerations, we obtain the lemma which will prove crucial in characterizing the limit behavior of the rotor-router on the ring.

Lemma 2.8 (**agent domains**). *If at some time step t every domain has size at least $22k+2$ and $k \geq 5$, then after a sufficiently large number of steps the interiors of adjacent domains will differ by at most 7.*

Proof. If domains have sizes at least $22k + 2$ in step t, then we can define interiors and borders so that every interior has size at least $22k$. We already know from Lemma 2.5 that if initially every interior has size at least $22k$, then during the deployment every domain will have size at least $11k$. If a and b are neighbors and at time t^* $|I_a(t^*)| \geq 2|I_b(t^*)|$, then we will say that there is a *significant difference* between a and b. If there is a significant difference between two adjacent domains of a and b, then by Lemma 2.6, the border will never move towards the smaller domain and by Lemma 2.7, the border will eventually move towards the bigger domain. Thus, significant differences will eventually disappear. Now, if there is no significant difference between any interiors of neighboring domains, then by Lemma 2.5, the border can move in the wrong direction (towards the smaller domain) only if the difference is at most 7. On the other hand, by Lemma 2.7, if the difference is at least 4 for a sufficiently large number of consecutive time steps, then the border will move towards the bigger domain. Thus, if the difference between the sizes of two adjacent interiors is at least 8, then the border cannot move in the wrong direction and will eventually move in the correct direction. Thus, finally if the sizes of each domain are initially at least $22k + 2$, then after some number of time steps, the interiors of adjacent domains will differ by at most 7. □

2.3 Continuous-time approximation

To provide an asymptotic description of the behavior of agent domains in time, we introduce the continuous-time approximation of the agents' behavior. This is useful under the assumption that the sizes of all the domains are sufficiently large, i.e., that the change of size of $V_i(t)$ in the number of rounds of the order $|V_i(t)|$ is negligible with respect to $|V_i(t)|$.

Suppose that the domains of the agents are ordered along the ring as $V_0(t), V_1(t), \ldots, V_k(t)$. Assuming that only the i-th agent is moving, the agent will reach each of the endpoints of its domain every $1/(2|V_i(t)|)$ rounds. Consequently, within T rounds, the agent enlarges its domain by approximately $T/(2|V_i(t)|)$ to the left, and $T/(2|V_i(t)|)$ to the right, thus by about $T/|V_i(t)|$ in total. This movement is counteracted by the moves of the adjacent agents occupying domains V_{i-1} and V_{i+1}. Consequently, we define the *continuous-time approximation* of the rotor-router through the set of differential equations:

$$\frac{d\nu_i(t)}{dt} = \frac{1}{\nu_i(t)} - \frac{1}{2\nu_{i-1}(t)} - \frac{1}{2\nu_{i+1}(t)}, \qquad \text{for } 1 \leq i \leq k,$$

where $\nu_i(t) = |V_i(t)|$, for all $1 \leq i \leq k$. The interpretation of $\nu_0(t)$ and $\nu_{k+1}(t)$ depends on whether the whole ring has already been covered: if so, then $\nu_{k+1}(t) \equiv \nu_1(t)$ and $\nu_0(t) \equiv \nu_k(t)$; if not, i.e., if $|V_0(t)| > 0$, then we put $\nu_0(t) = \nu_{k+1}(t) = +\infty$.

Whereas the above differential model provides the basic intuition for many of the proofs, the main difficulty lies in taking into account the differences between the continuous-time model and the real rotor-router. In particular, we have to consider the position of the agent within its domain, the discrete changes of the domain size in time, and the initial pointer arrangement in the unvisited part of the ring.

3 Cover Time of The Multi-agent Rotor Router on the Ring

3.1 Worst-case initial placement

The following lemma introduces a sequence $\{a_i\}_{i=0}^{k+1}$, useful in analyzing initial placements in which all agents start from the same point of the ring. It corresponds to a normalized solution to the

continuous-time model of the rotor-router (i.e., $a_i(t) = \nu_i(t)/\sum_j \nu_j(t)$), subject to the constraint that the proportions of domain sizes do not change in time (i.e., $\frac{da_i(t)}{dt} = 0$), and specific boundary conditions.

Lemma 3.1. *For any $k > 3$ there exists a sequence of positive numbers $(a_0, a_1, \ldots, a_k, a_{k+1})$ which satisfies the following properties:*

(1) $a_0 = +\infty$,

(2) $a_{k+1} = a_k < a_{k-1} < \ldots < a_1$,

(3) $\sum_{i=1}^{k} a_i = 1$,

(4) $a_i \cdot a_1 = \frac{2}{a_i} - \frac{1}{a_{i-1}} - \frac{1}{a_{i+1}}$, *for all* $1 \leq i \leq k$,

(5) $\frac{1}{4(H_k+1)} \leq a_1 \leq \frac{1}{H_k}$, *where* $H_k = 1 + \frac{1}{2} + \ldots + \frac{1}{k}$ *denotes the k-th harmonic number,*

(6) $\frac{1}{4i(H_k+1)} \leq a_i$, *for all* $1 \leq i \leq k$.

Proof. For a fixed $c > 0$, consider the recursively defined sequence $\{b_i(c)\}_{i=0}^{+\infty}$: $b_0 = 0$, $b_1 = c$, $b_{i+1} = 2b_i - b_{i-1} - \frac{1}{b_i}$, where we write $b_i \equiv b_i(c)$ to simplify notation. Let $d_i = b_i - b_{i-1}$. Then, $d_1 = c$, and $d_{i+1} = d_i - \frac{1}{b_i}$. Expanding this recurrence, we have:

$$d_{i+1} = c - \left(\frac{1}{b_1} + \ldots \frac{1}{b_i}\right).$$

$$b_{i+1} = c - \left(\frac{1}{b_1} + \ldots \frac{1}{b_i}\right) + b_i = (i+1)c - \left(\frac{i}{b_1} + \frac{i-1}{b_2} + \ldots + \frac{1}{b_i}\right).$$

First, by a simple inductive argument we observe from the above that for sufficiently large values of $c = b_1$, arbitrarily many of the initial elements of sequences $\{b_i(c)\}$ and $\{d_i(c)\}$ are positive.

Next, fix $i \geq 3$ and suppose that $d_j > 0$, for all $1 \leq j \leq i$. Then:

$$b_i \leq ic.$$

Thus:

$$d_{i+1} \leq c - \left(\frac{1}{c} + \ldots \frac{1}{ic}\right) = c - \frac{H_i}{c}.$$

From the relation $d_{i+1} > 0$, we obtain $H_i < c^2$, so $i < e^{c^2+1}$. This implies that by adjusting $c \in (0, +\infty)$, we can arbitrarily choose the number of positive initial elements of sequence $\{d_i(c)\}$. Taking into account that $d_i(c)$, for any fixed index i, is a continuous function of the parameter c, by the intermediate value theorem, there must exist a value of c such that $d_{k+1}(c) = 0$, or equivalently, that $b_{k+1} = b_k$. From now on, we use this value of c, only. Observe that $d_{k+1} = 0$ implies that $c = \sum_{i=1}^{k} 1/b_i$.

Now, define $a_i \equiv 1/(cb_i)$, for all $0 \leq i \leq k+1$. Such a sequence $\{a_i\}$ immediately satisfies conditions (1), (2), and (3). Condition (4) is obtained directly by observing that $a_1 = \frac{1}{c^2}$ and applying the replacement $b_i = 1/(ca_i)$ to the defining recursion of $\{b_i\}$.

Condition (5) may be restated as $H_k \leq c^2 \leq 4(H_k + 1)$. We have already established that the first of these relations holds, since otherwise we would have $d_{k+1} < 0$.

We will first show by induction that $d_i > c - \frac{2H_{i-1}}{c}$ for all $1 \leq i \leq e^{c^2/4}$. Indeed, the claim holds for $i = 1$. Suppose it holds for all $1 \leq j \leq i$. Then:

$$b_j = \sum_{l=1}^{j} d_l > cj - \frac{2}{c}\sum_{l=1}^{j} H_{l-1} = cj - \frac{2}{c}(jH_j - j),$$

$$d_{i+1} = c - \sum_{l=1}^{i} \frac{1}{b_l} > c - \sum_{l=1}^{i} \frac{1}{cl - \frac{2}{c}(lH_l - l)}.$$

Since $i < e^{c^2/4}$, then for any $l < i$ we have:

$$H_l < \log l + 1 < \frac{c^2}{4} + 1,$$

$$\frac{2}{c}(lH_l - l) < l\frac{c}{2}.$$

Thus

$$d_{i+1} > c - \sum_{l=1}^{i} \frac{1}{cl - l\frac{c}{2}} = c - \frac{2H_i}{c},$$

and the inductive claim holds. Now if $i < e^{\frac{c^2}{4}-1}$, then:

$$d_i > c - \frac{2H_{i-1}}{c} > c - \frac{2\log i + 2}{c} > c - \frac{c^2}{2c} = \frac{c}{2}.$$

Thus $k > e^{\frac{c^2}{4}-1}$, and we have:

$$c^2 \leq 4(\log k + 1) \leq 4(H_k + 1).$$

Since $b_i \leq ic$ then $a_i \geq 1/(ic^2)$ thus sequence $\{a_i\}$ satisfies condition (6). □

We are now ready to analyse a specific initialization, for which the k-agent rotor-router covers the ring particularly slowly.

Theorem 3.2. *In the case when all the agents are initially placed at the same node v, a group of k agents explores the ring of size n in time $\Theta(\frac{n^2}{\log k})$ when $k < n^{1/11}$, when all pointers are initialized along the shortest path to v.*

Proof. Consider a scenario with K agents on an N-node ring. Since $C(R[K-1]) \geq C(R[K]) \geq C(R[K+1])$, and the cover time is also monotonous with respect to the size of the ring, without affecting asymptotic bounds we can assume that K is even an N is odd, i.e., $K = 2k$ and $N = 2n-1$. By induction, we can show that the number of agents at node v will be even at all times, and the arrangement of pointers on the ring (except for node v) is symmetric with respect to the axis of symmetry passing through v. Consequently, the cover time for the N-node ring with K agents is asymptotically the same as the cover time of a n-node path with k agents, starting from an initial placement of all agents on one of the end-points v of the path.

Let $R[k]$ be this deployment on the path P_n. We now propose a delayed deployment D of $R[k]$ in which, starting from a certain moment in time, the domains of all agents are separate. Let the domains be ordered along the path according to decreasing numbers, i.e., the agent with domain

V_k is the one located closest to the starting point v, while the agent with domain V_1 is the furthest from v, i.e., it is the only agent to explore previously unvisited nodes of the path. The goal of the formalization below is to define the delayed deployment so that the ratios of domain sizes satisfy $|V_i| \sim a_i$, for $k \geq i \geq 1$, throughout time.

We will identify the path P_n with the integer interval $[1, n]$ (with $v = 1$), and domains with subsets of this interval. For $k \geq i \geq 1$, let $p_i = \sum_{j=i}^{k} a_i$. For a given value S, $n \geq S > 0$, we will call a configuration of agents and pointers on the path a *desirable configuration of length* S if it has the following properties:

- The position of the i-th agent on the path is $v_i = \lfloor p_i S \rfloor$.
- Each agent is at the right endpoint of its domain, i.e., $V_k = [1, v_k]$ and $V_i = [v_{i+1} + 1, v_i]$ for $k - 1 \geq i \geq 1$.
- For all the nodes on the path (including those containing agents), except for node 1, the pointer points to the left (towards node 1).

The evolution of the delayed deployment D is defined in two phases, as follows:

- *Phase* A. Form a desirable configuration with $S_0 = \frac{n}{\sqrt{k \log k}}$. To achieve this, release the agents one-by-one, starting from agent 1 to agent k, and perform exactly $(\lfloor p_i S_0 \rfloor - 1)^2$ moves with each agent, so that each agent i occupies position $\lfloor p_i S_0 \rfloor$ and all pointers on the path point to the left.
- *Phase* B. For successive $j = 0, 1, \ldots$, iterate the following procedure, until the path has been covered. Starting from an initial desirable configuration of some length S_j, form a new desirable configuration of length $S_{j+1} = S_j + \lceil k^4 a_1 a_k \rceil + 12k$ as follows:
  - B1. Starting from the current desirable configuration, release all agents simultaneously for $\lceil 2k^4 a_k S_j \rceil$ rounds.
  - B2. Adjust the positions of the agents, so as to reach the desirable configuration of length S_{j+1}. To achieve this, release the agents one-by-one, starting from agent 1 to agent k, allowing each agent i to move until it has reached position $\lfloor p_i S_{j+1} \rfloor$.

We denote by T the cover time of deployment D, by A, the total number of rounds of Phase A, by B_1, the total total number of rounds of phase B1, and by B_2, the total number of rounds of phase B2. We also remark that during phase B1 none of the agents is delayed, hence, by Lemma 2.3 we have:

$$B_1 \leq C(R[k]) \leq T = A + B_1 + B_2.$$

We begin by bounding time A. The agents are released sequentially in Phase A. The number of rounds required for each agent to reach its position is less than $\frac{n^2}{k \log k}$. Thus, $A < \frac{n^2}{\log k}$.

We now proceed to phase B (see Fig. 1 in the Appendix for an illustration). The size of the smallest domain in configuration S_0 is:

$$\left\lfloor \frac{n}{\sqrt{k \log k}} a_k \right\rfloor \geq \left\lfloor \frac{n}{\sqrt{k \log k}} \frac{1}{4(H_k + 1)k} \right\rfloor \geq \left\lfloor \frac{k^{11}}{k^{3/2} \log k \, (4 \log k + 8)} \right\rfloor \geq k^9,$$

where the last inequality holds for $k \geq 10^6$. Consider now the j-th step of the phase, starting from length $S = S_j$, and the change of the configuration within part B1 of this step. The number

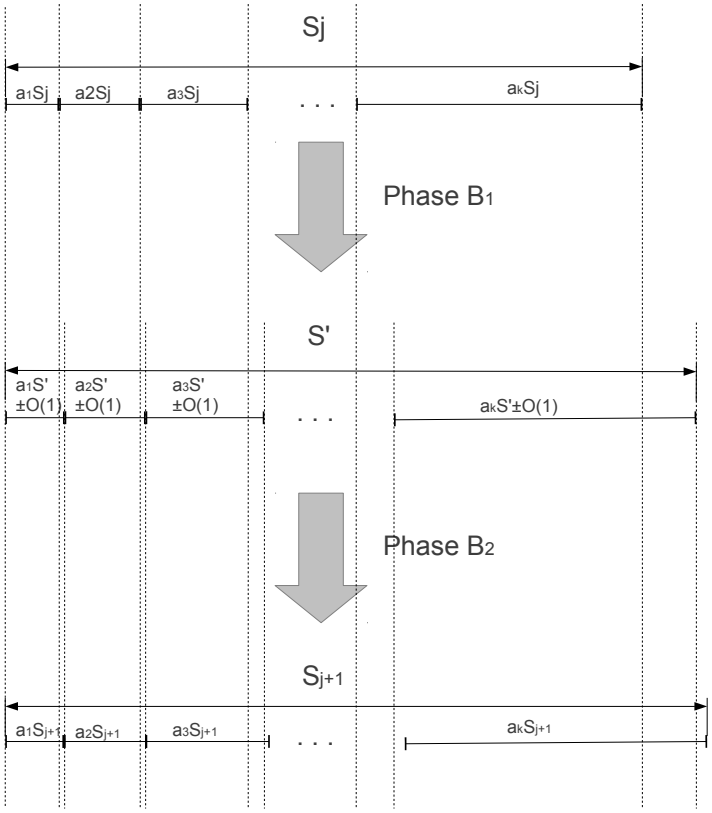

Figure 1: An iteration of Phase B of delayed deployment D (proof of Theorem 3.2)

of rounds used in part B1 of the step is $2a_kSk^4$. Let $|V_i|_j = \lfloor p_iS \rfloor - \lfloor p_{i+1}S \rfloor \geq a_iS - 1$ be the size of the domain of the i-th agent at the beginning of the j-th step, and let $|V_i|_j + g_i$ be its size after completion of part B1 of this step. In order to increase the size of its domain, the i-th agent needs to perform at least g_i traversals of its domain (such that during these traversals the size of this domain is at least $|V_i|_j$), where a traversal is understood as starting and ending at the right endpoint of the domain. These traversals require more than a_iSg_i rounds, whereas the total duration of part B1 of the j-th step is $\lceil 2a_kSk^4 \rceil$, hence we obtain $g_i < 2k^4$. Since the total size of all domains is non-decreasing in time, it follows that $\sum_{i=1}^{k} g_i \geq 0$, and so:

$$-2k^5 \leq g_i < 2k^4.$$

We now proceed to refine this bound on g_i. Initially, the size of the i-th domain is between $a_iS - 1$ and $a_iS + 1$. Thus, for the i-th agent, the number of completed traversals c_i of its domain during the considered part B1 is:

$$\frac{2a_kSk^4}{a_iS + 1 + 2k^4} \leq c_i \leq \frac{2a_kSk^4 + 1}{a_iS - 1 - 2k^5}.$$

If the i-th node performed c_i complete traversals, then it reached each of the boundaries of its domain at lest c_i times and one boundary could be reached $c_i + 1$ times. Thus, considering the change in size of domain g_i during the traversals of agents i, $i-1$ and $i+1$, we have:

$$2c_i - c_{i-1} - c_{i+1} - 2 \leq g_i \leq 2c_i + 1 - c_{i-1} - c_{i+1}$$

and introducing the bounds on c_i, c_{i-1}, c_{i+1} to the above:

$$2a_kSk^4\left(\frac{2}{a_iS+1+2k^4}-\frac{1}{a_{i-1}S-1-2k^5}-\frac{1}{a_{i+1}S-1-2k^5}\right)-2\leq g_i$$

$$g_i\leq\left(2a_kSk^4+1\right)\left(\frac{2}{a_iS-1-2k^5}-\frac{1}{a_{i-1}S+1+2k^4}-\frac{1}{a_{i+1}S+1+2k^4}\right)+1.$$

$$2a_kSk^4\left(\frac{2}{a_iS}\left(1-\frac{2k^4+1}{a_iS+1+2k^4}\right)-\frac{1}{a_{i-1}S}\left(1+\frac{2k^5+1}{a_{i-1}S-1-2k^5}\right)-\frac{1}{a_{i+1}S}\left(1+\frac{2k^5+1}{a_{i+1}S-1-2k^5}\right)\right)-2\leq g_i$$

$$g_i\leq 2a_kSk^4\left(\frac{2}{a_iS}\left(1+\frac{2k^5+1}{a_iS-1-2k^5}\right)-\frac{1}{a_{i-1}S}\left(1-\frac{2k^4+1}{a_{i-1}S+1+2k^4}\right)-\frac{1}{a_{i+1}S}\left(1-\frac{2k^4+1}{a_{i+1}S+1+2k^4}\right)\right)+2$$

We know that $a_iS\geq k^9$ and $a_k\leq a_i$

$$2a_kk^4\left(\frac{2}{a_i}\left(1-\frac{2}{k^5}\right)-\frac{1}{a_{i-1}}\left(1+\frac{2}{k^4}\right)-\frac{1}{a_{i+1}}\left(1+\frac{2}{k^4}\right)\right)-3\leq g_i$$

$$g_i\leq 2a_kk^4\left(\frac{2}{a_i}\left(1+\frac{2}{k^4}\right)-\frac{1}{a_{i-1}}\left(1-\frac{2}{k^5}\right)-\frac{1}{a_{i+1}}\left(1-\frac{2}{k^5}\right)\right)+3$$

$$2a_kk^4\left(\frac{2}{a_i}-\frac{1}{a_{i-1}}-\frac{1}{a_{i+1}}\right)-11-\frac{8}{k}\leq g_i\leq 2a_kk^4\left(\frac{2}{a_i}-\frac{1}{a_{i-1}}-\frac{1}{a_{i+1}}\right)+11+\frac{8}{k}$$

$$2a_ia_kk^4a_1-11-\frac{8}{k}\leq g_i\leq 2a_ia_kk^4a_1+11+\frac{8}{k}$$

The above analysis shows that the position of the i-th agent after phase B1 is upper-bounded by:

$$\lfloor p_iS\rfloor+\sum_{l=i}^{k}g_l<\lfloor p_iS\rfloor+\sum_{l=i}^{k}(a_la_1a_kk^4+11+8/k)\leq p_iS+p_ia_1a_kk^4+11k+8\leq\lfloor p_iS_{j+1}\rfloor-k+9<\lfloor p_iS_{j+1}\rfloor.$$

and lower-bounded by:

$$\lfloor p_iS\rfloor+\sum_{l=i}^{k}g_l\geq\lfloor p_iS\rfloor+\sum_{l=i}^{k}(a_la_1a_kk^4-11-8/k)\geq p_iS+p_ia_1a_kk^4a_1-11k-9\geq$$
$$\geq\lfloor p_iS_{j+1}\rfloor-23k-9>\lfloor p_iS_{j+1}\rfloor-24k.$$

Now, consider the duration of the phase B2. Each agent must adjust its position to the right, by a distance of at most $24k$. First, the right-most agent (agent 1) has to perform at most $24k$ traversals of its domain. As a result, the size of the domain of the penultimate agent (agent 2) can decrease by at most $24k$, hence it must perform at most $24k+24k=48k$ traversals to reach its position at the end of the step. In general, agent i has to perform at most $24ki\leq 24k^2$ traversals of its domain. The size of the i-th domain during phase B2 is at most $a_iS+2k^4+48k^2$. Thus, the duration of phase B2 is bounded by:

$$\sum_{i=1}^{k}\left(a_iS+2k^4+48k^2\right)24k^2<24Sk^2+48k^7+1152k^5.$$

Observe that the duration of part B1 of the step was $2a_kSk^4\geq\frac{2}{4k(H_k+1)}Sk^4>24Sk^2$ for $k>10^3$, because $a_k\geq\frac{1}{4k(H_k+1)}$ from Lemma 3.1. Thus, overall we have that the execution of B1 dominates

the complexity of the algorithm, $B_1 \in \Omega(B_2)$ and $B_1 \in \Omega(A)$. It follows that $C(R[k]) = \Theta(B_1)$. Now, in order to bound time B_1, observe that the j-th step of phase B results in the increase of S_j, the number of already covered nodes, by $\Theta(k^4 a_1 a_k)$, which means that phase B consists of $\Theta\left(\frac{n}{k^4 a_1 a_k}\right)$ steps. Since more than half of these steps are performed for $n/2 < S_j < n$, we obtain a tight bound on the cover time $B_1 \in \Theta(\frac{n^2}{a_1})$. Noting that $a_1 = \Theta(\frac{1}{H_k})$ by Lemma 3.1, we eventually obtain $B_1 \in \Theta(\frac{n^2}{\log k})$. Thus, $C(R[k]) \in \Theta(\frac{n^2}{\log k})$. □

We now show that the initialization considered above, with all agents starting from one node and all ports pointing to the left, is indeed asymptotically the worst possible. The proof of this theorem proceeds in two steps, first by considering agents starting from one node with an arbitrary placement of pointers on the ring, and then by extending this result to the general case through the application of delayed deployments.

Lemma 3.3. *In the case when all the agents are initially placed at the same node v, a group of k agents explores the ring of size n in time $O(\frac{n^2}{\log k})$ when $k < n^{1/11}$, regardless of the initial placement of pointers.*

Proof. We extend the proof of the upper bound from Theorem 3.2 to different initializations of pointers. We consider the case of the rotor-router deployment $R[k]$ on the n-node path with all agents initially positioned at the left endpoint of the path (but with arbitrary pointer initialization along the path). As in the proof of Theorem 3.2, we consider a delayed deployment with similarly defined phases A and B, using the same set of desirable configurations of length S_j. Note that in a desirable configuration, all the pointers along the path point to the left for all nodes which have already been visited by an agent at least once. In phase A, agents are released one-by-one, until the i-th agent reaches position $\lfloor p_i S_0 \rfloor$, after which the agent is stopped (this may happen after a smaller number of steps than in the proof of Theorem 3.2). In the j-th step of phase B, the only difference concerns the definition of part B1, where we add the condition that, upon reaching position $\lfloor p_i S_{j+1} \rfloor$ for the first time, the i-th agent stops and waits for the other agents to complete part B1 of the step. By induction, one can show that for $i > 1$, agent i will only stop moving in part B1 after agent $i-1$ has stopped moving, and consequently, it may never happen that a moving agent meets a stationary agent. The analysis of the time spent within parts A, B1 and B2 is performed as before, and we obtain $C(R[k]) = A + B_1 + B_2 = O(\frac{n^2}{\log k})$.

The analysis on the ring proceeds by a modification of the argument for a path, treating the ring as two sub-paths connected at the common node 1. In phase B, the deployments on both sub-paths are synchronized so that the agents a_k of the respective deployments arrive at node 1 simultaneously. If agent a_k of one of the sub-paths, say the left one, arrives before the agent a_k of the right sub-path, then all the agents of the left sub-path are stopped at their current locations until the other agent a_k arrives at node 1. (Note that the two sub-paths do not have to be performing the same step j of phase B at the same time.) This transformation of the deployment on the path does not affect asymptotic analysis, hence the cover time of the deployment on the ring is also $O(\frac{n^2}{\log k})$. □

Theorem 3.4. *For any initialization of the k-agent rotor-router system on the ring, the cover time is $O(\frac{n^2}{\log k})$, for $k \in O(poly(n))$.*

Proof. Let $R[k]$ be a deployment of the rotor-router on the ring. Fix a subset $P \subset V$ of $P = k^{2/3}$ points on the ring which are evenly spaced, i.e., $G[V \setminus P]$ is a set of disjoint paths of length at

most $n/k^{2/3}$. Consider a delayed deployment of $R[k]$, which begins with a phase in which the agents of $R[k]$ are activated and moved one by one, stopping each agent as soon as it has reached a node from P. Since the cover time of a path of length $O(n/k^{2/3})$ for a single agent is $O(n^2/k^{4/3})$, the duration of this phase is at most $O(n^2/k^{1/3})$. After this initial phase, by the pigeon-hole principle, there must exist a node $v \in P$ which contains $k' \geq k^{1/3}$ agents. We now continue the delayed deployment by releasing $k'' = \min\{k^{1/3}, n^{1/11}\}$ agents which are located at v, and permanently stopping (removing) all other agents. By Theorem 3.2, the path will be covered by the delayed deployment within $O\left(\frac{n^2}{\log k''}\right)$ rounds. By summing the duration of the two phases and using the slow-down lemma, we obtain the claim: $C(R[k]) \in O\left(\frac{n^2}{k^{1/3}} + \frac{n^2}{\log k''}\right) = O(\frac{n^2}{\log k})$, for $k \in O(poly(n))$. □

3.2 Best-case initial placement

We start by proposing the initialization with agents equally spaced along the path as a candidate for (asymptotically) best-case initial placement with $O\left((\frac{n}{k})^2\right)$ cover time. The proof is straightforward in the case if we assume that the adversary initially directs all pointers towards the nearest agent, so as to block it. However, the adversary may apply a different strategy, and there do indeed exist port arrangements which deflect agents from some section of the ring, leading to a larger value of cover time. In our proof we show such actions of the adversary do not affect the asymptotics of the cover time.

Theorem 3.5. *Consider an initialization of the rotor-router system on the ring with agents starting on a set of points $P = \{p_1, p_2, \ldots, p_k\}$, such that $G[V \setminus P]$ is a set of paths of length at most n/k. Then, the system covers all of the nodes of the ring in $O\left((\frac{n}{k})^2\right)$, regardless of the initial pointer arrangement.*

Proof. W.l.o.g, let $1 \leq p_1 < p_2 < \ldots < p_k \leq n$. Given a fixed initial pointer arrangement, let $x \in [1, n]$ be the node which is visited last by the rotor-router. To prove the claim, by the slow-down lemma, it suffices to construct a delayed deployment D of the rotor-router such that point x is visited by some agent within $O\left((\frac{n}{k})^2\right)$ rounds. We define deployment D as follows. Initially, we release all agents simultaneously, so that each agent moves left while the pointer of its current node points to the left, and stops as soon as it encounters a node whose pointer points to the right. Let q_i denote the position of the agent starting from p_i after this phase is complete; we have $p_i - n/k \leq q_i \leq p_i$, hence the duration of this phase is at most n/k. We also have $|q_{i+1} - q_i| \leq 2\lceil n/k \rceil$. After this initialization phase, the deployment proceeds in steps of duration $4\lceil n/k \rceil$. The deployment is defined so that at the start of each step, agent i is located at point q_i. We describe the deployment through the following procedure, performed simultaneously by each agent i. The agent moves (to the right), stopping when it has either reached point q_{i+1}, or a node whose pointer points to the left. It then waits until the end of the $2\lceil n/k \rceil$-th round of the step to synchronize with other agents, and then returns to node q_i, where it waits until the end of the step.

We observe that in each step such that agent i does not reach q_{i+1}, it reaches a node on the path $[q_i, q_{i+1}]$ which has not previously been visited by any agent. Suppose that i is such that $q_i < x < q_{i+1}$. It follows that node x will be visited by agent i within $|q_{i+1} - q_i| \leq 2\lceil n/k \rceil$ steps. Since the duration of each step is $4\lceil n/k \rceil$, the second phase of the delayed deployment takes at most $8\lceil n/k \rceil^2$ round. Overall, point x is covered within $O\left((\frac{n}{k})^2\right)$ rounds from the start of the process, and the claim follows. □

To prove that the equally-spaced initialization is the best possible, we provide a general case lower-bound of $\Omega\left((\frac{n}{k})^2\right)$ on cover time for all initializations. To do this, we introduce an auxiliary notion of a *good vertex* for an initialization of the rotor-router. Such vertices are shown to always exist (in fact, to be in the majority) and take a long time to cover, regardless of the initial placement of agents.

Definition 3.1. *For any placement of the k agents let $S = \{s_1, s_2, \ldots, s_k\}$ be the k not necessarily distinct starting vertices. We will consider the subset of* good vertices *of the cycle, defined as all nodes v which satisfy the following two constraints:*

1. *For all $1 \leq r \leq k$, $\left|\left[v, v + r\frac{n}{10k}\right] \cap S\right| \leq r$.*
2. *For all $1 \leq r \leq k$, $\left|\left[v, v - r\frac{n}{10k}\right] \cap S\right| \leq r$.*

The following lemma concerning the relation between good vertices and the starting positions of the agents, and proves useful in the analysis of the k-agent rotor-router, as well as the k-agent random walk.

Lemma 3.6. *For any initial placement $S = \{s_1, s_2, \ldots, s_k\}$ of the k agents, there are at least $0.8n - o(n)$ good vertices.*

Proof. Let V_1 and V_2 be the sets of vertices which satisfy constraints 1 and 2 above, respectively. We first show that $|V_1| \geq 0.9n - o(n)$. Consider an algorithm which starts from vertex 0 and scans the cycle in the increasing order of vertex numbers, as follows:

1. $v \leftarrow 0$
2. $B \leftarrow \emptyset$
3. While $(v < n - (n/10k))$, repeat:
   If $v \notin V_1$, then:
   (a) Let r be the smallest positive integer such that $|[v, v + r(n/k)) \cap S| > r$.
   (b) $B \leftarrow B \cup [v, v + r(n/10k))]$
   (c) $v \leftarrow v + r(n/k)$

   else $v \leftarrow v + 1$.

By the construction of set B, each new interval of the form $[v, v + r(n/10k))$, of length $r(n/10k)$ which is added to it, contains more than r elements of set S. Consequently: $|B \cap S| > 10k|B|/n$, so $|B| < 0.1n|S|/k = 0.1n$. On the other hand, we observe that for $v < n-(n/10k)$, $v \notin B \implies v \in V_1$, and so $|V_1| \geq n - o(n) - |B| \geq 0.9n - o(n)$. By a similar argument, we show that $|V_2| \geq 0.9n - o(n)$. From here, we obtain the sought bound on the number of good vertices: $|V_1 \cap V_2| \geq 0.8n - o(n)$. □

Theorem 3.7. *If $n \geq 440k^2$ and $k \geq 5$, then for any set of initial locations of k agents, there exists an initial arrangement of pointers on the ring such that the cover time of the rotor-router system is $\Omega\left((\frac{n}{k})^2\right)$.*

Proof. Let $S = \{s_1, s_2, \ldots, s_k\}$ be the k not necessarily distinct starting vertices. Let r_i be the number of vertices initially between s_i and s_{i+1}, and r_k be the number of vertices between s_k and s_1. Obviously $\sum_i r_i \geq n - k$. Thus $\sum_{\{i:r_i \geq \frac{n-k}{2k}\}} r_i \geq \frac{n-k}{2}$. If we take two middle quarters from each interval of length at least $\frac{n-k}{2k}$ then totally we will obtain at least $\frac{n-k}{4}$ nodes. Thus at least $n/4 - o(n)$ nodes are at distance at least $\frac{n-k}{8k}$ to the closest agent. If $n \geq 9k$ then $\frac{n-k}{8k} \geq \frac{n}{9k}$. Thus from Lemma 3.6 there are at least $0.05n - o(n)$ good nodes at distance at least $\frac{n}{9k}$ to the closest starting point of an agent. For sufficiently large n such node will exist. We will call this node v. Now we will use Lemma 2.1 and construct a delayed deployment $D1$. We will block all but one or two agents to ensure that each agent will have a domain of size at least $\frac{n}{20k}$ and at least $\frac{n}{10k}$ nodes will not be explored. We initiate all pointers negatively – in each node the pointer points away from the closest agent. We will describe the procedure in one direction. In the other direction procedure will be the same. Firstly we release the closet agent at the left of v until it reaches the node at distance $\frac{n}{20k}$ from v. Then we block the agent. Since the closest node to v is at distance at least $\frac{n}{9k}$ then after this procedure in interval $[v + \frac{n}{20k}, v + \frac{n}{10k}]$ there will be only one agent. Then we take the next closest agent at the left of v and release it until it reaches node $v + \frac{n}{10k}$. Again since v is a good node there will be only one agent in interval $[v + \frac{n}{10k}, v + \frac{n}{5k}]$. Then for i-th closest agent at the left of v for $i \geq 2$ we release it until it reaches node $v + (i-1)\frac{n}{10k}$. It is possible, that the agent will go to the other side of the ring. Then we block it at the node $v + \frac{n}{2}$ and continue procedure. We do the same procedure to the left and right from v. We end up with some agents at node $v + \frac{n}{2}$. We release them one-by-one. Assume that such agent a went to the left from v. We block him, when he is at distance $\frac{n}{10k}$ from the last agent placed to the left of v. Now each agent has a domain of size at least $\frac{n}{20k}$. Now we release all agents simultaneously. From Lemma 2.5 size of any domain will not drop below $\Omega(\frac{n}{20k})$. Assumptions of Lemma 2.5 are satisfied, because $\frac{n}{20k} \geq 22k + 2$. We also have a group of $\frac{n}{20k}$ not explored nodes. Since this group will be explored by agents having domains of sizes at least $\Omega\left(\frac{n}{20k}\right)$ it will take at least $\Omega\left(\frac{n^2}{400k^2}\right)$ time steps. Number of steps in $D1$ when all agents are released simultaneously is $\Omega\left(\frac{n^2}{k^2}\right)$, thus from Lemma 2.1 the cover time of not delayed k agents in the rotor-router model in this case will also be $\Omega\left(\frac{n^2}{k^2}\right)$. □

3.3 Comparison with the Random Walk

The question of the cover time of random walks starting from a worst-case initial placement has already been resolved in the literature. On the one hand, it is known that the speed-up of cover time for a k-agent random walk with respect to the single agent case is $\Omega(\log k)$ for any graph whose cover time is asymptotically equal to the maximum hitting time [3], regardless of the initial placement of agents. Since this is clearly the case for the ring [1], we have that the cover time of the k-agent random walk is $O(n^2/\log(k))$. On the other hand, the adversary may choose to place all agents at one node of the ring. Such an all-one-one initialization has a cover time of precisely $\Theta(n^2/\log(k))$ [3]. Thus, the cover time for k random walks on the ring with worst-case initialization is $\Theta(n^2/\log(k))$.

To establish an upper bound for the best-case scenario, we consider k random walks with initial positions given with equal spacing, i.e., with offsets $0, n/k, 2(n/k), \ldots, (k-1)\,(n/k)$ relative to some node. (For simplicity, we assume here that k divides n.) The following lemma implies that in this case the cover time is $O((n/k)^2)$.

Lemma 3.8. *Let $\alpha \geq 20$, $k \geq 2$ and let $t := \alpha^2 \cdot (n/k)^2 \cdot \log^2(k)$. Then, with probability at least $1 - k^{1-\alpha/20}$, k random walks starting from initial positions with equal spacing cover all the vertices*

of the ring within t steps.

Proof. Using standard results from probability, it follows that any random walk visits a vertex which is at least $\frac{1}{5} \cdot \sqrt{t}$ to the right of its starting vertex within t steps with probability at least $1/4$. Note that for any vertex $u \in V = \{0, \ldots, n-1\}$, there are at least $x-1$ random walks with distance between (n/k) and at most $x \cdot (n/k)$ to u. Putting $x = \frac{1}{10} \cdot \sqrt{t}/(n/k)$ we obtain the following upper bound for the event that u will *not* be covered:

$$\left(1 - \frac{1}{4}\right)^{\frac{1}{10} \cdot \sqrt{t}/(n/k) - 1} = \left(1 - \frac{1}{4}\right)^{\frac{\alpha}{10} \cdot \log(k) - 1} \leq k^{-\alpha/20}.$$

Now note that if for any vertex u of the set $S := \{0, n/k, 2(n/k), \ldots, (k-1)\,(n/k)\}$ there is a random walk that is initially placed to the left of u with distance at most $\frac{1}{10} \cdot \sqrt{t}/(n/k)$ and which traverses at least $\frac{1}{5} \cdot \sqrt{t}$ steps to the right within the first t steps, then all vertices are covered after t steps. Hence by taking the union bound over the set S we conclude that all vertices of the ring are covered with probability at least

$$1 - k \cdot k^{-\alpha/20} = 1 - k^{1-\alpha/20}.$$

□

We now prove a corresponding lower bound on the cover time in the best-case scenario, showing that the position with equal spacing is asymptotically the best possible. We first prove an auxiliary result which relies on the notion of good vertices introduced in the previous subsection.

Lemma 3.9. *Let $t = 10^{-4} \cdot (n/k)^2 \cdot \log^2(k)$, and u be any good vertex at distance at least $\frac{n}{10k}$ from the starting points of all random walks. Then, with probability at least $k^{-1/2}$, u is not covered after t steps by any of the k random walks.*

Proof. Consider first a random walk with distance $(n/k)/10 \leq d \leq 4 \cdot \sqrt{t}$ to u. The probability that the random walk reaches a point with distance at least $4 \cdot \sqrt{t}$ to u without visiting u before is equal to

$$\frac{d}{4 \cdot \sqrt{t}}.$$

Once the random walk has distance $4 \cdot \sqrt{t}$ to u, the probability that it does not visit u within t steps is at least $1/2$. Combining these insights, we obtain that a random walk with distance $d \leq 4 \cdot \sqrt{t}$ does not visit the vertex u within t steps with probability at least

$$\frac{d}{4 \cdot \sqrt{t}} \cdot \frac{1}{2}.$$

Consider now all random walks with distance less than $4 \cdot \sqrt{t}$. The number of these random walks

is $4 \cdot \sqrt{t}/(n/k) = (1/25) \log k$. The probability that none of these random walks covers u is at least

$$\begin{aligned}
\prod_{j=0}^{(1/25)\log k - 1} \frac{\frac{n}{10k} + j \cdot \frac{n}{10k}}{4 \cdot \sqrt{t}} \cdot \frac{1}{2} &\geq 4^{-(1/25)\log k} \prod_{j=1}^{(1/25)\log k} \frac{j \cdot \frac{n}{10k}}{\sqrt{t}} \\
&\geq 4^{-(1/25)\log k} \cdot \prod_{j=1}^{(1/25)\log k} \frac{j}{(1/10)\log k} \\
&\geq 10^{-(1/25)\log k} \cdot \prod_{j=1}^{(1/25)\log k} \frac{j}{(1/25)\log k} \\
&\geq 10^{-(1/25)\log k} \cdot \frac{((1/25)\log k)!}{((1/25)\log k)^{(1/25)\log k}} \\
&\geq 10^{-(1/25)\log k} \cdot e^{-(1/25)\log k},
\end{aligned}$$

where the last line follows from Stirling's approximation.

For a random walk with distance $d = c \cdot \sqrt{t}$, $c \geq 4$ to u, the probability to visit u is at most $e^{-c/2}$. Hence the probability that u is visited by none of the random walks with distance at least $4 \cdot \sqrt{t}$ is lower bounded by

$$\begin{aligned}
\prod_{j=(1/25)\log k}^{k} \left(1 - e^{-\frac{j \cdot \frac{n}{k}}{\sqrt{t}} \cdot \frac{1}{2}}\right) &= \prod_{j=(1/25)\log k}^{k} \left(1 - e^{-\frac{j}{\frac{1}{100}\log(k)} \cdot \frac{1}{2}}\right) \\
&= \prod_{j=(1/25)\log k}^{k} \left(1 - e^{-\frac{50j}{\log(k1)}}\right) \\
&= \prod_{j=(1/25)\log k}^{k} \left(1 - e^{-\frac{50j}{\log(k)}}\right)^{e^{\frac{50j}{\log(k)}} \cdot e^{-\frac{50j}{\log(k)}}} \\
&\geq e^{-\sum_{j=(1/25)\log k}^{k} e^{-\frac{50j}{\log(k)}}} \\
&\geq e^{-\frac{e^{-\frac{2}{\log(k)}}}{1 - e^{-50/\log(k)}}} \\
&\geq e^{-\frac{\log(k)}{50}} = k^{-1/50}.
\end{aligned}$$

Hence none of the k random walks will visit u with probability at least

$$10^{-(1/25)\log k} \cdot e^{-(1/25)\log k} \cdot k^{-1/50} \geq k^{-1/2}$$

□

The lower bound on cover time is completed when we prove the existence of a good vertex satisfying the conditions of Lemma 3.9. We do this taking into account Lemma 3.6.

Lemma 3.10. *For arbitrary starting positions of k random walks, we need at least $\Omega((n/k)^2 \log^2 k)$ steps to visit all n vertices with probability at least $1/2$.*

Proof. Let $S = \{s_1, s_2, \ldots, s_k\}$ be the k not necessarily distinct starting vertices. Fix $t = 10^{-4} \cdot (n/k)^2 \cdot \log^2(k)$. We define intervals $I_i, 0 \leq i < k/\log^2 k$, of the form $I_i = [(i-1)(n/k)\log^2 k, i(n/k)\log^2 k)$. The length of the union of all intervals with even indices is $I = \bigcup_{0 \leq j < k/2\log^2 k} I_{2j}$, and $|I| = 0.5n - o(n)$.

If F is the set of good vertices, then by Lemma 3.6, $|F| \geq 0.8n - o(n)$. Let H be the set of all nodes at distance at least $(n/k)/10$ to node from S; we have $|H| \geq 0.8n$. Thus $|I \cap F \cap H| \geq 0.1n - o(n)$ and $|I \cap F \cap H| \geq 0.09n$ for sufficiently large n. Since each interval I_i is of length $(n/k)\log^2 k$, at least $0.09k/\log^2 k$ intervals with even indices must contain a good vertex satisfying assumptions of lemma 3.9. We pick one such vertex from each interval. In this way, we obtain set S of $0.09k/\log^2 k$ vertices, at pairwise distances of at least $(n/k)\log^2 k$ from each other.

We denote by Y the event that none of random walks reached a distance more than $40 \cdot \sqrt{t}\log k$ to its origin. We note that $\mathbf{Pr}[Y] \geq 1 - k^{-40}$. We also denote by X the event, that every vertex in S is explored in time t by k random walks. Note, that $\mathbf{Pr}[X|Y] \leq \left(1 - k^{-1/2}\right)^{\frac{k}{\frac{1}{10}\log^2(k)}}$ because if event Y happened, then each vertex $s \in S$ remains uncovered with probability at least $k^{-1/2}$ and these events are independent for different vertices in S. We know, that

$$\mathbf{Pr}[X] \leq \mathbf{Pr}[Y]\,\mathbf{Pr}[X|Y] + 1 - \mathbf{Pr}[Y] \leq \left(1 - k^{-1/2}\right)^{\frac{k}{\frac{9}{100}\log^2(k)}} + k^{-40} \leq 1/2$$

The last inequality holds for $k > 1$. □

Now, the characterization of the cover time of k random walks in the best-case scenario follows directly from Lemmas 3.8 and 3.10.

Theorem 3.11. *The cover time of k random walks on the ring for best-case initial placement is $\Theta((n/k)^2 \log^2 k)$.* □

4 Return Time of the Rotor-Router on the Ring

The considerations of the rotor-router in the previous section concerned the time required to cover all nodes in the initialization phase. As a deterministic system with a finite number of states, the rotor-router eventually reaches its limit behavior, cycling through a finite number of configurations. In this section, we characterize this limit behavior of the rotor-router using the concept of *return time*, i.e. the maximum over $v \in V$ of the length of the longest time interval during which v is not visited by any agent of the rotor-router system in its limit behavior. We show that this performance parameter of the rotor-router on the ring achieves the best possible value of $\Theta(\frac{n}{k})$, regardless of the initial placement of the agents.

Theorem 4.1. *If $k \in O\left(n^{1/6}\right)$ then after sufficiently large number of time steps, k rotor-routers will visit every node of the n vertex ring once every $\Theta(\frac{n}{k})$ time steps.*

Proof. In this proof we will make use of delayed deployments of agents. When analyzing delayed deployments, we apply a different definition of a domain: for each agent a_i, we define its domain $V(a_i)$ as the union of the two maximal sub-paths of the ring adjacent to the location $v(a_i)$, consisting only of nodes whose pointers point towards $v(a_i)$. Note that, once the whole ring has been explored, for $k > 1$, this definition is equivalent to the definition of domains for the undelayed deployment $R[k]$, and is indistinguishable from the point of view of its future evolution in time. In particular, all the lemmas from Section 2.2 hold unchanged.

We will denote the undelayed deployment by $R[k]$ and a specific delayed deployment by D. The considered deployment D consists of two phases. In the first phase, we selectively release (and delay) agents until the whole ring has been covered, and each agent has a domain of size at least $22k + 2$; details of the construction are provided later. In the second phase, we release all agents. By Lemma 2.8, the size of every domain will eventually converge to $O\left(\frac{n}{k}\right)$. Thus, in deployment D every node will eventually be visited once every $O\left(\frac{n}{k}\right)$ time steps.

It remains to be shown that the same holds for the undelayed deployment $R[k]$. Let θ be the total number of rounds during which not all agents are active in D. The construction of D will be such that $\theta \in O\left(\frac{n}{k}\right)$. Now, let t be a time step such that after t every node is visited at least once every $c\frac{n}{k}$ time steps by some agent following deployment D, for some constant $c > 0$. Take any $t^* > t$. We have that in D, every node is visited at least once in the time interval $\left[t^*, t^* + c\frac{n}{k}\right]$. By Lemma 2.3 we have that for any v, $n_v^{R[k]}\left(t^* - \theta\right) \leq n_v^D(t^*)$, and $n_v^{R[k]}\left(t^* + c\frac{n}{k}\right) \geq n_v^D\left(t^* + c\frac{n}{k}\right) > n_v^D\left(t^*\right)$. Thus, for any time t^*, some agent following $R[k]$ visits v within the time interval $[t^* - \theta, t^* + c\frac{n}{k}]$, which contains t^* and is of duration $O\left(\frac{n}{k}\right)$. It follows directly that the refresh time of $R[k]$ is $O\left(\frac{n}{k}\right)$.

It remains to describe the construction for the first phase of deployment D to achieve domains of size at least $22k + 2$, so that not all agents are active at the same time in at most $O\left(\frac{n}{k}\right)$ rounds. We proceed as follows. First, we release all agents until all nodes of the ring have been covered. Next, we release agents one by one, progressing the agent until it has reached a point located a distance of at least $44k + 6$ from the nearest agent. Since the longest sub-path consisting of agents, which do not have a gap of length at least $2(44k + 6) + 1$ between them, is $88k^2 + 13k$, and the moving agent is equivalent to a single-agent rotor-router system, the agent will reach the endpoint of such a sub-path (and complete its movements) within $(88k^2 + 13k)^2$ steps. In total, the moves of all agents in this stage of the construction require $O(k^5)$ rounds. In the next stage, we deploy the agents one by one, so that each agent is moved until it is located at a distance of precisely $22k + 2$ from its location at the beginning of this stage. By a similar analysis, the duration of this stage is $O(k^3)$. Note that during this stage no two agents meet or pass each other on an edge, and so each agent is adjacent to a path of length $22k + 2$ with ports arranged towards its current location. Hence, we have achieved $|V(a_i)| \geq 22k + 2$ within a total of $O(k^5)$ steps, which is $O(n/k)$ for $k \in O\left(n^{1/6}\right)$.

□

No strong analogue of the above theorem holds for a system with k random walks. The only property which can be bounded is the expected time between two successive visits to a node, which is precisely equal to n/k on the ring (since the stationary distribution of each of the k walks is uniform with probability $1/n$ on each node). However, the random variable which describes the expected time between visits to a node has high variance.

5 Conclusions

We have shown that the rotor-router and random walk have similar speed-up characteristics w.r.t. the number of agents, at least in terms of cover time and return time on the ring. It is interesting to note that the worst-case speed-up on the ring is $\Theta(\log k)$ for both the k-agent random walk and the k-agent rotor-router, even though in the first case it is a consequence of the properties of probability distributions of independent Markovian processes, and in the latter case, it results directly from the interactions between the agents. On the other hand, the best-case speed-up of $\Theta(\frac{k^2}{\log^2 k})$ for the random walk takes into account a poly-logarithmic factor which results from the

probabilistic nature of the walk, whereas the speed-up for the rotor-router is simply $\Theta(k^2)$.

The techniques developed in this paper, in particular based on delayed deployments, may prove useful in possible extensions of this study to general graphs.

References

[1] D. Aldous and J. A. Fill. Reversible Markov Chains and Random Walks on Graphs, 1995. http://stat-www.berkeley.edu/users/aldous/RWG/book.html.

[2] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovasz, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science*, FOCS'79, pages 218–223, Washington, DC, USA, 1979. IEEE Computer Society.

[3] N. Alon, C. Avin, M. Koucký, G. Kozma, Z. Lotker, and M. R. Tuttle. Many random walks are faster than one. In *SPAA*, pages 119–128. ACM, 2008.

[4] E. Bampas, L. Gasieniec, N. Hanusse, D. Ilcinkas, R. Klasing, and A. Kosowski. Euler tour lock-in problem in the rotor-router model. In *DISC*, volume 5805 of *LNCS*, pages 423–435, 2009.

[5] E. Bampas, L. Gasieniec, R. Klasing, A. Kosowski, and T. Radzik. Robustness of the rotor-router mechanism. In *OPODIS*, volume 5923 of *LNCS*, pages 345–358, 2009.

[6] S. N. Bhatt, S. Even, D. S. Greenberg, and R. Tayar. Traversing directed eulerian mazes. *J. Graph Algorithms Appl.*, 6(2):157–173, 2002.

[7] C. Cooper, D. Ilcinkas, R. Klasing, and A. Kosowski. Derandomizing random walks in undirected graphs using locally fair exploration strategies. *Dist. Comp.*, 24(2):91–99, 2011.

[8] J. N. Cooper and J. Spencer. Simulating a random walk with constant error. *Combinatorics, Probability & Computing*, 15(6):815–822, 2006.

[9] B. Doerr and T. Friedrich. Deterministic random walks on the two-dimensional grid. *Combinatorics, Probability & Computing*, 18(1-2):123–144, 2009.

[10] K. Efremenko and O. Reingold. How well do random walks parallelize? In *APPROX-RANDOM*, volume 5687 of *LNCS*, pages 476–489, 2009.

[11] R. Elsässer and T. Sauerwald. Tight bounds for the cover time of multiple random walks. In *ICALP (1)*, volume 5555 of *LNCS*, pages 415–426, 2009.

[12] P. Fraigniaud, D. Ilcinkas, G. Peer, A. Pelc, and D. Peleg. Graph exploration by a finite automaton. *Theoretical Computer Science*, 345(2-3):331–344, 2005.

[13] T. Friedrich and T. Sauerwald. The cover time of deterministic random walks. In *COCOON*, volume 6196 of *LNCS*, pages 130–139, 2010.

[14] L. Gasieniec and T. Radzik. Memory efficient anonymous graph exploration. In *WG*, volume 5344 of *LNCS*, pages 14–29, 2008.

[15] V. Priezzhev, D. Dhar, A. Dhar, and S. Krishnamurthy. Eulerian walkers as a model of self-organized criticality. *Phys. Rev. Lett.*, 77(25):5079–5082, Dec 1996.

[16] O. Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), 2008.

[17] T. Sauerwald. Expansion and the cover time of parallel random walks. In *PODC*, pages 315–324. ACM, 2010.

[18] I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein. Distributed covering by ant-robots using evaporating traces. *IEEE Trans. Robotics and Automation*, 15:918–933, 1999.

[19] V. Yanovski, I. A. Wagner, and A. M. Bruckstein. A distributed ant algorithm for efficiently patrolling a network. *Algorithmica*, 37(3):165–186, 2003.