





Black Hole Search in Dynamic Tori

Adri Bhattacharya  

Indian Institute of Technology Guwahati, Assam, India

Giuseppe F. Italiano  

Luiss University, Rome, Italy

Partha Sarathi Mandal  

Indian Institute of Technology Guwahati, Assam, India

Abstract

We investigate the black hole search problem using a set of mobile agents in a dynamic torus. A black hole is defined as a dangerous stationary node that has the capability to destroy any number of incoming agents without leaving any trace of its existence. A torus of size $n \times m$ ($3 \leq n \leq m$) is a collection of n row rings and m column rings, and the dynamicity is such that each ring is considered to be 1-interval connected, i.e., in other words at most one edge can be missing from each ring at any round. The parameters which define the efficiency of any black hole search algorithm are: the number of agents and the number of rounds (or *time*) for termination. We consider two initial configurations of mobile agents: first, the agents are co-located, second, the agents are scattered. In each case, we establish lower and upper bounds on the number of agents and on the amount of time required to solve the black hole search problem.

2012 ACM Subject Classification Theory of computation \rightarrow Distributed algorithms

Keywords and phrases Black Hole Search, Time Varying Graphs, Dynamic Torus, Distributed Algorithms, Mobile Agents

Digital Object Identifier 10.4230/LIPIcs.SAND.2024.6

Related Version *Full Version*: <https://arxiv.org/abs/2402.04746> [3]

Funding *Adri Bhattacharya*: Supported by CSIR, Govt. of India, Grant Number: 09/731(0178)/2020-EMR-I.

Acknowledgements This work was done while Partha Sarathi Mandal was in the position of Visiting Professor at Luiss University, Rome, Italy.

1 Introduction

Given a network and a set of mobile agents, the black hole search problem (also termed as BHS problem) consists of locating a malicious stationary node that has the power to eliminate any number of incoming agents without leaving any trace of its existence. This problem is not new, and it readily has many real-life implications. For example, the black hole may be a node infected with a virus in a computer network, and in order to make the network safe, the infected node should be located for further action. The first task for any set of mobile agents ought to be to locate the black hole. To accomplish this task, at least one agent needs to visit the node; we aim at an efficient BHS algorithm, where the minimum number of agents gets consumed by the black hole so that at least one agent must remain alive in order to locate the black hole within finite time. This problem has been extensively studied in networks which are static, see, e.g., [1, 6, 8, 9, 14, 15, 17, 18, 20]. Recently, research on black hole search problem has been mainly focused on dynamic networks; in particular, the most relevant dynamic networks studied are *time-varying graphs*. These networks work on temporal domains, which are mainly considered to be discrete time steps. More precisely, the network is a collection of static graphs, in which some edges may disappear or reappear



© Adri Bhattacharya, Giuseppe F. Italiano, and Partha Sarathi Mandal;
licensed under Creative Commons License CC-BY 4.0

3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2024).

Editors: Arnaud Casteigts and Fabian Kuhn; Article No. 6; pp. 6:1–6:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

at each discrete time step, while the vertex set is fixed, with the additional constraint that at each time step the underlying graph remains connected (also termed as *1-interval connected*). Presently, apart from the black hole search on a dynamic ring [11] and on a dynamic cactus [2], nothing much is known about the black hole search problem on dynamic networks.

In this paper, we investigate this problem on a dynamic torus of size $n \times m$ (where each ring is 1-interval connected and without loss of generality $3 \leq n \leq m$), where a set of agents synchronously perform the same execution, with the goal of locating the black hole. Moreover, we consider that each node in the underlying torus has a whiteboard, used by the agents to write/read some information used by the agents while executing a certain black hole search algorithm.

We study two types of initial configurations of agents, under the assumption that each agent executing the black hole search algorithm, can communicate among themselves when they are at the same node, and they can also use the whiteboard present at each node of the underlying graph in order to communicate with other agents. In the first configuration, all agents are initially located at the same node; in the second configuration, the agents are scattered along the nodes of the underlying network. In both configurations, all the nodes where agents are initially located are not *dangerous*, i.e., they do not contain the black hole (they are *safe*). Our primary objective is to design an efficient BHS algorithm such that: (a) within a finite time, at least one agent survives, and (b) it gains knowledge of the black hole location.

2 Related Works and Our Contribution

2.1 Related work

Network exploration by mobile agents is one of the fundamental problems in this domain, and it was first introduced by Shannon [24]. After his pioneering work, this problem has been extensively studied in various topologies such as rings [23], trees [10], general graphs [7] under different models of communication (particularly, pebbles [12] and whiteboard [26]), synchrony (synchronous [7], semi-synchronous [4] and asynchronous [19]) and both in static [7] as well as dynamic networks (tori [22] and general graphs [21]).

The black hole search (BHS) problem is a special version of the exploration problem, where, in the worst case the underlying network needs to be explored in order to locate the black hole position. This problem was first introduced by Dobrev et al. [15], and after that has received a lot of attention: indeed, it has been studied for directed [8] as well as undirected graphs [6], and for different underlying networks, such as rings [6], tori [5], trees [9] and general graphs [15]. In addition, different communication models have been considered for this problem, including ‘Enhanced Token’ [13], ‘Pure Token’ [20] and whiteboard [16]. Moreover, this problem has also been explored for different initial agent configurations. In particular, Shi et al. [25] showed that, when the agents are co-located, a minimum of 2 co-located agents communicating via tokens can solve the BHS problem in hypercube, torus and complete network with $\Theta(n)$ moves, whereas in the case where k agents ($k > 3$) are scattered, then with only 1 token per agent it is shown that BHS can be solved in $O(k^2 n^2)$ moves. All these above papers discuss black hole search in a static network, and very little is known about the problem in dynamic networks. Di Luna et al. [11] first investigated this problem in a dynamic ring, and they showed that when the agents are co-located, then in face-to-face communication with 3 agents there is an optimal algorithm that works in $\Theta(n^2)$ moves and $\Theta(n^2)$ rounds (where n is the size of the ring). Next, with whiteboard communication, they reduced the complexity to $\Theta(n^{1.5})$ rounds and $\Theta(n^{1.5})$ moves. Lastly,

when the agents are initially scattered and each node has a whiteboard, then again with 3 agents they showed that at least $\Theta(n^2)$ moves and $\Theta(n^2)$ rounds are required for any BHS algorithm. In each case, they gave an optimal algorithm. Next, Bhattacharya et al. [2] studied the BHS problem in a dynamic cactus graph, and proposed an agent optimal algorithm when at most one edge can be dynamic; in the case when at most k (> 1) edges can be dynamic, they proposed a lower bound of $k + 2$ and an upper bound of $2k + 3$ agents.

In this paper, we further investigate the BHS problem in a dynamic torus, with the aim of providing an efficient BHS algorithm. To the best of our knowledge, this is the first work where the BHS problem is explored in the case of a dynamic torus. Previously, Gotoh et al. [22] studied the exploration problem under link presence detection and no link presence detection in dynamic tori, whereas Chalopin et al. [5] studied the BHS problem in a static torus and gave tight bounds on the number of agents and tokens when the agents are initially scattered.

2.2 Our Contribution

We investigate the BHS problem in a dynamic torus for two initial configurations: first, when the set of agents are initially co-located, and next, when the agents can be initially scattered in different nodes. When the agents are initially co-located, we provide the following results.

- We establish the impossibility of correctly locating the black hole with $n + 1$ agents.
 - We show that with $n + c$ (where $c \geq 2$ and $c \in \mathbb{Z}^+$) co-located agents, any BHS algorithm requires at least $\Omega(m \log n)$ rounds.
 - With $n + 3$ agents we present a BHS algorithm that works in $O(nm^{1.5})$ rounds.
 - With $n + 4$ agents we present an improved BHS algorithm that works in $O(mn)$ rounds.
- The following results are obtained when the agents are initially scattered.
- We establish the impossibility of correctly locating the black hole with $n + 2$ agents.
 - We show that with $k = n + c$ (where $c \geq 3$ and $c \in \mathbb{Z}^+$) scattered agents, any BHS algorithm requires $\Omega(mn)$ rounds.
 - With $n + 6$ agents we present a BHS algorithm that works in $O(nm^{1.5})$ rounds.
 - With $n + 7$ agents we present a $O(mn)$ round optimal BHS algorithm.

The list of results are summarised in the following Table 1.

■ **Table 1** Summary of Results where LB, UB and IC represent lower bound, upper bound and initial configuration of the agents, respectively.

IC	Bound	# Agents	Rounds	Results
Colocated	LB	$n + 2$	$\Omega(m \log n)$	Cor 1 & Thm 3
	UB	$n + 3$	$O(nm^{1.5})$	Thm 7
	UB	$n + 4$	$O(nm)$	Thm 8
Scattered	LB	$n + 3$	$\Omega(nm)$	Cor 3 & Thm 5
	UB	$n + 6$	$O(nm^{1.5})$	Thm 9
	UB	$n + 7$	$O(nm)$	Thm 10

Organisation. The remainder of the paper is organised as follows. In Sections 3 and 4, we explain the model and prove the lower bound results. Next, in Section 5, we discuss some preliminary notation and basic subroutines which will be used by our algorithms. Further, in Sections 6 and 7, we present and analyse our algorithms for the co-located and scattered case. Finally, we list some concluding remarks in Section 8.

Due to the restrictions in the page limit, the pseudocodes of the algorithms, proofs of the theorems and lemmas are omitted and can be found in the full version of this paper [3], which also includes a detailed explanation of the states and predicates used in our algorithms.

3 Model and Problem Definition

Graph Model. The dynamic graph is modelled as an undirected time-varying graph (or formally known as *temporal graph*) $\mathcal{G} = (G, V, E, \mathbb{T}, \rho)$, where V is the set of vertices (or nodes), E is the set of edges in G , \mathbb{T} is defined to be the *temporal domain*, which is defined to be \mathbb{Z}^+ as in this model we consider discrete time steps, also $\rho : E \times \mathbb{T} \rightarrow \{0, 1\}$ is defined as the *presence* function, which indicates the presence of an edge at a given time. The graph $G = (V, E)$ is the underlying static graph of the dynamic graph \mathcal{G} , also termed as *footprint* of \mathcal{G} . More specifically, the footprint $G = (V, E)$ is a torus of size $n \times m$, where n represents the number of rows and m represents the number of columns, we define $V = \{v_{i,j} \mid 0 \leq i \leq n-1, 0 \leq j \leq m-1\}$ and E is the set of edges, where the horizontal and vertical edges are $\{(v_{i,j}, v_{i,j+1 \bmod m})\}$ and $\{(v_{i,j}, v_{i+1 \bmod n,j})\}$, respectively. By the node $v_{i,j}$ we invariably mean $v_{i \bmod n, j \bmod m}$ and these modulus functions are ignored further in this paper. In order to restrict self-loop or multiple edges without loss of generality, we assume $3 \leq n \leq m$. A row ring R_i (resp, a column ring C_j) is the subgraph of G induced by the set of vertices $\{v_{i,j} \mid 0 \leq j \leq m-1\}$ (resp, $\{v_{i,j} \mid 0 \leq i \leq n-1\}$). The adversary has the ability to make an edge reappear or disappear at any particular time step with the added constraint that, irrespective of how many edges disappear or reappear, each row and column ring at any time step must be connected; in other words, each row and column ring in \mathcal{G} is *1-interval connected* (so at any time, the adversary can make at most one edge disappear from each row and column ring, in order to maintain the 1-interval connectivity property). A disappeared edge is termed as a *missing edge* in this paper.

Every node $v_{i,j} \in G$ is labelled by a unique Id (i, j) , whereas each node in G has 4 ports adjacent to it, where the ports corresponding to the edges $(v_{i,j}, v_{i,j-1})$, $(v_{i,j}, v_{i,j+1})$, $(v_{i,j}, v_{i-1,j})$, $(v_{i,j}, v_{i+1,j})$, are denoted by *west*, *east*, *south*, *north*, respectively. In addition, corresponding to each port of a node $v_{i,j}$ of G a *whiteboard* of storage of $O(1)$ -bits is placed. The purpose of the whiteboard is to store and maintain certain information, such as the node Id or agent Id or the agent's course of traversal (depending on the amount of storage the whiteboard can store). Any incoming agent can read the existing information or write any new information corresponding to a port along which it travels to the next node. Fair mutual exclusion to all incoming agents restricts concurrent access to the whiteboard. In this paper, we consider our temporal graph \mathcal{G} to be an oriented dynamic torus, i.e., each row and column ring in \mathcal{G} has an orientation. In other words, the nodes of a row (or a column) ring are marked in an increasing order along a counter-clockwise direction. The network G has a malicious node or unsafe node also termed as a *black hole*, which vanishes any incoming agent without leaving any of its trace. The remaining nodes in G are not malicious, hence they are termed as *safe nodes*.

Agent Model. A set of k agents $A = \{a_1, a_2, \dots, a_k\}$ are assigned the task to locate the black hole in \mathcal{G} . We consider two initial configurations in this paper: first, the set of A agents are initially *co-located* at a safe node (the node in G at which they are co-located is termed as *home*), second, the agents are initially *scattered* along safe nodes in \mathcal{G} . Each agent in A has a distinct Id of size $\lceil \log k \rceil$ bits taken from the set $[1, k]$, and every agent has computational capabilities so that it can communicate with other agents when they are at

the same node at the same time. Each agent has knowledge of the torus size, i.e., both n and m are known to the agents, and also, each agent has an internal memory of $O(\log m)$ bits. An agent moves from one node to another using the edges at each round; furthermore, any number of agents can concurrently move along an edge at any round. These actions are atomic in nature, so an agent cannot recognise the other agents concurrently passing through the same edge at the same round, but it can see and communicate with all the other agents present at the current node at the same round. These agents operate in *synchronous* rounds, so in each round, every agent becomes active and takes a local snapshot of its current node. The snapshot includes the presence of the ports of its current node at the current round, the agent's local memory (which contains the amount of information gathered by the agent while communicating with other agents), the set of agents present at the current node, and the contents of the whiteboard. Now, based on this information, the agent performs the following actions:

- *Look*: In this step, the agent takes the *snapshot* of the current node. This snapshot helps the agent gather the information about the Ids of other agents residing at the same node, the edges that currently exist at the current round and also the whiteboard information at the current node.
- *Compute*: On the basis of its earlier snapshot and local memory, the agent decides to stay at the current node or move to another node. The direction of its movement is also calculated in this step.
- *Move*: In this step, if the agent decides to move along a specific direction and if the corresponding edge is present, then it moves along this edge while updating the whiteboard (if required, based on the algorithm) to the next node in the subsequent round.

Since, the agents operate in *synchronous* rounds, so each agent gets activated at each round and performs the LCM cycle. So, the time taken by any algorithm is calculated in terms of the *rounds*.

Configuration. A configuration C_r at a round r is defined to be the amalgamation of the presence of the number of agents at a node, the local memory of each agent and contents of the whiteboard at the start of round r . The transformation from C_{r-1} to C_r depends on multiple factors: first, the execution of the algorithm; second, the adversarial choices of edges disappeared and reappeared in round $r - 1$. C_0 is the initial configuration, where, in the co-located case, the initial safe node is chosen by the adversary, whereas in the scattered case, the adversary arbitrarily places the agents along the safe nodes.

► **Definition 1** (Black Hole Search). *Given a dynamic torus \mathcal{G} of size $n \times m$, an algorithm \mathcal{A} for a set of k agents solves the BHS problem if at least one agent survives and terminates. The terminating agent must know the exact position of the black hole in the footprint of \mathcal{G} .*

The measures of the complexity for the BHS problem are as follows: the number of agents or *size*, required to successfully execute \mathcal{A} , the *time* or the number of rounds required to execute \mathcal{A} . Note that in this paper, we have assumed the fact that whenever an agent correctly locates the black hole, the algorithm terminates, so all the other agents executing any action get terminated immediately.

4 Lower Bound Results

In this section, we present the lower bound results on the number of agents and number of rounds in both scenarios when the agents are either initially co-located or scattered.

4.1 Co-located Agents

The next theorem gives impossibility result on the number of agents when they are co-located.

► **Theorem 1.** *Given a dynamic torus \mathcal{G} of size $n \times m$, there does not exist a BHS algorithm which correctly locates the black hole with $k = n + 1$ co-located agents and each node in \mathcal{G} contains a whiteboard of $O(1)$ bits.*

► **Corollary 1.** *Any BHS algorithm on a dynamic torus \mathcal{G} of size $n \times m$ requires at least $k = n + 2$ co-located agents to correctly locate the black hole when each node in \mathcal{G} has a whiteboard of $O(1)$ bits.*

Next lemma gives a lower bound on the round complexity for any exploration algorithm on a dynamic ring, where at least 2 agents are initially co-located.

► **Lemma 1.** *In a dynamic ring of size $n > 3$ in presence of whiteboard, any exploration algorithm with l ($l \geq 2$) co-located agents require at least $\Omega(n)$ rounds to explore such a ring.*

► **Theorem 2** ([11]). *In a dynamic ring of size $n > 3$, any BHS algorithm with 3 co-located agents in presence of whiteboard requires $\Omega(n^{1.5})$ rounds, even if the agents have distinct Ids.*

The following corollary follows from Lemma 1 and Theorem 2.

► **Corollary 2.** *In a dynamic ring of size $n > 3$, any BHS algorithm with at least 4 co-located agents in presence of whiteboard requires $\Omega(n)$ rounds, even if the agents have distinct Ids.*

The next theorem gives a lower bound on the round complexity for any BHS algorithm operating on a dynamic torus with k co-located agents.

► **Theorem 3.** *Any BHS algorithm with $k = n + c$ co-located agents, where $c \in \mathbb{Z}^+$ and $c \geq 2$, on a $n \times m$ dynamic torus requires at least $\Omega(m \log n)$ rounds.*

Proof. Given a dynamic torus of size $n \times m$ (with $3 \leq n \leq m$) and $k = n + c$ agents are initially co-located at a safe node, observe by Corollary 2, l (where $l \geq 4$) agents can perform BHS in presence of whiteboard along a row ring of size m in at least $\Omega(m)$ rounds. Now, let us consider there exists an algorithm \mathcal{H} which is tasked to perform BHS along the dynamic torus \mathcal{G} , so concurrently exploring a set of rings by a set of l agents is always a better strategy rather than exploring a ring one at a time by a set of agents. Hence, we consider \mathcal{H} instructs a set of l agents to explore a set of rings concurrently. So, if t (where $t \leq \frac{k}{l}$) rings are concurrently explored by the set of k agents, then as each ring in \mathcal{G} is 1-interval connected, so the adversary has the ability to block an agent each in every t such rings. This means the remaining agents left to explore for the next concurrent exploration is at least $k - \frac{k}{l}$, where each of these concurrent exploration requires $\Omega(m)$ rounds and the number of rings till now explored is $\frac{k}{l}$. In the next concurrent exploration, at least $\frac{k - \frac{k}{l}}{l}$ row rings can be explored in $\Omega(m)$ rounds, which further blocks this many agents, and the remaining agents left to explore remaining graph is $k - \frac{k}{l} - \frac{k - \frac{k}{l}}{l}$, whereas the total number of row rings explored yet is $\frac{k}{l} + \frac{k - \frac{k}{l}}{l}$. Continuing this way, we can define a recursion relation on the remaining number of agents, $T(\alpha) = T(\alpha - 1)(1 - \frac{1}{l})$ and $T(1) = k - \frac{k}{l}$, where $T(\alpha)$ resembles that at the α -th iteration this many agents are left to explore the remaining part of \mathcal{G} and each such concurrent exploration for black hole requires $\Omega(m)$ rounds. So, for α many iterations \mathcal{H} requires $\alpha\Omega(m) = \Omega(\alpha m)$ rounds. Now, we try to approximate the value of α . Observe, when $T(\alpha) \leq 7$, then either the whole torus is explored in the worst case for the black hole or there is no further concurrency possible because in order to concurrently explore at least two

rings in $\Omega(m)$ rounds, a minimum of 8 agents (as 4 agents are at least required to explore a ring in $\Omega(m)$ rounds) are required to be left available, so if at most 7 agents are remaining that means no concurrency is possible for any BHS algorithm. Hence, for $T(\alpha) \leq 7$, we approximate the value of α .

$$\begin{aligned} T(\alpha) \leq 7 &\implies T(\alpha - 1) \left(1 - \frac{1}{l}\right) \leq 7 \implies T(\alpha - 1) \leq \frac{7l}{l-1} \\ &\implies T(\alpha - 2) \left(1 - \frac{1}{l}\right) \leq \frac{7l}{l-1} \leq 7 \left(\frac{l}{l-1}\right)^2 \dots \implies T(1) \leq 7 \left(\frac{l}{l-1}\right)^{\alpha-1} \\ &\implies k \left(1 - \frac{1}{l}\right) \leq 7 \left(\frac{l}{l-1}\right)^{\alpha-1} \implies k \leq 7 \left(\frac{l}{l-1}\right)^\alpha \implies \frac{\log k - \log 7}{\log \left(\frac{l}{l-1}\right)} \leq \alpha \end{aligned}$$

This implies $\alpha \approx \log n$, as $k = n + c$ and $l \geq 4$. Hence, this means that for any algorithm \mathcal{H} , in order to either explore the whole dynamic torus for a black hole or to stop concurrent exploration, at least $\alpha \approx \log n$ many concurrent exploration needs to be performed, where each iteration takes $\Omega(m)$ rounds. This concludes that the total number of rounds at least required by any algorithm with $k = n + c$, co-located agents is $\Omega(m \log n)$. ◀

4.2 Scattered Agents

Next theorem shows the impossibility of BHS with $k = n + 2$ scattered agents.

► **Theorem 4.** *Given a dynamic torus \mathcal{G} of size $n \times m$, there does not exist any BHS algorithm which can correctly locate the black hole with $k = n + 2$ scattered agents, the result holds as well even if the nodes in \mathcal{G} has a whiteboard.*

► **Corollary 3.** *Any BHS algorithm on a dynamic torus \mathcal{G} of size $n \times m$ requires at least $k = n + 3$ scattered agents to correctly locate the black hole when each node in \mathcal{G} has a whiteboard of $O(1)$ bits.*

Following theorem is inspired from Theorem 4.2 in [22], which gives the lower bound on the round complexity for any BHS algorithm with k scattered agents along \mathcal{G} .

► **Theorem 5.** *Any BHS algorithm with $k = n + c$ scattered agents, where $c \in \mathbb{Z}^+$ and $c \geq 3$, on a $n \times m$ dynamic torus \mathcal{G} requires at least $\Omega(mn)$ rounds.*

5 Preliminaries

In this section, we explain all the subroutines, definitions and ideas used in our BHS algorithm, but first, we explain the contents maintained by the agents on the whiteboard.

Whiteboard. The following data is stored and maintained in the whiteboard by the agents. For each $dir \in \{east, west, north, south\}$ with respect to each $v_{i,j} \in \mathcal{G}$ we define the function $f : \{east, west, north, south\} \rightarrow \{\perp, 0, 1\}$,

$$f(dir) = \begin{cases} \perp, & \text{if an agent is yet to visit the port } dir \\ 0, & \text{if no agent has marked the port } dir \text{ as safe} \\ 1, & \text{if the port } dir \text{ is marked safe, i.e., the node along } dir \text{ is not black hole} \end{cases}$$

Cautious Walk. This is a fundamental movement strategy used in a network with a black hole, and it is used as a building block of all our algorithms. In this strategy, if two agents are together, then this strategy ensures that only one among them enters the black hole while the other survives. On the contrary, if only a single agent is present, then whenever it visits a new node, it leaves some mark behind on the whiteboard so that whenever another agent tries to visit this node along the same edge, it finds the mark and does not enter the black hole.

This walk is performed in three rounds, where if an agent a_1 (say) is alone (resp, with another agent a_2 , say) then in the first round a_1 decides to move one step along $e = (u, v)$ from u to v by marking $f(e) = 0$ (while a_2 waits) and if it is safe, i.e., does not contain the black hole, then in the next round a_1 returns to u and marks the edge e safe by writing $f(e) = 1$, then in the third step a_1 (resp, a_2) moves to v . This strategy ensures that no two agents enter the black hole along the edge e .

Stuck. An agent a_1 is defined to be *stuck* while exploring a 1-interval connected ring for two reasons.

- First, if while performing *cautious* walk along an edge $e = (u, v)$, a_1 at round r marks $f(e) = 0$ at u and moves to v , while v is safe and a_1 tries to return to u at round $r + 1$ to mark $f(e) = 1$, finds e to be missing, in this situation a_1 is *stuck* at v until e reappears.
- Second, if while moving along dir , a_1 finds e to be missing. In this situation, if more than one agent is simultaneously trying to move along dir at the same round and if a_1 is the lowest Id among them, then a_1 is *stuck* until e reappears, or, if a_1 is alone, then in that case also a_1 is *stuck* until e reappears.

5.1 Subroutines

In this section, we will discuss the sub-routines used as a building block in our BHS algorithms for both the co-located and scattered initial configurations. We have followed some of the pseudocode convention from the papers [11] and [22]. In this paper, we use three kinds of MOVE procedure in our algorithms, first, $\text{MOVE}(dir \mid p_1 : s_1; p_2 : s_2; \dots; p_k : s_k)$, second, $\text{MOVE}(dir \rightarrow f(dir) \mid p_1 : s_1; p_2 : s_2; \dots; p_k : s_k)$, and lastly, $\text{MOVE}(dir \rightarrow f(dir) \rightarrow s_i \mid p_1 : s_1; p_2 : s_2; \dots; p_k : s_k)$, where p_i is the predicate corresponding to the state s_i and $f(dir)$ represents the value with respect to dir (where $dir \in \{east, west, north, south\}$) in the whiteboard, so depending on the algorithm we use either of these MOVE procedures. The agent at each round, first takes a snapshot at its current location, and thereafter checks the predicates p_1, \dots, p_k one after another. If no predicate is satisfied, then in the first MOVE procedure, the agent moves along the direction dir , in the second MOVE procedure the agent moves along dir while updating the whiteboard of the current node along dir to $f(dir)$, and lastly, in the third MOVE procedure, in addition to moving along dir and updating the whiteboard, it also moves directly in to the state s_i . On the other hand, if some predicates are satisfied, then the agent chooses the first satisfied predicate (say) p_i , and the procedure stops, and the agent moves into state s_i corresponding to p_i . The predicate and state of the form $p_j : time + i \rightarrow s_j$ indicates that if p_j is satisfied then the agent enters the state s_j after $time + i$ rounds, whereas the predicate and the state of the form $p_j : f(dir) \rightarrow s_j$, indicates that if p_j is satisfied then the agent performs the action $f(dir)$ and then moves to the state s_j . Further, these procedures are again executed in the subsequent rounds. In the following part, we define the algorithm $\text{CAUTIOUS-WAITMOVEWEST}()$.

CAUTIOUS-WAITMOVEWEST(j, l): This algorithm works on 1-interval connected ring R_i (say), where the main purpose is to make a certain number of agents reach the node $v_{i,j}$ along the C_j -th column from any initial configuration. Further, whenever an agent reaches the desired node, and it is not stuck, it waits at that node until further instruction is provided.

The algorithm works as follows: for the first $4(l-1)m$ rounds, if an agent a_1 is instructed to perform CAUTIOUS-WAITMOVEWEST(j, l) along R_i , then it starts the following procedure, if the agent a_1 (say) is initially with another agent a_2 (say) and since a_1 is the lowest Id among them, a_1 starts cautious walk along *west* until it either gets stuck or has reached the desired node. On the other hand the task of a_2 is to follow a_1 until a_1 is stuck. While a_1 is stuck, a_2 performs the following action. If a_1 is stuck due to a missing edge along *west*, then a_2 instead of waiting reverses its direction to *east* and continues to perform *cautious* walk. Otherwise, if a_1 is stuck while returning back to mark a port safe along *west* which it has in the last round marked unsafe while exploring and, then a_2 waits for at most $3m$ rounds since the round it encountered this situation, and then reverses its direction and continues to perform *cautious* walk. On the other hand, if a_1 is alone, then it performs *cautious* walk until it either reaches the desired node or it is stuck. If a_1 catches another agent stuck, and if it is not the lowest Id among them, then it performs a similar action, as explained earlier in case of a_2 . After $4(l-1)m$ rounds have passed, the agents enter the state *Return*, in which each agent not stuck due to a missing edge tries to reach the node $v_{i,j}$.

► **Observation 1** ([11]). *Given a dynamic ring R and a cut U , where $|U| > 1$, if its footprint is connected by edges e_c and e_{cc} (where e_c and e_{cc} are the clockwise and counter-clockwise edges, respectively) to nodes in $V \setminus U$ (where V is the set of vertices not in U). If all the agents at a round r are at U , and do not try to cross along e_c , whereas there exists an agent which tries to cross along e_{cc} , then the adversary has the ability to prevent any agent from crossing U .*

CAUTIOUS-WAITMOVEWEST() ensures that this situation does not arise, as when an agent is stuck on e_c (or e_{cc}), another agent after finding this situation waits for at most $3m$ rounds (depending on the fact that whether the earlier agent is stuck while backtracking or it is stuck because it has encountered a missing edge along *west*), and then reverses its movement towards e_{cc} (or e_c), while the other agent remains stuck. Hence, there exists a round r where an agent each is trying to cross e_c , and another agent is trying to cross e_{cc} .

The following lemma ensures the correctness of our algorithm in detecting the black hole when the algorithm terminates.

► **Lemma 2.** *If an agent executing CAUTIOUS-WAITMOVEWEST() terminates while moving along a specific direction, then it correctly locates the black hole.*

Next corollary states that in the worst case at most two agents can enter the black hole while executing CAUTIOUS-WAITMOVEWEST().

► **Corollary 4.** *CAUTIOUS-WAITMOVEWEST() ensures that at most two agents enters the black hole.*

The following lemma shows the complexity and correctness of reaching the desired node while executing CAUTIOUS-WAITMOVEWEST().

► **Lemma 3.** *If two agents along a safe row ring R_i of size m ($m \geq 3$) execute our algorithm, then at least one agent reaches the desired node within $7m$ rounds.*

► **Lemma 4.** *If three agents are executing CAUTIOUS-WAITMOVEWEST($j, 3$) along R_i and $v_{i,j}$ is the black hole, then at most 2 agents enter the black hole whereas the adversary has the ability to stop the third agent from detecting the black hole location.*

The next corollary states that if the black hole is located at the current ring, then within $15m$ (or $15n$) rounds, the black hole is detected by a set of 4 agents.

► **Corollary 5.** *A set of 4 agents, executing CAUTIOUS-WAITMOVEWEST(j, l) (where $l \geq 4$) along R_i can correctly locate the black hole in at most $15m$ rounds, where $v_{i,j}$ is the black hole node.*

The following corollary states that while executing CAUTIOUS-WAITMOVEWEST(j, l) the worst situation happens with exactly 2 agents entering the black hole, which is when the desired node to reach is $v_{i,j}$ and it is also the black hole node.

► **Corollary 6.** *CAUTIOUS-WAITMOVEWEST(j, l) ensures that exactly 2 agents can be consumed by the black hole when the desired node $v_{i,j}$ is also the black hole node.*

The next lemma states that if at any round after the first $4m$ rounds since the start CAUTIOUS-WAITMOVEWEST(j, l) along R_i , if there still exists at least 3 agents yet to reach the desired node $v_{i,j}$, then our algorithm ensures that in a period of $4m$ rounds since the last agent has reached the desired node, at least one among these set of agents yet to reach the desired node, reaches $v_{i,j}$.

► **Lemma 5.** *After the first $4m$ rounds have elapsed executing CAUTIOUS-WAITMOVEWEST(j, l) (where $l > 2$), if at least 3 agents are still present along R_i then it takes at most $4m$ rounds for an agent among them to reach the desired node, since the last agent has reached the desired node.*

The following corollary follows from Lemmas 3 and 5.

► **Corollary 7.** *Our algorithm ensures that among l agents operating along R_i at least $l - 2$ agents reach the desired node within $4(l - 1)m$ rounds.*

► **Lemma 6.** *Among the remaining two agents which enter state Return after $4(l - 1)m$ rounds has elapsed, at least one among them reaches the desired node.*

The following theorem follows from Corollary 7 and Lemma 6.

► **Theorem 6.** *If l agents ($l \geq 2$) are in a safe ring R_i and they perform CAUTIOUS-WAITMOVEWEST(j, l), then at least $l - 1$ agents reach and stay on $v_{i,j}$ within $4(l - 1)m + 3m$ rounds, since the start of execution of our algorithm.*

CAUTIOUS-MOVE($west, j$): This algorithm is a special version of our earlier algorithm, it has two stages, and requires at least 2 agents. The first stage is exploration and works for $3m$ rounds, and the second stage is Exit. The algorithm works as follows: the lowest Id agent becomes the *Leader*, whereas the other agents become the *Follower*. The *Leader* explores new nodes in first stage and *Follower* follows the *Leader* until it either finds the *Leader* to be stuck or *Leader* stops reporting either due to a missing edge or it has entered the black hole. Whenever, the *Follower* finds the edge is missing and *Leader* is not reporting and $time < 3m$ it waits until the missing edge reappears or till $time = 3m$, whereas if it finds that the edge exists and *Leader* is not reporting then it terminates the algorithm, by declaring the node in which *Leader* has explored is the black hole node. On the other hand, whenever the *Leader* is also stuck due to a missing edge along its moving direction, then

both *Leader* and *Follower* wait until $time = 3m$. Whenever $time > 3m$, both *Leader* and *Follower* enter the second stage, i.e., state *Exit*, in which, irrespective of the fact that they are stuck or not, they try to return to their desired node, i.e., the node along C_j -th column, while returning each agent irrespective of *Leader* or *Follower* is instructed to mark the port of each node along their movement to 1 if not already marked so. Whenever the agents, while returning back, encounter a missing edge, the lowest Id agent waits, and other agents change direction.

The following two lemmas ensures that if l agents start executing CAUTIOUS-MOVE(), then among them eventually $l - 1$ agents reach the desired node within at most $3lm$ rounds, since the start of its execution.

► **Lemma 7.** *If l ($l \geq 2$) agents execute CAUTIOUS-MOVE(*west*, j) along a safe ring R_i , then at least $l - 1$ agents reach $v_{i,j}$ within $3lm$ rounds.*

► **Corollary 8.** *If l agents enter the state *Exit*, then at least $l - 1$ agents reach $v_{i,j}$ by at most $3(l - 1)m$ rounds.*

CAUTIOUSDOUBLEOSCILLATION[11] We have used this BHS ring exploration algorithm as a sub-routine in our BHS Torus exploration algorithm. This algorithm uses 3 agents to explore the ring and successfully detect the black hole, if the black hole node is along that ring, otherwise, it explores the ring. Among these three agents, one agent is recognised as the *Leader*, whereas the remaining two agents are known to be as AVANGUARD and RETROGUARD, respectively. The only difference is that both AVANGUARD and RETROGUARD while exploring a new node marks the corresponding ports safe or unsafe in whiteboard (where a port is safe implies that the adjacent node along a that port does not contain the black hole), so this means if RETROGUARD enters the black hole while exploring a sector of \sqrt{m} nodes along R_i (or \sqrt{n} nodes along C_j) then using the whiteboard instead of a pebble, LEADER can detect its location. As stated in [11], three agents executing CAUTIOUSDOUBLEOSCILLATION requires $O(m^{1.5})$ rounds to detect the black hole along a 1-interval connected ring of size m .

6 Co-located Agents

In this section, we propose two BHS algorithms on $n \times m$ dynamic torus. First algorithm requires $n + 3$ agents and works in $O(nm^{1.5})$ rounds, whereas the second algorithm requires $n + 4$ agents and works in $O(nm)$ rounds.

6.1 BHS with $n + 3$ agents

The set of $n + 3$ agents, $A = \{a_1, a_2, \dots, a_{n+3}\}$ are initially located at a safe node $v_{i,j}$, also termed as *home*. Initially from *home*, a_1 and a_2 executes the algorithm CAUTIOUS-MOVE(*north*, i), whereas a_3 and a_4 executes CAUTIOUS-MOVE(*south*, i). Once $12n$ rounds have passed, at least 3 out of these 4 agents return to $v_{i,j}$ (refer corollary 8). Whenever 3 among 4 agents return back to *home*, the first three lowest Id agents become LEADER, AVANGUARD and RETROGUARD and they are instructed to perform CAUTIOUSDOUBLEOSCILLATION along R_i . Now, as per Lemmas 14 and 15 in [11] it takes at most $T = 25m^{1.5} + 7(m + \sqrt{m})$ rounds to locate the black hole along R_i . So, after T rounds since the start of CAUTIOUSDOUBLEOSCILLATION, if the algorithm hasn't terminated (or the black hole is not detected) then these agents are instructed to return to $v_{i,j}$ which is the desired node, irrespective of the fact, whether they are stuck or not. While returning,

if an agent encounters a missing edge, then the lowest Id agent waits, whereas the other agent changes direction. Using this strategy, in at most $6m$ rounds, at least 2 among 3 agents return to $v_{i,j}$ (as this is similar to state *Exit* in algorithm CAUTIOUS-MOVE(), hence by corollary 8 this bound holds). After which they all together start executing CAUTIOUS-WAITMOVESOUTH($i - 1, 4$), which enables at least 3 among $n + 3$ agents reach the node $v_{i-1,j}$ and continue the same process. This process iterates for each R_i , where $0 \leq i \leq n - 1$.

► **Lemma 8.** *Our BHS algorithm ensures that there always exist 3 agents to perform CAUTIOUSDOUBLEOSCILLATION along R_i , where $0 \leq i \leq n - 1$.*

An iteration of our BHS algorithm is defined to be the collection of steps the set of agents perform from the node $v_{t,j}$ (where suppose $v_{i,j}$ be the initial starting node) to reach the node $v_{t+1,j}$ (where $t > 0 \ t \in \mathbb{N}^+$). More precisely, the agents at $v_{t,j}$, first perform CAUTIOUSDOUBLEOSCILLATION along R_t , then they try to return back to $v_{t,j}$, after which each agent along C_j try to reach the node $v_{t+1,j}$ while executing CAUTIOUS-WAITMOVESOUTH($t - 1, 4$), this whole process is defined to be one iteration. Now, the next lemma gives the number of rounds required by the set of $n + 3$ while executing our BHS algorithm to perform one iteration.

► **Lemma 9.** *It takes at most $T + 6m + 15n$ rounds to perform one iteration of the BHS algorithm with $n + 3$ agents.*

The following lemma and theorem gives the correctness and complexity of our BHS algorithm.

► **Lemma 10.** *Our algorithm correctly locates the black hole.*

► **Theorem 7.** *A group of $n + 3$ agents executing the BHS algorithm along a dynamic torus of size $n \times m$ correctly locates the black hole in $O(nm^{1.5})$ rounds.*

6.2 BHS with $n + 4$ agents

In this case the set of $n + 4$ agents, $A = \{a_1, a_2, \dots, a_{n+4}\}$ agents are initially co-located at *home* = $v_{i,j}$, say. The algorithm in this case is similar to the earlier BHS algorithm with $n + 3$ agents; the only difference is that here in order to explore R_t , instead of 3, 4 agents are used, where the lowest and second lowest Id agents at $v_{t,j}$ perform CAUTIOUS-MOVE(*west*, j) and the third lowest and fourth lowest Id agents are instructed to perform CAUTIOUS-MOVE(*east*, j), instead of CAUTIOUSDOUBLEOSCILLATION. The pseudocode is explained in Algorithm 1.

■ **Algorithm 1** BHS with $n + 4$ agents.

```

1 Instruct  $a_1$  and  $a_2$  to perform CAUTIOUS-MOVE(north,  $i$ ).
2 Instruct  $a_3$  and  $a_4$  to perform CAUTIOUS-MOVE(south,  $i$ ).
3 if time >  $12n$  then
4   for  $t = i; t \leq i - 1; t --$  do
5     Instruct the lowest and second lowest Id agents at  $v_{t,j}$  to perform
6     CAUTIOUS-MOVE(west,  $j$ ).
7     Instruct the third lowest and fourth lowest Id agents at  $v_{t,j}$  to perform
8     CAUTIOUS-MOVE(east,  $j$ ).
9      $\triangleright$  time1 is defined as the time since the last call of CAUTIOUS-MOVE.
10    if time1 >  $12m$  then
11      Perform CAUTIOUS-WAITMOVESOUTH( $t - 1, 5$ ).

```

► **Lemma 11.** *At least 3 among 4 agents executing CAUTIOUS-MOVE(*west*, *j*) and CAUTIOUS-MOVE(*east*, *j*) along R_t at some i -th iteration of the for loop of Algorithm 1, reach $v_{t,j}$ within $12m$ rounds since the start of CAUTIOUS-MOVE() in the current iteration, where $0 \leq t \leq n-1$.*

► **Lemma 12.** *Our BHS algorithm, with $n+4$ agents, ensures that in every iteration of the for loop of our algorithm, there always exists 4 agents to perform CAUTIOUS-MOVE(*west*, *j*) and CAUTIOUS-MOVE(*east*, *j*).*

The following lemma and theorem gives the correctness and complexity of our algorithm.

► **Lemma 13.** *A set of $n+4$ agents executing Algorithm 1, correctly locates the black hole.*

► **Theorem 8.** *A group of $n+4$ agents executing Algorithm 1 along a dynamic torus of size $n \times m$ correctly locates the black hole in $O(nm)$ rounds.*

7 Scattered Agents

This section proposes two BHS algorithms on an $n \times m$ dynamic torus. Our first algorithm works with $n+6$ agents and requires $O(nm^{1.5})$ rounds, whereas our second algorithm works with $n+7$ agents and requires $O(nm)$ rounds.

7.1 BHS with $n+6$ agents

A set of $n+6$ agents, $A = \{a_1, a_2, \dots, a_{n+6}\}$ are initially scattered along different nodes of the torus \mathcal{G} , i.e., the agents are arbitrarily placed, where there may be more than one agent at a node or there can be single agent at each $n+6$ nodes in \mathcal{G} .

At the first step, each agent performs CAUTIOUS-WAITMOVEWEST(0, 6) from any initial configuration, so after $time1 = 23m (g(6)+3m = 20m+3m)$ rounds has elapsed since the start of CAUTIOUS-WAITMOVEWEST(0, 6), each agent currently along C_0 is further instructed to perform CAUTIOUS-WAITMOVESOUTH(0, 6), so after $time2 = 23n (g(6) + 3n = 20n + 3n)$ rounds has elapsed since CAUTIOUS-WAITMOVESOUTH(0, 6), if at least 3 agents have reached the node $v_{0,0}$, then 3 lowest Id agents at $v_{0,0}$ become LEADER, AVANGUARD and RETROGUARD, respectively and are then instructed to perform CAUTIOUSDOUBLEOSCILLATION along R_0 . Hence, within T rounds from the start of CAUTIOUSDOUBLEOSCILLATION either the black hole is detected and the algorithm terminates or the ring R_0 is explored. After T rounds since the start of CAUTIOUSDOUBLEOSCILLATION, these 3 agents are instructed to return to $v_{0,0}$ by marking each node along their movement till $v_{0,0}$ to 1 (as the ring is explored and there is no black hole in this ring, so an agent can mark each port as safe, if not already marked so). So, by corollary 8 in at most $6m$ rounds, at least 2 among these 3 agents return to $v_{0,0}$, after which, each agent in G are instructed to perform CAUTIOUS-WAITMOVEWEST(0, 6).

On the other hand, if two agents have reached $v_{0,0}$ after $23n$ rounds have elapsed since CAUTIOUS-WAITMOVESOUTH(0, 6), then the lowest Id agent *cautiously* walks along *west* whereas the other agent *cautiously* walks along *east*. If along their movement they *catches* another agent trying to move along the same direction, then they together perform CAUTIOUS-MOVE() in the same direction. After $3m$ rounds has passed since they have started *cautious* walk, these agents along R_0 are instructed to return to $v_{0,0}$ by marking each port along their movement to 1. So, within $6m$ rounds, if 3 agents are along R_0 , then at least 2 among them returns, or if 2 agents are along R_0 , then at least 1 among them returns, further each agent along G is again instructed to perform CAUTIOUS-WAITMOVEWEST(0, 6). This process iterates for each R_i ring (where $0 \leq i \leq n-1$), depending on whether 2 or 3 agents have reached the node $v_{i,0}$.

The following lemma states that if 2 agents eventually reach the node $v_{i,0}$ at some i -th iteration, then this implies that among the $n + 6$ agents, already 3 agents have entered the black hole from three different directions without the black hole getting detected.

► **Lemma 14.** *If 2 agents reach $v_{i,0}$ at the i -th iteration after the execution of CAUTIOUS-WAITMOVESOUTH($i, 6$) when $time2 > 23n$, and the algorithm has not terminated yet, then this implies exactly 3 agents has entered black hole from three different directions.*

This corollary gives the possible directions along which 3 agents might have entered the black hole without still detecting it, while executing our algorithm.

► **Corollary 9.** *If 3 agents enter black hole from 3 directions without detecting it, then 2 among these 3 directions are east and west, whereas the 3rd is either north or south.*

The following lemma states that if eventually 2 agents reach the node $v_{i,0}$ at some i -th iteration while executing our BHS algorithm, then this implies that there must be another agent stuck somewhere at a node along R_i other than the fact that 3 agents have already entered the black hole and an agent each is already stuck along the remaining $n - 1$ row rings. Otherwise only 2 agents must not have reached the node $v_{i,0}$.

► **Lemma 15.** *Our BHS algorithm with $n + 6$ agents, ensures that if at the i -th iteration after $time2 > 23n$ only 2 agents are present at $v_{i,0}$, then there exists another agent stuck somewhere along R_i .*

The following lemma and theorem gives the correctness and complexity of our algorithm.

► **Lemma 16.** *Our BHS algorithm with $n + 6$ agents correctly locates the black hole.*

► **Theorem 9.** *A group of $n + 6$ agents executing the BHS algorithm along a dynamic torus of size $n \times m$ correctly locates the black hole in $O(nm^{1.5})$ rounds.*

7.2 BHS with $n + 7$ agents

In this case the set of $n + 7$ agents, $A = \{a_1, a_2, \dots, a_{n+7}\}$ are scattered along the nodes of \mathcal{G} . The BHS algorithm with $n + 7$ agents is almost similar to the earlier BHS algorithm with $n + 6$ agents. The differences are as follows: at each iteration the agents are instructed to perform CAUTIOUS-WAITMOVEWEST($0, 7$) instead of CAUTIOUS-WAITMOVEWEST($0, 6$). Next, while exploring a ring R_i at the i -th iteration at least 3 agents reach $v_{i,0}$ within $time2 > 27m$ ($g(7) + 3m = 24m + 3m$), whereas in earlier BHS algorithm with $n + 6$ agents at least 2 agents reach $v_{i,0}$, within $time2 > 23m$. Next, if 3 agents reach $v_{i,0}$, then our earlier algorithm, 2 agent scenario is similar to 3 agent scenario in this case. In our BHS algorithm with $n + 6$ agents, both agents are instructed to walk *cautiously* along *west* and *east*, respectively, but now as we have one more agent, the two lowest Id agents among them perform CAUTIOUS-MOVE(*west*, i), while the other walks *cautiously* along *east*. Otherwise, if 4 agents reach $v_{i,0}$, then this case is again similar to our 3 agent scenario in our earlier BHS algorithm with $n + 6$ agents, in which these 3 agents perform CAUTIOUSDOUBLEOSCILLATION whereas in this algorithm as we have one more agent, so two lowest Id agents perform CAUTIOUS-MOVE(*west*, 0) and the other two agents (i.e., 3rd lowest and 4th lowest Id agents) perform CAUTIOUS-MOVE(*east*, 0), and all these process iterates for each row ring.

► **Lemma 17.** *If 3 agents reach $v_{i,0}$ when $time2 > 27n$, and the algorithm has not terminated, then 3 agents have entered the black hole from three different directions.*

► **Lemma 18.** *Our BHS algorithm with $n + 7$ agents ensures that if at the i -th iteration after $time2 > 27n$ only 3 agents are present at $v_{i,0}$, then there exists another agent stuck somewhere along R_i .*

Lemmas 17 and 18 are just a consequence of Lemmas 14 and 15. Also, Corollary 9 holds for this algorithm as well.

► **Lemma 19.** *Our BHS algorithm with $n + 7$ agents correctly locates the black hole.*

► **Theorem 10.** *A group of $n + 7$ agents executing our algorithm along a dynamic torus of size $n \times m$ correctly locates the black hole in $O(nm)$ rounds.*

8 Conclusion

In this paper, we have considered the black hole search problem on a dynamic torus, in which each row and column are 1-interval connected. We have considered two types of initial configuration of the agents, and in each case, gave the bounds (both upper and lower bound) on number of agents and complexity in order to locate the black hole. To be specific, when the agents are initially co-located, first, we give lower bounds of $n + 2$ and $\Omega(m \log n)$ on number of agents and rounds, respectively. Next, with $n + 3$ agents, we design a BHS algorithm that works in $O(nm^{1.5})$ rounds, whereas with $n + 4$ agents, we propose an improved algorithm that works in $O(nm)$ rounds.

When the agents are scattered, we give a lower bound of $n + 3$ and $\Omega(mn)$ on a number of agents and rounds, respectively. Next, we propose two BHS algorithms, first, works with $n + 6$ agents in $O(nm^{1.5})$ rounds and second, works with $n + 7$ agents in $O(nm)$ rounds (round optimal algorithm).

Moreover, in this paper we have considered that each node in the dynamic torus is labeled. A possible future work in this direction can be first to remove this assumption and give a BHS algorithm and check if the bounds get changed. Secondly, for both these cases, finding an agent optimal algorithm is another possible direction which can be pondered in to.

References

- 1 Balasingham Balamohan, Paola Flocchini, Ali Miri, and Nicola Santoro. Time optimal algorithms for black hole search in rings. *Discrete Mathematics, Algorithms and Applications*, 3(04):457–471, 2011.
- 2 Adri Bhattacharya, Giuseppe F Italiano, and Partha Sarathi Mandal. Black hole search in dynamic cactus graph. In *International Conference and Workshops on Algorithms and Computation, WALCOM 2024*, pages 288–303. Springer, 2024.
- 3 Adri Bhattacharya, Giuseppe F Italiano, and Partha Sarathi Mandal. Black hole search in dynamic tori. *arXiv preprint arXiv:2402.04746*, 2024.
- 4 Sebastian Brandt, Jara Uitto, and Roger Wattenhofer. A tight lower bound for semi-synchronous collaborative grid exploration. *Distributed Computing*, 33:471–484, 2020.
- 5 Jérémie Chalopin, Shantanu Das, Arnaud Labourel, and Euripides Markou. Black hole search with finite automata scattered in a synchronous torus. In *Distributed Computing: 25th International Symposium, DISC 2011, Rome, Italy, September 20-22, 2011. Proceedings 25*, pages 432–446. Springer, 2011.
- 6 Jérémie Chalopin, Shantanu Das, Arnaud Labourel, and Euripides Markou. Tight bounds for black hole search with scattered agents in synchronous rings. *Theoretical Computer Science*, 509:70–85, 2013.
- 7 Reuven Cohen, Pierre Fraigniaud, David Ilcinkas, Amos Korman, and David Peleg. Label-guided graph exploration by a finite automaton. *ACM Transactions on Algorithms (TALG)*, 4(4):1–18, 2008.
- 8 Jurek Czyzowicz, Stefan Dobrev, Rastislav Kráľovič, Stanislav Miklík, and Dana Pardubská. Black hole search in directed graphs. In *Structural Information and Communication Complexity*:

16th International Colloquium, SIROCCO 2009, Piran, Slovenia, May 25-27, 2009, Revised Selected Papers 16, pages 182–194. Springer, 2010.

- 9 Jurek Czyzowicz, Dariusz Kowalski, Euripides Markou, and Andrzej Pelc. Searching for a black hole in synchronous tree networks. *Combinatorics, Probability and Computing*, 16(4):595–619, 2007.
- 10 Shantanu Das, Dariusz Dereniowski, and Christina Karousatou. Collaborative exploration of trees by energy-constrained mobile robots. *Theory of Computing Systems*, 62:1223–1240, 2018.
- 11 Giuseppe Antonio Di Luna, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Tight bounds for black hole search in dynamic rings. *arXiv preprint arXiv:2005.07453*, 2020.
- 12 Yann Disser, Jan Hackfeld, and Max Klimm. Tight bounds for undirected graph exploration with pebbles and multiple agents. *Journal of the ACM (JACM)*, 66(6):1–41, 2019.
- 13 S Dobrev, P Flocchini, R Kralovic, and N Santoro. Exploring a dangerous unknown graph using tokens. In *Proceedings of 5th IFIP International Conference on Theoretical Computer Science*, pages 131–150, 2006.
- 14 Stefan Dobrev, Paola Flocchini, Rastislav Kráľovič, P Ružička, Giuseppe Prencipe, and Nicola Santoro. Black hole search in common interconnection networks. *Networks: An International Journal*, 47(2):61–71, 2006.
- 15 Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Searching for a black hole in arbitrary networks: Optimal mobile agents protocols. *Distributed Computing*, 19:1–99999, 2006.
- 16 Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Mobile search for a black hole in an anonymous ring. *Algorithmica*, 48:67–90, 2007.
- 17 Stefan Dobrev, Rastislav Kráľovič, Nicola Santoro, and Wei Shi. Black hole search in asynchronous rings using tokens. In *Algorithms and Complexity: 6th Italian Conference, CIAC 2006, Rome, Italy, May 29-31, 2006. Proceedings 6*, pages 139–150. Springer, 2006.
- 18 Stefan Dobrev, Nicola Santoro, and Wei Shi. Using scattered mobile agents to locate a black hole in an un-oriented ring with tokens. *International Journal of Foundations of Computer Science*, 19(06):1355–1372, 2008.
- 19 Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *Algorithmica*, 65:562–583, 2013.
- 20 Paola Flocchini, David Ilcinkas, and Nicola Santoro. Ping pong in dangerous graphs: Optimal black hole search with pebbles. *Algorithmica*, 62:1006–1033, 2012.
- 21 Tsuyoshi Gotoh, Paola Flocchini, Toshimitsu Masuzawa, and Nicola Santoro. Exploration of dynamic networks: tight bounds on the number of agents. *Journal of Computer and System Sciences*, 122:1–18, 2021.
- 22 Tsuyoshi Gotoh, Yuichi Sudo, Fukuhito Ooshita, Hirotugu Kakugawa, and Toshimitsu Masuzawa. Exploration of dynamic tori by multiple agents. *Theoretical Computer Science*, 850:202–220, 2021.
- 23 Shota Nagahama, Fukuhito Ooshita, and Michiko Inoue. Ring exploration of myopic luminous robots with visibility more than one. *Information and Computation*, 292:105036, 2023.
- 24 Claude E Shannon. Presentation of a maze-solving machine. *Claude Elwood Shannon Collected Papers*, pages 681–687, 1993.
- 25 Wei Shi, Joaquin Garcia-Alfaro, and Jean-Pierre Corriveau. Searching for a black hole in interconnected networks using mobile agents and tokens. *Journal of Parallel and Distributed Computing*, 74(1):1945–1958, 2014.
- 26 Yuichi Sudo, Daisuke Baba, Junya Nakamura, Fukuhito Ooshita, Hirotugu Kakugawa, and Toshimitsu Masuzawa. A single agent exploration in unknown undirected graphs with whiteboards. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98(10):2117–2128, 2015.