Tight Bounds for Scattered Black Hole Search in a Ring[*]

Jérémie Chalopin[1], Shantanu Das[1], Arnaud Labourel[1], and Euripides Markou[2]

[1] LIF, CNRS & Aix-Marseille University, Marseille, France.
{jeremie.chalopin,shantanu.das,arnaud.labourel}@lif.univ-mrs.fr
[2] Department of Computer Science and Biomedical Informatics, University of
Central Greece, Lamia, Greece. emarkou@ucg.gr

Abstract. We study the problem of locating a particularly dangerous node, the so-called *black hole* in a synchronous anonymous ring network with mobile agents. A black hole destroys all mobile agents visiting that node without leaving any trace. Unlike most previous research on the *black hole search* problem which employed a colocated team of agents, we consider the more challenging scenario when the agents are identical and initially scattered within the network. Moreover, we solve the problem with agents that have constant-sized memory and carry a constant number of identical tokens, which can be placed at nodes of the network. In contrast, the only known solutions for the case of scattered agents searching for a black hole, use stronger models where the agents have non-constant memory, can write messages in whiteboards located at nodes or are allowed to mark both the edges and nodes of the network with tokens.

We are interested in the minimum resources (number of agents and tokens) necessary for locating all links incident to the black hole. In fact, we provide matching lower and upper bounds for the number of agents and the number of tokens required for deterministic solutions to the black hole search problem, in oriented or unoriented rings, using movable or unmovable tokens.

1 Introduction

We consider the problem of exploration in unsafe networks which contain malicious hosts of a highly harmful nature, called *black holes*. A black hole is a node which contains a stationary process destroying all mobile agents visiting this node, without leaving any trace [9]. In the *Black Hole Search* (BHS) problem the goal for a team of agents is to locate the black hole within finite time, with the additional constraint that at least one of the agents must remain alive. It is

---

[*] Part of this work was done while E. Markou was visiting the LIF research laboratory in Marseille, France. Authors J. Chalopin, S. Das and A. Labourel are partially supported by ANR projects SHAMAN and ECSPER.

usually assumed that all the agents start from the same location and have distinct identities. In this paper, we do not make such an assumption and study the problem for identical agents starting from distinct locations within the network. We focus on minimizing the resources required to find the black hole.

The only way of locating a black hole is to have at least one agent visiting it. However, since any agent visiting a black hole is destroyed without leaving any trace, the location of the black hole must be deduced by some communication mechanism employed by the agents. Four such mechanisms have been proposed in the literature: a) the *whiteboard* model in which there is a whiteboard at each node of the network where the agents can leave messages, b) the *'pure' token* model where the agents carry tokens which they can leave at nodes, c) the *'enhanced' token* model in which the agents can leave tokens at nodes or edges, and d) the time-out mechanism (only for synchronous networks) in which one agent explores a new node while another waits for it at a safe node.

The most powerful inter-agent communication mechanism is having whiteboards at all nodes. Since access to a whiteboard is provided in mutual exclusion, this model could also provide the agents a symmetry-breaking mechanism: If the agents start at the same node, they can get distinct identities and then the distinct agents can assign different labels to all nodes. Hence in this model, if the agents are initially co-located, both the agents and the nodes can be assumed to be non-anonymous without any loss of generality. The BHS problem has been studied using whiteboards in asynchronous networks, with the objective of minimizing the number of agents required to locate the black hole. Note that in asynchronous networks, it is not possible to answer the question of whether or not a black hole exists in the network, since there is no bound on the time taken by an agent to traverse an edge. Assuming the existence of (exactly one) black hole, the minimum sized team of co-located agents that can locate the black hole depends on the maximum degree Δ of a node in the network (unless the agents have a complete map of the network). In any case, the prior knowledge of the network size is essential to locate the black hole in finite time.

In the case of synchronous networks two co-located distinct agents can discover one black hole in any graph by using the time-out mechanism, without the need of whiteboards or tokens. Furthermore it is possible to detect whether a black hole actually exists or not in the network. Hence, with co-located distinct agents, the issue is not the feasibility but the time efficiency of black hole search (see [3, 5, 15, 16] for example). However when the agents are scattered in the network (as in our case), the time-out mechanism is not sufficient to solve the problem anymore.

Most of the previous results on black hole search used agents whose memory is at least logarithmic in the size of the network. This means that these algorithms are not scalable to networks of arbitrary size. This paper considers agents modeled as finite automata, i.e., having a constant number of states. This means that these agents can not remember or count the nodes of the network that they have explored. In this model, the agents cannot have prior knowledge of the size of the network. For synchronous ring networks of arbitrary size, con-

taining exactly one black hole, we present deterministic algorithms for locating the black hole using scattered agents each having constant-sized memory. We are interested in minimizing both the number of agents and the number of tokens required for solving the BHS problem.

We use the 'pure' token model. Note that the 'pure' token model can be implemented with $O(1)$-bit whiteboards (assuming that only a constant number of tokens may be placed on a node at the same time), while the 'enhanced' token model can be implemented with $O(\log \Delta)$-bit whiteboards. In the previous results using the whiteboard model, the capacity of each whiteboard is always assumed to be of at least $\Omega(\log n)$ bits, where n is the number of nodes of the network. Unlike the whiteboard model, we do not require any mutual exclusion mechanism at the nodes of the network. We distinguish movable tokens (which can be picked up from a node and placed on another) from unmovable tokens (which can not be picked up once they are placed on a node). For both types of tokens, we provide matching upper and lower bounds on both the number of agents and the number of tokens per agent, required for solving the black hole search problem in synchronous rings.

Related Works: The exploration of an unknown graph by one or more mobile agents is a classical problem initially formulated in 1951 by Shannon [18] and it has been extensively studied since then. In unsafe networks containing a single dangerous node (black hole), the problem of searching for it has been studied in the asynchronous model using whiteboards and given that all agents initially start at the same safe node (e.g., [6, 8, 9]). It has also been studied using 'enhanced' tokens in [7, 10, 19] and in the 'pure' token model in [13]. It has been proved that the problem can be solved with a minimal number of agents performing a polynomial number of moves. Notice that in an asynchronous network the number of the nodes of the network must be known to the agents otherwise the problem is unsolvable [9]. If the network topology is unknown, at least $\Delta + 1$ agents are needed, where Δ is the maximum node degree in the graph [8].

In asynchronous networks, with scattered agents (not initially located at the same node), the problem has been investigated for arbitrary topologies [2, 14] in the whiteboard model while in the 'enhanced' token model it has been studied for rings [11, 12] and for some interconnected networks [19].

The issue of efficient black hole search has been studied in synchronous networks without whiteboards or tokens (only using the time-out mechanism) in [3, 5, 15, 16] under the condition that all distinct agents start at the same node.

The problem has also been studied for co-located agents in directed graphs with whiteboards, both in the asynchronous [4] and synchronous cases [16]. A different dangerous behavior is studied for co-located agents in [17], where the authors consider a ring and assume black holes with Byzantine behavior, which do not always destroy a visiting agent.

In all previous papers (apart from [13]) studying the Black Hole Search problem using tokens, the 'enhanced' token model is used. The weakest 'pure' token model has only been used in [13] for co-located agents in asynchronous networks.

In all previous solutions to the problem using tokens, the agents are assumed to have non-constant memory.

Our Contributions: Unlike previous studies on BHS, we consider the scenario of anonymous (i.e., identical) agents that are initially scattered in an anonymous ring. We focus our attention on very simple mobile agents. The agents have constant-size memory, they carry a constant number of identical tokens which can be placed at nodes and, (apart from using the tokens), they can communicate with other agents only when they meet at the same node. We consider four different scenarios depending on whether the tokens are movable or not, and whether the agents agree on a common orientation. We present deterministic optimal algorithms and provide matching upper and lower bounds for the number of agents and the number of tokens required for solving BHS (See Table 1 for a summary of results). Surprisingly, the agreement on the ring orientation does not influence the number of agents needed in the case of movable tokens but is important in the case of unmovable tokens.

The lower bounds presented in this paper are very strong in the sense that they do not allow any trade-off between the number of agents and the number of tokens for solving the BHS problem. In particular we show that:

- Any constant number of agents, even having unlimited memory, cannot solve the BHS problem with less tokens than depicted in all cases of Table 1.
- Any number of agents less than that depicted in all cases of Table 1 cannot solve the BHS problem even if the agents are equipped with any constant number of tokens and they have unlimited memory.

Meanwhile our algorithms match the lower bounds, are time-optimal and since they do not require any knowledge of the size of the ring or the number of agents, they work in any anonymous synchronous ring, for any number of anonymous identical agents (respecting the minimal requirements of Table 1). Due to space limitations, proofs and formal algorithms are omitted and can be found in the full version of the paper [1].

Tokens are	Ring is	Resources necessary and sufficient		References in the paper
		# agents	# tokens	
Movable	Oriented	3	1	Theorem 1, 2 and 5
	Unoriented			
Unmovable	Oriented	4	2	Theorem 1, 3 and 6
	Unoriented	5	2	Theorem 1, 4 and 7

Table 1. Summary of results for BHS in synchronous rings

2 Our Model

Our model consists of an anonymous, synchronous ring network with $k \geq 2$ identical mobile agents that are initially located at distinct nodes called *homebases*. Each mobile agent owns a constant number t of identical tokens which can be placed at any node visited by the agent. The tokens are indistinguishable. Any token or agent at a given node is visible to all agents on the same node, but not visible to agents on other nodes. The agents follow the same deterministic algorithm and begin execution at the same time and being in the same initial state. In all our protocols a node may contain at most two tokens at the same time. At any node of the ring, the ports leading to the two incident edges are distinguishable and locally labelled (e.g. as 1 and 2) and an agent arriving at a node knows the port-label of the edge through which it arrived. In the special case of an oriented ring, the ports are consistently labelled as `Left` and `Right` (i.e., all ports going in the clockwise direction are labelled `Left`). In an unoriented ring, the local port-labeling at a node is arbitrary and each agent in its first step chooses one direction as `Left` and in every subsequent step, it translates the local port-labeling at a node into `Left` and `Right` according to its chosen orientation.

In a single time unit, each mobile agent completes one *step* which consists of the *Look*, *Compute* and *Move* stages (in this order). During the Look stage, an agent obtains information about the configuration of the current node (i.e., agents, tokens present at the node) and its own configuration (i.e., the port through which it arrived and the number of tokens it carries). During the Compute stage, an agent can perform any number of computations (i.e., computations are instantaneous in our model). During the Move stage, the agent may put or pick up a token at the current node and then either move to an adjacent node or remain at the current node. Since the agents are synchronous they perform each stage of each step at the same time. We call a token *movable* if it can be put on a node and picked up later by any mobile agent visiting the node. Otherwise we call the token *unmovable* in the sense that, once released, it can occupy only the node where it was released.

Formally we consider a mobile agent as a finite Moore automaton $\mathcal{A} = (\mathcal{S}, S_0, \Sigma, \Lambda, \delta, \phi)$, where $\mathcal{S}$ is a set of $\sigma \geq 2$ states among which there is a specified state S_0 called the *initial* state; $\Sigma \subseteq \mathcal{D} \times \mathcal{C}_v \times \mathcal{C}_A$ is the set of possible configurations an agent can see when it enters a node; $\Lambda \subseteq \mathcal{D} \times \{\texttt{put}, \texttt{pick}, \texttt{no action}\}$ is the set of possible actions by the agent; $\delta : \mathcal{S} \times \Sigma \to \mathcal{S}$ is the transition function; and $\phi : \mathcal{S} \to \Lambda$ is the output function. $\mathcal{D} = \{\texttt{left}, \texttt{right}, \texttt{none}\}$ is the set of possible directions through which the agent arrives at or leaves a node (`none` represents no move by the agent). $\mathcal{C}_v = \{0, 1\}^\sigma \times \{0, 1, 2\}$ is the set of possible configurations at a node, consisting of a bit string that denotes for each possible state whether there is an agent in that state, and an integer that denotes the number of tokens at that node (in our protocols at most 2 tokens reside at a node at any time). Finally, $\mathcal{C}_A = \{1, 2\} \times \{0, 1\}$ is the set of possible configurations of an agent, i.e., its orientation and whether it carries any tokens or not.

Notice that all computations by the agents are independent of the size n of the network and the number k of agents. There is exactly one black hole in the network. An agent can start from any node other than the black hole and no two agents are initially colocated[3]. Once an agent detects a link to the black hole, it marks the link permanently as dangerous (i.e., disables this link). We require that at the end of a black hole search scheme, all links incident to the black hole (and only those links) are marked dangerous and that there is at least one surviving agent. Note that our definition of a successful BHS scheme is slightly different from the original definition, since we consider finite state agents. The time complexity of a BHS scheme is the number of time units needed for completion of the scheme, assuming the worst-case location of the black hole and the worst-case initial placement of the scattered agents.

3 Impossibility Results

We first show that one unmovable token does not suffice to solve the problem. This result provides a lower bound on the number of tokens necessary for solving BHS.

Theorem 1. *For any constant k, there exists no algorithm that solves BHS in all oriented rings containing one black hole and k or more scattered agents, when each agent is provided with only one unmovable token. The result holds even if the agents have unlimited memory.*

We prove the above theorem by showing that no two agents can gather at the same node, either before or after placing their token. Further, if an agent puts down its only token, all other surviving agents would put down their respective tokens at the same time. An adversary could select the size of the ring and the initial locations of the agents in such a way that the tokens released by the agents are equidistant apart from each other, and thus the locations of the tokens does not convey any information about the location of the black hole.

We now derive some lower bounds on the number of agents necessary to solve the BHS problem.

Lemma 1. *During any execution of any BHS algorithm, if a link to the black hole is correctly marked, then at least one agent must have entered the black hole through this link.*

To solve the BHS problem in a ring, both links leading to the black hole need to be marked as dangerous. Thus, we immediately arrive at the following result.

Theorem 2. *Two mobile agents carrying any number of movable (or unmovable) tokens each, cannot solve the BHS problem in an oriented ring, even if the agents have unlimited memory.*

---

[3] Since there is no symmetry breaking mechanism at a node, two agents starting at the same node and in the same state, would behave as a single (merged) agent.

When the tokens are unmovable, even three agents are not sufficient to solve BHS. If there are exactly three agents each having t tokens (for some constant t), we can show that no two agents can meet before at least one of the three agents has fallen into the black hole. The agent that falls into the black hole may have left at most t tokens in its path, but this is not sufficient to indicate the exact location of the black hole since the agents may be initially located at an arbitrarily large distance from the black hole. Thus, the two surviving agents cannot identify both links incident to the black hole (they may identify at most one of these links).

Theorem 3. *Three mobile agents carrying a constant number of unmovable tokens each, cannot solve the BHS problem in an oriented ring, even if agents have unlimited memory.*

In unoriented rings, even four agents do not suffice to solve the BHS problem with unmovable tokens. In fact we show a stronger result that it is not even possible to identify just one of the links to the black hole, using four agents. An adversary can construct a large unoriented ring of odd size with an axis of symmetry such that there are two agents on each side of the axis and the black hole lies on the axis. In this case, at least two agents may fall into the black hole (one from each side), before any two agents meet. Due to the symmetry of the resulting configuration, the two surviving agents would not be able to gather at a node (and none of them could, by itself, identify any link to the black hole). Thus, we have the following result:

Theorem 4. *In an unoriented ring, four agents carrying any constant number of unmovable tokens each, cannot correctly mark any link incident to the black hole, even when the agents have unlimited memory.*

4 A BHS Scheme with Movable Tokens

We first consider the case when the agents have movable tokens. If each agent has a movable token it can perform a cautious walk [9]. The `Cautious-Walk` procedure consists of the following actions: Put the token at the current node, move one step in the specified direction, return to pick up the token, and again move one step in the specified direction (carrying the token). After each invocation of the Cautious Walk, the agent looks at the configuration of the current node[4] and decides whether to continue performing Cautious Walk.

We show that only three agents are sufficient to solve BHS, when they have one movable token each. Algorithm 1 achieves this, both for oriented and unoriented rings. The procedure `Mark-Link` permanently marks as dangerous the specified link.

Theorem 5. *Algorithm 1 solves the BHS problem in an unoriented ring with $k \geq 3$ agents having constant memory and one movable token each.*

---

[4] Recall that only the tokens put on the node are counted, not the tokens carried by the agent itself.

Algorithm 1: BHS-Ring-1

/* BHS in any ring using $k \geq 3$ agents having 1 movable token each */

repeat CautiousWalk (Left);
until *current node has a token and no agent* **or** *next link is marked Dangerous*;
Mark-Link(*Left*);
repeat CautiousWalk (Right);
until *current node has a token and no agent* **or** *next link is marked Dangerous*;
Mark-Link(*Right*);
---

5   BHS Schemes with Unmovable Tokens

For agents having only unmovable tokens, we use the technique of *Paired Walk* (called *Probing* in [3]) for exploring new nodes. The procedure is executed by two co-located agents with different roles and the same orientation. One of the agents called the *leader* explores an unknown edge while the other agent, called *follower* waits for the leader. If the other endpoint of the edge is safe, the leader immediately returns to the previous node to inform the follower and then both move to this new node. On the other hand, if the leader does not return in two time steps, the follower knows that the next node is the black hole.

In order to use the *Paired Walk* technique, we need to gather two agents at the same node and then break the symmetry between them, so that distinct roles can be assigned to each of them. The basic idea of our algorithms is the following. We first identify the two homebases that are closest to the black hole (one on each side). These homebases are called *gates*. The gates divide the ring into two segments: one segment contains the black hole (thus, is dangerous); the other segment contains all other homebases (and is safe). Initially all agents are in the safe part and an agent can move to the dangerous part only when it passes through the gate node. We ensure that any agent reaching a gate node, waits for a partner agent in order to perform the *Paired Walk* procedure. We now present two BHS algorithms, one for oriented rings and the other for unoriented rings.

5.1   Oriented Rings

In an oriented ring, all agents may move in the same direction (i.e., Left). During the first phase of the algorithm each agent places a token on its homebase, moves left until the next homebase (i.e., next node with a token) and then returns to its homebase to put down the second token. During this phase one agent will fall into the black hole and there will be a unique homebase with a single token (a "gate" node) and the other homebases will have two tokens each. However, the agents may not complete this phase of the algorithm at the same time. Thus during the algorithm, there may be multiple homebases that contain a single token. Whenever an agent reaches a "single token" node, it waits for a partner and then performs *Paired Walk* in the left direction. One of the agents of a pair (the leader) eventually falls into the black hole and the other agent (the follower)

marks the edge leading to the black hole and returns to the gate node, waiting for another partner. When another agent arrives at this node, these two agents perform *Paired Walk* in the opposite direction to find the other incident link to the black hole. The algorithm sketched below ensures that exactly one leader agent falls into the black hole from each side while performing *Paired Walk*.

Algorithm BHS-Ring-2:
During the algorithm, an agent a performs the following actions.

1. Agent a puts a token and moves left until the next node with a token (state CHECK-LEFT) and then returns to its homebase v (state GO-BACK) and puts its second token.
2. If there are no other agents at v, the agent moves left until it reaches a node containing exactly one token (state ALONE) and then waits for other agents arriving at this node (state WAITING).
3. Otherwise, if there is a WAITING (or ALONE) agent b at node v, the agents a and b form a (LEADER, FOLLOWER) pair.
4. If an ALONE agent meets a WAITING agent (and there are no other agents), they form a (LEADER, FOLLOWER) pair.
5. A LEADER agent performs *Paired Walk* until it falls into the black hole or it sees a link marked dangerous. In the latter case it moves to the gate node (state SEARCHER) and participates in *Paired Walk* in the other direction (state RIGHT-FOLLOWER).
6. A FOLLOWER agent performs *Paired Walk* until the corresponding leader falls into the black hole or they see a link marked dangerous. In the former case, the agent (state RIGHT-LEADER) moves to the gate node and waits for a partner to start *Paired Walk* in the other direction.
7. When a WAITING agent a meets a RIGHT-LEADER, agent a becomes a RIGHT-FOLLOWER and participates in the *Paired Walk*.
8. The algorithm has some additional rules to ensure that no two LEADERs are created at the same node at the same time. No agent becomes a LEADER if there is already another LEADER at the same node (In this case, the agent become a SEARCHER and eventually a RIGHT-FOLLOWER when it reaches the gate node).

When the algorithm BHS-Ring-2 is executed by four or more agents starting from distinct locations, the following properties hold:

- Exactly one CHECK-LEFT agent falls into the black hole.
- There is at least one LEADER agent and each LEADER has exactly one FOLLOWER.
- No two LEADER agents are created at the same time on the same node and thus, two LEADERs can not reach the black hole at the same time.
- There is exactly one RIGHT-LEADER agent and it falls into the black hole through the edge on the left side of the black hole.
- An agent in any state other than CHECK-LEFT, LEADER, or RIGHT-LEADER, never enters the black hole.

Theorem 6. *Algorithm BHS-Ring-2 correctly solves the black hole search problem in any oriented ring with 4 or more agents having constant memory and carrying two unmovable tokens each.*

5.2 Unoriented Rings

For unoriented rings, we need at least 5 agents with two unmovable tokens each. The algorithm for unoriented rings with unmovable tokens is similar to the one for oriented rings, except that each agent chooses an orientation. When two agents meet and one has to follow the other, we assume that the state of the agent contains information about the orientation of the agent (i.e., the port at the current node considered by the agent to be *Left*). Thus, when two agents meet at a node, one agent (e.g. the Follower) can orient itself according to the direction of the other agent (e.g. the Leader).

Algorithm BHS-Ring-3:
Each agent puts one token on its homebase, goes on *its left* until it sees another token and then returns to its homebase. Now the agent goes on its right until it sees a token and then returns again to the homebase. The agent now puts its second token on its homebase. During this operation exactly two agents will fall into the black hole. Each surviving agent walks to its left until it sees a node u with a single token. At this point the agent has to wait, since either there is a black hole ahead, or u is the homebase of an agent b that has not returned yet to put its second token.

It may happen that two agents arrive at node u at the same time from opposite directions. In this case, both agents can wait until another agent arrives. Note that in this case, the ring is safe on both directions until the next homebase and thus, an agent b (whose homebase is u) would arrive within a finite time. When agent b arrives, only one of the waiting agents (the one having the same orientation as b) changes to state LEADER and pairs-up with agent b. A similar case occurs when an agent a is waiting and two agents (both ALONE) arrive from different directions. Among these two agents, the one having the same orientation as agent a pairs up with agent a and starts the *Paired Walk* procedure.

As before there can be multiple leader-follower pairs performing *Paired Walk* in different parts of the ring. Note that no two LEADERs can be created at the same node at the same time. Thus, two LEADERs may not enter into the black hole at the same time from the same direction. After the first LEADER enters the black hole from one direction, the corresponding FOLLOWER agent marks the link as a dangerous link and thus, no other agents enter the black hole from the same direction.

We ensure that each LEADER agent has exactly one FOLLOWER agent. When the LEADER agent falls into the black hole, the corresponding FOLLOWER agent becomes the RIGHT-LEADER. The objective of the RIGHT-LEADER is to discover the other link incident to the black hole. The RIGHT-LEADER agent moves to the other end of the ring until the node with one token. Since we assume there are at least five agents, there must be either an unpaired

agent at one of the gates or, there must be another (LEADER, FOLLOWER) pair that has already detected and marked the other link leading to the black hole. If the RIGHT-LEADER does not find a RIGHT-FOLLOWER at the first gate, it performs a *slow* walk to the other gate and returns again to the former gate. During the slow walk, it moves at one-third the speed of any other agent (i.e., waits two steps after each move). This ensures that it will meet another agent in at least one of the two gates. These two agents now starts the *Paired Walk* procedure in the other direction.

The following properties can be verified:

1. Exactly two agents fall into the black hole before placing their second token.
2. There is at least one LEADER and each LEADER has a corresponding FOLLOWER.
3. There is either one or two RIGHT-LEADER agents (with opposite orientations).
4. At most one LEADER or RIGHT-LEADER enters the black hole from each direction.
5. An agent in any other state never enters the black hole after placing its second token.

Due to the above properties, we know that at most 4 agents may fall into the black hole. We now show that both links to the black hole are actually discovered and marked as dangerous, during the algorithm.

Theorem 7. *Algorithm BHS-Ring-3 correctly solves the black hole search problem in unoriented ring with 5 or more agents having constant memory and carrying two unmovable tokens each.*

6 Conclusions

In this paper, we solved the *scattered* BHS problem using the optimal number of agents and the optimal number of tokens per agent, while requiring only constant-size memory. Thus, all resources used by our algorithms are independent of the size of the network. Further, all the algorithms presented in the paper have a time complexity of $O(n)$ steps, so, they are asymptotically time-optimal for BHS in a ring. The results of this paper show that the constant memory limitation has no influence on the resource requirements since the (matching) lower bounds hold even if the agents have unlimited memory. It would be interesting to investigate if similar tight results hold for BHS in other network topologies. We would also like to investigate the difference between 'pure' and 'enhanced' token model in terms of the minimum resources necessary for black hole search in higher degree networks.

References

1. J. Chalopin, S. Das, A. Labourel, and E. Markou. Tight bounds for black hole search with scattered agents in synchronous rings. *ArXiv Preprint*, (arXiv:1104.5076v1), 2011.

2. J. Chalopin, S. Das, and N. Santoro. Rendezvous of mobile agents in unknown graphs with faulty links. In *Proceedings of 21st International Conference on Distributed Computing*, pages 108–122, 2007.

3. C. Cooper, R. Klasing, and T. Radzik. Searching for black-hole faults in a network using multiple agents. In *Proceedings of 10th International Conference on Principles of Distributed Systems*, LNCS 4305, pages 320–332, 2006.

4. J. Czyzowicz, S. Dobrev, R. Kralovic, S. Miklik, and D. Pardubska. Black hole search in directed graphs. In *Proceedings of 16th International Colloquium on Structural Information and Communication Complexity*, pages 182–194, 2009.

5. J. Czyzowicz, D. Kowalski, E. Markou, and A. Pelc. Complexity of searching for a black hole. *Fundamenta Informaticae*, 71(2,3):229–242, 2006.

6. S. Dobrev, P. Flocchini, R. Kralovic, G. Prencipe, P. Ruzicka, and N. Santoro. Optimal search for a black hole in common interconnection networks. *Networks*, 47(2):61–71, 2006.

7. 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.

8. S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Searching for a black hole in arbitrary networks: Optimal mobile agents protocols. *Distributed Computing*, 19(1):1–19, 2006.

9. S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Mobile search for a black hole in an anonymous ring. *Algorithmica*, 48:67–90, 2007.

10. S. Dobrev, R. Kralovic, N. Santoro, and W. Shi. Black hole search in asynchronous rings using tokens. In *Proceedings of 6th Conference on Algorithms and Complexity*, pages 139–150, 2006.

11. S. Dobrev, N. Santoro, and W. Shi. Scattered black hole search in an oriented ring using tokens. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium*, pages 1–8, 2007.

12. S. Dobrev, N. Santoro, and W. Shi. Using scattered mobile agents to locate a black hole in an un-oriened ring with tokens. *International Journal of Foundations of Computer Science*, 19(6):1355–1372, 2008.

13. P. Flocchini, D. Ilcinkas, and N. Santoro. Ping pong in dangerous graphs: Optimal black hole search with pure tokens. In *Proceedings of 22nd International Symposium on Distributed Computing*, LNCS 5218, pages 227–241, 2008.

14. P. Flocchini, M. Kellett, P. Mason, and N. Santoro. Map construction and exploration by mobile agents scattered in a dangerous network. In *Proceedings of IEEE International Symposium on Parallel & Distributed Processing*, pages 1–10, 2009.

15. R. Klasing, E. Markou, T. Radzik, and F. Sarracco. Hardness and approximation results for black hole search in arbitrary graphs. *Theoretical Computer Science*, 384(2-3):201–221, 2007.

16. A. Kosowski, A. Navarra, and C. Pinotti. Synchronization helps robots to detect black holes in directed graphs. In *Proceedings of 13th International Conference on Principles of Distributed Systems*, pages 86–98, 2009.

17. R. Kràlovic and S. Miklìk. Periodic data retrieval problem in rings containing a malicious host. In *Proceedings of 17th International Colloquium on Structural Information and Communication Complexity*, pages 156–167, 2010.

18. C. E. Shannon. Presentation of a maze-solving machine. In *Proceedings of 8th Conference of the Josiah Macy Jr. Found. (Cybernetics)*, pages 173–180, 1951.

19. W. Shi. Black hole search with tokens in interconnected networks. In *Proceedings of 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 670–682, 2009.