Mobile Search for a Black Hole in an Anonymous Ring

Stefan Dobrev[1], Paola Flocchini[2], Giuseppe Prencipe[3], and Nicola Santoro[4]

[1] Slovak Academy of Sciences, stefan@ifi.savba.sk
[2] University of Ottawa, flocchin@site.uottawa.ca
[3] Università di Pisa, prencipe@di.unipi.it
[4] Carleton University, santoro@scs.carleton.ca

Abstract. We address the problem of *mobile agents searching a ring network* for a highly harmful item, a *black hole*, a stationary process destroying visiting agents upon their arrival. No observable trace of such a destruction will be evident. The location of the black hole is not known; the task is to unambiguously determine and report the location of the black hole. We answer some natural computational questions: *How many agents are needed to locate the black hole in the ring ? How many suffice? What a-priori knowledge is required?* as well as complexity questions, such as: *With how many moves can the agents do it ? How long does it take ?*

Keywords: Mobile Agents, Distributed Computing, Ring Network, Hazardous Search.

1 Introduction

The most widespread use of autonomous mobile agents in network environments, from the World-Wide-Web to the Data Grid, is clearly to *search*, i.e., to locate some required "item" (e.g., information, resource, ...) in the environment. This process is started with the specification of what must be found and ends with the reporting of where it is located.

The proposed solutions integrate their algorithmic strategies with an exploitation of the capabilities of the network environment; so, not surprising, they are varied in nature, style, applicability and performance (e.g., see [3,4,11, 14,16]). They do however share the same assumption about the "item" to be located by the agents: it poses no danger, it is *harmless*.

This assumption unfortunately does not always hold: the item could be a local program which severely damages visiting agents. In fact, protecting an agent from "host attacks" (i.e., harmful items stored at the visited site) has become a problem almost as pressing as protecting a host (i.e., a site) from an agent attack (e.g., see [17,18]). Still, this problem has not been taken into account so far by any of the existing solutions.

In this paper we address the problem of searching for a highly harmful item whose existence we are aware of, but whose whereabouts are unknown. The item

J. Welch (Ed.): DISC 2001, LNCS 2180, pp. 166–179, 2001.

is a stationary process which disposes of visiting agents upon their arrival; no observable trace of such a destruction will be evident. Because of its nature, we shall call such an item a *black hole*.

The task is to unambiguously determine and report the location of the black hole (following this phase, a "rescue" activity would conceivably be initiated to deal with such a destructive process). In the distributed computing literature, there have been many studies on computing in presence of undetectable faulty components (e.g., [5,10] which can be rephrased in terms of computations in presence of black holes). However, a mentioned, this problem has never been investigated before. We are interested in understanding the basic algorithmic limitations and factors. The setting we consider is the simplest symmetric topology: the anonymous ring (i.e., a loop network of identical nodes). In this setting operate mobile agents: the agents have limited computing capabilities and bounded storage[1], obey the same set of behavioral rules (the "protocol"), and can move from node to neighboring node. We make no assumptions on the amount of time required by an agent's actions (e.g., computation, movement, etc) except that it is finite; thus, the agents are *asynchronous*. Each node has a bounded amount of storage, called *whiteboard*; $O(\log n)$ bits suffice for all our algorithms. Agents communicate by reading from and writing on the whiteboards; access to a whiteboard is done in mutual exclusion.

Some basic computational questions naturally and immediately arise, such as: *How many agents are needed to locate the black hole ? How many suffice? What a-priori knowledge is required?* as well as complexity questions, such as: *With how many moves can the agents do it ? How long does it take ?*

In this paper, we provide some definite answers to each of these questions. Some answers follow from simple facts. For example, if the existence of the black hole is a possibility but not a certainty, it is impossible[2] to resolve this ambiguity. Similarly, if the ring size n is not known, then the black-hole search problem can not be solved. Hence, n must be known. Another fact is that at least two agents are needed to solve the problem.

A more interesting fact is that if the agents are *co-located* (i.e., start from the same node) and *anonymous* (i.e., do not have distinct labels), then the problem is unsolvable. Therefore, to find the black hole, co-located agents must be *distinct* (i.e., have different labels); conversely, anonymous agents must be *dispersed* (i.e., start from different nodes). In this paper we consider both settings.

We first consider *distinct co-located* agents. We prove that *two* such agents are both *necessary and sufficient* to locate the black hole. Sufficiency is proved constructively: we present a distributed algorithm which allows locating the black hole using only two agents. This algorithm is *optimal*, within a factor of two, also in terms of the *amount of moves* performed by the two agents. In fact, we show the stronger result that $(n-1)\log(n-1)+O(n)$ moves are needed regardless of the number of co-located agents, and that with our algorithm two agents can solve the problem with no more than $2n\log n+O(n)$ moves.

[1] $O(\log n)$ bits suffice for all our algorithms.
[2] i.e., no deterministic protocol exists which always correctly terminates.

We also focus on the minimal *amount of time* spent by co-located agents to locate the black hole. We easily show that $2n-4$ (ideal) time units are needed, regardless of the number of agents; we then describe how to achieve such a time bound using only $n-1$ agents. We generalize this technique and establish a *general trade-off* between time and number of agents.

We then consider *anonymous dispersed* agents. We prove that, *two* anonymous dispersed agents are both *necessary and sufficient* to locate the black hole if the ring is *oriented*. Also in this case the proof of sufficiency is constructive: we present an algorithm which, when executed by two or more anonymous agents dispersed in an unoriented ring, allows finding the black hole with $O(n \log n)$ moves. This algorithm is *optimal* in terms of *number of moves*; in fact, we prove that any solution with k anonymous dispersed agents requires $\Omega(n \log(n-k))$ moves, provided k is known; if k is unknown, $\Omega(n \log n)$ moves are always required.

We also show that *three* anonymous dispersed *agents* are *necessary and suffice* if the ring is *unoriented*. Sufficiency follows constructively from the result for oriented rings. Due to space restrictions, some of the proofs will be omitted.

2 Basic Results and Lower Bound

2.1 Notation and Assumptions

The network environment is a set A of *asynchronous* mobile agents in a ring $\mathcal{R}$ of n *anonymous* (i.e., unlabeled[3]) nodes. The size n of $\mathcal{R}$ is known to the agents; the number of agents $|A| = k \geq 2$ might not be a priori known. The agents can move from node to neighboring node in $\mathcal{R}$, have computing capabilities and bounded storage, obey the same set of behavioral rules (the "protocol"), and all their actions (e.g., computation, movement, etc) take a finite but otherwise unpredictable amount of time. Each node has two ports, labelled *left* and *right*; if this labelling is globally consistent, the ring will be said to be *oriented*, *unoriented* otherwise. Each node has a bounded amount of storage, called *whiteboard*; $O(\log n)$ bits suffice for all our algorithms. Agents communicate by reading from and writing on the whiteboards; access to a whiteboard is done in mutual exclusion. A *black hole* is a stationary process located at a node, which destroys any agent arriving at that node; no observable trace of such a destruction will be evident to the other agents.

The location of the black hole is unknown to the agents. The *Black-Hole Search* (BHS) problem is to *find the location of the black hole*. More precisely, BHS is solved if at least one agent survives, and all surviving agents know the location of the black hole (*explicit termination*). Notice that our lower bounds are established requiring only that at least one surviving agent knows the location of the black hole (the difference is only O(N) moves/time).

First of all notice that, because of the asynchrony of the agents, we have that:

[3] Alternatively, they all have the same label.

Fact 1. *It is impossible to distinguish between slow links and a black hole.*

This simple fact has several important consequences; in particular:

Corollary 1.
1. *It is impossible to determine (using explicit termination) whether or not there is a black hole in the ring.*
2. *Let the existence of the black hole in $\mathcal{R}$ be common knowledge. It is impossible to find the black hole if the size of the ring is not known.*

Thus, we assume that both the existence of the black hole and the size n of the ring are common knowledge to the agents. The agents are said to be *co-located* if they all start from the same node; if they initially are in different nodes, they are said to be *dispersed*.

Fact 2. *Anonymous agents starting at the same node collectively behave as one agent.*

Corollary 2. *It is impossible to find the black hole if the agents are both co-located and anonymous.*

Thus, we assume that if the agents are initially placed in the same node, they have distinct identities; on the other hand, if they start from different locations there is at most one agent starting at any given node. Finally, observe the obvious fact that if there is only one agent the BHS problem is unsolvable; that is

Fact 3. *At least two agents are needed to locate the black hole.*

Thus, we assume that there are at least two agents. Let us now introduce the complexity measure used in the paper. Our main measures of complexity are the *number of agents*, called *size*, and the total *number of moves* performed by the agents, which we shall call *cost*. We will also consider the amount of *time* elapsed until termination. Since the agents are asynchronous, "real" time cannot be measured. We will use the traditional measure of *ideal time* (i.e., assuming synchronous execution where a move can be made in one time unit); sometimes we will also consider *bounded delay* (i.e., assuming an execution where a move requires at most one time unit), and *causal time* (i.e., the length of the longest, over all possible executions, chain of causally related moves). In the following, unless otherwise specified, "time" complexity is "ideal time" complexity.

2.2 Cautious Walk

At any time during the search for the black hole, the ports (corresponding to the incident links) of a node can be classified as (a) *unexplored* – if no agent has moved across this port, (b) *safe* – if an agent arrived via this port or (c) *active* – if an agent departed via this port, but no agent has arrived via it.

It is always possible to avoid sending agents over *active* links using a technique we shall call *cautious walk*: when an agent moves from node u to v via an *unexplored* port (turning it into *active*), it must immediately return to u (making the port *safe*), and only then go back to v to resume its execution; an agent

needing to move from u to v via an *active* port must wait until the port becomes *safe*. In the following, by the expression *moving cautiously* we will mean moving using cautious walk. *Cautious walk* reduces the number of agents that may enter the black hole in a ring to 2 (i.e., the degree of the node containing the black hole). Note that this technique can be used in any asynchronous algorithm $\mathcal{A}$, at a cost of $O(n)$ additional moves, with minimal consequences:

Lemma 1 ([8]). *Let $\mathcal{A}'$ be the algorithm obtained from $\mathcal{A}$ by enforcing* cautious walk. *For every execution $\mathcal{E}'$ of $\mathcal{A}'$ there exists a corresponding execution $\mathcal{E}$ of $\mathcal{A}$ such that $\mathcal{E}$ is obtained from $\mathcal{E}'$ by deleting only the additional moves due to* cautious walk.

Let us remark that cautious walk is a general technique that can be used in any topology; furthermore, it has been shown that *every* black-hole location algorithm for two agents *must* use cautious walk [8].

2.3 Lower Bound on Moves

In this section we consider the minimum number of moves required to solve the problem. The existence of an asymptotic $\Omega(n \log n)$ bound can be proven by carefully adapting and modifying the (rather complex) proof of the result of [10] on rings with a faulty link. In the following, using a substantially different argument, we are able to to obtain directly a more precise (not only asymptotic) bound, with a simpler proof. In fact we show that, regardless of the setting (i.e., collocation or dispersal) and of the number of agents employed, $(n-1)\log(n-1)+O(n)$ moves are required.

In the following, we will denote by $\mathcal{E}^t$ and $\mathcal{U}^t$ the explored and unexplored area at time t, respectively. Moreover, z^t denotes the central node of $\mathcal{E}^t$; that is, given x^t and y^t, the two *border nodes* in $\mathcal{E}^t$ that connect $\mathcal{E}^t$ to $\mathcal{U}^t$, z^t is the node in $\mathcal{E}^t$ at distance $\lceil |\mathcal{E}^t|/2 \rceil - 1$ from x^t.

Definition 1. *A* causal chain *from a node v_p to a node v_q has been executed at time t, if $\exists d \in \mathbb{N}, \exists u_1, u_2, \ldots, u_d \in V$ and times $t_1, t'_1, t_2, t'2, \ldots, t_d, t'_d$ such that*

- $t < t_1 < t'_1 < t_2 < t'_2 < \ldots < t_d < t'_d$,
- $v_p = u_1$, $v_q = u_d$ *and* $\forall i \in \{1, 2, \ldots, d-1\} : u_i$ *is a neighbor of* u_{i+1}, *and*
- $\forall i \in \{1, 2, \ldots, d-1\}$ *at time* t_i *an agent moves from node* u_i *to node* u_{i+1} *and reaches* u_{i+1} *at time* t'_i.

Lemma 2. *Let $|\mathcal{U}^t| > 2$ at a given time $t \geq 0$, and $k \geq 2$.*

1. *Within finite time, at least two agents will leave the explored area $\mathcal{E}^t$ in different directions.*
2. *A finite time after they have left $\mathcal{E}^t$, say at $t' > t$, a* causal chain *is executed from one of the two border nodes of $\mathcal{E}_{t'}$ to z_t.*

Theorem 1. *At least $(n-1)\log(n-1)+O(n)$ moves are needed to find a black hole in a ring, regardless of the number of agents.*

3 Co-located Distinct Agents

In this section we consider the case when all agents are co-located but distinct; i.e., they start at the same node, called the *home base*, and have distinct identities. The distinct labels of the agents allows any tie to be deterministically broken. As a consequence, if the ring is unoriented, the agents can initially agree on the directions of the ring. Thus, in the rest of this section, we assume w.l.g. that the ring is oriented.

Let $0, 1, \ldots n-1$ be the nodes of the ring in clockwise direction ($0, -1, \ldots -(n-1)$ in counter-clockwise direction) and, without loss of generality, let us assume that node 0 is the home base.

3.1 Agent-Optimal Solution

At least two agents are needed to locate the black hole (Fact 3). We now consider the situation when there are *exactly* two agents, l and r, in the system, and they are co-located.

The algorithm proceeds in phases. Let E_i and U_i denote the explored and unexplored nodes in phase i, respectively. Clearly, E_i and U_i partition the ring into two connected subgraphs, with the black hole located somewhere in U_i.

Algorithm 1 (Two Agents)

Start with round number $i = 1$, $E_1 = \{0\}$, and $U_1 = \{1, 2, \ldots, n-1\}$.

1. Divide U_i into two continuous disjoint parts U_i^l and U_i^r of almost equal sizes. Since U_i is a path, this is always possible. (We may assume U_i^l is to the left of 0 while U_i^r is to the right.)
2. Let agents l and r explore (using *Cautious Walk*) U_i^l and U_i^r, respectively. Note that, since both of them are within E_i and since U_i is divided into two continuous parts, the agents can safely reach the parts they have to explore.
3. Since U_i^l and U_i^r are disjoint, at most one of them contains the black hole; hence, one of the agents (w.l.g. assume r) successfully completes step 2. Agent r then moves across E_i and follows the *safe* ports of U_i^l until it comes to the node w from which there is no *safe* port leading to the left.
4. Denote by U_{i+1} the remaining unexplored area. (All nodes to the right of w, up to the last node of U_i^r explored by r, are now explored – they form E_{i+1}.) If$|U_{i+1}| = 1$, agent r knows that the black hole is in the single unexplored node and **terminates**. Otherwise U_{i+1} is divided into U_{i+1}^l and U_{i+1}^r as in step 1. Agent r leaves on the whiteboard of w a message for l indicating the two areas U_{i+1}^l and U_{i+1}^r. Note that $O(\log n)$ bits are sufficient to code this message.
5. Agent r traverses E_{i+1} and starts exploring U_{i+1}^r. (Proceeds to the next round – increment i and go to step 2 ...)
6. When (if) l returns to w, it finds the message and starts exploring U_{i+1}^l. (Proceeds to the next round – increment i and go to step 2 ...) ◁▷

Theorem 2. *Two agents can find the black hole performing* $2n \log n + O(n)$ *moves (in time* $2n \log n + O(n)$*).*

From Fact 3 and Theorems 1 and 2, it follows that

Corollary 3. *Algorithm 1 is size-optimal and cost-optimal.*

3.2 More Than Two Agents: Improving the Time

In this section we study the effects of having $k > 2$ agents in the home base.

We know that increasing k will not result in a decrease of the total number of moves; in fact, the lower bound of Theorem 1 is independent of the number of agents and is already achieved, within a factor of two, by $k = 2$. However, the availability of more agents can be exploited to improve the *time complexity* of locating the black hole.

The following theorem shows a simple lower bound on the time needed to find the black hole, regardless of the number of agents in the system.

Theorem 3. *In the worst case,* $2n - 4$ *time units are needed to find the black hole, regardless of the number of agents available.*

We now show that the lower bound can be achieved employing $n - 1$ agents. Let $r_1 \dots r_{(n-1)}$ be the $n - 1$ agents.

Algorithm 2 ($n - 1$ Agents)

Each agent r_i is assigned a location $i + 1$; its task is to verify whether that is the location of the black hole. It does so in two steps, executed independently of the other agents.
Step 1: It first goes to node i in clockwise direction and, if successful, returns to the home base (phase 1).
Step 2: It then goes in counter clockwise direction to node $-(n - i - 2)$ and, if successful, returns to the home base: the assigned location is where the black hole resides.
Clearly, only one agent will be able to complete both steps, while the other $n-2$ will be destroyed by the black hole. ◁▷

Theorem 4. *The black hole can be found in time* $2n-4$ *by* $n-1$ *agents starting from the same node.*

Thus, by Theorems 3 and 4, it follows that

Corollary 4. *Algorithm 2 is time-optimal.*

We now show how to employ the idea used for the time-optimal algorithm to obtain a trade-off between the number of agents employed and the time needed to find the black hole. Let q $(1 \leq q \leq \log n)$ be the trade-off parameter.

Algorithm 3 (Time-Size Tradeoff)

Two agents (called *explorers*) are arbitrarily chosen; their task is to mark all the safe ports before the black hole. They do so by leaving the home base in opposite directions, moving cautiously; each will continue until it is destroyed by the black hole.
The other agents start their algorithm in pipeline with the two explorers, always leaving from safe ports. The algorithm proceeds in q rounds.
In each round $n^{1/q}-1$ agents $(r_1 \dots r_{n^{1/q}-1})$ follow an algorithm similar to Algorithm 2 to reduce the size of the unexplored area by a factor of $n^{1/q}$. The unexplored area is in fact divided into $n^{1/q}$ segments $S_1, S_2, \dots, S_{n^{1/q}}$ of almost equal size (e.g., at the first phase the segment S_i is $(i-1)n^{(q-1)/q}+1, \dots, in^{(q-1)/q}$). Agent r_i verifies the guess that the black hole belongs to segment S_i by checking the nodes around S_i (first the right one, then the left one).
Clearly only one agent, say r_i, will be able to locate the segment containing the black hole. When r_i verifies its guess, arriving to the node to the left of S_i, the agents r_j with $j < i$ are trying to enter S_i from the left, while the agents r_j for $j > i$ are still trying to enter S_i from the right. To use these agents in the next round, r_i has to "wake them up": before returning to the home base, r_i moves left (possibly entering S_i) up to the last safe port, awakening all r_j with $j < i$, so that they can correctly proceed to the next round.
The process is repeated until the black hole is located in round q. ◁▷

Notice that, except for the two exploring agents, all agents survive.

Theorem 5. *Let* $1 \leq q \leq \log n$. *The black hole can be found using* $n^{1/q}+1$ *agents in time* $2(q+1)n - o(n)$.

4 Dispersed Anonymous Agents

In this section we examine the case when the agents are anonymous but dispersed (i.e., initially there is at most one agent at any given location). The number k of agents is not known a priori.

4.1 Basic Properties and Lower Bounds

A simple but important property is that, although anonymous, the agents can uniquely identify each other by means of purely local names. This is easily achieved as follows. Each agent a will think of the nodes as numbered with consecutive integers in the clockwise direction, with its starting node (its "homebase") as node 0. Then, when moving, agent a will keep track of the relative distance d_a from the homebase: adding $+1$ when moving clockwise, and -1 otherwise. Thus, when a encounters at the node (at distance) $d_a = -3$ an agent b which is at distance $d_b = +2$ from its own homebase, a is able to unambiguously determine that b is the unique agent whose homebase is node -5 (in a's view of the ring).

Lemma 3. *Each agent can distinguish and recognize all other agents.*

Another simple but important property is that, unlike the case of co-located agents, with dispersed agents there is a major difference between oriented and unoriented rings. In fact, if the ring is unoriented, two agents no longer suffice to solve the problem: they could be located in the nodes next to the black hole, and both made to move towards it. In other words,

Fact 4. *At least three dispersed agents are needed to locate the black hole in the unoriented ring.*

Thus, when dealing with the unoriented ring, we assume that there are at least three dispersed agents.

We now establish a lower bound (that we will prove to be tight in the next section) on the cost for locating the black hole; the lower bound is established for oriented rings and, thus, applies also to the unoriented case.

Theorem 6. *The cost of locating the black hole in oriented rings is at least* $\Omega(n \log n)$.

We now consider the case when every agent is endowed with a priori knowledge of k. This additional knowledge would provide little relief, as indicated by the following lower bound.

Theorem 7. *If k is known a priori to the agents, the cost of locating the black hole in oriented ring is* $\Omega(n \log(n-k))$.

The proof of Theorem 6 considers a worst-case scenario: an adversarial placement of both the black hole and the agents in the ring. So, one last question is whether, knowing k we could fare substantially better under a (blind but) favorable placement of the agents in the ring; i.e., assuming that k is known a priori and that *we* can place the agents, leaving to the adversary only the placement of the black hole. Also in this case, the answer is substantially negative. In fact, the application of the proof technique of Theorem 1 (with the initial explored region set to be the smallest connected region containing all agents, which is clearly of size at most $n - n/k$) yields a lower bound of $\Omega(n \log(n/k)) = \Omega(n(\log n - \log k)$, which, for reasonably small k, is still $\Omega(n \log n)$.

4.2 Oriented Rings: A Cost-Optimal Algorithm

In this section we describe a cost-optimal algorithm for the oriented ring where $k \geq 2$ anonymous agents are dispersed. The algorithm is composed of three distinct parts: pairing, elimination, and resolution.

The basic idea is to first form pairs of agents and then have the pairs search for the black hole.

Algorithm 4 (Pairing)

1. Move along the ring clockwise using cautious walk, marking (direction and distance to the starting node) the visited nodes, until arriving to a node visited by another agent.
2. Chase that agent until you come to a) a node visited by two other agents or b) the last safe node marked by the agent you are chasing
   Case a) **Terminate** with status *alone*.
   Case b) Form a pair: Leave a mark *Join me* and **terminate** with status *paired-left*
3. When, during your cautious walk, you encounter the mark *Join me*, clear this mark and **terminate** with status *paired-right*.
4. If you meet a paired agent, **terminate** with status *alone*. ◁▷

The agents with status *paired-* will then execute the algorithm to locate the black hole. The agents terminating with status *alone* will be passive in the remainder of the computation.

Lemma 4. *At least one pair is formed during the pairing phase. The pairing phase lasts at most* $3n - 6$ *time units, its cost is at most* $4n - 7$.

Note that, if the pairing algorithm starts with k agents, any number of pairs between 1 and $\lfloor k/2 \rfloor$ can be formed, depending on the timing. For example $\lfloor k/2 \rfloor$ pairs are formed when the "even" (as counting to the left from the black hole) agents are very slow, and the "odd" agents are fast and catch their right neighbors.

Since agents can distinguish themselves using local names based on their starting nodes (Lemma 3), also the pairs can be given local names, based on the node where the pair was formed (the "homebase"). This allows a pair of agents to ignore all other agents. Using this fact, a straightforward solution consists of having each pair independently execute the location algorithm for two agents (Algorithm 1). This however will yield an overall $O(n^2 \log n)$ worst-case cost.

To reduce the cost, the number of active pairs must be effectively reduced. The reduction is done in a process, called *elimination*, closely resembling leader election. In this process, the number of homebases (and thus pairs) is reduced to at most two.

The two agents in the pair formed at node v will be denoted by r_v and l_v, and referred to as the right and the left agent, respectively; v will be their homebase.

Algorithm 5 (Elimination)

The computation proceeds in logical rounds. In each round, the left agent l_v* cautiously moves to the left until it is destroyed by the black hole (case 0), or it reaches a homebase u with higher (case 1) or equal (case 2) round number. In case (1), l_v* returns to v which it marks *Dead*. In case (2), l_v* marks u as *Dead* and returns to v; if v is not marked *Dead*, it is promoted to the next round.

Similarly, agent r_v* cautiously moves to the right until it finds (if it is not destroyed by the black hole) the first homebase u in equal or higher round; it then returns back to v. If the current level of v (its level could have risen during the travel of r_v*) is not higher then the level of u, v is marked *Dead*; otherwise, if v is not marked *Dead*, r_v* travels again to the right (it is now in a higher round).
To prevent both agents of a pair entering the black hole, both l_v* and r_v* maintain a counter and travel to distance at most $\lfloor (n-1)/2 \rfloor$. If one of them has traveled such a distance without finding another homebase with the same or higher round, it returns back to v, and v is marked *Selected*. ◁▷

The rule of case 1 renders stronger a homebase (and, thus, a pair) in a higher logical round; ties are resolved giving priority to the right node (case 2 and the handling of the right agent). This approach will eventually produce either one or two *Selected* homebases. If an agent returns to a homebase marked *Dead*, it stops any further execution. When a homebase has been marked *Selected*, the corresponding agents will then start the resolution part of the algorithm by executing Algorithm 1, and locating the black hole. Note that, for each of the two pairs, the execution is started by a single agent; the other agent either has been destroyed by the black hole or will join in the execution upon its return to the homebase. Summarizing, the overall algorithm is structured as follows.

Algorithm 6 (Overall)

1. Form pairs of agents using Algorithm 4.
2. Reduce the number of pairs using Algorithm 5.
3. Find the location of the black hole using Algorithm 1. ◁▷

Theorem 8. *In oriented rings, the black hole can be found by $k \geq 2$ dispersed anonymous agents in $O(n \log n)$ time and cost.*

Thus, by Theorems 6 and 8, it follows that

Corollary 5. *Algorithm 6 is cost-optimal.*

4.3 Oriented Rings: Considerations on Time

In the previous section we have shown that the lower bound on the cost is tight, and can be achieved by two agents. This implies that the presence of multiple agents does not reduces the cost of locating the black hole. The natural question is whether the presence of more agents can be successfully exploited to *reduce the time* complexity.

Unlike the case of co-located agents, now the agents have to find each other to be able to distribute the workload. Note that, if the agents are able to quickly *gather* in a node, Algorithm 3 can be applied. As a consequence, in the remainder of this section, we focus on the problem to quickly group the agents.

If the number of agents k is known, the gathering problem can be easily solved by the following algorithm:

Algorithm 7 (Gathering – k known)

1. Each agent travels to the right using cautious walk.
2. When arriving at a node already visited by another agent, it proceeds to the right via the safe port.
3. If there is no safe port, it tests how many agents are at this node; if the number of agents at the node is $k-1$, the algorithm terminates. ◁▷

Eventually, since all agents travel to the right, all but one agent (which will reach the black hole) will be at the same node (in the worst case, the left neighbor of the black hole). Since, using cautious walk, it takes at most 3 time units to safely move to the right, and since there are at most $n-2$ such possible moves, this yields the following lemma:

Lemma 5. *If the numbers of agents k is known, $k-1$ agents can gather in an oriented ring in time $3n-6$.*

This strategy can not be applied when k is unknown. In fact, while the agents can follow the same algorithm as in the previous case, they have no means to know when to terminate (and, thus, to switch to Algorithm 3).

Actually, if *causal* time complexity is considered (i.e., length of the longest chain of causally related moves, over all possible executions of the algorithm), the additional agents can be of little help in the worst case:

Lemma 6. *The* causal time *complexity of locating the black hole in an oriented ring, using k agents is at least $n(\log n - \log k) - O(n)$.*

However, if the *bounded delay* time complexity is considered (i.e., assuming a global clock and that each move takes at most one time unit), the additional agents can indeed help. Initially, all agents are in state *alone*.

Algorithm 8 (Gathering – k unknown)

Rules for alone agent r:

1. Cautiously walk to the right until you meet another agent r'.
2. If r' is in state *alone*, form a group $\mathcal{G}$ (r and r' change state to *grouped*) and start executing the group algorithm.
3. Otherwise (r' is in state *grouped*, belonging to the group $\mathcal{G}'$, formed at the node g') join the group $\mathcal{G}'$: Go to g' and set your state to *Join*$[g']$.

Rules for group $\mathcal{G}$ formed at node g, consisting of $|\mathcal{G}|$ agents:

Execute Algorithm 3 using $|\mathcal{G}|$ agents, with the following actions taken after finishing each phase and before starting the next one:

1. If any of your agents have seen agents of another group $\mathcal{G}'$ whose starting node g' is to the right of g, join group $\mathcal{G}'$ by sending all your agents to g', with state *Join*$[g']$.
2. Otherwise add all the agents waiting at g with state *Join*$[g]$ to $\mathcal{G}$ and execute the next phase of Algorithm 3 using the updated number of agents. ◁▷

Theorem 9. *In oriented rings the black hole can be located by $k = n^{1/q}$ agents in* bounded delay *time complexity $O(qn^{1/q})$.*

4.4 Unoriented Ring

If the ring is unoriented, at least three dispersed agents are needed to locate the black hole (Fact 4). Thus we assume that there are at least three dispersed agents.

It is easy to convert a solution for oriented rings into one for the unoriented ones, at the cost of twice the number of moves and of agents.

Lemma 7. *Let $\mathcal{A}$ be an algorithm for oriented ring which, using p agents solves problem $\mathcal{P}$ in time T and cost C. Then there is an algorithm $\mathcal{A}'$ for unoriented ring which, using $2p-1$ agents, solves $\mathcal{P}$ in time T and complexity at most $2C$.*

Note that lemma 7 can be applied to all previous algorithms presented for scattered agents, except Algorithm 7.

From Theorem 8, Lemma 7, and Fact 4, it follows that:

Theorem 10. *Three (anonymous dispersed) agents are necessary and sufficient to locate the black hole in an unoriented ring.*

References

1. L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Gathering of mobile agents with incomparable labels. Tech. Rep. ECHO-01, Center for e-Computing, 2001.
2. M.A. Bender and D. Slonim. The power of team exploration: Two robots can learn unlabeled directed graphs. In *FOCS '94*, pages 75–85, 1994.
3. B. Brewington, R. Gray, and K. Moizumi. Mobile agents in distributed information retieval. *Intelligent Information Agents*, pages 355–395, 1999.
4. W. Chen, C. Leng, and Y. Lien. A novel mobile agent search algorithm. *Information Sciences*, 122(2-4):227–240, 2000.
5. K. Diks, A. Malinowski, and A. Pelc. Reliable token dispersal with random faults. *Parallel Processing Letters*, 4:417–427, 1994.
6. K. Diks, A. Malinowski, and A. Pelc. Token transfer in a faulty network. *Theoretical Informatics and Applications*, 29:383–400, 1995.
7. K. Diks and A. Pelc. Fault-tolerant linear broadcasting. *Nordic Journal of Computing*, 3:188–201, 1996.
8. S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Finding a black hole in an arbitrary network. Tech.Rep. ECHO-03, Center for e-Computing, 2001.
9. O. Goldreich and L. Shrira. Electing a leader in a ring with link failures. *Acta Informatica*, 24:79–91, 1987.
10. O. Goldreich and L. Shrira. On the complexity of computation in the presence of link failures: The case of a ring. *Distr. Computing*, 5:121–131, 1991.
11. J. Hammer and J. Fiedler. Using mobile crawlers to search the web efficiently. *International Journal of Computer and Information Systems*. To appear.
12. F. Hohl. A framework to protect mobile agents by using reference states. In *ICDCS 2000*, 2000.
13. N. R. Jennings. On agent-based software engineering. *Art. Intelligence*, 117(2):277–296, 2000.

14. L.M. Kirousis, E. Kranakis, D. Krizanc, and Y.C. Stamatiou. Locating information with uncertainty in fully interconnected networks. In *DISC '00*, pages 283–296, 2000.
15. K. Moizumi and G. Cybenko. The travelling agent problem. *Mathematics of Control, Signal and Systems*, 1998.
16. D. Rus, R. Gray, and D. Kotz. Autonomous and adaptive agents that gather information. In *AAAI '96*, pages 107–116, 1996.
17. T. Sander and C. F. Tschudin. Protecting mobile agents against malicious hosts. In *Mobile Agents and Security*, LNCS 1419, pages 44–60, 1998.
18. K. Schelderup and J. Ines. Mobile agent security - issues and directions. In *6th Int. Conf. on Intell. and Services in Networks*, LNCS 1597, pages 155–167, 1999.