OPODIS'02
Int. Conf. On Principles On DIstributed Systems
December 11–13, Reims, France

BLACK HOLE SEARCH BY MOBILE AGENTS IN HYPERCUBES AND RELATED NETWORKS

S. Dobrev* P. Flocchini† R. Královič‡ G. Prencipe§ P. Ružička¶ N. Santoro‖

Abstract: *Mobile agents operating in networked environments face threats from other agents as well as from the hosts (i.e., network sites) they visit. A* black hole *is a harmful host that destroys incoming agents without leaving any trace. To determine the location of such a harmful host is a dangerous but crucial task, called* black hole search. *The most important parameter for a solution strategy is the number of agents it requires (the* size*); the other parameter of interest is the total number of moves performed by the agents (the* cost*). Any solution requires $\Omega(n \log n)$ moves in general networks; the same lower bound holds for rings. In this paper we show that this lower bound does not hold for hypercubes and related networks. In fact, we present a general strategy which allows two agents to locate the black hole with $O(n)$ moves in* hypercubes, cube-connected cycles, star graphs, wrapped butterflies, chordal rings, *as well as in multidimensional* meshes *and* tori *of restricted diameter.*

Keywords: *Mobile Agents, Black Holes Search, Distributed Search, Optimal Protocols, Cube Networks.*

1 Introduction

The use of mobile agents is becoming increasingly popular when computing in networked environments, ranging from Internet to the Data Grid, both as a theoretical computational paradigm and as a system-supported programming platform.

Computing in such environments is termed *distributed mobile computing*[1]. In its terminology, a network site is a *host*; local processes are *stationary agents*; mobile agents *navigate* moving from host to neighboring host, and perform computations at each host, according to a predefined set of behavioral rules called *protocol*, the same for all agents; the agents are *asynchronous* in their actions (e.g., computation, movement, etc) (i.e., the amount of time required by an action is finite but otherwise unpredictable). The hosts provide a storage area called *whiteboard* for incoming agents to communicate and compute, and its access is held in fair mutual exclusion.

The major *practical* concern in these systems is definitely *security* [Che98, GBH98, Opp99, SO99]. Among the severe security threats faced in distributed mobile computing environments, two are particularly troublesome: *harmful agent* (that is, the presence of malicious mobile processes), and *harmful host* (that is, the presence at a network site of harmful stationary processes). The former problem is particularly acute in unregulated non-cooperative settings such as Internet (e.g., e-mail transmitted viruses). The latter not only exists in those settings, but also in environments with regulated access and where agents cooperate towards common goals (e.g., sharing of resources or distribution of a computation on the Grid [ACF+02]). In fact, a local (hardware or software) failure might render a host harmful.

The problem posed by the presence of a harmful host has been intensively studied from a programming point of view (e.g., see [Hoh98, Hoh00, SC99, ST98, VC99]), and recently also from an algorithmic prospective [DFPS01, DFPS02]. Obviously, the first step in any solution to such a problem must be to *identify*, if possible, the harmful host; i.e., to determine and report its location; following this phase, a "rescue" activity would conceivably be initiated to deal with the destructive process resident there. Depending on the nature of the danger, the task to identify the harmful host might be difficult, if not impossible, to perform.

Consider the presence in the network of a *black hole*: a host which *disposes* of visiting agents upon their arrival, leaving *no observable trace* of such a destruction [DFPS01, DFPS02]. The task is to unambiguously determine and report the location of the black hole, and will be called *black hole search*.

*School of Information Technology and Engineering, *University of Ottawa*, sdobrev@site.uottawa.ca.
†School of Information Technology and Engineering, *University of Ottawa*, flocchin@site.uottawa.ca.
‡Department of Computer Science, *Comenius University, Bratislava*, kralovic@dcs.fmph.uniba.sk
§Dipartimento di Informatica, *University of Pisa*, prencipe@di.unipi.it.
¶Department of Computer Science, *Comenius University, Bratislava*, ruzicka@dcs.fmph.uniba.sk
‖School of Computer Science, *Carleton University*, santoro@scs.carleton.ca.
[1]the term "mobile computing" alone is traditionally used in reference to wireless (e.g., "ad-hoc") networks.

Note that this type of highly harmful host is not rare; for example, the undetectable crash failure of a site in a asynchronous network turns such a site into a black hole. Hence the problem has an immediate practical relevance.

The searching agents start from the same safe site; the task is successfully completed if, within finite time, at least one agent survives and knows the location of the black hole. The research concern is to determine under what conditions and at what cost mobile agents can successfully accomplish this task.

Some answers follow from simple facts. For example, if the network is not biconnected, the problem is unsolvable[2]; hence, we will only consider biconnected networks. Similarly, at least two agents are needed to solve the problem.

The problem has been investigated and its solutions characterized for *ring* networks [DFPS01]. Subsequently, the problem has been studied also for arbitrary networks and different solutions and matching lower bounds were presented, depending on the amount of topological information available to the agents [DFPS02]. In particular, if the agents have full knowledge of the network topology, *two agents* are sufficient, and can locate the black hole using $\Theta(n \log n)$ moves.

A natural question to ask is whether the $O(n \log n)$ bound for two agents with full topological knowledge of a general network can be improved for networks with special topologies. A negative result holds for rings where $\Omega(n \log n)$ moves are needed by any two-agents solution [DFPS01].

In this paper we show that the negative result for rings does not generalizes. On the contrary, we present a general technique for efficient black hole location and prove that its application leads to $\Theta(n)$ protocols for most of the frequently used interconnection networks: *hypercubes*, *cube-related networks*, *chordal rings*, and multidimensional *tori* and *meshes*.

The paper is organized as follows. In the next section we present the model, definitions and basic properties. Section 3 contains the technique, together with the concepts leading to it. In section 4 we show that the algorithm has $\Theta(n)$ cost for most commonly used interconnection networks. The paper concludes in section 5.

2 Definitions and Basic Properties

Framework

Let $G = (V, E)$ be a simple 2-connected graph; let $n = |V|$ be the size of G, $E(x)$ be the links incident on $x \in V$, $d(x) = |E(x)|$ denote the degree of x, and Δ denote the maximum degree in G. If $(x, y) \in E$ then x and y are said to be neighbors. The nodes of G can be *anonymous* (i.e., without unique names).

At each node x, there is a distinct label (called port number) associated to each of its incident links (or ports); let $\lambda_x(x, z)$ denote the label associated at x to the link $(x, z) \in E(x)$, and λ_x denote the overall injective mapping at x. The set $\lambda = \{\lambda_x | x \in V\}$ of those mappings is called a *labelling* and we shall denote by (G, λ) the resulting edge-labelled graph.

Operating in (G, λ) is a set $\mathcal{A}$ of distinct autonomous mobile agents. The agents can move from node to neighboring node in G, have computing capabilities and bounded computational storage ($O(\log n)$ bits suffice for all our algorithms), obey the same set of behavioral rules (the *protocol*). The agents are *asynchronous* in the sense that every action they perform (computing, moving, etc) takes a finite but otherwise unpredictable amount of time. Initially, both agents are in the same node h, called *home base*.

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 gained fairly in mutual exclusion.

Black Hole Search

A *black hole* is a node where resides a stationary process 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.

The main measure of complexity of a solution protocol $\mathcal{P}$ is the number of agents used to locate the black hole, called the *size* of $\mathcal{P}$. Since one agent may immediately wander into the black hole, we have:

Observation 1 *At least two agents are needed to locate the black hole.*

On the other hand,

Observation 2 *[DFPS02] With full topological knowledge, two agents suffice to locate the black hole.*

[2] i.e., no deterministic protocol exists which always correctly terminates.

To have *complete topological knowledge* of (G, λ), means that the following information is available to all agents: (1) Knowledge of the labelled graph (G, λ); (2) Correspondence between port labels and the link labels of (G, λ); (3) Location of the home base in (G, λ).

Example: Consider a 5×9 mesh with the source node at position $(2, 3)$ from the lower left corner. (1) means the agents know they are in 5×9 mesh. Note that (1) implies the knowledge of N. (2) means agents know for each node which links lead to *north*, *east*, *south* and *west*. This knowledge implies ability to optimally route between any two nodes, even if there are given "forbidden" nodes which have to be avoided. (3) means the agents know they start at position $(2, 3)$ from the lower left corner.

This information is stronger then the more common *topological awareness* (i.e., knowledge of the class of the network, but not of its size nor of the source location – e.g. being in a mesh, starting from an unknown position), but it allows us to present a general algorithm for all biconnected networks. Moreover, we show that for many useful topologies (hypercubes, hypercube-related topologies, meshes, tori, ...) the additional information (network size, position of the source) required by *complete topological knowledge* can be cheaply determined from *topological awareness* and thus our assumption is not restrictive for those networks.

We will be interested in size optimal solutions with full topological knowledge. The measure of complexity of a size-optimal solution protocol $\mathcal{P}$ is the total number of moves performed by the agents, called the *cost* of $\mathcal{P}$.

Cautious Walk

At any moment of the execution of a protocol, the ports will be classified as *unexplored* – no agent has been sent/received via this port, *explored* – an agent has been received via this port or *active* – an agent has been sent via this port, but no agent has been received via it. Obviously, an explored port does not lead to a black hole; on the other hand, both unexplored and active ports might lead to it. To minimize the number of casualties (i.e., agents entering the black hole), we will not allow any agent to leave through an active port. To prevent the execution from stalling, we will require any active port not leading to the black hole, to be made explored as soon as possible.

This is accomplished as follows: Whenever an agent a leaves a node u through an unexplored port p (transforming it into active), upon its arrival to the node v, and before proceeding somewhere else, a returns to u (transforming that port into explored). This technique is called *Cautious Walk* and has been employed in [DFPS01, DFPS02]. Note that *Cautious Walk* would increase the cost of any asynchronous algorithm by at most $2n$.

3 The Solution Strategy

The novel and optimal solution strategy we propose exploits the notion of *traversal pairs* of a network; it employs it within the usual cooperative approach of dynamically dividing the work between the agents.

3.1 Traversal Pairs

The key element of our optimal solution strategy is the notion of *Traversal Pair* (shortly TP), which describes how the graph will be explored by the agents, and will be used by the agents to avoid "dangerous" parts of the network.

In the rest of the paper, we denote by O_G an arbitrary fixed total ordering $v_1 < v_2 \cdots < v_n$ of the nodes of G.

Definition 1 (Traversal Pair) *Let $G = (V, E)$ be an n-node graph with a total ordering O_G of its nodes. Let π_a and π_b be two walks in G starting from v_1 and v_n, respectively, and visiting the nodes of G in the order $v_1, v_2, \ldots, v_n$ and $v_n, v_{n-1}, \ldots, v_1$, respectively. Then $\pi = (\pi_a, \pi_b)$ is called v_1-v_n* traversal pair *of G with respect to O_G.*

The above definition binds a v_1-v_n TP to the ordering O_G. Throughout the paper we will need the following more general notions:

Definition 2

- *If, for fixed nodes $u, v \in V$, there exists an ordering O_G of V with $v_1 = u$ and $v_n = v$ such that there exists u-v TP of G with respect to O_G, we say that G has a u-v TP.*
- *We say that G is traversable if it has u-v TP for any $u, v \in V$,.*
- *If, for a fixed $u \in V$, there exists a neighbor v of u such that G has u-v TP, then we say that G has TP from u.*

We will call π_a (π_b) *left* (*right*) traversal.

As we will see later, a black hole location algorithm with home base v is based on a TP from v. However, in order to achieve good complexity, we need the TP to have good properties:

Definition 3 *Let $\pi = (\pi_a, \pi_b)$ be a u-v TP of G. Let G_i^a (G_i^b) denote the subgraph of G induced by vertices $v_1, v_2, \ldots, v_i$ ($v_n, v_{n-1}, \ldots, v_i$). Let $r(G_i^a)$ ($r(G_i^b)$) be the depth of the BFS tree of G_i^a (G_i^b) rooted at v_1 (v_n).*
We define the size *of π, denoted as $s_\pi(G)$ as $\max(|\pi_a|, |\pi_b|)$, the maximum of the lengths of walks π_a and π_r.*
The radius *of π, denoted as $r_\pi(G)$, is defined as $\max_{i=1}^{n}(\max(r(G_i^a), r(G_i^b)))$.*

Note that if a graph G which has Hamiltonian circuit, then G is traversable, with $s_\pi(G) \leq n$ and $r_\pi(G) \leq n$.

The following lemma shows that our notion of a traversable graph actually coincides with biconnectivity. (The proof is in the appendix.)

Lemma 3 *A graph G is traversable if and only if it is biconnected. Moreover, for every biconnected graph there exists a TP with linear size.*

Proof. To show the "only if" direction consider a traversable graph G. For the sake of contradiction let v be an articulation point of G. There exists a TP in G from every vertex u, so let's choose a TP π from a vertex $u \neq v$. There is some vertex w which is in a different biconnected component of G than u. It is clear that in both π_a and π_b, v is explored before w which is a contradiction.

The "if" direction is shown by an inductive construction. Consider a biconnected graph G and a vertex v. Let G_i denote the induced subgraph of G for which a TP has been already constructed ; $G_0 = \{v\}$. If $G_i = G$ we are done, otherwise take a TP π^i for G_i. Consider π_a^i. Let u be the first appearance of a vertex with a neighbor not in G_i. Since G is biconnected, there exists an "ear" – a path $u = u_0, u_1, \ldots, u_k = w$ such that $u, w \in G_i$ and $u_j \notin G_i$ for $1 \leq j < k$. The existence of an ear follows readily from the biconnectivity: u has one neighbor in G_i and another outside G_i and there is a circle containing both of them. Part of this circle outside G_i forms an ear. Moreover, as u was chosen to be the first appearance of a vertex with a neighbor outside G_i in π_a^i, it must be the case that w is explored after u in π_a^i. Given an ear we extend the TP π^i as follows. After the first occurrence of u in π_a^i we insert the sequence $u_1, u_2, \ldots, u_{k-1}, u_{k-2}, \ldots, u_1, u$ and the rest of π_a^i is left unchanged. With π_b^i, the situation is somewhat different: just before the first occurrence of u we add the preceding vertices of π_b^i in reverse order until an occurrence of w. Then we add the sequence $u_{k-1}, \ldots, u_1, v$ and the rest stays unchanged.

To show the size of π first note that both π_a and π_b move only along edges of a (spanning) tree. Therefore it is sufficient to show that every edge is traversed at most twice in both π_a and π_b. For π_a this fact follows from the simple observation that in the process of adding an ear only edges from that ear are added to the traversal. For π_b it is, however not the case, but it is easy to show that π_b^i is always a depth-first traversal of a tree. Hence the "return" phase between u and w uses only edges that were so far traversed only once. □

The following notation will be used in the rest of the paper. We will denote by $V[i,j]$ for $i < j$ the set of nodes $\{v_i, v_{i+1}, \ldots, v_j\}$. $G_{\widehat{ij}}$ is the graph induced by the nodes in $V \setminus V[i,j]$. $\pi_a[i,j]$ ($\pi_b[i,j]$) is defined as the segment of π_a (π_b) between the first occurrences of v_i and v_j.

3.2 Algorithm SplitWork

The protocol uses two agents, a and b, starting from the same node v_1; a TP (π_a, π_b) of G from v_1 is available to both. The algorithm proceeds in logical rounds. In each round the agents follow a usual cooperative approach of dynamically dividing the work between them: the unexplored area is partitioned into two parts of (almost) equal size. Each agent explores one part without entering the other one; exploration and avoidance are directed by the traversal pair. Since the parts are disjoint, one of them does not contain the black hole and the corresponding agent will complete its exploration. When this happens, the agent (reaches the last safe node visited by the other agent and there) partitions whatever is still left to be explored leaving a note for the other agent (should it be still alive). This process is repeated until the unexplored area consists of a single node: the black hole.

At any time, an agent will be either exploring its part of the network, or searching for the other agent to perform another partition, or destroyed by the black hole.

Let V_e denote the *safe* nodes[3]. At the beginning V_e consists only of the home base v_1. Let V_u be the set of unexplored nodes. (Initially, $V_u = V[2,n]$).

Partitioning $V_u = V[i,j]$ -

1. The agent performing the partition sets $V_a = V[i,k]$ and $V_b = V[k+1,j]$, where $k = \lfloor (i+j)/2 \rfloor$ (see Figure 1);
2. it then writes a note informing the other agent of the partition, and leaves to explore its assigned set.

Reaching and Exploring the Partition -

[3] A node is safe if there is a safe link incident to it, or it is the home base. The safe nodes represent the explored part of the network.

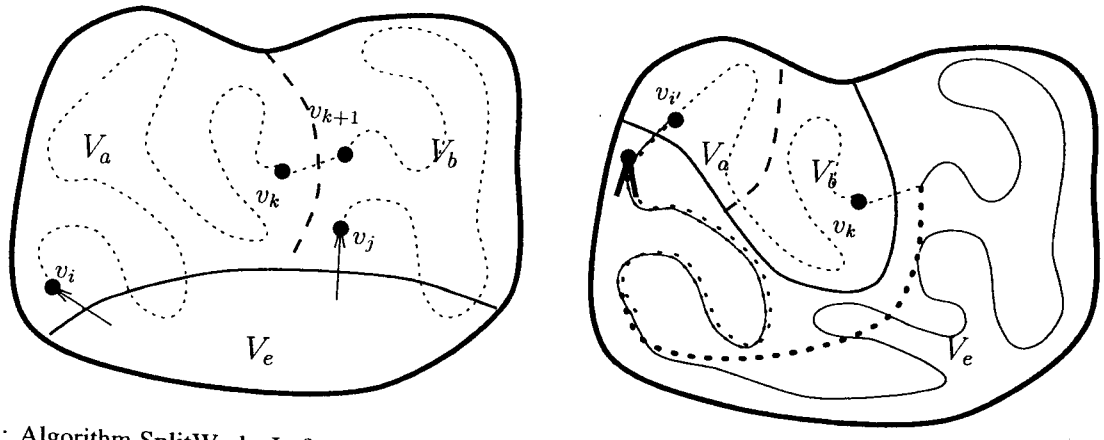

Figure 1: Algorithm SplitWork. Left: Beginning of a phase. Right: After finishing its part, the right agent finds the node from which the right agent departed to $v_{i'}$.

1. If a is the agent moving towards its new partition: a returns to v_{i-1} using the shortest possible route avoiding the new V_b; it then departs towards v_i and starts exploring V_a using *cautious walk* on $\pi_a[i,k]$.
2. If b is the agent moving towards its new partition: b returns to v_{j+1} using the shortest possible route avoiding the new V_a; it then departs towards v_j and starts exploring V_b using *cautious walk* on $\pi_b[j,k+1]$.
3. During the cautious walk, if the agent finds a note from the other agent informing it of the new partition V_a and V_b, it will reach and explore the new assigned part. **Note.** Since V_a and V_b do not overlap, one of them does not contain the black hole and the corresponding agent will finish its exploration.
4. When an agent completes the exploration of its part, it will search for the other agent to compute the new partition.

Searching for the other agent -

1. If a is searching for b: a goes to v_{j+1} (the node from which b departed towards its unexplored part) using the shortest possible route avoiding V_b. It then follows the safe links of the path $\pi_b[j,k+1]$ until it reaches the last safe vertex reached by b. Let $v_{j'}$ be the vertex to which b has departed. Then now $V_u = V[i,j']$.
2. If b is searching for a: b goes to v_{i-1} (the node from which a departed towards its unexplored part) using the shortest possible route avoiding V_a. It follows the safe links of the path $\pi_a[i,k]$ until it reaches the last safe vertex reached by a. Let $v_{i'}$ be the vertex to which a has departed. Then, now $V_u = V[i',j]$.
3. The agent computes the new partition of V_u.

Termination - When computing the new partition, if V_u contains a single node, that node is the black hole.

Lemma 4 *Let a graph G has a TP π from v of size $s_\pi(G)$ and radius $r_\pi(G)$. Then two agents placed at v can locate the black hole in G using $O(s_\pi(G) + r_\pi(G)\log n)$ moves.*

Proof. *Correctness.* Note that from the definition of TP and the way the algorithm works, it never happens that two agents depart to the same non-safe node[4]. This means that one agent will always survive. The fact that the agents never wait ensures progress of the algorithm. Since in each round the number of the unexplored nodes is halved, after $\log n$ rounds there is a single unexplored node and the algorithm terminates.

Complexity. We focus on the number of moves. The time complexity cannot be higher, and since there are only 2 agents, neither it could be asymptotically lower.

In each round, the only steps of the algorithm when agents move are

(1) Exploring the assigned area (w.l.g assume b explored whole G_b, while a explored only part of G_a.

(2) Moving to the "beginning" of G_a.

(3) Chasing the other agent through the area it newly explored.

[4]The algorithm uses the TP to be able to safely explore V_a without wandering into V_b, and vice versa. TP allows us to specify in a unified format the way V_a and V_b are explored, regardless of the actual values of V_a and V_b, which depend of the specifics of the particular execution.

(4) Moving to the "starting" node for the next round.

Let v_h be the node containing the black hole. Note that the total exploration path performed by the agent a during step (1) over all rounds is at most $|\pi_a[1,h]|$. (The bound is $|\pi_b[h,n]+1|$ for b.) Clearly, the total cost of step (1) over all rounds is less then $2s_\pi(G)$. Using similar arguments the same bound holds also for the total cost of step (3).

The cost of steps (2) and (4) for one agent in one round is clearly bound by $2r_\pi(G)$.

Combining with the fact that there are at most $\lceil \log n \rceil$ rounds results in $O(s_\pi(G) + r_\pi(G)\log n$ bound in the number of moves. □

4 BH Search in Interconnection Networks

In the previous section we have presented a general algorithm for black hole location by two agents which are provided with a traversal pair of the network.

In this section we first of all present a general technique for *constructing* such a pair. We then show that that the traversal pairs constructed by this technique have low size and radius in all the common interconnection networks; thus, when used in the proposed algorithm, they lead to size and cost optimal solutions to the black hole search problem. Finally, we show that, for those networks, the requirement of complete topological knowledge can be relaxed to the conventional one of topological awareness.

4.1 TP composition

In this subsection we present a technique for construction of TP based on hierarchical decomposition of the graph, making use of the TPs of the graph's components. The technique is not optimal, but since it is sufficient to find a traversal pair of linear size and $O(n/\log n)$ diameter, it leads to linear black hole search algorithms for most of the commonly used interconnection networks.

Let $H = (V_H, E_H)$ be a biconnected graph with $|V_H| = k$. Let $\pi_H = (\pi_a^H, \pi_r^H)$ be a traversal pair of H. We will denote by $s_\pi(H)$ the size and by $r_\pi(H)$ the radius of this TP.

Let $F_1 = (V_1, E_1)$, $F_2 = (V_2, E_2), \ldots, F_k = (V_k, E_k)$ be a set of traversable biconnected graphs. Let us denote by $s_\pi(F_i)$ the maximal (with respect to u, v) size of u-v TP of F_i and by $r_\pi(F_i)$ the maximal radius of a u-v TP of F_i. Define $r_\pi(F) = \max_{i=1}^k(r_\pi(F_i))$. Let $d(G)$ denote the diameter of graph G and let $d(F) = \max_{i=1}^k(d(F_i))$.

Definition 4 *We say that $G = (V, E)$ is a TP-composition of H and $F_1, F_2, \ldots, F_k$ if and only if $V = \cup_{i=1}^k V_i$ and E satisfies:*

1. $\cup_{i=1}^k E_i \subset E$
2. *If $(v_i, v_j) \in E_H$ then there exists an edge $(u_i, u_j) \in E$ such that $u_i \in V_i$ and $u_j \in V_j$.*
3. *Let v_{l_i} and v_{r_i} are the nodes from which v_i is for the first time visited in π_a^H and π_b^H, respectively. Then $\forall i : 2 \leq i < k$ holds: There are two different nodes $u, u' \in V_i$ such that u has a neighbor in V_{l_i} and u' has a neighbor in V_{r_i}.*

Moreover, if $\forall i : 1 \leq i \leq k$, $\forall u \in V_i$ holds: if $(v_i, v_j) \in E_H$ then $\exists u' \in V_i$ such that the distance from u to u' in F_i is less than c, and $\exists w \in V_j$ such that $(u', w) \in E$, we say that G has dilation c.

Informally, we replace a vertex v_i of H by graph F_i; the connectivity requirements are designed in order to allow the TP of H to be extended to TP of G.

Lemma 5 *Let $G = (V, E)$ is a TP-composition of H with $F_1, F_2, \ldots, F_k$. Then G has a TP from any vertex u_1 of V_1 such that u_1 has a neighbor in V_k, of size at most $d(F)s_\pi(H) + \sum_{i=1}^k s_\pi(F_i)$ and radius at most $r_\pi(F) + (d(F) + 1)r_\pi(H)$.*

Moreover, if G has dilation c, then G has TP of size at most $cs_\pi(H) + \sum_{i=1}^k s_\pi(F_i)$ and radius at most $r_\pi(F) + cr_\pi(H)$.

Finally, if G has dilation 1, then G has TP of radius at most $\max(r_\pi(H) + d(F_1), r_\pi(F_1))$.

Proof. We show how to construct π_a^G (the left traversal of G), π_b^G is constructed analogously.

Assume now that we have arrived for the first time to a node u_i of V_i (works also for $i = 1$). That corresponds to the first occurrence of v_i in π_a^H. Let v_{a_i} (v_{b_i}) be the node from which v_i was first time reached in π_a^H (π_b^H) and let v_j be the node following the first occurrence of v_i in π_a^H (j could be $i+1$, but may also be any number smaller than i).

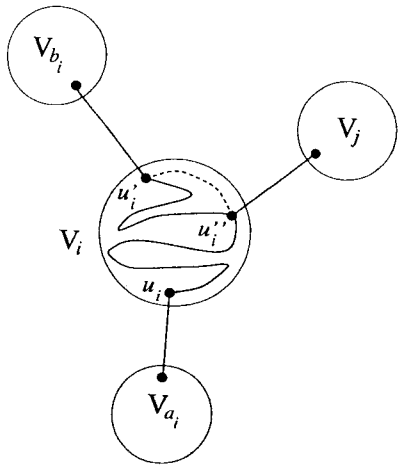

Figure 2: Extending the traversal in F_i.

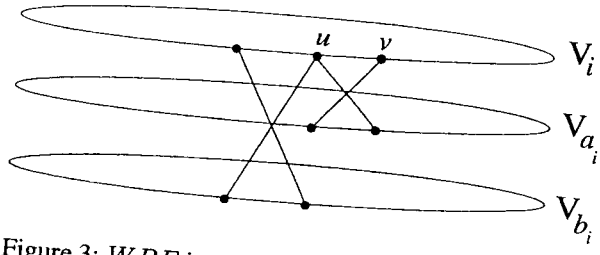

Figure 3: WBF is a uniform TP-composition of a hypercube and a cycle.

Let u'_i be any node of V_i different from u_i which has a neighbor in V_{a_i} (from the third point of Definition 4 we know that such node must exist) and u''_i be any node of V_i which has a neighbor in V_j.

We extend π_a^G from u_i by left u_i-u'_i traversal of F_i ($\pi_a^{F_i}$), followed by a path from u'_i to u''_i (if $u'_i \neq u''_i$) and then crossing the edge to V_j (see Figure 2).

If $j \neq i+1$, the node v_j has already been visited in π_a^H, which means that also all nodes in V_j have been visited. Let v'_j be the next node in π_a^H and let w be a node from V_j which has a neighbor in $V_{j'}$. We extend π_a^G by adding the shortest path (in F_j) leading to w, and adding the link to $V_{j'}$.

The right traversal is constructed analogously: We first leave u_1 for u'_b in V_b. When we come for the first time to a vertex u'_i of V_i, we use the right u'_i-u_i traversal ($\pi_b^{F_i}$) to visit the nodes of V_i in the exactly opposite order. The already visited components are crossed using shortest paths.

Size: Exploring a component F_i for the first time costs $s_\pi(F_i)$ moves, plus at most $d(F_i) < d(F)$ for crossing from u'_i to u''_i, if needed. Each next occurrence of v_i in π_l^H ($\pi_b(H)$) corresponds to additional crossing of F_i, of length at most $d(F)$. Summing up over the whole length of π_l^H ($\pi_b(H)$) produces the result. If G has dilation c, then all crossings can be done with at most c links (including the link for crossing to the next component).

Radius: Let $w_i \in V_i$ be any node of G. We will show how to construct a path from w_i to the home base u_1 such that it visit only the nodes which were in $\pi_a(G)$ visited before w_i. The case for $\pi_b(G)$ is analogous.

Note that if $i > 1$ then all components F_j for $j < i$ have already been fully explored. Let u_i be the first visited node of F_i (home base if $i = 1$). The path from w_i to u_1 is constructed as follows: (1) First take a safe path from w_i to u_i. From the definition of $r_\pi(F_i)$ we know that there is such a path, of length at most $r_\pi(F_i) \leq r_\pi(F)$. (2) The path from u_i to u_1 is constructed by embedding the path from v_i to v_1 in H into G: An edge (v_x, v_y) is replaced by an edge (u_x, u'_y), followed by the shortest path from u'_y to u_y, where u'_y is the neighbor of u_x in F_y and u_y is any node of F_y which has a neighbor in the next component on the path. (If $y = 1$, we go to the home base and are finished.) There are at most $r_\pi(H)$ edges in the path from v_i to v_1, and each is replaced by a path of length at most $1 + d(F)$. Summing with the cost of (1) results in $r_\pi(F) + (d(F)+1)r_\pi(H)$.

If G has dilation c then the term $d(F)$ can be replaced by $c-1$. If G has dilation 1 then each node of F_i (hence also w_i) is connected to all neighboring components. That means that we do not need to add the path from w_i to u_i, but can leave F_i directly from w_i. When we arrive to F_1, we still need to come to u_1, hence the total length is $r_\pi(H) + d(F_1)$. However, if $i = 1$, F_1 is not yet fully explored; the shortest path from w_i to u_1 may not be safe, so we need to use the safe one of length $r_\pi(F_1)$, instead of $d(F_1)$.

Quite often a more limited composition will be sufficient: □

Definition 5 *We say that G is uniform TP-composition of H and F if G is TP composition of H with $F_1, F_2, \ldots, F_r$, where $\forall i : 1 \leq i \leq r$ $F = F_i$.*

Directly applying Lemma 5 yields:

Corollary 6 *Let G be a uniform TP-composition of H and F such that $s_\pi(F) \leq c|F|$ for some constant c. If there is a TP for H of size $s_\pi(H) \leq q|H|$ where $q \geq 2c$ then there is a TP for G of size $s_\pi(G) \leq q|G|$.*

Proof. Lemma 5 bounds the size of TP for G to be at most $d(F)s_\pi(H) + ks_\pi(F)$. Since F is biconnected it holds $d(F) \leq |F|/2$ yielding $s_\pi(G) \leq q|H||F|/2 + kc|F|$. As $|H||F| = k|F| = |G|$ we get $s_\pi(G) \leq q|G|$. □

4.2 Results for Specific Topologies

Lemma 5 and Corollary 6 can be used to find good traversal pairs in a number of graphs.

Lemma 7 *Let G be a d-dimensional torus with n vertices and diameter $diam(G)$. Then G is traversable with a TP of size at most $4n$ and radius $diam(G)$.*

Proof. We denote $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ and ε_i a vector with a single 1 at position i. Torus is a Cayley graph over a group $\mathbb{Z}_{dim_1} \times \cdots \times \mathbb{Z}_{dim_d}$ where all $dim_i > 2$, with generators $\pm\varepsilon_i$. As Cayley graphs are vertex transitive, it is sufficient to show the existence of a TP from one vertex.

If $d = 1$ then G is a cycle and there is a traversal of size $2n$ and radius $n/2 = diam(G)$. Now consider a d-dimensional torus for $d > 1$. W.l.o.g we may assume $dim_1 \leq dim_2 \leq \cdots \leq dim_d$. The diameter of G is $\frac{1}{2}\sum_{i=1}^{d} dim_i$. Let F be the cycle in first dimension, i.e. of length dim_1. F is biconnected and traversable with $s_\pi(F) = 2dim_1$ and $r_\pi(F) \leq dim_1$.

We show that G has a uniform TP-composition with dilation 1 of H and F where H is a $(d-1)$-dimensional torus with dimensions $dim_2, \ldots, dim_d$. The first two conditions in Definition 4 are trivial. The fact that G has dilation 1 follows directly from the commutativity of the group. Consider a vertex u in a set V_i. The set V_i consists of vertices $u + c\varepsilon_1$ for all c. If u has a neighbor in some other component V_k, say, $v = u + \varepsilon_j$ then every $u + c\varepsilon_1$ has neighbor $u + c\varepsilon_1 + \varepsilon_j = v + c\varepsilon_1$ in V_k. The third condition of Definition 4 is a consequence of G having a dilation 1.

We prove the bound on the size and radius of the TP by induction on the number of dimensions. The first step (a ring) is trivial. Following the induction hypothesis, H has a TP with size $s_\pi(H) = 4|H|$ and radius $r_\pi(H) = diam(H)$. Using Corollary 6 we conclude that G has a TP of size at most $4n$. To bound the diameter, we combine the fact that G has dilation 1 with Lemma 5 yielding $r_\pi(G) \leq \max\{r_\pi(H) + diam(F), r_\pi(F)\}$. Since $diam(F) = dim_1/2$, the term $r_\pi(H) + diam(F) = \frac{1}{2}\sum_{i=1}^{d} dim_i = diam(G)$. The result follows from the fact that $r_\pi(F) \leq dim_1 \leq dim_1/2 + dim_2/2$. □

Lemma 8 *Let G be a d-dimensional hypercube. Then G is traversable with a TP of size at most $4n$ and radius $diam(G)$.*

Proof. It is the same as the proof of Lemma 7 with all $dim_i = 2$. This time, however, we set F to be a cycle of length four induced by the first two dimensions and then H is a $(d-2)$-dimensional hypercube. The $diam(F) = 2$, $r_\pi(F) \leq 4$ and $diam(H) = d - 2$. The basis of the induction are cases $d = 2$ and $d = 3$. □

Lemma 9 *Cube-connected cycles $CCC(d)$ and wrapped butterfly $WBF(d)$ are traversable with a TP of size $4n$ and radius $O(d^2)$.*

Proof. It is sufficient to show that both topologies are uniform TP-compositions of a d-dimensional hypercube with a cycle of length d. The size then comes from Corollary 6 and Corollary 8 and radius from Lemma 5 as $r_\pi(G) \leq r_\pi(F) + (diam(F) + 1)r_\pi(H) \leq d + (d/2 + 1)d$.

To prove the TP-composition property consider the cycles corresponding to a particular hypercube vertex (i.e. induced by "shift" operations) in both topologies. The only nontrivial part to show is the condition 3 in Definition 4.

CCC: Consider a circle V_i in $CCC(d)$. Every vertex $v \in V_i$ has exactly one neighbor outside V_i, and any two distinct vertices in the circle V_i have their outside neighbors in different circles (corresponding to neighbors in the hypercube along appropriate dimensions). The condition 3 follows from the fact that $v_{l_i} \neq v_{r_i}$.

$WBF(d)$: Condition 3 directly follows from the fact that in each circle V_i and V_j in $WBF(d)$ there are two different nodes $u, v \in V_i$ which have a neighbor in V_j (see Figure 3). □

Lemma 10 *The star graph $S(d)$ has a TP of size at most $4n$ and radius at most $2^{d-1} + 1$.*

Proof. The star graph $S(d)$ is a Cayley graph over the symmetric group $\mathbb{S}_d$ generated by the involutions $(1, q)$ for $1 < q \leq d$. Let $S(k, d)$ be a Cayley graph over the coset group $\mathbb{S}_d|\mathbb{S}_k$ with generators $g \circ (1, q)[\mathbb{S}_k]$ where $g \in \mathbb{S}_k$ and $k < q \leq d$. Clearly, $S(1, d) = S(d)$ and $S(d-1, d) = K_d$ is a complete graph with d vertices. We can visualize the vertices of $S(k, d)$ as strings of length $d - k$ consisting of different symbols from the alphabet $\{1, 2, \ldots, d\}$. The edges of $S(k, d)$ connect vertices which differ in exactly one place.

Now we show that $S(k, d)$, $2 \leq k < d - 1$ is a uniform TP-composition of $H = S(k + 1, d)$ and $F = K_{k+1}$. We have to prove that the three conditions from Definition 4 are fulfilled. The first one is trivial. For the remaining two we show that if there is an edge $(v_i, v_j) \in E_H$, there are two pairs of vertices $(u_i, u_j) \in E$, $(u'_i, u'_j) \in E$ such that $u_i, u'_i \in V_i$ and $u_j, u'_j \in V_j$. Consider an edge $(v_i, v_j) \in E_H$ where $v_i = \alpha a \beta$, $v_j = \alpha c \beta$; here α, β stand for strings.

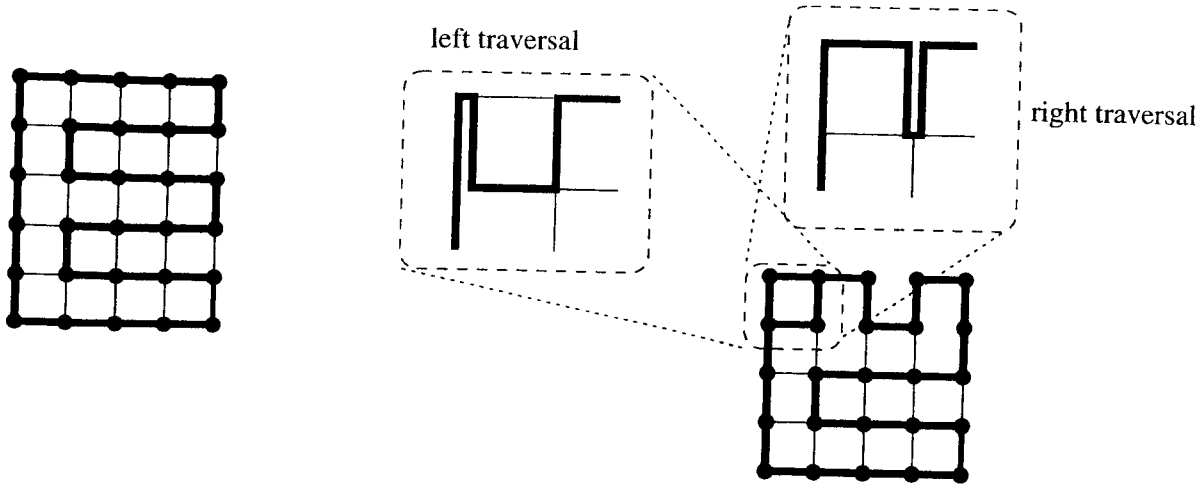

Figure 4: Left: Hamiltonian circuit defining the left/right traversal pair for a mesh with at least one side of even length. Right: Left/right traversals of a mesh with both sides of odd length (only the top two rows differ from the even case).

As $k \geq 2$, there are two distinct symbols g, g' not present in α, β and different from q, c. Let $u_i = b\alpha a\beta$, $u'_i = g'\alpha q\beta$, $u_j = g\alpha c\beta$ and $u'_j = g'\alpha c\beta$. It is easy to see that $(u_i, u_j) \in E$ and $(u'_i, u'_j) \in E$.

As a next step we shall prove that $S(k,d)$, $2 \leq k < d-1$ has a TP of size $2d!/k!$ and radius $2^{d-k}-1$. For $k = d-1$ the statement clearly holds. As $S(k,d)$ is a uniform TP-composition of $S(k+1,d)$ and K_{k+1} we get the size from Corollary 6 and the radius from Lemma 5, as $r_\pi(G) \leq r_\pi(F) + (d(F)+1)r_\pi(H) = 1 + 2 \cdot (2^{d-k-1}-1) = 2^{d-k}-1$.

In order to finish the proof we have to bridge the gap between $S(d) = S(1,d)$ and $S(3,d)$. Similar arguments as above lead to conclusion that $S(d)$ is a uniform TP-composition of $S(3,d)$ and a circle of length 6 and the result follows. □

Lemma 11 *Let G be a d-dimensional mesh with n vertices and diameter $diam(G)$. Then G is traversable with a TP of size at most $4n$ and radius $diam(G)$.*

Proof. (Sketch.) By induction on the number of dimensions. The basis of induction are $d = 2$ and $d = 3$. The TP for a $2D$ mesh is depicted in Figure 4. Its size is n for a mesh with at least one side even, and $n+2$ for all sides odd. It is not difficult to see that the radius of this TP is no more that $diam(G)+1$.

The TP for a $3D$ mesh is depicted in Figure 5. The left traversal starts by going to the topmost $2D$ submesh using the $(0,0)$ column. The $2D$ meshes are then traversed from the top to the bottom using a left traversal for $2D$ mesh from $(0,1)$, ending at $(1,0)$, returning to $(0,1)$ and going down. The right traversal traverses $2D$ meshes from the bottom to the top, and returns by the $(0,0)$ column. Each $2D$ mesh is traversed using right traversal for $2D$ mesh starting at $1,0)$ and ending at $(0,1)$, then returning to $(1,0)$ and moving one level up. Again, it is easy to see that the size of this TP is $O(n)$ and its radius is $O(diam(G))$.

A d dimensional mesh for $d \geq 4$ is a uniform TP composition of a $2D$ mesh F with a $d-2$ dimensional mesh H with dilation 1. The proof (as well as of the result bounds on its size and diameter) is analogous to the tori and hypercube case. □

Combining the results of this subsection with Lemma 4 yields the main theorem of this paper:

Theorem 12 *Two agents can locate the black hole in $O(n)$ moves in all of the following topologies: tori and meshes of diameter $O(n/\log n)$, hypercubes, CCC, wrapped butterflies and star graphs.*

4.3 Relaxing the Knowledge Requirements

In deriving our results, we have assumed that the agents have complete topological knowledge; that is, the agents know not only the network topology type and labelling (e.g., torus with "N-S-E-W" labelling), but also the actual size n of the network and the location of the home base.

However, for the class of networks under investigation, our stronger assumptions can be dropped, since the additional information can be efficiently acquired. This is achieved by adding a precomputation phase, in which agents compute the size of the network and the location of the home base (knowledge of the topology class e.g. CCC, or mesh is still assumed, as well as a knowledge of globally consistent labelling, e.g. being able to distinguish between cycle and hypercube edges in CCC, or identify north, east, south, west in a 2-dimensional mesh).

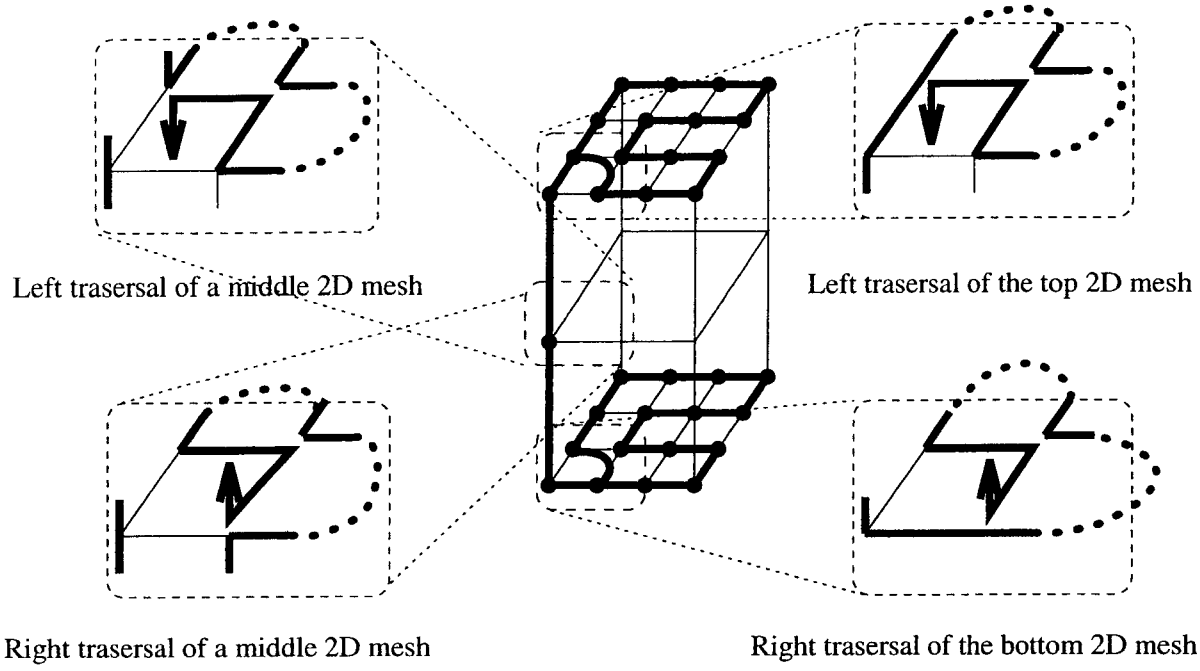

Figure 5: TP for a 3D mesh.

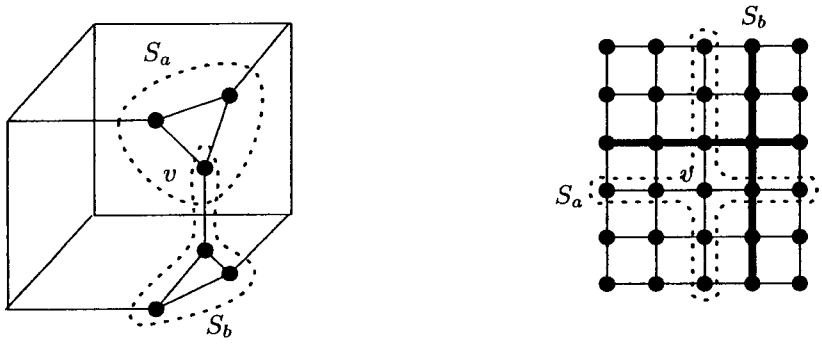

Figure 6: Left: S_l and S_r in a CCC. Right: S_l and S_r in a two dimensional mesh.

For vertex symmetric topologies (tori, hypercubes, CCC, wrapped butterflies and star graphs), the problem of identifying the location of the home base is irrelevant, as all nodes are alike. For hypercubes and star graphs the size immediately follows from the degree of nodes; in tori and meshes the number of dimensions can be determined that way.

We present a general scheme for determining n and location of the home base (if relevant), and show how to apply it to CCC and 2-dimensional meshes, the extensions to wrapped butterflies and multidimensional meshes and tori are quite straightforward.

The general scheme

(1) Choose two disjoint sets of vertices in G: S_a and S_b such that $S_a \cap S_b = \{v\}$ (v is the home base) and it is possible to determine the size of the network (and the location of the home base, if needed) from each of them independently. See Figure 6, left.

(2) If no such sets can be found, explore some neighborhood S' of v in a way that at least one agent survives. S' is chosen such that for every $|S'| - 1$ node subset S'' of S' there exist S_a and S_a such that $S_a \cap S_a \subset S''$ and n and location of v can be determined from each of them. See Figure 6, right, for an example for q 2-dimensional

mesh: S' consists of the four direct neighbors of v. The cross S_b is chosen to intersect S_a in two neighbors of v which are known to be safe.

(3) The agents a and b explore S_a and S_b, respectively, and return to the home base. The way S_a and S_b were chosen ensures that at least one of them (w.l.o.g. assume that b) succeeds.

(4) b goes to the last safe node visited by a and leaves a mark with the meaning "Stop exploring S_a, a already know n and location of v. Join me in the Algorithm SplitWork." and starts executing the Algorithm SplitWork.

(5) Let v_i be the node to which a was traveling when b left the mark for it. The first assignment of V_a and V_b will not be $V[2..\lfloor n/2 \rfloor]$ and $V[\lfloor n/2 \rfloor + 1, n]$, but $V[2..i-1]$ and $V[i+1..n]$. Furthermore, if a is still blocked at i when b finishes its part, b will "switch" with a (i.e. b will start exploring from the left, while a will be asked to explore from the right). This prevents both agents disappearing in i if the black hole is there.

Note that the cost of such precomputation is $O(|S'| + |S_1| + |S_2|)$, which is for all relevant topologies $O(n)$.

5 Conclusions

We have presented a novel concept, traversal pairs of a biconnected graph, and shown how to use it to obtain a *size-optimal* black hole searching technique. We have shown that this technique leads to solutions which are also *cost-optimal* for all the common interconnection networks.

The outstanding open question is to determine for what other types of networks $\Theta(n)$ cost can be achieved by two searching agents.

References

[ACF+02] W. Allcock, A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke. The data grid: Towards an architecture for the distributed management and analysis of large scientific dataset. *J. of Network and Computer Applications*, 2002.

[Che98] David M. Chess. Security issues in mobile code systems. In *Proc. Conf. on Mobile Agent Security*, LNCS 1419, pages 1–14, 1998.

[DFPS01] S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Mobile agents searching for a black hole in an anonymous ring. In *Proc. of 15th Int. Symposium on Distributed Computing (DISC 2001)*, pages 166–179, 2001.

[DFPS02] S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Finding a black hole in an arbitrary network: optimal mobile agents protocols. In *Proc. of 21st ACM Symposium on Principles of Distributed Computing (PODC 2002)*, pages 153–162, 2002.

[GBH98] M.S. Greenberg, J.C. Byington, and D. G. Harper. Mobile agents and security. *IEEE Commun. Mag.*, 36(7):76 – 85, 1998.

[Hoh98] F. Hohl. Time limited blackbox security: Protecting mobile agents from malicious hosts. In *Proc. of Conf on Mobile Agent Security*, LNCS 1419, pages 92–113, 1998.

[Hoh00] F. Hohl. A framework to protect mobile agents by using reference states. In *Proc. of the 20th Int. Conf. on Distributed Computing Systems (ICDCS 2000)*, 2000.

[Opp99] R. Oppliger. Security issues related to mobile code and agent-based systems. *Computer Communications*, 22(12):1165 – 1170, 1999.

[SC99] S.K.Ng and K.W. Cheung. Protecting mobile agents against malicious hosts by intention spreading. In *Proc. 1999 Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'99)*, pages 725–729, 1999.

[SO99] K. Schelderup and J. Ones. Mobile agent security - issues and directions. In *Proc. 6th Int. Conf. on Intelligence and Services in Networks*, LNCS 1597, pages 155–167, 1999.

[ST98] T. Sander and C. F. Tschudin. Protecting mobile agents against malicious hosts. In *Proc. of Conf on Mobile Agent Security*, LNCS 1419, pages 44–60, 1998.

[VC99] Jan Vitek and Giuseppe Castagna. Mobile computations and hostile hosts. In D. Tsichritzis, editor, *Mobile Objects*, pages 241–261. University of Geneva, 1999.