Fundamenta Informaticae 71 (2-3) (2006) 229–242
IOS Press

Complexity of searching for a black hole

Jurek Czyzowicz [*]
Département d'informatique
Université du Québec en Outaouais
Gatineau, Québec
J8X 3X7, Canada
jurek@uqo.ca

Dariusz Kowalski [†]
Department of Computer Science
The University of Liverpool
Chadwick Building, Peach Street
Liverpool L69 7ZF, UK
D.R.Kowalski@csc.liv.ac.uk

Euripides Markou [‡]
Department of Informatics & Telecommunications
National Kapodistrian University of Athens
Panepistimiopolis 15784
Athens, Greece
emarkou@cs.ntua.gr

Andrzej Pelc [§]
Département d'informatique
Université du Québec en Outaouais
Gatineau, Québec
J8X 3X7, Canada
pelc@uqo.ca

Abstract. A black hole is a highly harmful stationary process residing in a node of a network and destroying all mobile agents visiting the node, without leaving any trace. We consider the task of locating a black hole in a (partially) synchronous network, assuming an upper bound on the time of any edge traversal by an agent. The minimum number of agents capable to identify a black hole is two. For a given graph and given starting node we are interested in the fastest possible black hole search by two agents, under the general scenario in which some subset of nodes is safe and the black hole can be located in one of the remaining nodes. We show that the problem of finding the fastest possible black hole search scheme by two agents is NP-hard, and we give a 9.3-approximation for it.

Keywords: approximation algorithm, black hole, graph, mobile agent, NP-hard problem

[*]Research supported in part by NSERC grant
[†]Research supported in part by grants from KBN (4T11C04425) and UE DELIS
[‡]This work was done during this author's stay at the Research Chair in Distributed Computing of the Université du Québec en Outaouais, as a postdoctoral fellow
[§]Research supported in part by NSERC grant and by the Research Chair in Distributed Computing of the Université du Québec en Outaouais
Address for correspondence: E. Markou, Department of Informatics & Telecommunications, National Kapodistrian University of Athens, Panepistimiopolis 15784, Athens, Greece

1. Introduction

1.1. The background and the problem

Security of mobile agents working in a network environment is an important issue which has recently attracted attention of many researchers. Protecting mobile agents from “host attacks”, i.e., harmful items stored in nodes of the network, has become almost as urgent as protecting a host, i.e., a node of the network, from an agent’s attack [12, 13]. Various methods of protecting mobile agents against malicious hosts have been discussed, e.g., in [7, 8, 10, 12, 13, 14].

In this paper we consider hostile hosts of a particularly harmful nature, called *black holes* [1, 2, 3, 4, 5]. A black hole is a stationary process residing in a node of a network and destroying all mobile agents visiting the node, without leaving any trace. Since agents cannot prevent being annihilated once they visit a black hole, the only way of protection against such processes is identifying the hostile node and avoiding further visiting it. Hence we are dealing with the issue of locating a black hole: assuming that there is at most one black hole in the network, at least one surviving agent must find the location of the black hole if it exists, or answer that there is no black hole, otherwise. The only way to locate the black hole is to visit it by at least one agent, hence, as observed in [3], at least two agents are necessary for one of them to locate the black hole and survive. Throughout the paper we assume that the number of agents is minimum possible for our task, i.e., 2, and that they start from the same node.

In [2, 3, 4, 5] the issue of efficient black hole search was extensively studied in many types of networks. The underlying assumption in these papers was that the network is totally asynchronous, i.e., while every edge traversal by a mobile agent takes finite time, there is no upper bound on this time. In this setting it was observed that, in order to solve the problem, the network must be 2-connected, in particular black hole search is infeasible in trees. This is because, in asynchronous networks it is impossible to distinguish a black hole from a “slow”link incident to it. Hence the only way to locate a black hole is to visit all other nodes and learn that they are safe. (In particular, it is impossible to answer the question of whether a black hole actually exists in the network, hence [2, 3, 4, 5] worked under the assumption that there is exactly one black hole and the task was to locate it.)

Totally asynchronous networks rarely occur in practice. Often a (possibly large) upper bound on the time of traversing any edge by an agent can be established. Hence it is interesting to study black hole search in such partially synchronous networks. Without loss of generality, this upper bound on edge traversal time can be normalized to 1 which yields the following definition of the time of a black hole search scheme: this is the maximum time taken by the scheme, i.e. the time under the worst-case location of the black hole (or when it does not exist in the network), assuming that all edge traversals take time 1. This was the scenario adopted in [1], and we use it in the present paper as well.

The partially synchronous scenario makes a dramatic change to the problem of searching for a black hole. Now it is possible to use the time-out mechanism to locate the black hole in any graph, with only two agents, as follows: agents proceed along edges of a spanning tree. If they are at a safe node v, one agent goes to the adjacent node and returns, while the other agent waits at v. If after time 2 the first agent has not returned, the other one survives and knows the location of the black hole. Otherwise, the adjacent node is known to be safe and both agents can move to it. This is in fact a variant of the *cautious walk* described in [3] but combining it with the time-out mechanism makes black hole search feasible in any graph. Hence the issue is now not the feasibility but the time efficiency of black hole search, and the present paper is devoted to this problem.

In all previous papers on black hole search it was assumed that the starting node s is safe (otherwise the agents would be immediately anihilated) and the black hole can be located in any other node. However, in practice, some other nodes may also be known as safe, e.g., because they were already verified in some previous network exploration. Hence we adopt a more general scenario in which an arbitrary subset of nodes of the network, containing the starting node, is safe, and the black hole can be located in one of the remaining nodes.

The time of a black hole search scheme should be distinguished from the time complexity of the algorithm producing such a scheme. While the first was defined above for a given input consisting of a network and a starting node, and is in fact the larger of the numbers of time units spent by the two agents in the worst case, the second is the time of producing such a scheme by the algorithm. In other words, the time of the scheme is the time of walking and the time complexity of the algorithm is the time of thinking.

1.2. Our results

We show that the problem of finding the fastest possible black hole search scheme by two agents in an arbitrary graph is NP-hard, and we give a 9.3-approximation for this problem, i.e., we construct a polynomial time algorithm which, given a graph with a subset of safe nodes and a starting node as input, produces, in polynomial time, a black hole search scheme whose time is at most 9.3 times larger than the time of the fastest scheme for this input.

2. Model and terminology

We consider a graph G with node s which is the starting node of both agents. We assume that a subset S of nodes containing s cannot contain a black hole. These nodes are called *safe*. Each of the remaining nodes can contain a black hole; they are called *unsafe*. We assume that there is at most one black hole in the network. This is a node which destroys any agents visiting it. A black hole search scheme (*BHS-scheme*) for the input (G, S, s) is a pair of sequences of edge traversals (moves) of each of the two agents, with the following properties.

- Each move takes one time unit (if the agent moved faster, it waits till the end of the time unit at the target node).
- Upon completion of the scheme there is at least one surviving agent, i.e., an agent that has not visited the black hole, and this agent either knows the location of the black hole or knows that there is no black hole in the graph. The surviving agents must return to s.

The time of a black hole search scheme is the number of time units until the completion of the scheme, assuming the worst-case location of the black hole in the complement of S (or its absence, whichever is worse). It is easy to see that the worst case for a given scheme occurs when there is no black hole in the network or when the black hole is the last unvisited node outside of S, both cases yielding the same time. A scheme is called *fastest* for a given input if its time is the shortest possible for this input.

An unsafe node is called *explored* at a given step of a BHS-scheme if the remaining agents know if it is a black hole.

Any BHS-scheme must have the following property: after a finite number of steps, at least one agent stays alive and all unsafe nodes are explored (there is at most one black hole, so once the black hole has been found, all unsafe nodes are explored).

The *explored territory* at step t of a BHS-scheme is the set of explored nodes. At the beginning of a BHS-scheme the explored territory is empty. We say that a *meeting* occurs in node v at step t when the agents meet at node v and exchange information which *strictly increases* the explored territory. Node v is called a *meeting point*.

In any step of a BHS-scheme, an agent can traverse an edge or wait in a node. Also the two agents can meet. If at step t a meeting occurs, then the explored territory at step t is defined as the explored territory *after* the meeting. The sequence of steps of a BHS-scheme between two consecutive meetings is called a *phase*.

3. Preliminary results

Lemma 3.1. In a BHS-scheme, an edge towards an unsafe unexplored node cannot be traversed by both agents.

Proof:
Suppose that an edge f towards an unexplored node v has been traversed by an agent and while v remains unexplored (which means that the two agents have not yet met after the traversal of f), the other agent traverses f. If v is a black hole, then both agents vanish, which means that this is not a BHS-scheme. □

Hence in a BHS-scheme, an unsafe node can be explored only in the following way: an agent visits it and then a meeting is scheduled. Whether it occurs or not (in the latter case the agent vanished in the black hole) the node becomes explored.

Lemma 3.2. During a phase of a BHS-scheme an agent can visit at most *one* unsafe unexplored node.

Proof:
Suppose that an agent visits two unexplored nodes. If one of them is a black hole and hence the agent vanishes then there is no way for the other agent to locate the black hole without vanishing, which means that this is not a BHS-scheme. □

Therefore an unsafe node could be explored in the next phase only if it is adjacent to the explored territory. Recall that the explored territory increases only at scheduled meeting points.

Lemma 3.3. At the end of each phase, the explored territory is increased by one or two nodes.

Proof:
By the end of a phase the explored territory is increased by at least one node. By Lemma 3.2, an agent can visit at most one unsafe unexplored node during a phase, thus both agents can visit a total of at most two unexplored nodes during a phase. □

We define a *single exploration phase* (*SE-phase*) to be a phase in which exactly one node is explored. Similarly, we define a *double exploration phase* (*DE-phase*) to be a phase in which exactly two nodes are explored. In view of Lemma 3.3, every phase is either a SE-phase or a DE-phase.

A node p is called a *limit* of the explored territory at step t if it is either safe or explored, and it is adjacent to an unexplored node.

A way of exploring exactly one node in a phase is the following: one of the agents walks through safe or explored territory to its limit p, while the other agent walks through the safe or explored territory to p, visits an unexplored node and returns to p. If we assume that both agents are at a limit p of the explored territory at step t and v is an unexplored node adjacent to p, we define the following procedure:

probe(v**)**: one agent traverses edge (p, v) and returns to node p to meet the other agent that waits. If they do not meet at step $t + 2$ then the black hole has been found.

4. The NP-hardness of the black hole search problem

In this section we prove that the following optimization problem is NP-hard:

The BHS problem.

Input: Graph G with a subset S of safe nodes, starting node $s \in S$.
Find the fastest BHS scheme for the input (G, S, s).

In order to prove NP-hardness of the BHS problem, we present a reduction from the NP-complete Hamiltonian Cycle Problem (HC problem) to the decision version of the BHS problem, which we call dBHS.

The HC problem.

Input: Graph G
Question: Does G contain a Hamiltonian cycle?

The dBHS problem.

Input: graph G' with a subset S of safe nodes, starting node $s \in S$, positive integer X.
Question: Does there exist a BHS scheme for the input (G', S, s), with time at most X?

4.1. Construction

Let a graph G with n nodes and e edges be an instance of the HC problem. We construct a new graph G' as follows. Call the nodes of graph G old nodes. In each edge of G we add 2 new unsafe nodes adjacent to endpoints of this edge and $M = 4e + 5n - 1$ new safe nodes between them, as in Figure 1. Let s be any node of the old n nodes. All old nodes except s are considered unsafe. Hence the set S of unsafe nodes consists of all old nodes except s and all nodes adjacent to old nodes.

The instance of the dBHS problem is the graph G' with $n' = n + (M + 2)e$ nodes, the set S of $l = n + 2e - 1$ unsafe nodes, node s as a starting node, and the integer $X = M(n + 1) - 1$.

The construction of this instance from the graph G can be clearly done in polynomial time. It remains to prove that the answers "yes" to the HC problem for the input graph G and the answer "yes" to the dBHS problem for the constructed input are equivalent.

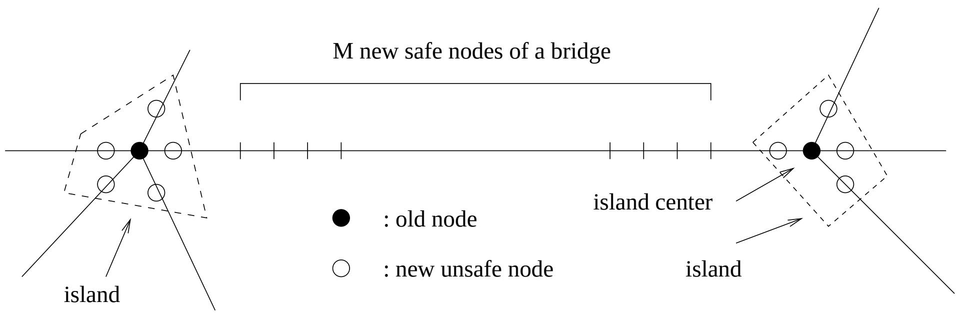

Figure 1. Construction of a dBHS problem instance

4.2. Analysis of the reduction

Fix a graph G with n nodes and e edges and the graph G' with n' nodes where l of them are unsafe, constructed as in the previous section. Each node v of graph G corresponds to a node v' of graph G' called *old* node. Any set of nodes of graph G' consisting of an old node v' together with all adjacent nodes is called an *island* (see Figure 1). Node v' is called the *center* of the island. Let v_1, v_2 be two nodes in G and v_1', v_2' be the corresponding old nodes in G'. The construction of G' implies that v_1, v_2 are adjacent in G if and only if there is a path in G' with endpoints v_1', v_2' which does not pass through any island except those with centers v_1', v_2'. We call this path in G' a *bridge*.

Let I be an island. After every meeting of the agents we define the following partition of the set of n islands into sets $\mathcal{PE}$ of *partially explored islands* and $\mathcal{U}$ of *unexplored islands*:

- $I \in \mathcal{PE}$, when I has at least one of its nodes explored
- $I \in \mathcal{U}$, otherwise

The above partition is well defined at any moment in which a meeting is scheduled. In the beginning we have $|\mathcal{PE}| = 1$, $|\mathcal{U}| = n - 1$. At a meeting at time t after a phase in a BHS-scheme one of the following can happen: a) sets $\mathcal{PE}$, $\mathcal{U}$ do not change, b) one island is moved from $\mathcal{U}$ to $\mathcal{PE}$, or c) two islands are moved from $\mathcal{U}$ to $\mathcal{PE}$. We call the above types of phases *0-phase*, *1-phase* and *2-phase* respectively. We say that the agents *discover* an island I at time t if and only if I is moved from $\mathcal{U}$ to $\mathcal{PE}$ at t. We call t the time of *discovery* of island I.

Upon completion of any BHS scheme, all islands must have been discovered (i.e. moved from set $\mathcal{U}$ to set $\mathcal{PE}$). Consider a BHS scheme σ and let k_1 be the number of 1-phases and k_2 the number of 2-phases. Since any island is moved from $\mathcal{U}$ to $\mathcal{PE}$ at the end of exactly one of the above phases in the worst case (i.e. when there is no black hole), we have $k_1 + 2k_2 = n - 1$.

Lemma 4.1. If graph G has a Hamiltonian cycle, then there exists a BHS scheme on graph G' starting at node s, with time at most $M(n + 1) - 1$.

Proof:
The Hamiltonian cycle in G corresponds to a cycle C in G' which passes exactly once through every

center of an island. The two agents agree on a direction of cycle C and they explore graph G', starting from s, as follows:

They explore by probing the nodes which are adjacent to the center of the island except the two nodes which are on C and they return to the center of the island. Then they explore by probing the unsafe node of the bridge on C in the chosen direction and they walk along the bridge, till they get to the last safe node of it. Subsequently, they explore the adjacent unsafe node v (in the next island) by probing, walk to it, explore the center of this island by probing and walk to it. They repeat the above procedure in every island on C until they reach again node s.

The two agents cross n bridges using the above procedure. They need $M + 9$ time units to cross every bridge in C, except the last one which leads to s for which they need $M + 7$ time units, since s is a safe node. Every edge in G corresponds to a bridge in G'. Since there are n bridges in cycle C, there are $e - n$ bridges not crossed by the two agents. For each not crossed bridge the agents explore 2 nodes (one from each endpoint of the bridge). Therefore they spend 4 time units in every bridge which is not in cycle C. The total time is at most $(n-1)(M+9) + M + 7 + 4(e-n) = M(n+1) - 1$. □

We say that a BHS scheme σ on G' is *reduced* to a BHS scheme σ' iff at any step t_i where a meeting occurs in σ the agents in σ' are at the same node as in σ and the explored territory is the same. Observe that the time of σ' is not longer than the time of σ.

We call a BHS scheme on G' *regular* if and only if it has the following property. Take any meeting point p_i in σ at time t_i and any island I whose center is unexplored at t_i and the phase ending at t_i does not explore any node of I. Then p_i must be at distance at least $M + 2$ from the center of I.

Lemma 4.2. Every BHS scheme on G' can be reduced to a regular one.

Proof:
Take the first meeting m which occurs in a BHS scheme σ at step t_i at a node p_i, with the following property: there is an island I whose center is unexplored at t_i and the phase ending at t_i does not explore any node of I. Suppose also that the distance between node p_i and the center of I is less than $M + 2$.

Consider the node p'_i which is at distance $M + 2$ from the center of island I on the bridge including node p_i. We will transform the BHS scheme σ to a BHS scheme σ' where the two agents meet at p'_i at step $t'_i < t_i$ and then they walk together till they reach node p_i at step t_i.

Since m is the first meeting with the above property in σ, the meeting m' before m in σ could not have taken place in a node between p'_i and the center of I. Therefore there are steps between meetings m' and m at which the two agents were at node p'_i (not neccesarily together) in σ. Consider the last time t_1 before step t_i when one of the agents, say R_1, was at node p'_i.

If at time t_1 the other agent R_2 is not in a node between nodes p'_i and the center of I then, in scheme σ' the agent R_1 waits since step t_1 until it meets R_2 and then they walk to node p_i.

If at time t_1 the other agent R_2 is in a node between nodes p'_i and the center of I then consider the last time t_2 before step t_1 when agent R_2 was at node p'_i. In scheme σ' the agent R_2 waits since step t_2 until step t_1, meets the other agent and then they walk to node p_i. □

Lemma 4.3. Any 1-phase in a regular BHS scheme on G' requires more than M time units.

Proof:
In view of regularity of the scheme, in a 1-phase there is at least one agent which has to cover a distance of at least $M + 1$ to reach a node of an island $I \in \mathcal{U}$. □

Lemma 4.4. Any 2-phase in a regular BHS scheme on G' requires more than $2M$ time units.

Proof:
Consider a 2-phase after which two islands I, I' are discovered. Suppose that nodes $v \in I$ and $u \in I'$ are explored in this 2-phase. In view of regularity of the scheme, the meeting point p before that 2-phase is at a distance at least $M+2$ from any of the centers of I, I'. Therefore p is at a distance at least $M+1$ from nodes v, u. Suppose that p is at a distance x from node v and y from node u. Since the centers of islands I, I' are still unexplored, the bridge that each agent uses to go to the new meeting point p' is the same as that used to reach the island. Since $x + y > 2M$, the agents need at least $2M$ time units to complete the 2-phase. □

In view of Lemmas 4.3 and 4.4, the time needed in any regular BHS scheme for discovering all islands is at least $Mk_1 + 2Mk_2 \geq (n-1)M$.

Lemma 4.5. Consider a regular BHS scheme σ on G'. Suppose that σ contains a 2-phase ϕ in which islands I, I' are discovered and at the end of ϕ the two agents meet at a node p. Then the distance between node p and the center of at least one of the islands I, I' is at least $M+2$.

Proof:
Since the meeting point before ϕ is at a distance at least $M+2$ from any of the centers of I, I' (by regularity of the scheme) and each agent has to cross the same bridge to go to p as that used to reach the island, the distance between node p and the center of at least one of the islands I, I' is at least $M+2$. □

Let $I_1, I_2, ..., I_n$ be the enumeration of islands in G' in the order of discovery by a BHS-scheme σ. Let $v_1, v_2, ..., v_n$ be the sequence of nodes in G corresponding to the centers of these islands. Node v_1 corresponds to node s of G'.

Lemma 4.6. Consider a regular BHS-scheme σ in which all islands are discovered during 1-phases. Let t be the time of discovery of I_{n-1}. If $t < (n-1)M$ then the sequence $v_1, v_2, ..., v_{n-1}$ is a path in G.

Proof:
After every meeting just before a 1-phase we may always suppose by symmetry that the same agent R_1 goes to the completely unexplored island (we may need just to interchange the names of the agents in some meeting points in σ). Each 1-phase takes at least M time units. There are $n-2$ 1-phases.

Suppose that the sequence $v_1, v_2, ..., v_{n-1}$ is not a path in G. Then there exists $i < n-1$ such that v_i and v_{i+1} are not adjacent in G. Therefore agent R_1 moves from island I_i to island I_{i+1} via another island. By regularity of the scheme, the meeting point after discovering I_{i-1} is at distance at least M from the center of I_i. Hence discovering I_i and I_{i+1} takes total time at least $3M$, which implies that the total time spent by agent R_1 on the discovery of islands $I_2, ..., I_{n-1}$ is at least $(n-1)M$. □

Lemma 4.7. If G has no Hamiltonian cycle, then any regular BHS scheme σ on G', starting at a center of an island, requires time at least $(n+1)M$.

Proof:
First suppose there is a 2-phase in scheme σ. Take the last 2-phase ϕ in σ and let I, I' be the islands

discovered during ϕ. Let p be the meeting point at the end of ϕ. In view of Lemma 4.5 the distance between p and the center of at least one of the islands I, I' is at least $M + 2$. Let I be that island. Consider the first phase ψ after ϕ in which a node v of I is explored. Phase ψ cannot be a 2-phase (since I was already discovered before ψ). Let p' be the meeting point at step t_ψ just before ψ. If $p' = p$ then the distance between p' and the center of I is at least $M + 2$. If $p' \neq p$ then ψ does not immediately follow ϕ and hence, by regularity of σ the distance between p' and the center of I is at least $M + 2$.

- Suppose that by the end of ψ all islands are discovered.

  If ψ is a 0-phase, then the two agents have spent time at least $(n - 1)M$ until step t_ψ. An agent needs $2M$ additional time units to go from p' to the center of I and return to s. Hence the total time is at least $(n + 1)M$.

  If ψ is a 1-phase, then it means that together with node v of island I, a node u of another island $J \in \mathcal{U}$ is explored. The two agents have spent time at least $(n - 2)M$ until step t_ψ. Since at step t_ψ the center of the island I is unexplored, any path which connects islands I, J and can be used by the agents has length at least $2M$. Hence phase ψ ends at time at least nM. After phase ψ the center of J is still unexplored. Therefore an agent needs at least M time units to go there and return to s. Therefore the total time is at least $(n + 1)M$.

- Suppose that after phase ψ there are islands still undiscovered.

  If ψ is a 0-phase then it lasted at least M time units.

  If ψ is a 1-phase then it lasted at least $2M$ time units for the same reason as before.

  Since ϕ was the last 2-phase of σ, the last discovery phase of σ must be a 1-phase. Call it χ. The two agents have spent time at least $(n - 1)M$ before the start of χ. By regularity of σ an agent needs $2M$ additional time units to go to the center of the last discovered island and return to s. Hence the total time is at least $(n + 1)M$.

If there is no 2-phase in the BHS scheme σ then consider the following cases:

- v_n is adjacent to v_{n-1} and to v_1 in G.
  Since there is no Hamiltonian cycle in G, the sequence $v_1, v_2, ..., v_{n-1}$ cannot be a path in G. In view of Lemma 4.6 the time of discovery of I_{n-1} is at least $(n-1)M$. By regularity of the scheme, the meeting point after discovering I_{n-1} is at distance at least M from the center of I_n. Hence discovering I_n and returning to s takes time at least $2M$ which implies that the total time of the scheme is at least $(n + 1)M$.

- v_n is not adjacent to v_{n-1}.
  By regularity of the scheme, I_{n-2} is discovered in time at least $(n - 3)M$ and the meeting point after discovering I_{n-2} is at a distance at least M from the center of I_{n-1}. Since v_n is not adjacent to v_{n-1}, discovering I_{n-1}, I_n and returning to s takes a total time of at least $4M$ which implies that the total time of the scheme is at least $(n + 1)M$.

- v_n is not adjacent to v_1 in G.
  By regularity of the scheme, I_{n-1} is discovered in time at least $(n - 2)M$ and the meeting point after discovering I_{n-1} is at a distance at least M from the center of I_n. Since v_n is not adjacent to v_1, discovering I_n and returning to s takes a total time of at least $3M$ which implies that the total time of the scheme is at least $(n + 1)M$.

In all cases we showed that the time of the scheme is at least $(n+1)M$. This concludes the proof. □

We can now prove the main result of this section.

Theorem 4.1. The BHS problem is NP-hard.

Proof:
It is enough to show that the answers "yes" to the HC problem for the input graph G and the answer "yes" to the dBHS problem for the constructed input are equivalent. By Lemma 4.1, if G has a Hamiltonian cycle then there exists a BHS scheme on G', starting at s, with time at most $(n+1)M-1$. Conversely, suppose that there is a BHS scheme on G', starting at s, with time at most $(n+1)M-1$. By Lemma 4.2 it can be reduced to a regular BHS scheme, whose time is also at most $(n+1)M-1$. By Lemma 4.7, graph G has a Hamiltonian cycle. □

5. An approximation algorithm for the BHS problem

In this section we give an approximation algorithm for the BHS problem. The algorithm is based on the construction of a *Steiner Tree* of the input graph G, where the unsafe nodes of G along with the starting node s are the required nodes. Recall that a Steiner Tree for a graph $G=(V,E)$ with the set $R \subseteq V$ of required nodes is any subtree of G containing R.

Algorithm Tree
  construct a minimum Steiner Tree T containing all the unsafe nodes and node s;
  explore((T,s))

Let G be a graph with a set S of safe nodes and a starting point s. We construct a Steiner Tree T where the unsafe nodes of G along with node s play the role of required nodes for the Steiner Tree. We can construct such a Steiner Tree in polynomial time with approximation ratio α, where $\alpha = 1+\frac{\ln 3}{2} < 1.55$ ([9], [11]). More specifically, if x is the number of unsafe nodes in G plus one for node s, and y is the number of safe nodes in T (excluding node s), while y^* is a minimum number of safe nodes (excluding node s) needed for the optimal Steiner Tree, then $(x+y) \leq 1.55(x+y^*)$. We then use the procedure explore(T,s).

Procedure explore((T,v))
  for every unexplored node z adjacent to v **do**
    probe(z);
  end for
  if every node is explored **then**
    repeat walk(s) **until** both agents are at s
  else
    $next$:= relocate(v);
    explore($(T,next)$)
  end if

The high-level description of the procedure explore is the following. Let v be the meeting point of the two agents after a phase (initially $v = s$); the unexplored children of v are explored by calling procedure probe; this is repeated for any child of v. The precise formulation of the algorithm is given below.

Function relocate(v) takes as input the current node v where both agents reside and returns the new location of the two agents. If there is an unexplored node adjacent to a child of v then the agents go to that child. Otherwise the two agents go to the parent of v.

Function relocate(v)
  case 1.1: $\exists$ an unexplored node adjacent to $w \in children(v)$
    walk(w);
    relocate := w
  case 1.2: every node adjacent to any child of v is explored
    let t be the parent of v;
    walk(t);
    relocate := t

The time-complexity of Algorithm Tree is polynomial in the size of G and is dominated by the time of constructing the Steiner Tree. Procedure explore is in fact a depth first search type algorithm with the only difference that any unsafe node is visited using a cautious way. The time spent on traversals of any edge (u, v) (v is a child of u) of the tree T is at most 4 units: the worst case is when edge (u, v) leads to an unexplored node v which is not a leaf in T, therefore the agents spend 2 time units for probing v, 1 time unit to walk to v and another time unit to return to node u - after the exploration of the descedants of v. The total time needed by the BHS-scheme produced by Algorithm Tree is less than $4(x + y)$.

Lemma 5.1. Any BHS scheme for the graph G requires at least $\frac{4}{3}(x + y^*)$ traversals of edges.

Proof:
Take a BHS scheme σ. Let A_i^*, for $1 \leq i$, denote the set of edges in σ which are traversed exactly i times. Let $a_i^* = |A_i^*|$, $A^* = \bigcup_i A_i^*$ and $a^* = |A^*|$.

Let ϕ be a phase in σ starting at meeting point m_ϕ and ending at meeting point m'_ϕ. Let p_ϕ, p'_ϕ be the unsafe nodes explored in ϕ by agents R_1, R_2 respectively (possibly $p_\phi = p'_\phi$).

Let $B_\phi \subseteq A_1^*$ be the set of edges traversed by R_1 since the start of ϕ until R_1 reaches node p_ϕ at time t. Let $C_\phi \subseteq A_1^*$ be the set of edges traversed by R_1 since t to the end of ϕ. Let $B'_\phi \subseteq A_1^*$ be the set of edges traversed by R_2 since the start of ϕ until R_2 reaches node p'_ϕ at time t'. Let $C'_\phi \subseteq A_1^*$ be the set of edges traversed by R_2 since t' to the end of ϕ (see Figure 2).

We have:

$$\max\{|B_\phi|, |C_\phi|, |B'_\phi|, |C'_\phi|\} \geq \frac{|B_\phi| + |C_\phi| + |B'_\phi| + |C'_\phi|}{4} \qquad (1)$$

Notice that if ϕ is a SE-phase then at least one of $B_\phi, C_\phi, B'_\phi, C'_\phi$ is empty and the relation (1) still holds.

We will prove that we can remove any one of the sets $B_\phi, C_\phi, B'_\phi, C'_\phi$, in every phase ϕ in σ and the resulting graph will still contain a Steiner Tree (with the set of required nodes consisting of the unsafe ones and of the node s).

Let $< \phi_1, \phi_2, ..., \phi_k >$ be the enumeration of phases in the order that they appear in σ. In each phase ϕ_i we calculate $B_{\phi_i}, C_{\phi_i}, B'_{\phi_i}, C'_{\phi_i}$ and we remove the set with the maximum number of edges.

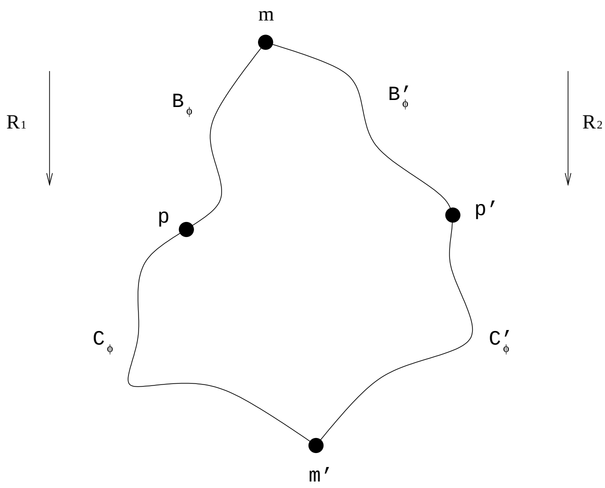

Figure 2. The sets $B_\phi, C_\phi, B'_\phi, C'_\phi$ consist of edges traversed once in σ, namely only in phase ϕ

Let G_ϕ be the graph resulting from G after removing sets of edges in all phases prior to ϕ.

Consider a phase ϕ. Let P be the path followed by an agent which departs from m_ϕ, explores p_ϕ and reaches m'_ϕ. Let P' be the path followed by the other agent which departs from m_ϕ, explores p'_ϕ and reaches m'_ϕ. All edges in both these paths belong to G_ϕ because they are traversed either only in phase ϕ or traversed at least twice in σ. This implies that nodes $m_\phi, m'_\phi, p_\phi, p'_\phi$ are in the same component of G_ϕ. After removing one of the sets $B_\phi, C_\phi, B'_\phi, C'_\phi$ in phase ϕ, nodes $m_\phi, m'_\phi, p_\phi, p'_\phi$ are still connected by one of the previous paths, say P, and the remaining part of P'. Moreover $m_\phi, m'_\phi, p_\phi, p'_\phi$ cannot be disconnected in any later phase, since P and the remaining part of P' contain edges traversed only in ϕ or traversed at least twice in σ.

Since $s = m_{\phi_1}$ and $m'_{\phi_i} = m_{\phi_{i+1}}$, for $1 \leq i \leq k-1$, the resulting graph after all the removals of sets of edges done as above still contains a Steiner Tree (with the set of required nodes consisting of the unsafe ones and of the node s).

Let $b^*_{\phi_i}$ be the number of edges removed in phase ϕ_i and let $a^*_1(\phi_i)$ be the number of all edges traversed only in ϕ_i. We have $b^*_{\phi_i} \geq \frac{a^*_1(\phi_i)}{4}$. Since after all removals the resulting graph still contains a Steiner Tree, we have $a^* - b^*_{\phi_1} - b^*_{\phi_2} - ... - b^*_{\phi_k} \geq x + y^*$. Therefore $a^* - \frac{a^*_1}{4} \geq x + y^*$. The total number of traversals in σ is bounded as follows:

$$\sum_i i \cdot a^*_i \geq 2 \cdot (a^* - a^*_1) + a^*_1.$$

Hence we get

$$\frac{3}{4} \cdot \sum_i i \cdot a^*_i \geq \frac{3}{2} \cdot (a^* - a^*_1) + \frac{3}{4} \cdot a^*_1 \geq \frac{3}{4} \cdot a^*_1 + a^* - a^*_1 \geq x + y^*.$$

Thus $\sum_i i \cdot a^*_i \geq \frac{4}{3}(x + y^*)$. □

Theorem 5.1. Algorithm Tree is an approximation algorithm for the BHS problem with ratio 9.3.

Proof:
The time needed by a fastest BHS-scheme for the graph G is $OPT \geq \frac{\sum_i i \cdot a_i^*}{2}$, since there are two agents. Hence the time needed by algorithm Tree is at most:

$$4(x + y) \leq 4 \cdot 1.55 \cdot (x + y^*) \leq 3 \cdot 1.55 \cdot \sum_i i \cdot a_i^* \leq 9.3 \cdot OPT.$$

□

6. Conclusion

We showed that the black hole search problem is NP-hard and we gave a polynomial approximation algorithm to solve it. A natural open problem is to decrease the approxiamtion ratio: 9.3 is relatively high. Another interesting issue is to increase the potential number of black holes. In this case two agents are not enough: the number of agents must be larger than the maximum number of black holes. Also connectivity requirements on the graph have to be imposed, in order to make locating all black holes feasible. A natural generalization of our approach would be to find good approximation algorithms for the black hole search problem with at most k holes, using $l > k$ agents (whenever it is feasible on a given input).

References

[1] J. Czyzowicz, D. Kowalski, E. Markou, A. Pelc, Searching for a black hole in tree networks, *Proc. 8th International Conference on Principles of Distributed Systems (OPODIS'2004)*, LNCS 3544, 67–80.

[2] S. Dobrev, P. Flocchini, R. Kralovic, G. Prencipe, P. Ruzicka, N. Santoro, Black hole search by mobile agents in hypercubes and related networks, *Proc. of Symposium on Principles of Distributed Systems (OPODIS 2002)*, 171-182.

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

[4] S. Dobrev, P. Flocchini, G. Prencipe, N. Santoro, Searching for a black hole in arbitrary networks: Optimal Mobile Agents Protocols, *Proc. 21st ACM Symposium on Principles of Distributed Computing (PODC 2002)*, 153-161.

[5] S. Dobrev, P. Flocchini, G. Prencipe, N. Santoro, Multiple agents rendezvous on a ring in spite of a black hole, *Proc. Symposium on Principles of Distributed Systems (OPODIS 2003)*, 34–46.

[6] M. Halldórsson, S. Ueno, H. Nakao, Y. Kajitani, Approximating Steiner trees in graphs with restricted weights. *Networks* 31 (1998) 283–292.

[7] F. Hohl, Time limited black box security: Protecting mobile agents from malicious hosts, *Proc. Conf. on Mobile Agent Security* (1998), LNCS 1419, 92-113.

[8] F. Hohl, A framework to protect mobile agents by using reference states, *Proc. 20th Int. Conf. on Distributed Computing Systems (ICDCS 2000)*, 410-417.

[9] Minimum Steiner Tree. In `http://www.nada.kth.se/∼viggo/wwwcompendium/node78.html`

[10] S. Ng, K. Cheung, Protecting mobile agents against malicious hosts by intention of spreading, *Proc. Int. Conf. on Parallel and Distributed Processing and Applications (PDPTA'99)*, 725-729.

[11] G. Robins, A. Zelikovsky, Improved steiner tree approximation in graphs. *Proc. 10th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'2000)*, 770–779.

[12] T. Sander, C.F. Tschudin, Protecting mobile agents against malicious hosts, *Proc. Conf. on Mobile Agent Security* (1998), LNCS 1419, 44-60.

[13] K. Schelderup, J. Ines, Mobile agent security – issues and directions, *Proc. 6th Int. Conf. on Intelligence and Services in Networks*, LNCS 1597 (1999), 155-167.

[14] J. Vitek, G. Castagna, Mobile computations and hostile hosts, in: *Mobile Objects*, D. Tsichritzis, Ed., University of Geneva, 1999, 241-261.