Distributed Black Virus Decontamination and Rooted Acyclic Orientations

Jie Cai
School of Computer Science
Carleton University
Ottawa, ON, Canada K1S 5B6
Email: jie.cai@carleton.ca

Paola Flocchini
School of EECS
University of Ottawa
Ottawa, ON, Canada, K1N 6N5
Email: paola.flocchini@uottawa.ca

Nicola Santoro
School of Computer Science
Carleton University
Ottawa, ON, Canada K1S 5B6
Email: santoro@scs.carleton.ca

***Abstract*—In a network supporting mobile agents, a particular threat is that posed by the presence of a *black virus* (BV), a harmful entity capable of destroying any agent arriving at the site where it resides, and of then moving to all the neighbouring sites. A moving BV can only be destroyed if it arrives at a site where an anti-viral agent is located. The objective for a team of mobile anti-viral system agents, called cleaners, is to locate and permanently eliminate the BV, whose initial location is unknown. The goal is to perform this task with the minimum number of network infections and agent casualties. The problem of optimal *black virus decontamination* (BVD) has been investigated for special classes of highly regular network topologies; a (centralized) solution exists for networks of known arbitrary topology.**

In this paper, we consider the BVD problem in networks of arbitrary and unknown topology; we prove that it can be solved optimally in a purely decentralized way by asynchronous agents provided with 2-hop visibility. In fact, we prove that our proposed protocols always correctly decontaminate the network with the minimum number of system agents' casualties and network infections. Furthermore, we show that the total number of system agents is also optimal.

Finally, we prove an interesting correspondence between the BVD problem and the problem of computing a rooted acyclic orientation of a given graph with minimum outdegrees. As a consequence, our protocols provide a distributed optimal solution to this graph optimization problem.

***Index Terms*—Black Virus, Graph Exploration and Decontamination, Mobile Agent.**

I. Introduction

In networked systems supporting mobile agents, a malicious agent can cause computer nodes to malfunction by contaminating or infecting them; additionally, a malicious host can harm incoming agents for various purposes (e.g., see [16]).

The theoretical focus has been on a particularly harmful host, called *black hole*: a network node infected by a process which destroys any incoming agent without leaving any detectable trace of the destruction. The primary concern has been on locating its position, for isolation and later disarming; this problem, called *black hole search* (BHS), has been extensively studied (e.g., [6], [7], [8], [11], [12], [10], [19], [23]). In regards to harmful agents, the theoretical work has focused on the problem called *intruder capture* (IC) (also known as *graph decontamination* and *connected graph search*): an extraneous mobile agent, the *intruder*, moves through the network infecting the visited sites; the task is to decontaminate the network using a team of system agents avoiding recontamination. Also this problem has been extensively studied (e.g.,[2], [3], [13], [14], [15], [18], [21], [22], [25], [26], [24], [27]).

Let us point out that a black hole is a presence which is harmful to agents but it is stationary, that is, it does not propagate in the network and so it is not harmful to other sites; on the other side, the intruder is mobile and harmful to the network sites, but does not cause any harm to system agents.

Recently some investigations have started to focus on an harmful entitiy, called *black virus* (BV), that combines some of the destructive power of black holes with some of the mobility of intruder [4]. Like a black hole, a BV destroys any agent arriving at the network site where it resides. When this occurs, unlike a black hole which is stationary, the BV moves spreading clones to all neighbouring sites; A black virus (clone) is destroyed only if it moves to a node where an anti-viral system agent is present; in this case, the agent is able to deactivate and permanently remove that (instance of the) BV.

The task of permanently removing any presence of the BV from the network using a team of system agents, called *black virus decontamination* (BVD), is dangerous for the system agents performing it, since any agent arriving at a node where an instance of the BV resides will be destroyed; it is obviously dangerous for all the nodes where the BV will spread to. The search is for solution protocols, that is algorithmic strategies that would enable the team of system agents, once injected in the system at a network site, to move in the network so that within finite time any presence of the BV is removed from the network. The goal of a solution protocol is to minimize the *spread* of the BV i.e., the number of node infections by the BV's; note that, since each instance of the BV has to be eventually removed and each removal requires the destruction of at least one agent, the spread also measures the number of agent *casualties*. The other important cost measure is the *size* of the team, i.e. the number of agents employed by the solution.

The clones of a BV have the same harmful capabilities of the original BV; two versions of the BVD problem have been examined, depending on whether or not are *sterile*, that is, unable to produce clones. In this paper we consider the model of sterile clones.

This problem has been studied in networks with special topologies, namely *grids*, *tori*, and *hypercubes* [4], for which optimal solutions have been designed. In the case of networks arbitrary topology, the problem has been investigated and solved assuming full topological knowledge of the graph [5]; in other words, the solution is centralized.

In this paper, we consider asynchronous networks of arbitrary unknown topology and we show that the BVD problem can be solved optimally in a purely decentralized way by asynchronous agents provided with 2-hop visibility. In fact, we prove that our proposed protocols always correctly decontaminate the network with the minimum number of system agents' casualties and network infections. Furthermore, we show that the total number of system agents is also optimal.

Finally, we show an interesting correspondence between the BVD problem and a special instance of the graph optimization problem of determining an orientation with minimum maximum outdegree (e.g., [1], [28]). In fact, we prove that any optimal solution to the BVD problem can be used to compute a rooted acyclic orientation with minimum outdegree; as a consequence, our protocols provide a distributed optimal solution to this graph optimization problem.

II. Model

The environment is a network supporting mobile agents; its topology is modelled as a simple undirected connected graph $G = (V, E)$ with $n = |V|$ nodes (or sites) and $m = |E|$ edges (or links). We denote by $E(v) \subseteq E$ the set of edges incident on $v \in$ V, by $N(v) \subset V$ the set of its neighbours, by $d(v) = |E(v)|$ its degree, and by $\triangle(G)$ (or simply $\triangle$) the maximum degree in G. Every node v has a distinct identity $id(v)$. The links incident to a node are labelled with distinct port numbers. The labelling mechanism could be totally arbitrary among different nodes; without loss of generality, we assume the link labels for node v form the set $l_v = \{1, 2, 3, ..., d(v)\}$.

A team $\mathcal{A} = \{A_1, \ldots, A_k\}$ of mobile system agents, called *cleaners*, provided with decontamination capabilities, is injected in the network at a node h called the *home base*. Each agent $A \in \mathcal{A}$ is a computational entity with its own local memory and a unique identifier $id(A)$ from some totally ordered set; it can move from node to neighbouring node. The agents can have different roles (i.e., states), but they all operate according to the same protocol. More than one agent can be at the same node at the same time. Communication among the agents is face-to-face: two (or more) agents can communicate only when at the same node; there are no a priori restrictions on the amount of exchanged information. The agents have no a priori knowledge of the network G nor of its size. The agents are provided with *limited visibility*: when at a node v an agent see the 2-neighbourhood $N^2(v)$ of v, including the node identities and the edge labels.

In the network there is a node infected by a *black virus* (BV), a process endowed with reactive capabilities for destruction and spreading. It is harmful not only to the node where it resides but also to any agent arriving at that node. More precisely, a BV destroys any agent arriving at the network site where it resides.

A BV can be deactivated only by a cleaner; even then, it can harm the neighbouring networks sites. More precisely, when deactivated the BV can release *clones* that spread to all the neighbouring nodes. The clones of a BV have the same harmful capabilities of the original BV but are *sterile*, that is, unable to produce clones.

If a BV clone arrives at a node with no cleaner, it infects the node and becomes resident there; if the node is occupied by a cleaner, the BV clone is destroyed before it can cause any harm. We assume that multiple copies of the BV clones at the same node are merged into one; i.e., at any time at each node there is at most one BV clone.

With respect to time and synchronization, there are no global clocks, and the duration of any activity (e.g., processing, communication, moving) by the agents, the BV, and its clones is finite but unpredictable; in other words, the system is *asynchronous* and any needed synchronization must be achieved by the agents' protocol.

The Black Virus Decontamination (BVD) problem is to permanently remove the BV and its clones from the network using the team of cleaners.

A protocol defining the actions of the cleaners solves BVD in G if, within finite time, at least one cleaner survives and the network is free of BVs, regardless of the location of the home base and of the black virus, and regardless of the duration of the actions of the agents, of the BV, and of its clones. A protocol solves BVD if it solves it in every network G. Let $\mathcal{P}$ denote the set of all solution protocols of the BVD problem. A solution protocol is *monotone* if no recontamination occurs in any of its execution; that is, once a node is visited by a cleaner, it will not be (re)contaminated by a BV clone.

The goal of a solution protocol $P \in \mathcal{P}$ is to decontaminate the network minimizing the *spread* of the black virus, i.e., the number of node infected by the BV and its clones. Note that, since each instance of the BV (original or clone) has to be eventually removed and since each removal requires the destruction of at least one cleaner, the spread also measures the number of cleaner *casualties*.

Given a solution protocol P and a network $G = (V, E)$, let $spread(P, G)$ denote the maximum number of casualties incurred when executing P in G in the worst case (i.e., over all possible initial locations of the BV and of the home base, and all possible execution delays); then

$$spread(G) = Min_{P \in \mathcal{P}}\{spread(P, G)\}$$

denotes the minimum amount of casualties possible to decontaminate G in the worst case. A solution protocol P is *worst-case optimal* if $spread(P, G) = spread(G)$ for all G.

A finer cost measure is the maximum number of casualties $spread(P, G, v)$ incurred over all possible executions and locations of BV starting from homebase $v \in V$; thus

$$spread(G, v) = Min_{P \in \mathcal{P}}\{spread(P, G, v)\}$$

denotes the minimum amount of casualties possible to decontaminate G when starting from v. A solution protocol P is *everywhere optimal* if for all $G = (V, E) \in \mathcal{G}$, and all $v \in V$, $spread(P, G, v) = spread(G, v)$; clearly everywhere optimality implies worst-case optimality.

For a (worst-case or everywhere) optimal protocol P, an important cost measure is the size of the team of cleaners $size(P, G)$; we denote by $size(G) = Min\{size(P, G)\}$ the smallest team-size over all optimal solution protocols P.

III. Basic Properties and Bounds

Let us first state a few basic observations on the properties of the universe under investigation.

A. Monotonicity and Sequentiality

To decontaminate a network, the team of agents must explore the network until the BV is found. Indeed, the execution of any solution protocol P in a network G where no BV is present, a *BV-free execution*, generates an exploration of all the nodes of G.

Once the BV is found at a node u, clones of the BV will move to every neighbour of u. This means that, any previously explored neighbour of u will be contaminated unless an agent is there at that time. A solution protocol is said to be *monotone* if, in every execution it avoids recontamination of already visited nodes. The interest in monotone protocols is because monotonicity is necessary for optimality [4]:

Lemma 3.1: [4] Every (worst-case) optimal protocol is monotone.

As a consequence, in our search for spread-optimal solutions, we must restrict ourselves to solution protocols that are monotone.

Consider now how the exploration of the network is performed when executing a solution protocol. We say that a solution protocol P is *sequential* if it prescribes the nodes to be explored one a time, and the exploration of a new node to be started only after the exploration of the previous node has been completed. A very important fact is that, due to the asynchrony of the system, sequentiality is not a handicap for optimality.

Lemma 3.2: There exist everywhere optimal sequential protocols.

Proof: Let $\mathcal{O}$ be the set of everywhere optimal protocols, and let $P \in \mathcal{O}$. If P is sequential, the lemma holds. So let P be not sequential; that is, in some BV-free execution it require some agents to move concurrently to more that one unexplored node. Since the system is asynchronous, delays can force these nodes to be reached by the agents in any order, including a strictly sequential one, without altering the everywhere optimality of P. Consider now the protocol P_1 identical to P except that those agents are required to visit those nodes sequentially instead of concurrently. Clearly also $P_1 \in \mathcal{O}$. By replacing in $P_i \in \mathcal{O}$ $(i \geq 1)$ a concurrent visit of unexplored nodes by a sequential visit of the same nodes, we can construct a protocol $P_{i+1} \in \mathcal{O}$ with one less concurrent visit. This sequence of everywhere optimal protocols is finite; by construction, the last protocol $P_s \in \mathcal{O}$ in the sequence is sequential, proving the Lemma. ■

Thus, by Lemmas 3.1 and 3.2, in our search for spread-optimal solutions, we can restrict ourselves to protocols that are sequential (i.e., in which the nodes are visited one at a time) and monotone (i.e., all the visited neighbours of the node being explored are be protected with a cleaner).

B. Residual Degrees and Feasible Permutations

Let $\mathcal{P}$ be the set of all monotone sequential solution protocol. Consider a protocol $P \in \mathcal{P}$, and a graph $G = (V, E) \in \mathcal{G}$. Since P is sequential, the nodes of G are visited for the first time one at a time, starting from the home base. Let $P[G, h] \equiv [x_0, x_1, x_2, ..., x_{n-1}]$ be the resulting ordered sequence in a BV-free execution of P in G starting from $h = x_0$.

We call *residual degree* of x_i in $P[G, h]$ the number of neighbours of x_i following it in the sequence $P[G, h]$; i.e., $\rho(x_i) = |\{x_j \in V(x_i) : n > j > i\}|$. We say that $\rho(P[G, h]) = \max_{n > i > 0}\{\rho(x_i)\}$ is the residual degree of the entire sequence $\pi_P[h]$.

Based on the notion of residual degree, a lower estimate on the number $spread(P, G, h)$ of casualties created, in the worst case, by the cleaners executing P in $G = (V, E)$ starting from $h \in V$ can be easily derived.

Lemma 3.3: $spread(P, G, h) \geq \rho(P[G, h]) + 1$.

Proof: In a monotone protocol, when visiting a new node, the still unexplored nodes are unprotected. Thus, if the BV is at x_i, all the neighbours of x_i still unexplored, i.e. the set $\{x_j \in V(x_i) : n > j > i\}$, will become contaminated. Since one casualty is required to decontaminate each of them, the total number of casualties, including the one occurred at x_i, is precisely $\rho(x_i) + 1$. ■

By Lemma 3.3, it follows that to minimize spread, we need to find a protocol that has minimum residual degree in all graphs and for all choices of the home base. That is, our quest is for a protocol $P \in \mathcal{P}$ such that $\forall G = (V, E), \forall v \in V, \rho(P[G, h]) = spread(G, h)$.

To aid in our quest, we recall that the sequence $P[G, h]$ is a permutation of the n network nodes. Let us call a permutation $[x_0, x_1, x_2, ..., x_{n-1}]$ of the nodes of G *feasible* for x_0 if for all $1 \leq i \leq n - 1$ there exists a path in G from x_0 to x_i composed only of nodes whose index is smaller than i; let $\Pi(G, v)$ denote the set of all permutations feasible for node v, and $\Pi(G) = \cup_{v \in V} \Pi(G, v)$ the set of all feasible permutations of the nodes of G. Then, for a given graph G, each sequential solution protocol $P \in \mathcal{P}$ uniquely defines a set $F_P(G)$ of n feasible permutations, one for every possible choice of the home base. Conversely, any set $F \subseteq \Pi(G)$ of n feasible permutations, each for a different node of G, corresponds to the BV-free execution sequences of some sequential solution protocol $P_F \in \mathcal{P}$ in G. That is, if $\alpha = [x_0, x_1, x_2, ..., x_{n-1}] \in F \subseteq \Pi(G)$ then $\alpha = P[G, x_0]$ for some $P \in \mathcal{P}$.

In other words, to determine a spread-optimal decontamination strategy for G, it suffices to determine for each h a feasible permutation α for h such that $\rho(\alpha) = \rho(G, h)$. We remind the reader that the graph G is however not known to the cleaners.

Before proceeding let us establish an obvious but important property of residual degrees in feasible permutations.

Lemma 3.4: Given a permutation $\alpha = [z_0, z_1, ..., z_{n-1}]$ feasible for z_0, let also $\alpha_{i,j} = [z_0, z_1, ..., z_{i-1}, z_j, z_i, z_{i+1}, ..., z_{j-1}, z_{j+1}, ..., z_{n-1}]$ be feasible for z_0, where $0 < i < j \leq n-1$. Then $\rho(z_l, \alpha_{i,j}) \leq \rho(z_l, \alpha)$, for all $l \neq j$.

Proof: Permutation $\alpha_{i,j}$ is the obtained from α by moving z_j immediately before z_i and leaving the rest unchanged. Since $\alpha_{i,j}$ is feasible for z_0, for each z_p ($0 \leq p \leq n-1$ there is a path from z_0 to z_p composed only of predecessors of z_p in $\alpha_{i,j}$. This means that the the number of neighbours of z_p following it in $\alpha_{i,j}$ (i.e., its residual degree in $\alpha_{i,j}$) is the same in both α and $\alpha_{i,j}$ for $0 \leq p \leq i-1$ and for $j+1 \leq p \leq n-1$. Furthermore, since z_j appears before z_i in $\alpha_{i,j}$, for $i \leq p \leq j-1$ the residual degree of z_p in $\alpha_{i,j}$ is the is either one less than or equal to its residual degree in α, depending on whether or not $(z_i, z_p) \in E$ (i.e., they are neighbours). ■

IV. Decontamination with local knowledge

We propose and analyze decontamination protocols that, like in [5], consist of two phases, *shadowed exploration* and *elimination*. In the shadowed exploration phase, starting from the homebase h, the agents explore sequentially the nodes of the graph, using agents (the *shadows*) to protect the already explored neighbours of the node to be visited, and thus ensure monotonicity of the solution. Once the BV node has been found, causing BV clones to spread to the unprotected neighbouring nodes, the elimination phase start by finally and safely remove all the clones.

In our case, unlike [5], the agents do not have a priori knowledge of the graph (e.g., a map); on the contrary, they have only local knowledge restricted to neighbours at distance at most two of the node where they currently are. The agents discover the network (and construct a map) as they move. Indeed the shadowed exploration phase is truly a distributed exploration and map-construction process.

In the following we present and analyze two decontamination protocols, the first based on a simple“greedy” strategy, the other using a “threshold” search strategy. We prove that both protocols are everywhere optimal; furthermore, the total number of agents used is asymptotycally optimal.

A. Greedy Strategy

1) General Description:

n the shadowed exploration phase, starting from h, the exploration sequence is constructed by the agents *while exploring*. Initially, only the homebase h and its 2-hop neighbourhood ia know to the agents.

At each step, the agents select a node, the *target*, in the map of the graph constructed so far. The target is chosen among the unexplored neighbours of the already explored nodes (the *fronteer*), according to a greedy criterion: the selected node is one with minimum residual degree; should there be multiple candidates, the one with the shortest distance[1] from the last target is chosen as the new target.

Once the target has been selected, agents (the *shadows*) move to occupy the explored neighbours of the target, so to protect them from BV clones. Once this operation is concluded, an agent (the *explorer*) moves to the target.

If the target was not a BV node, the current map of the network is updated so to include the information acquired from visiting the target, and a new step of the exploration process takes place. If instead the target was a BV node, the explorer dies, the BV is removed from the target, and BV clones move to all its neighbours. The clones arriving at the neighbours still unexplored (and thus unprotected by shadows), transform them into BV nodes; the clones arriving at the explored neighbours of the target are destroyed by the shadows located there.

Once this operation is completed, the elimination phase start. Note that at this time, the map contains all the new BV nodes and their neighbours. In the elimination phase, each BV node is decontaminated by an exploring agent moving there, once all its non-BV neighbours have been occupied by shadow agents.

2) Coordination and Synchronization:

The algorithm is described in Figure 1. Missing from the descriptions are the details about how the coordination among the agents and the synchronization required among the different operations (e.g. all the shadow agents must be in place *before* the explorer can move to the target node) are achieved. Time-out cannot be employed since the system is asynchronous and time delays unpredictable and unbounded. The solution is however simple. One of the agents is used to perform the role of coordinator and synchronizer; we shall call this agent the *leader*. It is the leader's task to mantain a map of the explored network and its distance-two boundaries, with the location of all agents. Another agent, the *explorer*, will be the one visiting the unexplored nodes until it finds the BV (and is destroyed).

At the beginning of each step, the leader and the explorer are at the same node. The leader starts the step by selecting the next target. v and determining its already-explored neighbours $N_{ex}(v) = \{z_1, ...z_p\}$ based on the information on the map. The explorer also computes the the target v and the neighbour z_p; it then moves to z_p and waits there for instructions from the leader. The leader meanwhile sequentially goes to the current locations[2] $\{y_1, ...y_{p-1}\}$ of the shadow agents needed for this step. When at location y_i, the leader tells the agent there to

[1] symmetry broken arbitrarily.

[2] Initially, all agents are in the homebase.

become a shadow and to go to node $z_i \in N_{ex}(v)$ to protect it; the leader itself goes to z_i, and waits there until the shadow agent arrives; it then proceeds to the location y_{i+1} of the next needed shadow agent. To reduce the total number of agents used by the protocol, the leader will act as the shadow of the last node $z_p \in N_{ex}(v)$; thus, after the leader determines that a shadow agent has arrived at z_{p-1}, it moves to z_p and acts as the shadow agent for that node. All movements of all the agents during this process is through the already explored part of the graph, and thus safe. Once both the leader and the explorer are at z_p, the leader gives the order and the explorer moves to the target node.

At this point two outcomes are possible: either the explorer returns to z_p or a clone arrives there. In the first case, the next step of the shadowed exploration takes place, with the leader updating the map with the information reported by the explorer and repeating the process just described.

In the second case, the leader disables the arriving clone and starts the coordination of the elimination phase. The leader starts the elimination phase by updating its map to include the locations of the new BV nodes (the unexplored neighbours of the last target v). It then starts the process by going sequentially to the current locations of the agents required to clean the BV nodes, notifying them of the current map, and assigning a BV node to each of them to clean. Each of those cleaners independently reaches the assigned BV node (by first going to v and then to its destination) without any further need of coordination.

3) Analysis:

We now show that the BV-free exploration sequence created by protocol GREEDY EXPLORATION has minimal residual degree.

Theorem 4.1: Let $\pi =< x_0, x_1, ..., x_{n-1} >$ be the BV-free exploration sequence created by procedure GREEDY EXPLORATION with $x_0 = h$. Then $\rho(\pi) = \rho[h]$.

Proof: Let $\alpha =< h, y_1, ..., y_{n-1} >$ be an optimal exploration sequence, i.e., a feasible permutation with minimal residual degree $\rho[h]$. If $\alpha = \pi$, the theorem holds. Thus let us consider the case $\alpha \neq \pi$.

Let $i \geq 1$ be the smallest index such that $x_i \neq y_i$, and let $x_i = y_j$. Consider now the sequence $\alpha_{i,j} = [y_0, y_1, ..., y_{i-1}, y_j, y_i, y_{i+1}, ..., y_{j-1}, y_{j+1}, ..., y_{n-1}] = [x_0, x_1, ..., x_{i-1}, x_i, y_i, y_{i+1}, ..., y_{j-1}, y_{j+1}, ..., y_{n-1}]$, where $y_0 = x_0 = h$. The feasibility of $\alpha_{i,j}$ follows from the feasibility of α and that, by construction, $y_j = x_i$ is a neighbour of some x_l with $l < i$. Thus, by Lemma 3.4 we have that $\rho(y_l, \alpha_{i,j}) \leq \rho(y_l, \alpha)$, for all $l \neq j$. That is, in $\alpha_{i,j}$ the residual degree of every node, except possibly for y_j, is not more than in α.

Consider now y_j in $\alpha_{i,j}$; and recall $\rho(y_j, \alpha_{i,j}) = \rho(x_i, \pi)$. Consider the step of the algorithm in which x_i is chosen; obviously at that time x_i belongs to the frontier. Since y_i is a neighbour of some x_l with $l < i$, in that step also y_i and/or x_l belong to the frontier. By definition, $\rho(y_i, \alpha) \leq \rho(\alpha)$ is a node with residual degree not exceeding the current threshold and thus $\rho(x_l, \alpha) = \rho(y_l, \alpha) \leq \rho(\alpha)$; this implies that the algorithm would not have chosen x_i in that step if $\rho(x_i, \pi) > \rho(\alpha)$; hence, $\rho(y_i, \alpha) \leq \rho(\alpha)$.

Summarizing, in $\alpha_{i,j}$ the residual degree of every node, including y_j, is not more than in α; that is, also $\alpha_{i,j}$ has minimal residual degree $\rho(\alpha)$ and is thus an optimal exploration sequence. In other words, if an optimal exploration sequence (e.g., α) coincides with π in the first i elements, then there exists an optimal exploration sequence (e.g., $\alpha_{i,j}$) that coincides with π in the first $i + 1$ elements. By repeating this argument, the optimality of π follows; i.e., $\rho(\pi) = \rho[h]$. ■

Consider now the size of the team of agents (including casualties) employed by the protocol.

Theorem 4.2: Protocol GREEDY EXPLORATION employs $\Delta + 1$ agents.

Proof: During the shadowed exploration phase, one exploring agent is used to move to the target, and one shadow is

```
GREEDY EXPLORATION

(* Initialization *)
All agents initially at homebase, h.
M = (V_M, E_M) := N^2(h);
(* initial map is the 2-neighborhood of h*)
V_ex := {h}; (* only the h is explored*)
V_un := V_M \ {h}; (* initial unexplored nodes*)
B := h; ((* only the h borders the unexplored nodes*)
Fr := N(h); (*unexplored frontier*)
π := [h]; (* the h is the first in the search sequence*)
Found:= FALSE;

(* Iteration *)
while Found=FALSE
    Forall v ∈ Fr do r(v) := |{u ∈ V_un : (u, v) ∈ E_M}|;
    Choose v ∈ Fr such that r(v) is minimum;
    N_ex(v) := {u ∈ V_ex : (u, v) ∈ E_M};
    Locate a shadow agent at each u ∈ N_ex(v);
    Move an exploring agent to v;
    if (v ≠ BV) then (* update Map and variables*)
      M := M ∪ N^2(v);
      V_ex := V_ex ∪ {v};
      V_un := V_M \ V_ex;
      B := {x ∈ V_ex : ∃y ∈ V_un(x, y) ∈ E_M};
      Fr := {x ∈ V_un : ∃y ∈ V_ex(x, y) ∈ E_M};
      π := π ∗ [v]; (*v is next in the search sequence*)
    else
      Found := TRUE
endwhile
Start ELIMINATION
```

Fig. 1. Greedy

used to protect each of the already explored neighbours of the target; hence, since the leader is also a shadow, at most $\Delta+1$ agents are needed in this phase. In the elimination phase, the total number of agents needed is at most $\rho[h]$ cleaners; since $\rho[h] < \Delta$, the claim holds. ∎

Lemma 4.1: Let G be a d-regular graph. Then any solution protocol needs at least $d+1$ agents.

Proof: In a d-regular graph G, $\rho(P[G,h]) = d-1$ for every homebase h. By Lemma 3.3, $spread(P,G,h) \geq \rho(P[G,h]) + 1$; since at least one agent must survive, the lemma follows. ∎

Thus, by Theorem 4.2 and Lemma 4.1, it follows that GREEDY EXPLORATION is worst-case optimal with respect to the team size.

B. Threshold Strategy

The protocol described in the previous section is optimal both in terms of casualties and number of agents used. In this section we describe and analyze a simple variant with the same properties. Also this algorithm is sequential and it chooses a single target at each step of the shadowed exploration phase. The coordination and synchronization is exactly as in the previous algorithm. The main idea of the algorithm is to select as a target, among the nodes of the frontier, not the one with smallest residual degree (like in the "greedy" protocol), but rather one with residual degree not greather than a *threshold*.

Initially the threshold is set to the smallest residual degrees of the neighbours of the homebase. In subsequent steps, should all frontier nodes have residual degree above the threshold, the threshold is increased to the smallest of those residual degrees. When more than one frontier node is within the threshold, the one closest to the last explored node is chosen.

The algorithm is described in Figure 2, where the coordination and synchronization details are omitted.

We will now show that π has minimal residual degree $\rho[h]$.

Theorem 4.3: Let $\pi =< x_0, x_1, ..., x_{n-1} >$ be the exploration sequence created by protocol THRESHOLD EXPLORATION from homebase $h = x_0$. Then $\rho(\pi) = \rho[h]$.

Proof: Let $\tau(i)$ be the value of the threshold τ when x_i was explored in the execution of THRESHOLD EXPLORATION generating π; thus, by construction, $\tau(j) \leq \tau(j+1)$ $(0 \leq j < n-1)$ and $\rho(\pi) = \tau(n)$

Let $\alpha =< y_0, y_1, ..., y_{n-1} >$ be an optimal exploration sequence, i.e., a feasible permutation for $h = y_0$ with minimal residual degree $\rho[h]$. If $\alpha = \pi$, the theorem holds. Thus let us consider the case $\alpha \neq \pi$. Let $i \geq 1$ be the smallest index such that $x_i \neq y_i$, and let $y_j = x_i$. Consider now the sequence $\alpha_{i,j}$ obtained by moving y_j before y_i; that is, $\alpha_{i,j} = [y_0, y_1, ..., y_{i-1}, y_j, y_i, y_{i+1}, ..., y_{j-1}, y_{j+1}, ..., y_{n-1}]$ = $[x_0, x_1, ..., x_{i-1}, x_i, y_i, y_{i+1}, ..., y_{j-1}, y_{j+1}, ..., y_{n-1}]$, where $y_0 = x_0 = h$.

The feasibility of $\alpha_{i,j}$ follows from the feasibility of α and that, by construction, x_i is a neighbour of some x_l with $l < i$.

THRESHOLD EXPLORATION

(* Initialization *)
All agents initially at homebase, h.
$M = (V_M, E_M) := N^2(h)$;
(* initial Map is the 2-neighborhood of homebase*)
$V_{ex} := \{h\}$; (* only the h is explored *)
$V_{un} := V_M \setminus \{h\}$; (* initial unexplored nodes *)
$Fr := N(h)$; (*unexplored frontier*)
$\pi := [h]$; (* the h is the first in the search sequence*)
$\tau := Min\{|\{u \in V_{un} : (u,v) \in E_M\}| : v \in Fr\}$
(* initial threshold *)
Current := h; Found:= FALSE;

(* Iteration *)
while Found=FALSE
  Forall $v \in Fr$ **do** $r(v) := |\{u \in V_{un} : (u,v) \in E_M\}|$;
  $\tau := Max\{\tau, Min\{r(v) : v \in Fr\}\}$;
(* update threshold *)
  Choose $v \in Fr$ closest to Current with $r(v) \leq \tau$
  $N_{ex}(v) := \{u \in V_{ex} : (u,v) \in E_M\}$;
  Locate a shadow agent at each $u \in N_{ex}(v)$;
  Move an exploring agent to v;
  if $(v \neq BV)$ **then** (* update map and variables*)
    $M := M \cup N^2(v)$; (* update map*)
    $V_{ex} := V_{ex} \cup \{v\}$;
    $V_{un} := V_M \setminus V_{ex}$;
    $Fr := \{x \in V_{un} : \exists y \in V_{ex}, (x,y) \in E_M\}$;
    Current:= v; $\pi := \pi * [v]$;
  else
    Found := TRUE
endwhile
Start ELIMINATION

Fig. 2. Threshold

Thus, by Lemma 3.4, we have that $\rho(y_l, \alpha_{i,j}) \leq \rho(y_l, \alpha)$, for all $l \neq j$. That is, in $\alpha_{i,j}$ the residual degree of every node, except possibly for y_j (i.e., x_i), is not more than in α and (due to the optimality of α) no more than $\rho[h]$. This also means that the first $i-1$ thresholds are all no more than $\rho[h]$; in particular, $\tau(i-1) \leq \rho[h]$.

Let now prove that $\rho(x_i, \alpha_{i,j}) \leq \rho[h]$. By definition, $\rho[h] \geq \rho(x_{i-1}, \alpha) = \rho(x_{i-1}, \pi) = \tau(i-1)$. Consider $\tau(i)$, i.e. the threshold when x_i is chosen constructing π; by construction $\tau(i-1) \leq \tau(i)$. If $\tau(i-1) = \tau(i)$, then trivially $\rho(x_i, \alpha_{i,j}) \leq \rho[h]$. Thus consider the case $\tau(i-1) < \tau(i)$. This case occurs only if the residual degree of all the nodes in fronteer, after the exploration of x_{i-1}, is greater than $\tau(i-1)$; then $\tau(i)$ is set to the smallest among the residual degrees of the nodes in the fronteer; in particular, $\tau(i) = \rho(x_i, \pi)$.

Since α and π concide for the first i elements, the number of unexplored neighbours of y_i after $x_{i-1} = y_{i-1}$ has been explored is precisely $\rho(y_i, \alpha)$ which is no more that $\rho[h]$, since

α is optimal. Observe that, when x_i is selected as the next target in π, also y_i belongs to the fronteer (it follows from feasibility of α); thus, the residual degree of y_i at that time is at least $\tau(i)$. In other words $\rho(x_i, \alpha_{i,j}) = \rho(x_i, \pi) = \tau(i) \leq \rho(y_i, \alpha) \leq \rho[h]$.

Summarizing, in $\alpha_{i,j}$ the residual degree of every node, including y_j, is not more than in α; that is, also $\alpha_{i,j}$ has minimal residual degree $\rho(\alpha)$ and is thus an optimal exploration sequence.

In other words, if an optimal exploration sequence (e.g., α) coincides with π in the first i elements, then there exists an optimal exploration sequence (e.g., $\alpha_{i,j}$) that coincides with π in the first $i+1$ elements. By repeating this argument, the optimality of π follows; i.e., $\rho(\pi) = \rho[h]$. ■

With precisely the same proof of Theorem 4.2, and by Lemma 4.1, we have that also this protocol uses an optimal total number of agents.

Theorem 4.4: Protocol THRESHOLD EXPLORATION employs at most $\Delta + 1$ agents

V. ROOTED ACYCLIC ORIENTATION WITH MINIMUM OUTDEGREE

In this section we establish an interesting connection between solutions of the BVD problem and the problem of determining rooted acyclic orientations of unoriented graphs with minimum outdegrees. As a consequence, our protocols provide a distributed optimal solution to this graph optimization problem. Due to space constraints, the proofs are omitted.

Given an undirected graph $G = (V, E)$, an *orientation* λ of G is an assignment of direction to each edge. Every orientation λ transforms G into a directed graph $\vec{G}_\lambda = (V, \vec{E}_\lambda)$. Let $\mathcal{D}(G)$ be the set of all directed graphs generated by acyclic orientations of G An acyclic orientation λ of G is said to be *rooted* if $\vec{G}_\lambda$ has a single source (i.e., exactly one node of zero in-degree). Let $\mathcal{R}(G, v)$ be the set of all directed graphs generated by acyclic orientations of G rooted in v, and let $\mathcal{R}(G) = \cup_{v \in V}\{\mathcal{D}(G, v)\}$.

Given a directed acyclic graph $\vec{G}$, let $d^+(u, \vec{G})$ be the out-degree of u in $\vec{G}$; and let $d^+(\vec{G}) = Max_{u \neq v}\{d^+(u, \vec{G}_v)\}$ be the maximum out-degree among the nodes. An acyclic orientation λ of G is said to be *optimal* if $d^+(\vec{G}_\lambda) \leq d^+(\vec{G}')$ for all $\vec{G}' \in \mathcal{D}(G)$; similarly an acyclic orientation λ of G rooted in v *optimal* if $d^+(\vec{G}_\lambda) \leq d^+(\vec{G}')$ for all $\vec{G}' \in \mathcal{R}(G, v)$.

The interesting connection between BV-decontamination and otimal rooted acyclic orientations is provided by the property discussed next.

As well known, to any directed acyclic graph $\vec{G}$ corresponds a *partial order* "$\preceq_{\vec{G}}$" on the nodes of the graph, where $x \preceq_{\vec{G}} y$ if and only if there is a directed path from x to y. A *linear extension* of "$\preceq_{\vec{G}}$" is any total order "$<$" on the nodes consistent with $\preceq_{\vec{G}}$; that is, if $x \preceq_{\vec{G}} y$ then $x < y$. The sequence of the nodes ordered according to $<$ defines a unique permutation $X_< = [x_0, x_1, x_2, ..., x_{n-1}]$ of the nodes. Let $\Gamma(\vec{G})$ denote the set of all permutations defined by the linear extensions of the partial order $\preceq_{\vec{G}}$.

Theorem 5.1: Let $\vec{G} \in \mathcal{R}(G, v)$ be a directed acyclic orientation of $G = (V, E)$ rooted in $v \in V$.

1) $\Gamma(\vec{G}, v) \subseteq \Pi(G, v)$
2) $\forall\ X \in \Gamma(\vec{G}, v),\ \rho(X) = d^+(\vec{G})$.

In other words, in a directed acyclic graph $\vec{G} \in \mathcal{R}(G, v)$ rooted in v, every linear extension of $\preceq_{\vec{G}}$ defines a feasible permutation; additionally, all these permutations have the same residual degrees, which coincides with the maximum out degree in $\vec{G}$. As a consequence

Theorem 5.2: Let $\vec{G} \in \mathcal{R}(G, v)$ be such that $\forall \vec{G}' \in \mathcal{R}(G, v), d^+(\vec{G}) \leq d^+(\vec{G}')$. Then $\forall\ X \in \Gamma(\vec{G}, v),\ \rho(X) = \rho(G, v)$

That is, the problem of finding an acyclic orientation of G with v as its only source and with the minimum out-degree possible is equivalent to the problem of determining an optimal feasible permutation for v.

In the previous sections we have seen two protocols that determine an optimal feasible permutation in a decentralized way. By exploiting the result of Theorem 5.2, we can use them to construct an optimal rooted acyclic orientation in a distributed way, using a single agent.

In our protocols, a BV-free exploration sequence $\pi =< x_0, x_1, ..., x_{n-1} >$ is created (different depending on the protocol) stating from the homebase x_0. When at x_i the explorer has enough information to determine what the next target (i.e., x_{i+1}) is; the explorer moves sequentially from x_i to x_{i+1}. Consider now single agent protocol ROOTED ORIENTATION, described in Figure 3. In this protocol, the single agent performs exactly the same operations as performed by the explorer in the BV decontamination protocol being used. The only difference is that now the agent, when visiting a node for the first time (starting from the homebase), orients as outgoing all the edges connecting that node to its still unexplored neighbours.

The selection of x_{i+1} when at x_i will be different depending on whether we follow the greedy strategy of protocol GREEDY EXPLORATION (Protocol GREEDY ROOTED ORIENTATION) with or the threshold strategy of protocol GREEDY EXPLORATION (Protocol THRESHOLD ROOTED ORIENTATION). Regardless of the strategy, the result is an optimal rooted orientation.

Theorem 5.3: Both GREEDY ROOTED ORIENTATION and THRESHOLD ROOTED ORIENTATION produce an optimal acyclic orientation rooted in the homebase.

In a single agent computation, the important cost measure is the number of *moves* performed by the agent.

Let $\pi =< x_0, x_1, ..., x_{n-1} >$ be the sequence obtained by the strategy employed, and let $dis_{ex}(x_i, x_{i+1})$ be the shortest distance between x_i and x_{i+1} in the explored part of the graph when x_i was visited for the first time. Then the number of

ROOTED ORIENTATION

(* Initialization *)
Explorer initially at homebase x_0.
$M = (V_M, E_M) := N^2(x_0)$; (*init map is hop-2 of x_0*)
$\vec{E} = \emptyset$.
$V_{ex} := \{x_0\}$; (*only the home base is explored*)
$V_{un} := V_M \setminus \{x_0\}$; (*init unexplored nodes*)
(* Iteration *)
while $V_{un} \neq \emptyset$
    determine x_i
    move to x_i
    ORIENT(x_i)
    $M := M \cup N^2(x_i)$; (*update map*)
    $V_{ex} := V_{ex} \cup \{x_i\}$; (*update explored nodes*)
    $V_{un} := V_M \setminus V_{ex}$; (*update unexplored nodes*)
endwhile

ORIENT(x)
    forall $y \in N(x) \cap V_{un}$ **do** $\vec{E} := \vec{E} \cup \{\overrightarrow{(x,y)}\}$

Fig. 3. Orient

moves the explores performs is precisely

$$\sum_{i=0}^{n-1}(dis_{ex}(x_i, x_{i+1}))$$

Since $dis_{ex}(x_i, x_{i+1}) < i+1$, it follows that the total number of moves for both protocols is less than $\frac{1}{2}n^2$. This upperbound is however quite coarse.

A more precise bound can be easily established for the THRESHOLD ROOTED ORIENTATION protocol as follows.

Theorem 5.4: The total number of moves of THRESHOLD ROOTED ORIENTATION is less than $2n\Delta$.

VI. CONCLUSIONS

In this paper, we have considered the Black Virus decontamination problem in networks of arbitrary and unknown topology. We have presented an optimal solution that works in a fully decentralized way by agents endouded with only local 2-hop visibility. The algorithm works in an asycnrhnous setting, with the minimum number of system agents' casualties, network infections and total number of employed agents. We have also shown a correspondence between this problem and the one of computing a rooted acyclic orientation of a given graph with minimum outdegrees. Due to this correspondence, with little modifications, our protocols provide a distributed optimal solution to this graph optimization problem.

REFERENCES

[1] Y. Asahiro,E. Miyano, H. Ono, and K. Zenmyo. Graph orientation algorithms to minimize the maximum outdegree. *Int. J. Found. Comput. Sci.* 18(2): 197-215, 2007.

[2] L. Barrière, P. Flocchini, P. Fraignaud, and N. Santoro. Capture of an intruder by mobile agents. *14th Symposium on Parallel Algorithms and Architectures* (SPAA), 200-209, 2002.

[3] L. Blin, P. Fraignaud, N. Nisse, and S. Vial. Distributed chasing of network intruders. *Theoretical Computer Science* 399 (1-2): 12-37. 2008.

[4] J. Cai, P. Flocchini, and N. Santoro. Decontaminating a network from a black virus. *Int. J. Networking and Computing*, 4(1): 151-173, 2014

[5] J. Cai, P. Flocchini, and N. Santoro. Black virus decontamination in arbitrary networks. *3rd Conference on Information Systems and Technologies*, Advances in Intelligent Systems and Computing 353, Springer, 991-1000, 2015.

[6] J. Chalopin, S.Das, A. Labourel, and E. Markou. Black hole search with finite automata scattered in a synchronous torus. *25th International Symposium on Distributed Computing (DISC)*, 432–446, 2011.

[7] C. Cooper, R. Klasing, and T. Radzik. Searching for black-hole faults in a network using multiple agents. *10th International Conference on Principles of Distributed Systems (OPODIS)*, 320–332, 2006.

[8] J. Czyzowicz, S. Dobrev, R. Královic, S. Miklík, and D. Pardubská. Black hole search in directed graphs. *16th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, 182–194, 2009.

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

[10] S. Dobrev, P. Flocchini, R. Královic, P. Ruzicka, G. Prencipe, and N. Santoro. Black hole search in common networks. *Networks*, 47(2):61–71, 2006.

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

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

[13] P. Flocchini, M.J. Huang, and F.L. Luccio. Decontamination of hypercubes by mobile agents. *Networks* 52 (3): 167-178, 2008.

[14] P. Flocchini, M.J. Huang, and F.L. Luccio. Decontaminating chordal rings and tori using mobile agents. *International Journal on Foundations of Computer Science* 18 (3): 547-563, 2006.

[15] P. Flocchini, F.L. Luccio, and L.X. Song. Size optimal strategies for capturing an intruder in mesh networks. *International Conference on Communications in Computing* (CIC), 200-206, 2005.

[16] P. Flocchini and N. Santoro. *Distributed Security Algorithms For Mobile Agents*. Chapter 3. in J. Cao and S.K. Das (Eds.) "Mobile Agents in Networking and Distributed Computing", Wiley, 2012.

[17] F.V. Fomin, D.M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science* 399(3): 236-245, 2008

[18] F.V. Fomin, D.M. Thilikos, and I. Todineau. Connected graph searching in outerplanar graphs. *7th International Conference on Graph Theory* (ICGT), 2005.

[19] P. Glaus. Locating a black hole without the knowledge of incoming link. *5th Int. Workshop on Algorithmic Aspects of Wireless Sensor Networks* (ALGOSENSOR), 128–138, 2009.

[20] D. Ilcinkas, N. Nisse and D. Soguet. The cost of monotonicity in distributed graph searching. *Distributed Computing* 22 (2), 117-127, 2009.

[21] N. Imani, H. Sarbazi-Azadb, and A.Y. Zomaya. Capturing an intruder in product networks. *Journal of Parallel and Distributed Computing* 67 (9): 1018–1028, 2007.

[22] N. Imani, H. Sarbazi-Azad, A.Y. Zomaya, and P. Moinzadeh. Detecting threats in star graphs. *IEEE Trans. Par. and Dist. Systems* 20 (4): 474-483 , 2009.

[23] R. Klasing, E. Markou, T. Radzik, and F. Sarracco. Approximation bounds for black hole search problems. *Networks*, 52(4):216–226, 2008.

[24] F.L. Luccio Contiguous search problem in Sierpinski graphs. *Theory of Computing Systems* 44(2): 186-204, 2009.

[25] F. Luccio and L. Pagli. A general approach to toroidal mesh decontamination with local immunity. *23rd IEEE International Parallel and Distributed Processing Symposium* (IPDPS), 1-8, 2009.

[26] F. Luccio, L. Pagli, and N. Santoro. Network decontamination in presence of local immunity. *Int. J. Foundation of Computer Science* 18(3): 457–474, 2007.

[27] N. Nisse. Connected graph searching in chordal graphs. *Discrete Applied Mathematics* 157 (12): 2603-2610, 2009

[28] M. Venkateswaran. Minimizing maximum indegree. *Discrete Applied Mathematics* 143: 374-378, 2004.