# A Review of Computation Solutions by Mobile Agents in an Unsafe Environment

Anis Zarrad
Department of Computer Science and Information Systems
Prince Sultan University, Riyadh,
Saudi Arabia

Yassine Daadaa
College of Computer and Information Sciences
Al-Imam Muhammad ibn Saud Islamic University, Riyadh,
Saudi Arabia

***Abstract*—Exploration in an unsafe environment is one of the major problems that can be seen as a basic block for many distributed mobile protocols. In such environment we consider that either the nodes (hosts) or the agents can pose some danger to the network. Two cases are considered. In the first case, the dangerous node is a called black hole, a node where incoming agents are trapped, and the problem is for the agents to locate it. In the second case, the dangerous agent is a virus; an agent moving between nodes infecting them, and the problem is for the "good" agents to capture it, and decontaminate the network.**

**In this paper, we present several solutions for a black–hole and network decontamination problems. Then, we analyze their efficiency. Efficiency is evaluated based on the complexity, and the effort is in the minimization of the number of simultaneous decontaminating elements active in the system while performing the decontamination techniques.**

***Keywords—Distributed algorithm; Mobile Agent; Network Decontamination; Black Hole Search; and Network Exploration***

## I. INTRODUCTION

Today's need to maintain network protection practices and challenges in the universe are interconnected. Virus protection represents a rising importance in network decontamination methods. Faults and viruses often spread in networked environments, where nodes represent hosts and edges represent connections between hosts, by propagating from neighboring sites.

Such a topic is known as exploration in unsafe environment. Exploration consists of having a set of agents collaboratively traverse an unknown network to collect relevant information. Network exploration has been extensively studied over the past fifty years due to its various applications in different areas such as engineering, computer science, and applied mathematics. Two major problems are discussed in this work; black-hole and network decontamination.

In network contamination, a node might behave incorrectly, and it could affect its neighbor to become contaminated as well, thus propagating faulty computations. The propagation patterns of faults can follow different dynamics, depending on the behavior of the affected site.

In this work we begin by giving a general overview about network exploration in unsafe environment and its applications. The rest of the paper is organized as follows. First, we summarize the backgrounds and related works, second we review some of the major solutions to date and their classification, and finally in section four we offer a conclusion.

## II. BACKGROUNDS AND RELATED WORKS

The main focus in this work is computation solutions by mobile agents in an unsafe environment. Also, we believe there is a need to highlight the most influential works related to mobile agent systems that occurred in the past in order to give credit to founder researches. The problem of exploration is well known, where agents need to collaborate in order to explore an unknown environment. For example, tasks in environments that is not suitable for human operation, navigating a robot through a terrain containing obstacles, and finding a path through a maze. In recent years, application such as searching for data stored at unknown nodes in a computer network using mobile software agents, and obtaining maps of existing networks (e.g., computer networks, sewage systems, unexplored caves) have been studied. Map construction is the related problem of exploring the network to return an exact map of its topology.

Previous work on exploration of labeled graphs has emphasized minimizing the cost of exploration in terms of the total number of edge traversals (moves), and the amount of memory used by the agent [1, 2, 12, 13, 38]. In [2], Awerbuch et al. studied how a mobile robot can learn an unknown environment in a piecemeal manner. The robot's goal is to learn a complete map of its environment, while satisfying the constraint that it must return every so often to its starting position (e.g., for refuelling).

Exploration of anonymous graphs is impossible if marking of the nodes is not allowed in some way. An exception is when the graph is acyclic, meaning the graph is a tree [14, 30]. Different models for marking the nodes have been used to solve the exploration problem. Pebbles which can be dropped and removed from a node was proposed by Bender et al. in [5], where it was shown that one pebble is enough to explore the graph if the robot knows an upper bound on the size of the graph, and (log log n) pebbles are necessary and sufficient otherwise. Among the various possible techniques of decontamination, two types have been identified in the literature [24] internal and external decontamination.

In internal decontamination a site can decontaminate itself (i.e., it can activate an antiviral software) when a certain condition of the neighborhood is verified. A clean site gets re-contaminated when some other condition of the neighboring

states is verified. This approach has been followed in [39], where a node becomes clean when the majority of its neighbors are clean, and a clean node becomes contaminated if any of its neighbors are contaminated. A similar approach has been taken in [36, 37], where a decontaminated node is immune to recontamination if the majority of its neighbors are clean. Internal decontamination has been specifically studied in the context of fault- tolerance to describe the mechanisms of the spread of faults and of auto-correction (see [40] for a survey). In all these studies the main objective is typically to determine the minimum size of a set of faulty nodes, which completely disrupts the system under given contamination/decontamination dynamics or, equivalently, the minimum size of a set of decontaminating nodes that can decontaminate the whole network under the same circumstances.

On the other hand, mobile agents moving in the network can perform external decontamination. There is an extensive literature on external decontamination either in specific topologies (see [3, 25]), under various assumptions on the capabilities of the agents, or in arbitrary topologies (see [7]). Typically, agents have memory, distinct identifiers, and the ability to communicate with other agents when they meet or exchange information writing on whiteboards (storage area located at the nodes). In all models investigated, agents can move from node to node (usually asynchronously and independently) decontaminating the sites they pass through. A clean site becomes contaminated if at least one of its neighbors is contaminated. External decontamination has been studied in the context of intruder capture to design algorithms to neutralize a virus in a network, or in graph search (e.g., [3, 31, 25, 6, 4]). The main goal of decontamination in these settings is usually to design a strategy that employs the minimum possible size of the team of cleaning agents.

## III. Black Hole Search – A Classification Review

In the last decade, there has been a great deal of work done on finding faults in networks using mobile entities. Most of the existing work deals with the black hole search problem for a single black hole [9, 10, 15, 16, 17, 18, 19, 20, 21, 22, 23, 27, 32, 33, 34]. Some work has been done for multiple black holes in [8, 35], although both of these papers use the synchronous model, as do some of the single black hole papers [9, 10, 33, 34]. Existing solutions can be classified based on the network model: synchronous and asynchronous.

### A. Synchronous Model

In [33], Klasing et al. consider the problem of designing a black hole scheme under the scenario of synchronous networks. Authors investigate the case when there may be at most one black hole in the network. The search is performed by exactly two agents, which start from the same node home-base and can communicate only when they are in the same node. At least one agent must report the information back to the home-base on the exact location of the black hole or whether one exists. In [9], Czyzowicz et al. studied the black hole problem in a (partially) synchronous network, assuming an upper bound on the time of any edge traversal by an agent. In this work, the minimum number of agents capable to identify a black hole is two for a given graph and a given starting node.

Czyzowicz et al. [9] looked to the fastest possible black hole search by two agents, under the general scenario in which some subsets of nodes are safe and the black hole can be located in one of the remaining nodes. Authors show that the problem of finding the fastest possible black hole search scheme by two agents is NP-hard, and they give a polynomial approximation to solve it.

In [10], Czyzowicz looked at the same problem in trees, and gave optimal black hole search algorithms for two extreme classes of trees: the class of lines and the class of trees in which any internal node (including the root which is the starting node) has at least two children. In [34], Klasing et al. consider the same problem assuming that the map of the network is given. Their objective is minimizing the number of agents that fall into the black hole and the time taken by the surviving agents to locate the black hole. The proposed algorithm explores the network via a spanning tree. In [8], Cooper et al. were the first to consider the general case of multiple black holes using k agents starting from the same node. The agents move through the network in synchronous steps and can communicate only when they meet in a node. In [35], Kosowski et al. look at a multiple black hole search: assuming a directed graph. The robots are associated with unique identifiers, they know the number of nodes in the graph (or at least an upper bound), and they know the number of edges leading to the black holes. Each node is associated with a whiteboard, separately considering the synchronous and the asynchronous cases.

### B. Synchronous Model

The asynchronous scenario was studied under different agent models (i.e., network knowledge, tokens, and whiteboard). In [19], Dobrev *et al*. provided solutions to the black hole search problem in anonymous rings using whiteboards for two settings; when the anonymous agents are co-located, and when they are dispersed. They proved that two such agents are necessary and sufficient to locate the black hole. In [18], Dobrev *et al*. provide a characterization of the impact that factors such as a priori network knowledge and consistency of the local port labeling have on the complexity of the black hole location problem. In [18] authors consider both the case of topological ignorance in systems where there is sense of direction and the case of complete topological knowledge of the network. Authors show that, in both cases, two agents suffice.

In [15, 20, 21], Dobrev *et al*. investigate the black hole search problem for two agents with a map and whiteboard searching for a single black hole. In [20], authors present a search protocol that improves the bounds from the worst case lower bound of $\Omega(n \log n)$ agents moves to $O(n + d \log d)$ agents moves, where $d$ is the diameter of the network. The result allows for $\Theta(n)$ moves for a large class of possibly unstructured networks with low diameter. In [15], Dobrev *et al*. show that with a map of the network, a team of two agents suffices, and the number of moves is in the worst case $O(n \log n)$. They also present a general strategy that allows two agents to locate the black hole with $O(n)$ moves in common interconnection networks such as hypercubes, cube connected cycles, star graphs, wrapped butterflies, and chordal rings, as well as in multidimensional meshes and tori of restricted

diameter. These results hold even if the networks are anonymous. In [19], authors use a technique based on pre-calculating the open vertex cover of cycles of a graph that allows them to solve the problem in $\Theta(n)$ for a large class of networks.

In contrast, in [32], Glaus studied the black hole search problem without the knowledge of the incoming link, and has shown that this modification has effects on the size of the solution. Glaus [32] provided the lower bound on the number of agents that are necessary to locate the black hole; any correct algorithm solving the black hole search problem without the knowledge of the incoming link needs at least $\frac{\Delta^2+\Delta+2}{2}+1$ agents. The algorithm uses the optimal number of agents in the worst case, however, the cost of the algorithm and bounds on the optimal cost of the solution were not shown in this paper.

In [22], Dobrev et al. consider the token model, where each agent has a bounded number of tokens available that can be carried, placed on, or removed from a node. All tokens are identical (i.e., indistinguishable), and no other form of communication or coordination is available to the agents. Authors first prove that a team of two agents is sufficient to locate the black hole infinite time even in this weaker coordination model. Furthermore, Dobrev et al. [22] prove that this can be accomplished using only $O(n \log n)$ moves in total, which is optimal, the same as with whiteboards. Finally, authors show that to achieve this result the agents need to use only $O(1)$ tokens each.

The previous strategy is generalized in [16], where Dobrev *et al*. look to the case of unknown graph. Authors present an algorithm that works in the token model and solves the black hole search problem with the minimal number of agents and with a polynomial number of moves. Dobrev *et al*. [16] algorithm works even if the agents are asynchronous, and if both the agents and the nodes are anonymous. More precisely, authors consider an unknown, arbitrary, anonymous network and a team of exploring agents starting their identical algorithm from the same node (home-based). The agents are anonymous, and they move from node to neighboring node asynchronously. Each agent has an indistinguishable token (or pebble) available that can be placed on, or removed from a node. The token can be placed on a node, either in the center or on an incident link. In the proposed algorithm, two tokens are never placed in the same location (node center or port), nor does an agent ever carry more than one token. Using only this tool for marking nodes and communicating information, authors show that with $(\Delta + 1)$ agents (where $\Delta$ is the maximal degree of the graph), the exploration can be successfully completed. The proposed algorithm allows at least one agent to survive and, within a finite time, the surviving agents will know the location of the black hole with the allowed level of accuracy. The number of moves performed by the agents when executing the proposed protocol is shown to be polynomial, and the proposed algorithm is rather complex.

In [23], Dobrev et al. show not only that a black hole can be located in a ring using tokens with scattered agents, but also that the problem is solvable even if the ring is unoriented. First authors prove that the black hole search problem can be solved using only three scattered agents. Then, Dobrev et al. [23] show that, with k (k > 4) scattered agents, the black hole can be located in O(kn+n log n) moves. Moreover, when k(k > 4) is a constant number, the move cost can be reduced to O(nlogn), which is optimal. These results hold even if both agents and nodes are anonymous.

In [17], Dobrev et al. consider k anonymous, asynchronous mobile agents in an anonymous ring with a black hole. The agents are aware of the existence, but not of the location of such a danger, and the network are totally asynchronous. In this setting it was observed that in order to solve the problem, the network must be 2-connected. A black hole search is not feasible in trees, because in asynchronous networks it is impossible to distinguish a black hole from an incident slow link. 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 authors worked under the assumption that there is exactly one black hole and the task was to locate it. In [27], Flocchini et al. prove that the pure token model is computationally as powerful as the whiteboard model for the black hole search problem. Furthermore, the complexity is exactly the same. Authors prove that a team of two asynchronous agents, each endowed with a single identical pebble (which can be placed only on nodes, and with no more than one pebble per node) can locate the black hole in an arbitrary network of known topology. This can be done with (n log n) moves, where n is the number of nodes.

## IV. External Decontamination- A Classification Review And Evaluation

Due to the manner in which the Network topology is contaminated, decontamination approaches may not be efficiency applied in term of complexity and resource efforts. Therefore we have created a novel classification for decontamination. Our classification space pays particular attention to the impact that the choices of some parameters of the model have on the efficiency of the solutions. In this section we review and compare such recent strategies solution according to three key characteristics: Network topology, effort minimization and complexity results produced.

### *A. Tree Topology*

The tree was the first topology to be investigated in the Decontamination. In [3], Barrière et al. showed that for a given tree *T*, the minimum number of agents needed to decontaminate *T* depends on the location of the homebase. The proposed solution is based on two observations. Consider node *A*, if *A* is not the homebase, the agents will arrive at *A* for the first time from some link *e*. Let $T_1(A), \ldots, T_i(A), \ldots, T_{d(A)-1}$ be the subtrees of *A* from the other incident links, where *d(A)* denotes the degree of *A*, let $m_i$ be the number of agents needed to decontaminate $T_i(A)$ once the agents are at *A*, and let $m_i \geq m_{i+1}$, $1 \leq i \leq d(A)-2$. The first observation is that to decontaminate *A* and all its other subtrees without recontamination, the minimum number *m (A)* of agents needed is $m(A) = m_1$ if $m_1 > m_2$ and $m(A) = m_1 + 1$ if $m_1 = m_2$. Consider now homebase *B*, let $m_j(B)$ be the minimum number

of agents needed to decontaminate the subtree $T_j(B)$, and let $m_j \geq m_{j+1}$, $1 \leq j \leq d(B)$. The second observation is that to decontaminate the entire tree starting from B the minimum number $m(B)$ of agents needed is $m(B) = m_1$ if $m_1 > m_2$ and $m(B) = m_1 + 1$ if $m_1 = m_2$.

Based on these two observations, Barrière et al. [3] first show how the determination of the optimal number of agents can be done through saturation. Simple information about the structure of the tree are collected from the leaves and propagated along the tree, until the optimal number of agents is known for each possible starting point. The most interesting aspect of this strategy is that it immediately yields a protocol for trees that uses the exact minimum number of agents. The technique to determine the minimum number of agents and the corresponding decontamination strategy is done in $O(n)$ time and exchanges $O(n)$ messages. The algorithm is also naturally distributed; the minimum number of agents and the decontamination strategy can be computed in a decentralized manner. The trees that require the largest number of agents are complete binary trees, where the number of agents is $O(\log(n))$. In contrast, in the line two agents are sufficient.

In [29], Flocchini et al. introduce decontamination with *temporal immunity* in a tree. The main difference between the classical decontamination model, and the *temporal immunity* model is that, a cleaner is able to decontaminate any infected node it visits. Once the cleaner departs, the decontaminated node is immune for a certain time $t \geq 0$ (i.e. $t = 0$ corresponds to the model without temporal immunity studied in the previous work) time units to viral attacks from infected neighbors. After the immunity time $t$ is elapsed, recontamination can occur. The minimum team size necessary to disinfect any given tree with immunity time $t$ is derived. Further, Flocchini et al. [29] show how to compute the minimum team size for all nodes of the tree and implicitly the solution strategy starting from each starting node. These computations use a total of $\Theta(n)$ time (serially) or $\Theta(n)$ messages (distributively). Authors then provide a complete structural characterization of the class of trees that can be decontaminated with $k$ agents and immunity time $t$; Flocchini et al. [29] do so by identifying the forbidden subgraphs and analyzing their properties. Finally, authors consider generic decontamination algorithms, protocols that work unchanged in a large class of trees with little knowledge of their topological structure. Flocchini *et al*. [29] prove that, for each immunity time $t \geq 0$, all trees of a maximum height $h$ can be decontaminated by a team of $k = \left\lfloor \frac{2h}{t+2} \right\rfloor$ agents whose only knowledge of the tree is the bound $h$.

### B. Hypercube Topology

Decontamination in a hypercube has been studied in [25], in which Flocchini et al. prove a lower bound on the number of agents necessary and sufficient to decontaminate a hypercube of size *n*, $\Theta(\frac{n}{\sqrt{\log n}})$. The employ of this optimal number in the *Local Model* has an interesting consequence, $\Theta(\frac{n}{\sqrt{\log n}})$ is the search number in the classical graph search problem. In the *Local Model* an agent located at a node can see only local information like state of the node, labels of the incident links, and other agent present at the node.

Four different strategies were proposed. In the first strategy (*Local model*), one of the agents acts as a coordinator for the entire cleaning process. The cleaning strategy is carried out on the broadcast tree of the hypercube. The main idea is to place enough agents on the home-base and to have them move, level by level, on the edges of the broadcast tree, led by the coordinator in such a way that no recontamination may occur. This strategy employs an optimal number of agents $\Theta(\frac{n}{\sqrt{\log n}})$, with $O(n \log n)$ moves, and runs in $O(n \log n)$ time steps. The second strategy is devised for a model where the agents are allowed to ``see" the state of their neighbors called *Visibility Model*. Visibility offer to the agent the capability to see whether neighboring node is guarded, clean, or contaminated, in some mobile agent system the visibility power could be easily achieved by probing the state of neighboring node before making a decision.

In this strategy, the computation is local, so there is no need for a coordinator and agents can move autonomously. In fact, the agents are still moving on the broadcast tree, but they do not have to follow the order imposed by the coordinator. In this setting the solution requires $\frac{n}{2}$ agents, but the time complexity is optimal ($\log n$ time steps), and requires the same number of moves $O(n \log n)$ . Finally, the last two strategies are devised for models that assume agents have cloning capabilities and can either ``see" the state of their neighbors or move in a synchronous setting. In both cases the bound on the number of moves becomes optimal decreasing to $n - 1$.

### C. Mesh Topology

In [28], Flocchini et al. consider the problem of decontaminating a $m \times$ n Mesh $(m \leq n)$. They show some lower bounds on the number of agents, number of moves, and time required to decontaminate an $m \times$ n Mesh $(m \leq n)$. At least $m$ agents, $mn$ moves, and $m + n - 2$ time units are required to solve the decontamination problem. The authors consider two models, one in which an agent has only local knowledge about the node where it resides, and the other in which an agent has visibility to see their neighboring nodes.

In the first model, the only knowledge that an agent has is the information written on the local whiteboard at its current location. The algorithm is described as follows. In the initialization phase, all agents have entered the network in the homebase which is the initial position ($P(0,0)$, upper left most node in the mesh), before the cleaning. Each searching agent is informed by a Synchronizer $S$ to move to its starting point along the first column, but $S$ stays at node $P(0,0)$. By the end of this step there will be one searcher in each node of the first column of the Mesh, while $S$ and one of the searching agents are at the node $P(0,0)$. In the cleaning phase; the Synchronizer $S$ moves *SOUTH* and forces the searching agents to move *EAST* one at a time. When the whole column of agents has moved to the next column, the Synchronizer will also move *EAST* to the next column. Then, it will move *NORTH* and continue to force the searching agents to move *EAST* one at a time. Again, when the whole column of agents has moved to the next column, the Synchronizer will also move *EAST* to the next column. These operations are repeated until the Mesh is

cleaned. This algorithm requires $(m+1)$ agents, $\frac{m^2+4mn-5m-2}{2}$ moves, and $(mn-2)$ time units.

In the second model, agents have the power of visibility. Each agent can see the other agents located at its adjacent neighboring nodes and may coordinate its searching operations according to its neighboring agents` moves. In other words, each agent moves independently without the need of a Synchronizer, and agents communicate with each other by using the whiteboard associated with each node in the Mesh. In the initialization phase, all the *m* agents are located at the node $P(0,0)$ In the cleaning phase, all agents wake up to be the searchers $s$. Each searcher $s$ will independently perform the following operations: $s$ reads the whiteboard on the current node. If the whiteboard has a ``*CLEAN*'' message, $s$ moves *SOUTH* to the next row. $s$ will contiguously move *SOUTH* until it reaches a node on which the Whiteboard is empty. If the whiteboard is empty, $s$ writes a ``*CLEAN*'' message on it. $s$ guards the current node until it can see that its neighbouring nodes *NORTH-NORTH*, *NORTH-WEST*, and *NORTH-SOUTH* except *NORTH-EAST* are all clean (or guarded), then $s$ moves *EAST* to the next column. $s$ will repeat this operation until it reaches the last column in the Mesh. The Mesh is cleaned when all the $m$ searching agents reach the last column. This algorithm requires $m$ Agents, $\frac{m^2+2mn-3m}{2}$ moves, and $(m+n-2)$ time unit, time and number of agents complexity are optimal.

In [11], Daadaa *et al.* describe an efficient network decontamination approach. The system consists of a two dimensional lattice that evolves like a cellular automata. A dynamic contamination process causes the spread of a virus (or a fault), and the presence of an agent on a cell guarantees local disinfection (or decontamination). Once disinfected, a cell stays immune to recontamination for a predetermined amount of time. The goal is to design the local rules for the agents and their initial placement so that the agents can decontaminate the entire system without allowing any cell to be re-contaminated. To be efficient, the decontamination should employ as few agents as possible. We design several strategies depending on the type of neighborhood, and on the ability of the agents to clone themselves. The efficiency of the proposed solution depends on the relationship between n and T, where n is the size of the mesh topology and T is time immunity.

### *D. Tori and Chordal Ring Topologies*

In [26], Flocchini et al. studied the decontamination problem in tori and chordal rings. It has been shown that any solution of the decontamination problem in a torus $T(h,k)$ with $h,k \geq 4$ requires at least $2 \times \min(h,k)$ agents, and in the *Local Model* it requires at least $2 \times \min(h,k)+1$ agents.

To match the lower bound a very simple strategy is employed. The idea is to deploy the agents to cover two consecutive columns and then keep one column of agents to guard from recontamination and have the other column move along the torus. In this setting the solution requires $2h+1$ agents, $hk-2h$ time units and $2hk-4h-1$ moves, where $h,k$ are the dimensions of the torus, $h \leq k$. As for the other topologies, visibility decreases time and slightly increases the number of agents. In the case of torus, it is interesting that in the *Visibility Model* all three complexity measures are optimal. This strategy employs $2h$ agents, $\left\lceil \frac{k-2}{2} \right\rceil$ time units and $hk-2h$ moves. These strategies were generalized to the case of d-dimensional tori.

The *Local* and *Visibility Models* have been also studied in the chordal ring topology. A chordal ring with $n$ nodes is defined as $C(< d_1 = 1, \dots, d_k >)$ and a link structure is defined as $(< d_1 = 1, \dots, d_k >)$ where $d_i < d_{i+1}$ and $d_k \leq \left\lfloor \frac{n}{2} \right\rfloor$. In [26], it is first shown that the smallest number of agents needed for the decontamination does not depend on the size of the chordal ring, but solely on the length of the longest chord. In fact, any solution of the contiguous decontamination problem in a chordal ring $C(< d_1 = 1, \dots, d_k >)$ with $4 \leq d_k \leq \sqrt{n}$ requires at least $2d_k$ agents in the *Local Model* and $d_k+1$ agents in the *Visibility Model*. In both models, the cleaning is preceded by a deployment stage after which the agents have to occupy $2d_k$ consecutive nodes. After the deployment, the decontamination stage can start. In the *Local Model*, nodes $x0$ to $x_{d_k+1}$ are constantly guarded by one agent each, forming a window of $d_k$ agents. This window of agents will shield the clean nodes from recontamination from one direction of the ring while the agents of the other window are moved by the coordinator (one at a time starting from the one occupying node $x_{d_k}$) along their longest chord to clean the next window in the ring. Also in the case of the chordal ring, the visibility assumption allows the agents to make their own decision solely on the basis of their local knowledge. An agent move to clean a neighbor only when this is the only contaminated neighbor.

In [36], Luccio et al. studied network decontamination on a k-dimensional torus$(n_1, , n_2, \dots, n_k)$ with $k \geq 1$ and $2 \leq n_1 \leq \cdots \leq n_k$. The decontamination is done by a set of agents moving on a network with local immunity. After an agent leaves from a vertex, this vertex remains uncontaminated as long as $m$ neighbors are uncontaminated. The problem of decontamination is studied for k-dimensional torus with an arbitrary immunity level $m$, upper and lower bounds are established on the number of agents and of their moves. The proposed approach has required the development of new concepts and algorithms, attaining general results that admit the previous result (where there is no local immunity, $(m=0)$ ) is for k-dimensional torus topology as special cases.

## V. CONCLUSION

In this work, we review the exploration in an unsafe environment topic. Two major problems Black-Hole and network decontamination were presented. In the Black-Hole we provide a brief description of synchronous and asynchronous models in order to give the reader a complete knowledge domain. In network decontamination an evaluation methodology was set in terms of a minimum number of system agents needed. We have established both lower and upper bounds in different network topology such as Mesh. Tree, Hypercube, and Chorded ring. The introduction of time immunity will have an important impact on the evaluation results.

## ACKNOWLEDGMENT

We would like to thank Paola Flocchini. We would also like to thank the anonymous referees for their useful comments and suggestions.

## REFERENCES

[1] S. Albers and M. R. Henzinger. Exploring Unknown Environments. In 29th Annual ACM Symposium on Theory of Computing (STOC), pages 416-425, 1997.

[2] B. Awerbuch, M. Betke, R. L. Rivest, and M. Singh. Piecemeal Graph Exploration by a Mobile Robot. Information and Computation, 152(2): 155-172, 1999.

[3] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an Intruder by Mobile Agents. In ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 200-209, 2002.

[4] L. Barrière, P. Fraigniaud, N. Santoro, and D. M. Thilikos. Searching Is Not Jumping. In 29th International Workshop on Graph Theoretic Concepts in Computer Science (WG), pages 34-45, 2003.

[5] M. A. Bender, A. Fernández, D. Ron, A. Dennunzio Sahai, and S. P. Vadhan. The Power of a Pebble: Exploring and Mapping Directed Graphs. Information and Computation, 176(1):1-21, 2002.

[6] L. Blin, P. Fraigniaud, N. Nisse, and S. Vial. Distributed Chasing of Network Intruders. Theoretical Computer Science, 399(1-2):12-37, 2008.

[7] R. Breish. Intuitive Approach to Speleotopology. Southwestern cavers, 6(5):72-78, 1967.

[8] C. Cooper, R. Klasing, and T. Radzik. Searching for Black-Hole Faults in a Network Using Multiple Agents. In International Conference On Principles Of Distributed Systems (OPODIS), pages 320-332, 2006.

[9] J. Czyzowicz, D. R. Kowalski, E. Markou, and A. Pelc. Complexity of Searching for a Black Hole. Journal Fundamenta Informaticae, 71(2-3):229-242, 2006.

[10] J. Czyzowicz, D. R. Kowalski, E. Markou, and A. Pelc. Searching for a Black Hole in Synchronous Tree Networks. Combinatorics, Probability and Computing, 16(4):595-619, 2007.

[11] Y. Daadaa, P. Flocchini, and N. Zaguia. Decontamination with Temporal Immunity by Mobile Cellular Automata. In International Conference on Scienti_c Computing (CSC), pages 172-178, 2011.

[12] S. Das, P. Flocchini, A. Nayak, and N. Santoro. Distributed Exploration of an Unknown Graph. In 12th International Colloquium on Structural Information Complexity (SIROCCO), pages 99-114, 2005.

[13] X. Deng and C. H. Papadimitriou. Exploring an Unknown Graph (Extended Abstract). In FOCS, pages 355-361, 1990.

[14] K. Diks, P. Fraigniaud, E. Kranakis, and A. Pelc. Tree Exploration with Little Memory. Journal of Algorithms, 51(1):38-63, 2002.

[15] S. Dobrev, P. Flocchini, R. Kralovic, P. Ruzicka, G. Prencipe, and N. Santoro. Black Hole Search in Common Interconnection Networks. Networks, 47(2):61-71, 2006.

[16] S. Dobrev, P. Flocchini, R. Kralovic, and N. Santoro. Exploring an Unknown Graph to Locate a Black Hole Using Tokens. In IFIP TCS, pages 131-150, 2006.

[17] S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Multiple Agents Rendezvous in a Ring in Spite of a Black Hole. In International Conference On Principles Of DIstributed Systems (OPODIS), pages 34-46, 2003.

[18] 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.

[19] S. Dobrev, P. Flocchini, G Prencipe, and N. Santoro. Mobile Search for a Black Hole in an Anonymous Ring. Algorithmica, 48(1):67-90, 2007.

[20] S. Dobrev, P. Flocchini, and N. Santoro. Improved Bounds for Optimal Black Hole Search with a Network Map. pages 111-122, 2004.

[21] S. Dobrev, P. Flocchini, and N. Santoro. Cycling Through a Dangerous Network: A SimpleEfficient Strategy for Black Hole Search. In 26th International Conference on Distributed Computing Systems (ICDCS), page 57, 2006.

[22] S. Dobrev, R. Kralovic, N. Santoro, and W. Shi. Black Hole Search in Asynchronous Rings Using Tokens. In 6th Conference on Algorithms and Complexity (CIAC), pages 139-150, 2006.

[23] S. Dobrev, N. Santoro, and W. Shi. Using Scattered Mobile Agents to Locate a Black Hole in an Un-Oriented Ring with Tokens. International Journal of Foundation of Computer Science, 19(6):1355-1372, 2008.

[24] P. Flocchini. Contamination and Decontamination in Majority-Based Systems. Journal of Cellular Automata, 4(3):183-200, 2009.

[25] P. Flocchini, M. J. Huang, and F. L. Luccio. Decontamination of Hypercubes by Mobile Agents. Networks, 52(3):167-178, 2008.

[26] P. Flocchini, M. Jun Huang, and F. L. Luccio. Decontaminating Chordal Rings and Tori using Mobile Agents. International Journal of Foundation of Computer Science, 18(3):547{563, 2007.

[27] P. Flocchini, D. Ilcinkas, and N. Santoro. Ping Pong in Dangerous Graphs: Optimal Black Hole Search with Pure Tokens. In 22nd International Symposium on Distributed Computing (DISC), pages 227-241, 2008.

[28] P. Flocchini, F. L. Luccio, and L. Xiuli Song. Size Optimal Strategies for Capturing an Intruder in Mesh Networks. In Communications in Computing, pages 200-206, 2005.

[29] P. Flocchini, B. Mans, and N. Santoro. Tree Decontamination with Temporary Immunity. In 19th International Symposium on Algorithms and Computation (ISAAC), pages 330-341, 2008.

[30] P. Fraigniaud, L. Gasieniec, D. R.Kowalski, and A. Pelc. Collective Tree Exploration. Networks, 48(3):166-177, 2006.

[31] P. Fraigniaud and N. Nisse. Connected Treewidth and Connected Graph Searching. In 7th Latin American Symposium on Theoretical Informatics (LATIN), pages 479-490, 2006.

[32] P. Glaus. Locating a Black Hole without the Knowledge of Incoming Link. In Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS), pages 128-138, 2009.

[33] R. Klasing, E. Markou, T. Radzik, and F. Sarracco. Approximation Bounds for Black Hole Search Problems. In International Conference On Principles Of DIstributed Systems (OPODIS), pages 261-274, 2005.

[34] R. Klasing, E. Markou, T. Radzik, and F. Sarracco. Hardness and Approximation Results for Black Hole Search in Arbitrary Networks. Theoretical Computer Science, 384(2-3):201-221, 2007.

[35] A. Kosowski, A. Navarra, and M. C. Pinotti. Synchronization Helps Robots to Detect Black Holes in Directed Graphs. In International Conference Of Principles Of Distributed Systems (OPODIS), pages 86-98, 2009.

[36] F. Luccio and L. Pagli. A General Approach to Toroidal Mesh Decontamination with Local Immunity. In 2009 IEEE International Symposium on Parallel & Distributed Processing, pages 1-8, 2009.

[37] F. Luccio, L. Pagli, and N. Santoro. Network decontamination in Presence of Local Immunity. International Journal of Foundation of Computer Science, 18(3):457-474, 2007.

[38] P. Panaite and A. Pelc. Exploring Unknown Undirected Graphs. Journal of Algorithms, 33(2):281-295, 1999.

[39] D. Peleg. Size Bounds for Dynamic Monopolies. Discrete Applied Mathematics,86:263-273, 1998.

[40] D. Peleg. Local Majorities, Coalitions and Monopolies in Graphs: a Review. Theoretical Computer Science, 282(2):231-257, 2002.