International Journal of Foundations of Computer Science

DECONTAMINATING CHORDAL RINGS AND TORI USING MOBILE AGENTS

PAOLA FLOCCHINI and MIAO JUN HUANG
*SITE, University of Ottawa**

and

FLAMINIA L. LUCCIO
Dip. di Matematica e Informatica, University of Trieste†

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

In this paper we consider a network where an intruder is moving "contaminating" the nodes it passes by, and we focus on the problem of decontaminating such a network by a team of mobile agents.
The contamination/decontamination process has the following asynchronous dynamics: when the team is deployed all nodes are assumed to be contaminated, when an agent transits on a node, it will clean the node, when the node is left with no agent, the node will be recontaminated as soon as at least one of its neighbours is contaminated.
We study the problem in asynchronous chordal ring networks and in tori. We consider two variations of the model: one where agents have only local knowledge, the other in which they have "visibility", i.e., they can "see" the state of their neighbouring nodes.
We first derive lower bounds on the minimum number of agents necessary for the decontamination. In the case of chordal rings we show that the number of agents necessary to perform the cleaning does not depend on the size of the network; in fact it is linear in the length of the longest chord (provided that it is not too long). In the case of a torus, the minimum number of agents is equal to $2 \cdot h$, where h is the smallest dimension.
We then propose optimal strategies for decontamination and we analyse the number of moves and the time complexity of the decontamination algorithms, showing that the visibility assumption allows us to decrease substantially both complexity measures. Another advantage of the "visibility model" is that agents move independently and autonomously without requiring any coordination.

*School of Information Technology and Engineering, University of Ottawa, 800 King Edward, Ottawa, Canada. {mhuang,flocchin}@site.uottawa.ca.
†Dip. di Matematica e Informatica, 34100, Trieste, Italy. luccio@dsm.univ.trieste.it

1. Introduction

1.1. Problem and Framework

In networked environments supporting mobile agents, security is a pressing concern and to tackle security issues is becoming more and more complex due to the growing potential threats a network can be faced with. A particularly important security concern is to protect a network from unwanted, and possibly dangerous intrusions. At an abstract level, an intruder is an alien process that moves on the network to sites unoccupied by the system's agents "contaminating" the nodes it passes by. The concern for the severe damage intruders can cause has motivated a large amount of research, especially on detection (e.g., see [1, 13, 23]). Rather than being interested in the detection of the presence of an intruder, we are interested instead in designing strategies for "decontaminating" a possibly infected network, by deploying a team of "cleaning agents". Decontaminating the network, while protecting it from recontamination, is a rather useful task; an efficient implementation of this task would have practical consequences for fighting the spread of viruses in the network.

We consider a networked environment where nodes are hosts and links represent connections between hosts. We assume the nodes of the network are initially *contaminated* and we want to deploy a team of agents to *clean* (or decontaminate) the whole network. The cleaning of a node occurs when an agent transits on the node; however, as soon as a node without an agent on it has a contaminated neighbour, it will become contaminated again. We are interested in monotone decontamination strategies, i.e., we want that once a node is clean, it remains clean until the whole network is decontaminated. More precisely, a team of agents is initially located at a node (the *homebase*) and agents can move from node to neighbouring node. At any point in time each node of the network can be in one of three possible states: *clean*, *contaminated*, *guarded*. Initially all nodes are contaminated except for the homebase (which is guarded). A node is guarded when it contains at least one agent. We say that a node is clean when an agent passes by it and all its neighbouring nodes are clean or guarded, contaminated otherwise. The solution of the problem is given by devising a strategy for the agents to move in the network in such a way that at the end all the nodes are clean.

The system is *asynchronous*, that is, every action the agents perform (i.e., computing, moving), takes a finite but otherwise unpredictable amount of time. In this setting efficiency is measured in terms of the number of agents to be involved, traffic (i.e., number of moves the agents have to perform), and time (or steps)[a]. We consider two variations of the model and accordingly propose lower bounds on the number of agents necessary for decontamination and tight strategies. In the first model (*Local model*), the only knowledge that an agent has is the information available at its current location (port labels, state of the node); in the second model

[a] As the system is asynchronous, we will measure ideal time, i.e., we assume - for the purpose of time complexity only - that it takes one unit of time for an agent to traverse a link.

(*Visibility model*) agents can "see" also the state of their neighbouring nodes.

Network decontamination could be fairly simple in some specific topologies, where the determination of the minimum number of agents required for the task could be easy. This is the case, for example, of the ring, where two agents starting from the same node can move in opposite direction and easily clean the whole network (with one agent the task would be obviously impossible). Determining the optimal number of agents and a tight strategy is however in general an NP-complete problem.

In this paper we consider *chordal ring* networks and tori. Chordal rings are a particular case of circulant graphs, and are also known in the literature as *distributed loop networks*. A chordal ring is a ring augmented by additional chords (each node has the same chord structure) that act as "shortcuts" of the external ring. Chordal rings constitute a common topology for interconnection networks and have been widely studied in the literature to analyse their fault-tolerant properties (for a survey see [5]). Subclasses of chordal rings have been studied under a variety of scenarios and for a large number of problems ranging from routing (e.g., [15]), to election (e.g., [2]), to broadcast (e.g., [17]).

1.2. Related Work

It is easy to see that the decontamination problem can be equivalently formulated in terms of an *intruder capture* problem, where an intruder moves arbitrarily fast in a network and a team of searching agents is deployed to capture it. The intruder capture problem has been extensively studied in the literature under the name of *graph search* in a model where the searchers may be placed and removed from any node of the graph, i.e., they are allowed to *"jump"* while they perform the searching task. This problem was first introduced by Breish [7] and Parson [20, 21], and after that several variations of the problem have been studied: among them, node search and edge search (see, e.g., [8, 14, 16, 18, 19, 22]), where the aim is to find a strategy that minimizes the number of searchers and leads the graph to a state in which all nodes (or nodes and edges) are simultaneously decontaminated. The size of the searching team is called *node-search number* $ns(G)$ (or *edge-search number* $es(G)$)) and the determination of the optimal size is an NP-complete problem in general.

Graph search, intruder detection, and decontamination are equivalent problems. The main difference in our setting is that the agents *cannot be removed from the network*: they can only move from a node to a *neighbouring node*; this assumption is obviously motivated by the fact that we are considering software agents that are able to move only on the edges of the network. In fact, we consider the *contiguous, monotone, decontamination* first introduced in [3] where: 1) the removal of agents is not allowed, 2) at any time of the search strategy, the set of clean nodes forms a connected subnetwork, and 3) a clean node cannot be recontaminated. The contiguous assumption considerably changes the nature of the problem and the classical results on node and edge search do not generally apply. Moreover, the problem is harder than the non-contiguous one as in [4] it has been proved that the contiguous searching number is always greater or equal to the non-contiguous searching

number; the relationship between the search numbers in the two models has been also studied in outerplanar graphs [12]. Finding the contiguous searching number is still an NP-complete problem for general graphs; some specific topologies have been studied, for example it has been shown that it can be solved in linear time in trees [3], moreover, optimal strategies have been studied in hypercubes and meshes [9, 10]. Also the arbitrary topology has been considered; in this case, some heuristics have been proposed [11] and a move-exponential solution has been given in [6].

1.3. Our Results

As mentioned above, for some topologies an efficient decontamination is easy to perform and finding the optimal number of agents is a trivial task. In general however, the problem is NP-complete. The ring is an extreme case, where decontamination is trivial. Adding extra chords to the ring highly complicates the decontamination problem and clearly only two agents are not sufficient anymore to clean the network. One interesting question that we address in this paper is whether, with the addition of the chords, the minimum number of agents is still constant, or it depends on the size of the network, or on the structure of the chords. Interestingly we show that, when the longest chord is not too long, none of these hypothesis is correct; in fact, the smallest number of agents needed for the decontamination solely depends on the *length* of the longest chord. After we derive the lower bound on the number of agents we describe and analyse two optimal strategies for two variations of the model.

More precisely, let $C(\langle d_1 = 1, d_2, ..., d_k \rangle)$ be a chordal ring network with n nodes and link structure $\langle d_1 = 1, d_2, ..., d_k \rangle$, where $d_i < d_{i+1}$ and $d_k \leq \lfloor \frac{n}{2} \rfloor$. We first show that, when $4 \leq d_k \leq \sqrt{n}$, the minimum number of agents required is $2 \cdot d_k$ in the *Local model*, and it is $2 \cdot d_k + 1$ in the *Visibility model*. In a similar fashion, we also derive a lower bound for the torus topology (which was unknown) in both models. We then describe agent-optimal decontamination algorithms for the chordal ring in the two models, and simple agent-optimal strategies for Tori.

One of our goals is to understand the power of visibility by determining whether such an assumption can indeed improve the performances of solutions to the problem, and how. In this respect, we have observed that with our strategies the visibility assumption allows to drastically decrease both the time and the number of moves in tori and chordal rings (provided that the longest chord does not exceed $\sqrt{n}$).

2. Lower Bounds for Decontamination

2.1. Chordal Rings

A circulant graph with n nodes $x_0, x_1, ..., x_{n-1}$ and link structure $\langle d_1, d_2, ..., d_k \rangle$, $d_i < d_{i+1}$, and $d_k \leq \lfloor \frac{n}{2} \rfloor$, is a graph where each node x_i is adjacent to all the nodes $x_{(i+d_j) \bmod n}$ and $x_{(i-d_j) \bmod n}$ for $1 \leq j \leq k$. A *chordal ring* is a circulant graph with $d_1 = 1$, i.e., it is an augmented ring, and will be denoted by $C(\langle d_1 = 1, d_2, ..., d_k \rangle)$. The links of the chordal ring are labeled with chordal sense of direction, i.e., asso-

ciated to link (x_i, x_j) at x_i is the distance $(j-i) \, mod \, n$ between x_i and x_j along the ring connection.

As the topology is fully symmetric, we can assume that the agents start from any node: the homebase. Let $x_0, \ldots x_{n-1}$ be the nodes of the external ring and w.l.g, let x_0 be the homebase.

In the following we consider the chordal ring as arranged in rows of size d_k where the last node of a row is connected to the first node of the following row and the last node is connected to the first. Depending on the size of the chordal ring, the last row could be incomplete. Observe that in this "matrix", going down a column corresponds to using the longest chord d_k.

The following lower bound holds in chordal rings where, in the arrangement described above, the number of columns is not smaller than the number of rows; in other words, we assume that $d_k \leq \frac{n}{d_k}$.

Theorem 1 *Any solution of the contiguous decontamination problem in a chordal ring $C(\langle d_1 = 1, d_2, ..., d_k \rangle)$ with n nodes and and $4 \leq d_k \leq \sqrt{n}$, requires at least $2 \cdot d_k$ searchers.*

Proof.

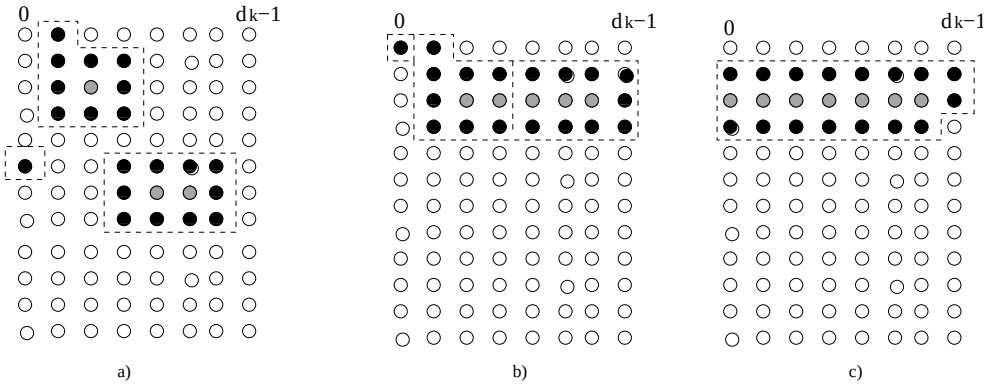

Figure 1: *a*) clean area formed by non attached blocks; *b*), *c*) clean area formed by attached blocks in $P(\langle 1, d_k \rangle)$. Black nodes are guarded, white nodes are contaminated, grey nodes are clean.

Let f be the number of clean (or guarded) nodes. Let us consider a subgraph $P(\langle 1, d_k \rangle) = (V, E')$ of the chordal ring $C = (V, E)$ $(E' \subseteq E)$ containing only the chords at distance 1 and d_k. Let us observe the placement of the f clean nodes in P. By definition, we know that at any point in time the clean nodes must be connected in C; however, in P they might form disconnected blocks. Clearly, the "perimeter" (i.e., the clean nodes of the blocks in contact with contaminated nodes) of these blocks in P must be guarded to avoid recontamination from the neighbouring contaminated nodes (through chords 1 and d_k). First notice that, following a simple geometric reasoning, the number of agents X needed to cover the perimeters of the clean blocks is greater than or equal to the number that it would

be required if these blocks were to be attached (i.e., forming a single block) (see Figure 1).

Second, it is easy to show that the perimeter of a single block is minimized when it is as close as possible to a rhombus as shown in Figure 2.

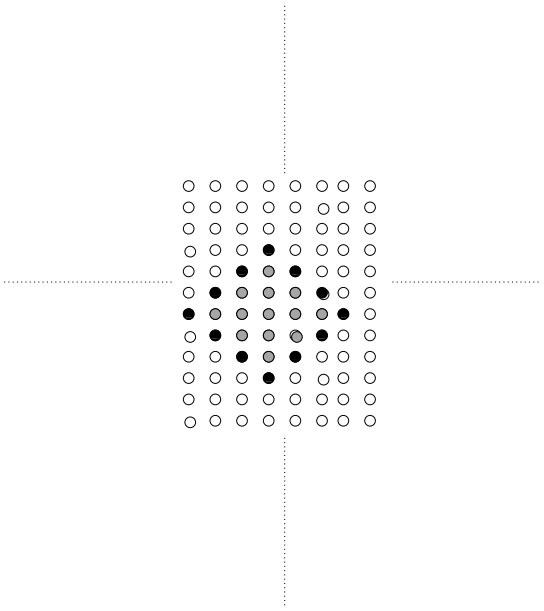

Figure 2: Block that, for a given area, has minimum perimeter.

Let us now compute the number of agents X needed to cover the perimeter of the rhombus; for any other possible shape, X will be a larger number.

Given a rhombus of side l, the perimeter is $4l-4$ and the area (composed of cleaned or guarded nodes) is $f = l^2+(l-1)^2$. Thus, the number of agents X needed to cover the perimeter of any shape is at least $4l-4$. From $f = l^2+(l-1)^2$ we derive that $l = \frac{1+\sqrt{2f-1}}{2}$; substituting l we have $X \geq 2\sqrt{2f-1}-2$.

Consider now a moment during the cleaning when there are $f = l^2+(l-1)^2 \geq \frac{d_k^2+2d_k+2}{2}$ clean nodes. This holds as long as $n \geq d_k^2 \geq \frac{d_k^2+2d_k+2}{2}$, i.e., $d_k \geq 4$.

From $f \geq \frac{d_k^2+2d_k+2}{2}$, it follows that the number of required agents is $X \geq 2\sqrt{2(\frac{d_k^2+2d_k+2}{2})}-1-2 = 2d_k$.

□

The previous lower bound holds for any decontamination algorithm (so, in particular it holds for the *Visibility Model*); we now show that, when the agents *do not* have visibility, the bound is actually higher since an additional agent is required to perform the decontamination.

Theorem 2 *In the* Local model*, any solution of the contiguous decontamination problem in a chordal ring $C(\langle d_1 = 1, d_2, ..., d_k\rangle)$ with n nodes and $4 \leq d_k \leq \sqrt{n}$, requires at least $2 \cdot d_k + 1$ searchers.*

Proof. In Theorem 1 we have shown that, at some point, the "perimeter" of the clean area is $2d_k$. Let us assume that the decontamination has reached such a point; i.e., there are $2 \cdot d_k$ agents covering the perimeter and let us assume that this number of agents suffices for the decontamination. Since n is big enough compared to $2d_k$ ($n \geq \sqrt{d_k}$), it is easy to see that, regardless of the shape of the clean area, at least one of the agents (say agent a) has more than one contaminated neighbour. At least one agent has to move to continue the cleaning. Since the agents have only local knowledge and cannot communicate with each other, let an adversary choose agent a for the next movement. We have a contradiction because the clean area will now be contaminated.

□

2.2. Tori

Following the same lines as the arguments of Theorems 1 and 2, we obtain a lower bound also for the torus, which was not known. The torus is a very common interconnection network; it is the product of two rings and it can be seen as a grid where the last node of a row is connected to the first of the same row, and last node of a column is connected to the first of the same column. Let $T(h,k)$ denote a torus with h rows and k columns, we have that:

Theorem 3 *Any solution of the contiguous decontamination problem in a torus $T(h,k)$ with $h,k \geq 4$ requires at least $2 \cdot min\{h,k\}$ searchers; in the* Local model *it requires at least $2 \cdot min\{h,k\} + 1$ searchers.*

Proof. The argument follows the same lines as the argument of Theorems 1 and 2. In this case, however, the "matrix" representation corresponds to the actual torus and there are no additional chords. The minimum perimeter - occurring if the shape of the block were to be a rhombus - would analogously be $4\sqrt{f} - 4$ and the area $f = l^2 + (l-1)^2$, thus the number of required agents would be $X \geq 2\sqrt{2f-1} - 2$. Assume $z = \min\{h,k\}$ and consider a moment during the cleaning when there are $f = l^2 + (l-1)^2 \geq \frac{z^2+2z+2}{2}$ clean nodes. This holds as long as $h \times k \geq z^2 \geq \frac{z^2+2z+2}{2}$, i.e., $z = \min\{h,k\} \geq 4$.

From $f \geq \frac{z^2+2z+2}{2}$, it follows that the number of required agents is $X \geq 2\sqrt{2(\frac{z^2+2z+2}{2}) - 1} - 2 = 2z = 2\min\{h,k\}$, thus the number of required agents is at least $2\min\{h,k\}$.

If there is no visibility (*Local model*) an additional agent is required following the same reasoning of Theorem 2. □

3. Optimal Decontamination Strategies

3.1. Chordal Rings

In this section we propose two algorithms respectively for the decontamination of a chordal ring of size n in the *Local* and in the *Visibility* models.

In the following discussion, we will assume that operation among indices are

modulo n (i.e., $i + d_j$ means $(i + d_j)$ mod n). We call *clockwise neighbors* (respectively, *counterclockwise neighbors*) of x_i, the set of neighbors $\{x_{i+1}, x_{i+d_2}, ..., x_{i+d_k}\}$ (respectively, $\{x_{i-1}, x_{i-d_2}, ..., x_{i-d_k}\}$). We call *window* of size s is a sequence of s nodes in consecutive clockwise positions along the ring.

3.1.1. Decontamination in the Local Model

The first strategy we present is for the *Local model*, where a special agent is designated as a coordinator. The main idea is that the searching agents are coordinated by this special agent that, moving back and forth, allows them to visit all nodes and safely protects the system from recontamination.

Cleaning strategy. In this strategy we employ $2d_k$ identical agents and a coordinator. We assume that the coordinator can communicate with an agent when they reside on the same node.

The cleaning must be preceded by a deployment stage after which the agents have to occupy $2d_k$ consecutive nodes. The simplest way to deploy the agents is for the coordinator to lead them to their respective starting position by moving along the external ring. During the deployment the agents also clean the $2d_k$ nodes and no recontamination occurs (i.e., the deployment is monotone). After the deployment, nodes x_0 to x_{2d_k-1} are guarded by one agent each, and the coordinator moves to node x_{d_k}, which is then guarded by two agents. Now the cleaning stage can start. During the cleaning stage, nodes x_0 to x_{d_k-1} are still 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.

Algorithm 1 CLEAN WITH A COORDINATOR**.**

Initially, nodes $x_0 \dots x_{2d_k-1}$ are guarded by an agent each and the coordinator is at x_{d_k}.

1.1 The cleaning starts at x_{d_k} and proceeds in the clockwise direction until x_{n-1-d_k} is reached. Let x_i $(d_k \leq i \leq n-1-d_k)$ be the node with an agent and the coordinator on it. The two agents move along link d_k of x_i, and when they both arrive to node x_{i+d_k}, one is left to guard x_{i+d_k} and the coordinator first goes back to x_i and then moves along link 1 of x_i to arrive at node x_{i+1}. The agent on any node x_j where $i \neq j$ has to wait on x_j for the coordinator to arrive.

1.2 When two agents are on node x_{n-d_k}, one agent terminates and the coordinator goes to notify all the agents on node x_{n-d_k+1} to x_{d_k-1} to terminate.

Note that in Step 1.2., when two agents are on node x_{n-d_k} all nodes from x_{n-d_k} to x_n are guarded. Note also that as the system is asynchronous, in the different phases the coordinator has to make sure every agent reaches its position before the new step can start: starting the new step without insuring that the agent has reached the prescribed position would result in a possible recontamination.

Correctness and Complexity. We first prove that our cleaning strategy is correct; i.e., that all nodes will be cleaned and that once a node has been cleaned, it will never be recontaminated.

Theorem 4 *Algorithm 1 cleans all the nodes of the chordal ring and a clean node will never be recontaminated.*

Proof. We first prove that a clean node will not be recontaminated. By induction. The cleaning starts at node x_{d_k}. Except for the neighbor x_{2d_k} of x_{d_k}, which is not guarded by an agent yet, all the other neighbors are guarded. By the strategy, the two agents on it move to x_{2d_k}, the only contaminated neighbor. Then one agent is left to guard this neighbor and the other goes back to x_{d_k}. So all neighbors of x_{d_k} are guarded; when the agent from x_{2d_k} comes back to x_{d_k} and then moves to x_{d_k+1}, node x_{d_k} becomes clean and no recontamination can occur.
Assume nodes x_{d_k} to x_{i-1} where $d_k \leq i-1 \leq n-2-d_k$ are clean and the cleaning is at node x_i. We show that x_i becomes clean and no recontamination can occur. By the first step of the strategy and by the induction hypothesis, node x_0 to x_{i-1} are clean or guarded. So the counterclockwise neighbors of x_i are all either guarded or clean. Moreover, we know nodes from x_i to x_{i-1+d_k} are guarded. So the clockwise neighbors of x_i are guarded except for x_{i+d_k}. By the strategy, when the two agents move to the only contaminated neighbor x_{i+d_k}, no recontamination can occur. One agent is left to guard x_{i+d_k}. So all neighbors of x_i are either guarded or clean; when the agent goes back to x_i and then moves to x_{i+1}, no agent is left to guard x_i and x_i becomes clean. No recontamination can occur to x_i.
At step 1.2, when the agent from x_{n-1-d_k} arrives at x_{n-d_k}, all the nodes from x_{n-d_k} to x_{d_k-1} are guarded and the others are clean from step 1.1. There is no contaminated node anymore and it is impossible to recontaminate a clean node. Since a clean node will never be recontaminated, by the cleaning strategy, after step 1.2 all agents terminate and all the nodes become clean. □

We now compute the number of moves performed by the agents during the cleaning.

Theorem 5 *The total number of moves performed by the agents is* $4n - 6d_k - 1$.

Proof. For cleaning a node x_i, four moves are performed by the agents. It takes one move for each of the two agents to arrive to x_{i+d_k}, one move back to x_i and then one move to x_{i+1}. So totally, $4(n-2d_k)$ moves are performed in the cleaning stage. It takes $2d_k - 1$ moves for the agent from x_{n-d_k} to notify agents on x_{n-d_k} to x_{d_k-1} to terminate. So totally, the number of moves for the entire process is $4(n-2d_k) + 2d_k - 1 = 4n - 6d_k - 1$. □
We now consider the ideal time complexity of the cleaning strategy. We recall that ideal time assumes that it takes one unit of time for an agent to traverse an edge.

Theorem 6 *The cleaning strategy takes* $3n - 4d_k - 1$ *time units.*

Proof. The cleaning process is carried out sequentially by the coordinator agent on each node. The time required is then equal to the number of moves of the coordinator, which is $3(n - 2d_k) + 2d_k - 1$. So totally, it takes $3(n - 2d_k) + 2d_k - 1$ $= 3n - 4d_k - 1$. □

Finally, it directly follows from the strategy that:

Theorem 7 *Our strategy employs* $2d_k + 1$ *agents.*

3.1.2. Decontamination with Visibility

In this section we consider the decontamination problem in the *Visibility model*: we assume that an agent located at a node can "see" whether its neighboring nodes are clean or guarded or contaminated. This capability could be easily achieved if the agents have communication power and send a message (e.g., a single bit) to their neighbouring nodes after cleaning a node or when guarding a node. The interesting aspect of this model is that this extra capability enables agents to correctly act without the need of being coordinated. We also assume that agents have distinct Ids, otherwise they cannot perform any meaningful computation starting from the same homebase (symmetry could not be broken).

Cleaning strategy. The idea of the algorithm is quite simple and all the agents follow the same local rule: as soon as an agent sees that all its neighbours are clean except for one, it moves there. Before starting the algorithm, however, a deployment phase is needed during which the agents move to occupy $2d_k$ consecutive nodes. As for the *Local model*, we want that they deploy in a monotone way, that is during the deployment they also start cleaning the nodes without allowing any recontamination. For example, we can assume they move along the external ring occupying one node each like in the deploy for the *Local model*.

Algorithm 2 CLEAN WITH VISIBILITY**.**

Initially all agents are in x_0, and they start the deployment to occupy $x_{n-(d_k-1)}, \ldots x_{d_k}$.
- When an agent on a node x_i "sees" that node x_i has only one contaminated neighbor, the agent moves to clean the contaminated neighbor; when the agent "sees" that all the neighbors are clean or guarded, it terminates; otherwise, it waits on the node.

Notice that the execution of the cleaning algorithm could actually start before the deployment is completed. In fact, as soon as an agent sees it has only one contaminated neighbour, it can start the cleaning.

Figure 3 shows a possible execution of the algorithm in a portion of a chordal ring $C(\langle 1, 2, 4\rangle)$. Figure 3 *a*) shows the guarded nodes (in black) after the deployment phase. At this point, the nodes indicated in the figure can independently and concurrently start the cleaning phase moving to occupy their only contaminated

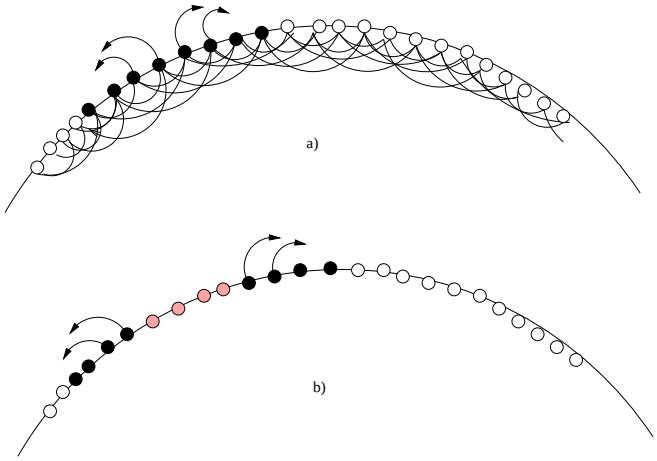

Figure 3: A chordal ring $C(\langle 1, 2, 4\rangle)$. *a*) The agents are deployed and four of them (the ones pointed by an arrow) could move to clean the neighbour. *b*) Four agents have moved to clean their only contaminated neighbour and four more (the ones pointed by an arrow) could now move.

neighbour. Figure 3 *b*) shows the new state of the network if they all move (the arrows indicate the nodes where the agents could move to clean their neighbour).

Correctness and Complexity. We first prove that our strategy is correct, i.e., that the network is clean and once a node has been cleaned, it will never be recontaminated.

Lemma 1 *In a chordal ring network with n nodes and link structure* $\langle d_1 = 1, d_2, ..., d_k\rangle$, $d_i < d_{i+1}$, *and* $d_k \leq \lfloor \frac{n}{2} \rfloor$, *within a window of size* $2d_k$, *there are at most* $2(d_k - d_{k-1})$ *nodes which have only one neighbor outside this window. These nodes are consecutive along the external ring.*

Proof. Let us arbitrarily pick a window W of size $2d_k$ and mark the first node to be $x_{n-(d_k-1)}$ and the the last node to be x_{d_k}. We can also see W as two windows W_1 and W_2 of size d_k and such that $W = W_1 \bigcup W_2$. Window W_1 covers nodes from $x_{n-(d_k-1)}$ to x_0 and W_2 covers nodes from x_1 to x_{d_k}. Inside W_1, nodes from $x_{n-d_k+d_{k-1}+1}$ to x_0 have only one neighbor outside $W = W_1 \bigcup W_2$ in the counterclock direction. Similarly, inside W_2, nodes from x_1 to $x_{d_k-d_{k-1}}$ have only one neighbor outside of W in the clockwise direction. So totally, there are at most $2(d_k - d_{k-1})$ nodes having only one neighbor outside of W. □

Observe now that if $n < 4d_k - 2d_{k-1}$ some of the nodes that had to be outside the window W, lie on the opposite subwindow (i.e., from W_1 to W_2 and vice versa). Thus, we may obtain the following:

Corollary 1 *If* $n \geq 4d_k - 2d_{k-1}$, *within a window of size* $2d_k$ *there are exactly* $2(d_k - d_{k-1})$ *consecutive nodes which have only one neighbor outside this window.*

We now prove the correctness of the strategy.

Theorem 8 *Algorithm 2 cleans all the nodes of the chorded ring and a clean node will never be recontaminated.*

Proof. We have to prove both that the strategy considers all the nodes of the chordal and that no recontamination occurs.
Let us first prove by induction that there is no recontamination, i.e., a clean node may never be a neighbour of a contaminated node. To prove this we also prove something else, i.e., that if an agent has left a node x_i reached during the deployment stage to move to a new node for the cleaning stage, then x_i is not a neighbour of a contaminated node. This second property is included in the first but considering it apart will help us during the proof.
We start the induction considering the initial state during which all the agent are placed on the homebase. There is only one guarded node and all the other nodes are contaminated, thus the two property hold. Consider the system after a set of moves during which both properties held and let us now consider a set of agents that execute a new move. There are two possible cases depending on the fact that an agent that moves is in the deployment or in the cleaning stage. If the agent is in the deployment stage, this holds because the deployment strategy is monotone. So the case we are really interested in is the one in which a node moves during the cleaning stage. Since no recontamination has occurred previously, the move is safe because an agent moves only if all the neighbours but one are clean or guarded and it moves towards the contaminated node. Thus, no recontamination occurs.
Let us now prove that the strategy cleans all the nodes, i.e., all nodes are eventually reached. After the deployment stage all nodes in a window W of size $2d_k$ starting from $x_{n-(d_k-1)}$ and ending in x_{d_k} are guarded and thus cleaned by an agent. By lemma 1 we know that there are $2(d_k-d_{k-1})$ consecutive nodes which have only one neighbor outside window W, thus agents guarding these nodes will eventually move to guard (d_k-d_{k-1}) nodes outside W on the left and (d_k-d_{k-1}) nodes on the right forming a clean window W' of size $4d_k-2d_{k-1}$. There are now other $2(d_k-d_{k-1})$ nodes of W' which have only one neighbor outside W' (this time the nodes are not consecutive, but they form two consecutive blocks of size (d_k-d_{k-1}), thus this procedure enlarges the window up to when the whole chordal ring is covered. □
We now consider the ideal time complexity of the cleaning strategy.

Theorem 9 *The cleaning strategy takes at most* $\left\lceil \frac{n-2d_k}{2(d_k-d_{k-1})} \right\rceil$ *time units.*

Proof. For simplicity, let us assume that the cleaning starts after the deployment is completed (as we have seen before some agents could actually start earlier). We also assume that $n > 2d_k$, otherwise nothing has to be done after the deployment phase. We can now divide our computation after the deployment phase into different sub-phases during which, using lemma 1, blocks of $2(d_k-d_{k-1})$ nodes are cleaned until the whole network is cleaned. Thus, in at most $\left\lceil \frac{n-2d_k}{2(d_k-d_{k-1})} \right\rceil$ time steps blocks of $2(d_k-d_{k-1})$ nodes are cleaned (one in each time step) up to when the whole network is cleaned. □

Theorem 10 *The total number of moves performed by the agents during the cleaning strategy is* $n-2d_k$*, which is optimal.*

Proof. By definition of the strategy, during the cleaning stage an agent moves to a node only if it "sees" that is it the only contaminated neighbour, thus it only moves to contaminated nodes. Moreover from Theorem 8 we know that the strategy is correct and all the contaminated nodes will be cleaned. Thus all contaminated nodes will be visited exactly once, and after the cleaning phase there are $n - 2d_k$ of them (the remaining nodes are guarded). Notice that this number is optimal since to clean $n - 2d_k$ contaminated nodes, at least $n - 2d_k$ moves are necessary. □

Finally, it directly follows from the strategy that:

Theorem 11 *Our strategy employs $2d_k$ agents.*

3.1.3. Comparison

Our results on the cost of the cleaning strategy are summarized in Table 1.

Chordal Ring	**Agents**	**Time**	**Moves**
Local	$2d_k + 1$ $(\star)$	$3n - 4d_k - 1$	$4n - 6d_k - 1$
Visibility	$2d_k$ $(\star)$	$\left\lceil \frac{n-2d_k}{2(d_k - d_{k-1})} \right\rceil$	$n - 2d_k$ $(\star)$

Table 1: Results for the Chordal Ring. The $(\star)$ indicates an optimal bound.

We can observe that with our strategies the visibility assumption allows us to drastically decrease the time and move complexities. In particular, the strategies for the *Visibility model* are optimal both in terms of number of agents and in terms of number of moves; as for the time complexity, visibility allows some concurrency. For example, in the case of the chordal ring the level of concurrency depends on the distance between the longest chord and the second longest. The higher the distance, the higher the concurrency is, and thus the improvement in time complexity.

3.2. Torus

In this section we briefly mention the decontamination strategies that match the lower bounds for the Tori. The algorithms are very similar to the ones for the mesh topology described in [10]. Without loss of generality, let us assume that the number of rows h is smaller than the number of columns k. The idea is to deploy the agents to cover two consecutive columns and then keep one column of agents to guard from decontamination and have the other column move along the torus. In the *Local model* the movement of the agents must be synchronized by a coordinator so to avoid recontamination, in the *Visibility model* the agents can safely move autonomously.

The cleaning is very similar to the one for the mesh, the only difference is that in the mesh it was not required to have a column of agents shielding one part of the network from recontamination since the cleaning could start from a border of the

mesh. The move and time complexities for the Local Model are the same as in [10], and are reported in Table 2 (where we consider the complexity of the cleaning only). For the Visibility Model the time complexity is half of the one for the mesh topology because here the two columns of agents proceeds independently and autonomously to clean the torus in both directions, thus halving the complexity.

Torus	**Agents**	**Time**	**Moves**
Local	$2h+1$ $(\star)$	$hk-2h$	$2hk-4h-1$
Visibility	$2h$ $(\star)$	$\lceil \frac{k-2}{2} \rceil$ $(\star)$	$hk-2h$ $(\star)$

Table 2: Results for the 2-dimensional Torus with dimensions h, k, $h \leq k$. The $(\star)$ indicates an optimal bound.

This cleaning strategy can be seen as a generalization of an optimal strategy for the ring (which is a 1-dimensional torus), where two agents would move in opposite directions).

Notice that, in the Visibility model all three complexity measures are optimal. In fact: 1) the number of agents is shown to be optimal in Theorem 3; 2) the ideal time is optimal because it corresponds to the number of contaminated nodes (after the agents are deployed on two consecutive columns) divided by the number of available agents ($\lceil \frac{k-2}{2} \rceil = \lceil \frac{kh-2h}{2h} \rceil$), which means that at each time step every agent is cleaning a node; 3) the number of moves is obviously optimal because $hk-2h$ is the number of contaminated nodes (after the deployment on two columns).

Interestingly, the simple strategy of this section, which would work in the particular case of the ring, can be generalized to d-dimensional tori.

Multi-dimensional Tori. Let $T(h_1, h_2, \ldots, h_d)$ be a d-dimensional torus and let $h_1 \leq h_2 \leq \ldots \leq h_d$. Let $R(n)$ denote a ring with n nodes.

A d-dimensional torus $T(h_1, h_2, \ldots, h_d)$ is the product of a torus $T(h_1, h_2, \ldots, h_{d-1})$ with a ring $R(h_d)$.

In other words, $T(h_1, h_2, \ldots, h_d)$ can be seen as a replica of h_d $(d-1)$-dimensional tori $T_1(h_1, h_2, \ldots, h_{d-1}), \ldots, T_{h_d}(h_1, h_2, \ldots, h_{d-1})$ connected in parallel cycles (see the example of dimension 3 in Figure 4). Each node in a replica of $T_i(h_1, h_2, \ldots, h_{d-1})$ is then connected to two corresponding nodes in the two "adjacent" replicas $T_{i-1}(h_1, h_2, \ldots, h_{d-1})$ and T_{i+1} $(h_1, h_2, \ldots, h_{d-1})$.

Let N be the number of nodes in the torus and $H = h_1 \cdot h_2 \cdot \ldots \cdot h_{d-1}$ (i.e., $H = \frac{N}{h_d}$). It can be easily verified that $2 \cdot H + 1$ agents can clean $T(h_1, h_2, \ldots, h_d)$ ($2 \cdot H$ in the Visibility Model). The idea is that the agents are first deployed to cover two consecutive replicas $T_i(h_1, h_2, \ldots, h_{d-1})$ and $T_{i+1}(h_1, h_2, \ldots, h_{d-1})$ (as in the 3-dimensional example of Figure 4) and then they are moved to clean, one at a time, the successive replicas - analogously to the case of the two-dimensional torus. In

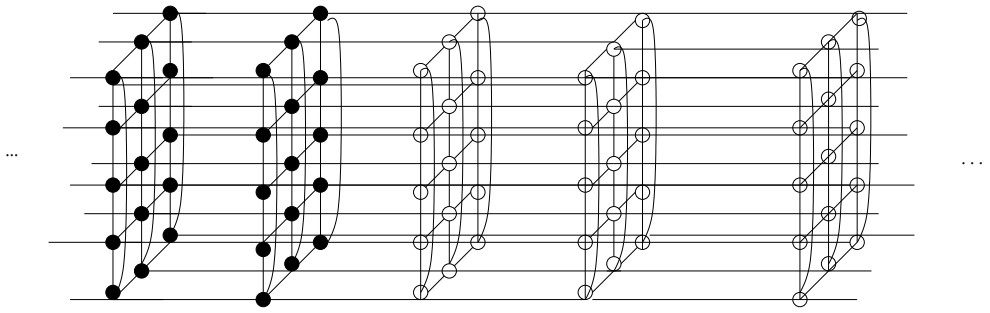

Figure 4: A portion of a 3-dimensional torus.

the Visibility Model the agents move autonomously and concurrently, while in the Local Model they must be leaded by the coordinator. The resulting complexities are reported in Table 3.

d-dim Torus	**Agents**	**Time**	**Moves**
Local	$2\frac{N}{h_d}+1$	$N-2\frac{N}{h_d}$	$2N-4\frac{N}{h_d}-1$
Visibility	$2\frac{N}{h_d}$	$(\lceil h_d-2\rceil)/2$	$N-2\frac{N}{h_d}$

Table 3: Results for a d-dimensional Torus $T(h_1, h_2, \ldots, h_d)$.

The two-dimensional torus can be seen as a particular case; in fact $T(h,k)$ is the product of two rings $R(h)$ and $R(k)$ and can be decontaminated with $2h+1$ agents ($2h$ in the Visibility Model). It is interesting to observe that the number of agents employed by this strategy grows very quickly when increasing the number of dimensions: in the ring 2 agents are enough, in the two dimensional torus $2\sqrt{N}$ are employed, in a d-dimensional torus their number grows to ${}^d\sqrt{N^{d-1}}$.

Although this number can appear quite high, we conjecture it is optimal. In fact the lower bound for the 2-dimensional Torus of Section 2.2 can probably be extended to the multi-dimensional Tori. We leave as an open problem to verify the conjecture or disprove it.

4. Remarks and Open Problems

In this paper we have considered the problem of decontamination in chordal rings. We have determined lower bounds on the number of agents required and decontamination strategies for two variations of the model.

Following similar reasonings as the ones of the lower bounds for the chordal ring, we have obtained lower bounds also for the Torus (which were not known); these

bounds can be easily matched by simple cleaning strategies.

We have observed that the visibility assumption allows us to drastically decrease the time and move complexities in both topologies. The improvements hold when the longest chord of the chordal ring is not too long ($d_k \leq \sqrt{n}$), otherwise our strategies are not as efficient, and the lower bound is not valid. Consider, for example, the case of $C(\langle 1, 8 \rangle)$ with 24 nodes. It is easy to see that the decontamination can be done with 6 agents only (placed in two consecutive "columns" in the matrix representation), while our bound would prescribe 16. The determination of the minimum number of agents needed when $d_k > \sqrt{n}$ and a matching strategy is an interesting open problem.

Another interesting problem would be to study a trade-off between time and number of agents in the Local Model. For example, if we add one extra agent to our strategy with coordinator, that is we employ $2d_k + 2$ agents, we can reduce the cleaning process time by half. The extra agent can move in opposite direction to clean the contaminated nodes thus two agents would be cleaning concurrently, in the two directions of the external ring.

Finally, we have left as an open problem the verification of our conjecture that the lower bounds for the 2-dimensional torus can be generalized to multi dimensional tori making the general algorithm of Section 3.2 optimal.

Acknowledgements

Work partially supported by NSERC, by Dr. Flocchini's University Research Chair, and by Abstract Interpretation Design and Application MIUR COFIN Project.

References

1. M. Asaka, S. Okazawa, A. Taguchi and S. Goto. A method of tracing intruders by use of mobile agents. *Proceedings of the 9th Annual Conference of the Internet Society (INET)*, San Jose, California, U.S.A., 1999.
2. H. Attiya, J. van Leewen, N. Santoro and S. Zaks. Efficient elections in chordal ring networks. *Algorithmica*, 4:437-446, 1989.
3. L. Barrière, P. Flocchini, P. Fraigniaud and N. Santoro. Capture of an intruder by mobile agents. *Proceedings of the 14-th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, Winnipeg, Manitoba, Canada, 200-209, 2002.
4. L. Barrière, P. Fraigniaud, N. Santoro and D.M. Thilikos. Searching is not jumping. *Proceedings of the 29th International Workshop on Graph Theoretic Concepts in Computer Science (WG)*, Elspeet, the Netherlands, Springer Verlag, Lecture Notes in Computer Science 2880, 34-45, 2003.
5. J-C Bermond, F. Comellas and D.F. Hsu. Distributed loop computer networks: a survey. *Journal of Parallel and Distributed Computing*, 24:2-10, 1995.
6. L. Blin, P. Fraigniaud, N. Nisse and S. Vial. Distributed Chasing of Network Intruders by Mobile Agents. *Proceedings of the 13th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, Chester, UK, 70-84, 2006.
7. R. Breish. An intuitive approach to speleotopology. *Southwestern cavers*, **VI** (5), 72-28, 1967.

8. J.A. Ellis, I.H. Sudborough and J.S. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113: 50-79, 1994.

9. P. Flocchini, M.J. Huang and F.L. Luccio. Contiguous search in the hypercube for capturing an intruder. *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Denver, Colorado, U.S.A.

10. P. Flocchini, F.L. Luccio and L.X. Song. Size Optimal Strategies for Capturing an Intruder in Mesh Networks. *Proceedings of the International Conference on Communications in Computing (CIC)*, Las Vegas, USA, 200-206, 2005.

11. P. Flocchini, A. Nayak and A. Shulz. Cleaning an arbitrary regular network with mobile agents *Proceedings of the 2nd International Conference on Distributed Computing & Internet Technology (ICDCIT)*, Bhubaneswar, India, 132-142, 2005.

12. F.V. Fomin, D.M. Thilikos and I. Todinca. Connected graph searching in outerplanar graphs. *Proceedings of the 7th Int. Colloquium on Graph Theory (ICGT)*, Electronic Notes in Discrete Mathematics **22**, 213-216, 2005.

13. N. Foukia, J.G.Hulaas and J. Harms. Intrusion Detection with Mobile Agents. *11th Annual Internet Society Conference (INET)*, 2001.

14. L. M. Kirousis and C. H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47:205-218, 1986.

15. D. Krizanc and F.L. Luccio. Boolean Routing on Chordal Rings. *Proceedings of the 2nd Colloquium on Structural Information and Communication Complexity (SIROCCO)*, Olympia, Greece, June 12-14, 1995.

16. A. Lapaugh. Recontamination does not help to search a graph. *Journal of the ACM*, 40(2): 224-245, 1993.

17. B. Mans. Optimal distributed algorithms in unlabeled tori and chordal rings. *Journal of Parallel and Distributed Computing*, 46(1): 80-90, 1997.

18. N. Megiddo, S. Hakimi, M. Garey, D. Johnson and C. Papadimitriou. The complexity of searching a graph. *Journal of the ACM*, 35(1): 18-44, 1988.

19. B. Monien and I.H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theoretical Computer Science*, **58**, 209-229, 1988.

20. T. Parson. Pursuit-evasion problem on a graph. *Theory and applications in graphs*, Lecture Notes in Mathematics, Springer-Verlag, 426-441, 1976.

21. T. Parson. The search number of a connected graph. *9-th Southeastern Conference on Combinatorics, Graph Theory and Computing*, Utilitas Mathematica, 549-554, 1978.

22. S. Peng, M. Ko, C. Ho, T. Hsu and C. Tang. Graph searching on some subclasses of chordal graphs. *Algorithmica*, 27: 395-426, 2000.

23. E.H. Spafford and D. Zamboni. Intrusion detection using autonomous agents, *Computer Networks*, 34(4), 547-570, 2000.

24. P. Yospanya, B. Laekhanukit, D. Nanongkai and J. Fakcharoenphol. Detecting and cleaning intruders in sensor networks. *Proceedings of the 8th National Computer Science and Engineering Conference*, 2004.