Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

More agents may decrease global work: A case in butterfly decontamination ☆

Fabrizio Luccio *, Linda Pagli

Dipartimento di Informatica, Università di Pisa, Italy

ARTICLE INFO

Article history:
Received 30 October 2015
Received in revised form 16 September 2016
Accepted 29 September 2016
Available online xxxx

Keywords:
Network decontamination
Distributed protocol
Agent
Work
Butterfly
Graph search

ABSTRACT

This paper is a contribution to network decontamination with a view inherited from parallel processing. At the beginning some or all the vertices may be contaminated. The network is visited by a group of decontaminating agents. When a decontaminated vertex is left by the agents, it can be re-contaminated only if the number of infected neighbors exceeds a certain immunity threshold m. The main goal of the studies in this line is to minimize the number $\mathcal{A}$ of agents needed to do the job and, for a minimum team, to minimize the number $\mathcal{M}$ of agent moves. Instead of $\mathcal{M}$ we consider the number $\mathcal{T}$ of steps (i.e. parallel moves) as a measure of time, and evaluate the quality of a protocol on the basis of its *work* $\mathcal{W} = \mathcal{AT}$. Taking butterfly networks as an example, we compare different protocols and show that, for some values of m, a larger team of agents may require smaller work.

1. Introduction and computational model

A team of *agents* travel along a network represented as a connected undirected graph G, with the task of decontaminating infected vertices. As the distribution of the infection in G is unknown, we consider the worst case in which all vertices are initially infected.

The agents enter the network at a same vertex called *homebase*. At any given time a vertex may contain any number of agents. At any given step several agents may traverse the same edge. If infected, a vertex v is decontaminated by an incoming agent that makes it *clean*. At a given time vertex v is *grey* if infected, *black* if contains one or more agents, and *white* if it is clean but no agent is there (i.e., it has been previously decontaminated).

We assume that a white vertex is re-contaminated and becomes grey if m or more of its neighbors are grey, where m is an *immunity threshold* fixed beforehand. The system is then said to have *m-immunity*. The presence of an agent guarantees that a black vertex cannot be re-contaminated regardless of the number of infected neighbors. A basic assumption is that when an agent moves from vertex u to vertex v on edge (u, v), it protects u from possible re-contamination by v. Our study draws rules and concepts from distributed algorithms [29] and from parallel algorithms [17].

Our aim is to study possible trade-offs between number of agents and total decontamination time, particularly if an increase of the number of agents by a factor h results in a decrease of time by more than h: a phenomenon studied in parallel algorithms in connection with accumulation of cache space beyond a certain threshold, or in special computational

☆ This work was supported in part by the Italian Ministry of Education, University, and Research (MIUR) under PRIN 2012C4E3KT national research project "AMANDA" Algorithmics for MAssive and Networked DAta.

* Corresponding author.
E-mail addresses: luccio@di.unipi.it (F. Luccio), pagli@di.unipi.it (L. Pagli).

http://dx.doi.org/10.1016/j.tcs.2016.09.027
0304-3975/

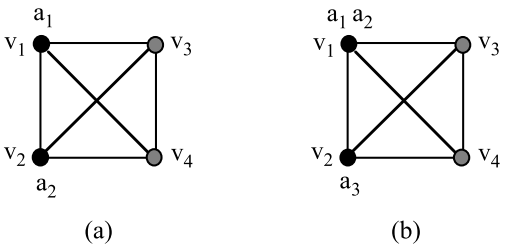

Fig. 1. A simple network with four vertices. (a) For $m = 2$ two agents a_1, a_2 can decontaminate the network. (b) For $m = 1$ three agents are necessary.

conditions, see [13,22]. In our model, however, the situation is completely different. We take butterfly networks as a working example and refer to *monotone* protocols where clean vertices are never re-contaminated [4,19].

1.1. Framework and related work

The studies on network decontamination on arbitrary graphs are based on the seminal paper [25] followed by many others, e.g. see [1,15]. A major line of research assumes that a white vertex is re-contaminated if one or more of its neighbors is contaminated, i.e. $m = 1$. This is done in [2,15,26] for arbitrary graphs, and a vast literature is directed to particular network topologies, e.g. see [23,21,9]. The problem is also known as "monotone connected graph search" and "intruder capture" [5,16,12].

As the assumption $m = 1$ may be too restrictive depending on the context, threshold rules for establishing a significant value of m have been studied as in [27]. In particular in [23] a vertex is re-contaminated according to a local value of m that equals the majority of its neighbors. The model has been extended for the first time in [21], to any value of m treated as a parameter, and further extended in [11] for tori and tries.

An important line of research deals with identifying hostile vertices [24]. Important studies are directed to the decontamination from black holes (BH) that reside in unknown vertices and delete any incoming agent or data without leaving a trace. The phenomenon has been widely studied in different contexts, see for example [18,10,7], and the excellent review [28]. More dangerous are black viruses (BV) residing in unknown vertices [6]. Including mobility aspects into the problem, a black virus destroys the arriving agents and spreads the infection to all neighboring sites.

In our problem the intruder is harmful for the network sites, but not for the agents; then the two problems are different. Still the techniques used for BH and BV might be useful to derive lower and upper bounds valid in our case. No BH and BV studies are known for butterflies to the best of our knowledge.

1.2. The computational model

Several points must be properly fixed to define our computational model.

1. *The system is organized in steps in which the agents move asynchronously and independently of one another.* At each step all the agents may compute and move to a neighboring vertex. We do not pretend these operations to be instantaneous or to take the same time, but all operations started in step i must be completed by step $i + 1$. In particular, for the agents moving at step i no assumption is made on when they reach the target vertex provided they are all in their next destination before step $i + 1$ begins.

To understand the rationale behind the rule, consider the simple network of Fig. 1. In part (a) two agents a_1, a_2 are respectively located in vertices v_1, v_2. For $m = 2$ the two agents can decontaminate vertices v_3, v_4 in one step, simply moving along edges (v_1, v_3) and (v_2, v_4) respectively. For $m = 1$, instead the two agents are unable to decontaminate the network. In fact if for example a_1 moves along the edge (v_1, v_3), the now white vertex v_1 is exposed to re-contamination in the same step from the grey vertex v_4 before agent a_2 has reached v_4 moving along the edge (v_2, v_4). The reader may immediately verify that also all the other possible moves would not help to decontaminate the network. For $m = 1$ three agents a_1, a_2, a_3 are necessary, as in part (b) of the figure where a_1, a_2 can be sent to v_3, v_4 respectively, while a_3 stands in v_2 to avoid re-contamination of v_1.

2. *To decide when one step is completed and the next step can be initiated two strategies are possible.* 2.1) In a distributed version of the protocols each agent broadcasts a message to all the others stating that its operations in the current step are completed. 2.2) In a synchronous centralized version of the protocols a maximum operating time for each agent is known, then a global clock accessible by all agents decides the timing.

3. *At each step, each agent computes its next destination.* The edges of the network are locally labeled. Each agent must know the number of the current step and has its own instructions on which edge to follow at that step. There is no need to give a complete map of the network to each agent.

4. *We direct the present study to the relation between number of agents and total number of steps.* Letting $\mathcal{A}$ be the number of agents in a decontaminating team, the main goal studied in distributed systems is to determine the smallest value of $\mathcal{A}$

Please cite this article in press as: F. Luccio, L. Pagli, More agents may decrease global work: A case in butterfly decontamination, Theoret. Comput. Sci. (2016), http://dx.doi.org/10.1016/j.tcs.2016.09.027