European Journal of Combinatorics 48 (2015) 88–99

Contents lists available at ScienceDirect

European Journal of Combinatorics

journal homepage: www.elsevier.com/locate/ejc

The maximum time of 2-neighbour bootstrap percolation: Algorithmic aspects☆

Fabrício Benevides [a], Victor Campos [b], Mitre C. Dourado [c], Rudini M. Sampaio [b], Ana Silva [a]

[a] Departamento de Matemática, Universidade Federal do Ceará, Fortaleza, Brazil
[b] Departamento de Computação, Universidade Federal do Ceará, Fortaleza, Brazil
[c] Instituto de Matemática, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brazil

ARTICLE INFO

Article history:
Available online 5 March 2015

ABSTRACT

In 2-neighbourhood bootstrap percolation on a graph G, an infection spreads according to the following deterministic rule: infected vertices of G remain infected forever and in consecutive rounds healthy vertices with at least 2 already infected neighbours become infected. Percolation occurs if eventually every vertex is infected. In this paper, we are interested to calculate the maximal time $t(G)$ the process can take, in terms of the number of times the interval function is applied, to eventually infect the entire vertex set. We prove that the problem of deciding if $t(G) \geq k$ is NP-complete for: (a) fixed $k \geq 4$; (b) bipartite graphs and fixed $k \geq 7$; and (c) planar graphs. Moreover, we obtain linear and polynomial time algorithms for trees and chordal graphs, respectively.

1. Introduction

We consider a problem in which an infection spreads over the vertices of a connected simple graph G following a deterministic spreading rule in such a way that an infected vertex will remain infected forever. Given a set $S \subseteq V(G)$ of initially infected vertices, we build a sequence $S_0 = S, S_1, S_2, \ldots$ in which S_{i+1} is obtained from S_i using such a spreading rule.

☆ This research was supported by Capes (STIC-AmSud 040/2012) and CNPq (Universal 478744/2013-7).
E-mail addresses: fabricio@mat.ufc.br (F. Benevides), campos@lia.ufc.br (V. Campos), mitre@dcc.ufrj.br (M.C. Dourado), rudini@lia.ufc.br (R.M. Sampaio), anasilva@mat.ufc.br (A. Silva).

http://dx.doi.org/10.1016/j.ejc.2015.02.012
0195-6698/

Under r-neighbour bootstrap percolation on a graph G, the spreading rule is a threshold rule in which S_{i+1} is obtained from S_i by adding to it the vertices of G which have at least r neighbours in S_i. We say that a set S_0 percolates G (or that S_0 is a percolating set of G) if eventually every vertex of G becomes infected, that is, there exists a t such that $S_t = V(G)$. In that case, we define $t_r(S)$ as the minimum t such that $S_t = V(G)$. And define, the *percolation time of* G as $t_r(G) = \max\{t_r(S) : S \text{ percolates } G\}$. In this paper, we shall focus on the case where $r = 2$ and in such a case we omit the subscript of the functions $t_r(S)$ and $t_r(G)$.

Bootstrap percolation was introduced by Chalupa, Leath and Reich [14] as a model for certain interacting particle systems in physics. Since then it has found applications in clustering phenomena, sandpiles [23], and many other areas of statistical physics, as well as in neural networks [1] and computer science [19].

There are two broad classes of questions one can ask about bootstrap percolation. The first, and the most extensively studied, is what happens when the initial configuration S_0 is chosen randomly under some probability distribution? One would like to know how likely percolation is to occur, and if it does occur, how long it takes.

The answer to the first of these questions is now well understood for various graphs. An interesting case is the one of the lattice graph $[n]^d$, in which d is fixed and n tends to infinity, since the probability of percolation under the r-neighbour model displays a sharp threshold between no percolation with high probability and percolation with high probability. The existence of thresholds in the strong sense just described first appeared in papers by Holroyd, Balogh, Bollobás, Duminil-Copin and Morris [25,5,4]. Sharp thresholds have also been proved for the hypercube (Balogh and Bollobás [3], and Balogh, Bollobás and Morris [6]). There are also very recent results due to Bollobás, Holmgren, Smith and Uzzell [10], about the time percolation takes on the discrete torus $\mathbb{T}_n^d = (\mathbb{Z}/n\mathbb{Z})^d$ for a randomly chosen set S_0.

The second broad class of questions is the one of extremal questions. For example, what is the smallest or largest size of a percolating set with a given property? The size of the smallest percolating set in the d-dimensional grid $[n]^d$ was studied by Pete and a summary can be found in [7]. Morris [28] and Riedl [30] studied the maximum size of minimal percolating sets on the square grid $[n]^2$ and the hypercube $\{0, 1\}^d$, respectively, answering a question posed by Bollobás. However, the problem of finding the smallest percolating set is NP-hard even on subgraphs of the square grid [2] and it is APX-hard even for bipartite graphs with maximum degree four [17]. Moreover, it is hard [15] to approximate within a ratio $O(2^{\log^{1-\varepsilon} n})$, for any $\varepsilon > 0$, unless $NP \subseteq DTIME(n^{polylog(n)})$.

Another type of question is: What is the minimum or maximum time that percolation can take, given that S_0 satisfies certain properties? Recently, Przykucki [29] determined the precise value of the maximum percolation time on the hypercube $2^{[n]}$ as a function of n, and Benevides and Przykucki [9,8] have similar results for the square grid $[n]^2$, also answering a question posed by Bollobás. In particular, they have a polynomial time dynamic programming algorithm to compute the maximum percolation time on rectangular grids [9].

In this paper, we investigate the computational complexity of $t(G)$, motivated by these recent results on the maximum percolation time. Here, we consider the decision version of the maximum time percolation problem, as stated below.

PERCOLATION TIME
Input: A graph G and an integer k.
Question: Is $t(G) \geq k$?

In Section 2, we prove that PERCOLATION TIME is NP-complete even when the input is restricted to certain cases. More precisely, we prove it is NP-complete for: general graphs even if $k \geq 4$ is fixed, that is, k is not part of the input; bipartite graphs and any fixed $k \geq 7$; and planar graphs and a given k, that is, k is part of the input. In Section 3, we provide polynomial time algorithms for general graphs when $k \leq 2$ and for chordal graphs, and a linear time algorithm for trees.

1.1. Related works and some notation

It is interesting to notice that infection problems appear in the literature under many different names and were studied by researches of various fields. A recent source on related topics is [16]. The