The Maximum Time of 2-neighbour Bootstrap Percolation in Grid Graphs and Parametrized Results

Thiago Marcilon(✉) and Rudini Sampaio

Dept. Computação, Universidade Federal do Ceará, Fortaleza, Brazil
{thiagomarcilon,rudini}@lia.ufc.br

Abstract. In 2-neighborhood 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 two already infected neighbors become infected. Percolation occurs if eventually every vertex is infected. The maximum time $t(G)$ is the maximum number of rounds needed to eventually infect the entire vertex set. In 2013, it was proved by Benevides et al. [10] that $t(G)$ is NP-hard for planar graphs and that deciding whether $t(G) \geq k$ is polynomial time solvable for $k \leq 2$, but is NP-complete for $k \geq 4$. They left two open problems about the complexity for $k = 3$ and for planar bipartite graphs. In 2014, we solved the first problem [24]. In this paper, we solve the second one by proving that $t(G)$ is NP-complete even in grid graphs with maximum degree 3. We also prove that $t(G)$ is polynomial time solvable for solid grid graphs with maximum degree 3. Moreover, we prove that the percolation time problem is fixed parameter tractable with respect to the parameter treewidth $+ k$ and maximum degree $+ k$. Finally, we obtain polynomial time algorithms for several graphs with few P_4's, as cographs and P_4-sparse graphs.

Keywords: 2-Neighbor bootstrap percolation · Maximum percolation time · Grid graph · Fixed parameter tractability · Treewidth

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 $S_0, S_1, S_2, \ldots$ in which $S_0 = S$ and S_{i+1} is obtained from S_i using such spreading rule.

Under r-neighbor 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 neighbors in S_i. We say that a set S infects a vertex v at time i if $v \in S_i \backslash S_{i-1}$. Let, for any set of vertices S and vertex v of G, $t_r(G, S, v)$ be the minimum t such that v belongs to S_t or, if there is no t such that v

E.W. Mayr (Ed.): WG 2015, LNCS 9224, pp. 169–185, 2016.
DOI: 10.1007/978-3-662-53174-7_13

belongs to S_t, then $t_r(G, S, v) = \infty$. Also, we say that a set S_0 infects 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 $S_t = V(G)$. If S is a percolating set of G, then we define $t_r(G, S)$ as the minimum t such that $S_t = V(G)$. Also, define the *percolation time of* G as $t_r(G) = \max\{t_r(G, S) : S \text{ is a percolating set of } G\}$. In this paper, we shall focus on the case where $r = 2$ and in such case we omit the subscript of the notations $t_r(G, S)$ and $t_r(G)$. Also, from the notation $t(G, S)$, when the parameter G is clear from context, it will be omitted.

Bootstrap percolation was introduced by Chalupa, Leath and Reich [15] as a model for certain interacting particle systems in physics. Since then it has found applications in clustering phenomena, sandpiles [20], 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? For example, vertices are included in S_0 independently with some fixed probability p. One would like to know how likely percolation is to occur, and if it does occur, how long it takes. The answer to these questions is now well understood for various types of graphs [5,7,8,13,22].

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 [6]. Morris [25] and Riedl [28] 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 [16] 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 [27] determined the precise value of the maximum percolation time on the hypercube $2^{[n]}$ as a function of n, and Benevides and Przykucki [11,12] 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 [11].

Here, we consider the decision version of the Percolation Time Problem, as stated below.

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

In 2013, Benevides et al. [10], among other results, proved that the Percolation Time Problem is polynomial time solvable for $k \leq 2$, but is NP-complete for $k \geq 4$ and, when restricted to bipartite graphs, it is NP-complete for $k \geq 7$.

Moreover, it was proved that the Percolation Time Problem is NP-complete for planar graphs. They left three open questions about the complexity for $k = 3$ in general graphs, the complexity for $3 \leq k \leq 6$ in bipartite graphs and the complexity for planar bipartite graphs.

In 2014, the first and the second questions were solved [24]: it was proved that the Percolation Time Problem is $O(mn^5)$-time solvable for $k = 3$ in general graphs and, when restricted to bipartite graphs, it is $O(mn^3)$-time solvable for $k = 3$, it is $O(m^2n^9)$-time solvable for $k = 4$ and it is NP-complete for $k \geq 5$.

In this paper, we solve the third question of [10]. We prove that the Percolation Time Problem is NP-complete for planar bipartite graphs. In fact, we prove a stronger result: the NP-completeness for grid graphs, which are induced subgraphs of grids, with maximum degree 3.

There are NP-hard problems in grid graphs which are polynomial time solvable for solid grid graphs. For example, the Hamiltonian cycle problem is NP-complete for grid graphs [23], but it is polynomial time solvable for solid grid graphs [30]. Motivated by the work of [11] for rectangular grids, we obtain in this paper a polynomial time algorithm for solid grid graphs with maximum degree 3.

Finally, we prove several complexity results for $t(G)$ in graphs with bounded maximum degree and bounded treewidth, some of which implies fixed parameter tractable algorithms for the Percolation Time Problem. Moreover, we obtain polynomial time algorithms for $(q, q-4)$-graphs, for any fixed q, which are the graphs such that every subset of at most q vertices induces at most $q-4$ P_4's. Cographs and P_4-sparse graphs are exactly the $(4,0)$-graphs and the $(5,1)$-graphs, respectively. These algorithms are fixed parameter tractable on the parameter q.

2 Percolation Time Problem in Grid Graphs with $\Delta = 3$

In this section, we prove that the Percolation Time Problem is NP-complete in grid graphs with maximum degree $\Delta = 3$. We also show that, when the graph is a grid graph with $\Delta = 3$ and $k = O(\log n)$, the Percolation Time Problem can be solved in polynomial time. But, first, let us define a S-infection path and, then, prove two lemmas that will be useful in the proofs.

Let $t(G, S, v)$ be the time where S infects v in G or, if S does not infect v in G, then let $t(G, S, v) = \infty$. Let S be a percolating set. A path $P = v_0, v_1, \ldots, v_n$ is a $\{S, G\}$-infection path if and only if, for all $0 \leq i \leq n-1$, $t(G, S, v_i) < t(G, S, v_{i+1})$. In both notations, when the parameter G is clear from context, it will be omitted

Note that if $t(G, S, v) = k$ then there is a $\{S, G\}$-infection path $v_1, \ldots, v_k = v$, where each vertex v_i is such that $t(G, S, v_i) = i$. The next lemma, which is valid for every graph with maximum degree 3, is the main technical lemma of this section. Due to space restrictions, its proof will be omitted.

Lemma 1. *Let G be any connected graph with $\Delta = 3$ and k a non-negative integer. Then, $t(G) \geq k$ if and only if G has an induced path P where either all*

vertices in $V(P)$ have degree 3 and $|E(P)| \geq 2k-2$ or all vertices in $V(P)$ have degree 3, except for one of his extremities, which has degree 2, and $|E(P)| \geq k-1$.

Before proving the NP-completeness result of this section, we use Lemma 1 to show that the Percolation Time Problem is polynomial time solvable for $k = O(\log n)$ when the graph has maximum degree 3.

Theorem 1. *If G is a graph with maximum degree 3, then deciding whether $t(G) \geq k$ can be done in polynomial time for $k = O(\log n)$.*

Proof (sketch of the proof). We can decide whether $t(G) \geq k$ by making use of a modified version of the depth-first search. This version of the depth-first search with maximum search depth l traverses all paths with $l + 1$ vertices starting from some vertex v. For each $v \in V(G)$, we will run this version of the depth-first search starting in v. If $d(v) = 2$, we run the modified depth-first search with maximum search depth $k - 1$. If $d(v) = 3$, we run the modified depth-first search with maximum search depth $2k - 2$. If there is a vertex v such that the depth-first search that starts in v finds a path that is an induced path, reaches the maximum depth and passes only by vertices of degree 3, except maybe for v, then, by Lemma 1, $t(G) \geq k$. Otherwise, $t(G) < k$.

Now, let us show that this algorithm runs in polynomial time. For each vertex v in G, there is at most $3 \cdot 2^{k-2}$ paths of length k in G that starts in v, for any k. In this case, since $k = O(\log n)$, there are at most $3 \cdot 2^{O(\log n)} = 3n^{O(1)}$ paths of length k in G that starts in v, which is a polynomial on n. Therefore, since the depth-first search traverses all paths with length equals to the maximum depth once for each vertex in $V(G)$, then our algorithm runs in time $O(n \cdot 2^k)$, which is polynomial in n since $k = O(\log n)$. ■

Thus, if $k = O(\log n)$, we can find whether $t(G) \geq k$ in polynomial time for every graph G with $\Delta(G) = 3$. However, the following theorem states that the Percolation Time Problem is NP-complete, even when G is restricted to be a grid graph with $\Delta = 3$.

Theorem 2. *Deciding whether $t(G) \geq k$ is NP-complete when the input G is restricted to be a grid graph with $\Delta(G) \leq 3$.*

Proof (sketch of the proof). Clearly, the problem is in NP. To prove that the problem is also NP-hard, we obtained a reduction from the Longest Path problem with input restricted to be grid graphs with maximum degree 3. The Longest Path problem with input restricted to be grid graphs with maximum degree 3 is a NP-complete problem because the Hamiltonian Path Problem with input restricted to be grid graphs with maximum degree 3 is also NP-complete [26] and there is a trivial reduction from the Hamiltonian Path Problem to the Longest Path problem that does not change the input graph: G has an Hamiltonian Path if and only if G has a path greater or equal to $n - 1$.

Consider the following reduction from the Longest Path Problem's instance (G, k) where G is restricted to be a grid graph with maximum degree 3 to the

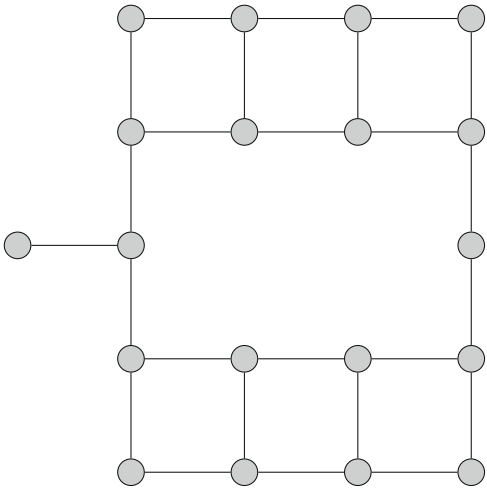

Fig. 1. Grid graph with $\Delta = 3$

Percolation Time Problem's instance $(G', 3k + 2)$ where G' is also a grid graph with maximum degree 3: Multiply the scale of the grid G by three. Each edge in G becomes a path in G' with 4 vertices where the vertices at the extremities are vertices that were originally in G. Let us call an *original vertex* the vertices in G' that were originally in G. After that, for each original vertex v, if $d(v) < 3$, add to G' $3 - d(v)$ vertices in any free position in the grid adjacent to v and link them to v. Thus, after we do that, each original vertex has degree 3 in G'. Henceforth, if a vertex in G' is not an original vertex at this point, then we will call it an *auxiliary vertex*. Note that each auxiliary vertex is adjacent to exactly one original vertex and each original vertex is adjacent to 3 auxiliary vertices.

After that, for each auxiliary vertex v, add a new vertex adjacent to v in the following manner: if the original neighbor of v is located above it, add a vertex adjacent to v at his left position, if there is not one there already, and link it to v. If the original neighbor of v is located below it, add a vertex adjacent to v at his right position, if there is not one there already, and link it to v. If the original neighbor of v is located at his left position, add a vertex adjacent to v at the position below it, if there is not one there already, and link it to v. If the original neighbor of v is located at his right position, add a vertex adjacent to v at the position above it, if there is not one there already, and link it to v. The Fig. 2 show how a 4×4 block will look like in G' before and after we add these vertices.

Then, for each auxiliary vertex v, if $d(v) = 2$, add a new vertex adjacent to v in the following position: if the original neighbor of v is at the left position of v, add a vertex adjacent to v at his right position. If the original neighbor of v is at the right position of v, add a vertex adjacent to v at his left position. If the original neighbor of v is below v, add a vertex adjacent to v above v. If the original neighbor of v is above v, add a vertex adjacent to v below v.

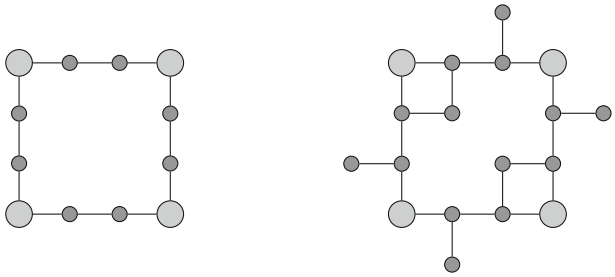

Fig. 2. 4×4 block before and after addition of the auxiliary vertices' neighbors

Thus, the construction of G' is finished. Since G is a grid graph and, every time an original vertex and an auxiliary vertex are in adjacent positions in the grid, they are linked, then G' is a grid graph.

Note that all original and auxiliary vertices have degree 3 and they are the only vertices that have degree 3. Let us call *corner vertex* all the vertices that have degree 2 in G'. Also, note that, for each corner vertex, there is exactly one original vertex at distance 2 of it, and, for each original vertex, there is exactly one corner vertex at distance 2 of it. This happens because each original vertex has degree exactly three. Let f be the bijective function that maps each original vertex to the corner vertex that is at distance 2 of it. The Fig. 3 shows the reduction applied to the grid graph of the Fig. 1. It is worth noting that, in G', a path P that has only original and auxiliary vertices and starts with an original vertex, it has length multiple of 3 if and only if it ends in an original vertex. Also, for each 3 consecutive vertices of this path, two are auxiliary vertices and one is an original vertex.

Now, let us prove that G has a path of length $\geq k$ if and only if $t(G') \geq 3k+2$.

Suppose that G is a grid graph with maximum degree 3 that has a path of length $\geq k$. Let us prove that $t(G') \geq 3k+2$. Since G has a path P of length $\geq k$, we have that G' has an induced path P of length $\geq 3k$ that passes by the same path that P passes, which implies that P passes only by original and auxiliary vertices. Note that, when an auxiliary vertex is in P, his auxiliary neighbor is also in P.

Let v and v' be the extremities of P and $f(v') = q'$. Since v is an original vertex, then let w be any auxiliary neighbor of v that is not in $V(P)$. Note that all neighbors of w, except v, are not in $V(P)$. Let r be the vertex auxiliary neighbor of v' that is in P and let P' be the induced path that we obtain from P by adding w, by removing v' and by adding all vertices in any smallest path between r and q', excluding r, that only have vertices not adjacent to w and passes only by original and auxiliary vertices. Since P is an induced path and we removed one vertex and added only one induced path that has either 1 or 3 vertices to create P', we have that P' is an induced path with length $\geq 3k+1$ where all of its vertices have degree 3, except for q', which has degree 2. Therefore, by Lemma 1, we have that $t(G') \geq 3k+2$.

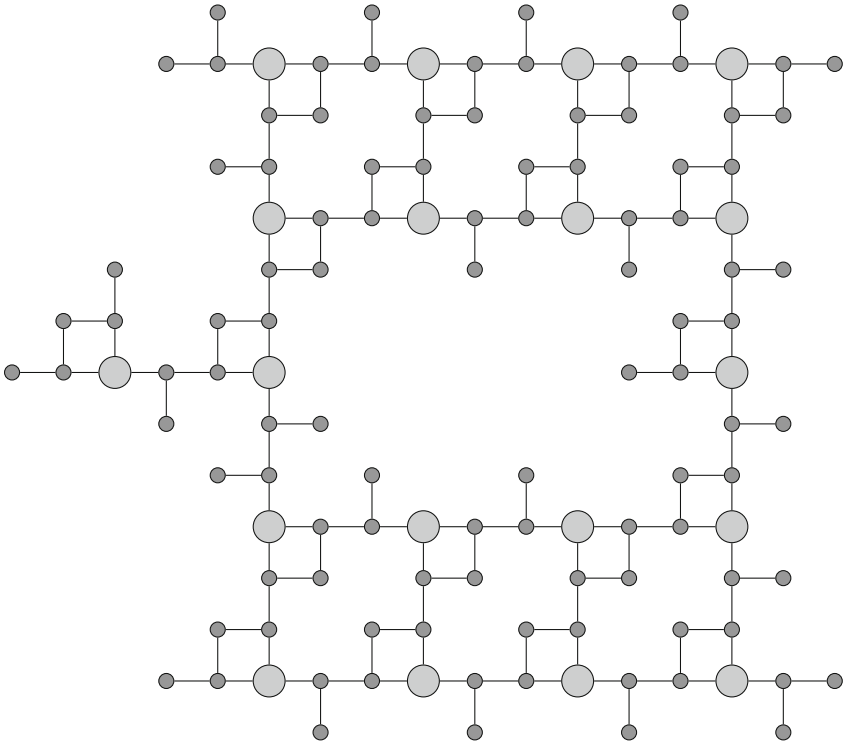

Fig. 3. Grid graph resulting from the reduction applied to the grid graph of the Fig. 1.

Now, suppose that G is a grid graph with maximum degree 3 such that, when we apply the reduction to G to create G', we have that $t(G') \geq 3k + 2$. Let us prove that G has a path of length $\geq k$. Since $t(G') \geq 3k+2$, applying the Lemma 1, we have that G' has an induced path P where either all vertices in $V(P)$ have degree 3 and $|E(P)| \geq 6k + 2$ or all vertices in $V(P)$ have degree 3, except for one of his extremities, which has degree 2, and $|E(P)| \geq 3k + 1$.

Firstly, suppose that G' has an induced path P where all vertices in $V(P)$ have degree 3 and $|E(P)| \geq 6k + 2$. Since, the only vertices that have degree 3 are the original and auxiliary vertices and for each three consecutive vertices in P there is one original vertex and two auxiliary vertices, it is easy to see that P has at least $k + 1$ original vertices and, thus, there is a path in G of length at least k.

Finally, suppose that G' has an induced path P where all vertices in $V(P)$ have degree 3, except for one of his extremities, which has degree 2, and $|E(P)| \geq 3k+1$. It is enough to analyze the case $|E(P)| = 3k+1$ because, if $|E(P)| > 3k+1$, any subpath of P of length $3k + 1$ that starts at the extremity of P that have degree 2 is an induced path where all of his vertices have degree 3, except for one of his extremities, which has degree 2, and has length $3k + 1$. So, let us say that P starts in the vertex that has degree 2. Since the only vertices that have

degree 2 are corner vertices, then P starts with a corner vertex. Let q be that corner vertex, let $q' = f^{-1}(q)$ and let v be the other extremity of P.

Suppose that P passes by q'. Since P is an induced path, then q' is the third vertex of P. Since q and q' are at distance 2 of each other and $|E(P)| = 3k + 1$, then v is an auxiliary vertex which his neighbor that is an original vertex, say v', is not in P. Let us append v' to P and remove all vertices between q and q', including q and excluding q'. So, since P starts at q', an original vertex, ends in v', another original vertex, and has length $3k$, then there is a path in G of length greater or equal to k.

Now, suppose that P does not pass by q'. Since $|E(P)| = 3k + 1$, then v is an auxiliary vertex which his neighbor that is an original vertex, say v', is in P. Let us remove q, appending q' in his place, and v from P. Thus, since P starts at q', an original vertex, ends in v', another original vertex, and has length $3k$, then there is a path in G of length greater or equal to k. ■

3 Percolation Time Problem in Solid Grid Graphs with $\Delta = 3$

A solid grid graph is a grid graph in which all of his bounded faces have area one. There are NP-hard problems in grid graphs that are polynomial time solvable for solid grid graphs. For example, the hamiltonian cycle problem is NP-hard for grid graphs [23], but, in 1997, it was proved that it is polynomial time solvable for solid grid graphs [30]. Motivated by the work of [11] for rectangular grids, we obtain in this section a polynomial time algorithm for solid grid graphs with maximum degree 3. However, the Percolation Time Problem for solid grid graphs with maximum degree 4 is still open.

Theorem 3. *For any solid grid graph G with $\Delta = 3$, $t(G)$ can be found in $O(n^2)$ time.*

Proof (sketch of the proof). If a solid grid graph has $\Delta = 3$, then, since it is $K_{1,4}$-free, it becomes a graph formed only by ladders L_k, which are grid graphs with dimensions $2 \times k$, and by paths, possibly linking these ladders by the vertices in their extremities. Let the extremities of a ladder be the four vertices that have only two neighbors in the ladder and let all the other vertices be the vertices internal to the ladder. In Fig. 4, there is an example of solid grid graph with $\Delta = 3$.

To find the percolation time of G, according to Lemma 1, it is enough to find both the longest induced path that starts with a degree 2 vertex and, then, passes only by vertices with degree 3, and the longest induced path that passes only by vertices with degree 3. Thus, since all of G's bounded faces have area one and, besides the ladders, G is composed only by paths, the only difficulty to calculate $t(G)$ is to find the longest induced paths in the ladders between any two extremities that passes only by vertices with degree 3. However, one can easily calculate the longest induced paths between any two extremities of a

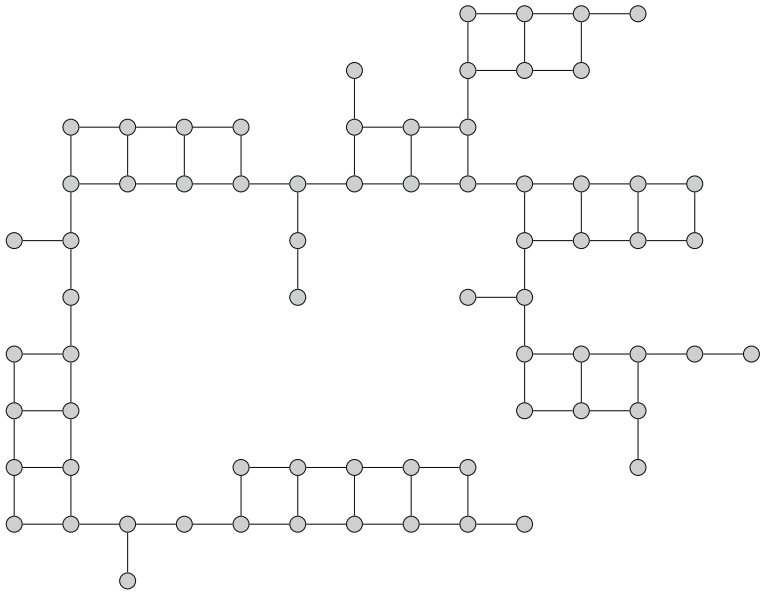

Fig. 4. A solid grid graph with maximum degree 3.

ladder L_k: if the two extremities are neighbors, the length of the longest induced paths between them is 1; if the two extremities are at distance $k-1$, the length of the longest induced paths between them is $(k-t)+2\cdot\lfloor(k-t+1)/4\rfloor-1+t$; if the two extremities are at distance k, the length of the longest induced paths between them is $(k-t)+2\cdot\lfloor(k-t-1)/4\rfloor+t$, where t is how many of the two others extremities have degree 2.

So, first, we will transform G in a weighted graph G' where G' is the same graph as G only with all the ladders replaced by weighted K_4's, where the weight of an edge between two vertices in a K_4 represents the length of a longest induced path between the corresponding extremities of the ladders in G that passes only by vertices with degree 3. The weight of all the other edges is 1. The Fig. 5 represents the transformation applied in the graph of the Fig. 4. Note that there is exactly one induced path between any two vertices in G', which length is equal to the longest induced path between the same two vertices in G. It is not hard to see that this transformation from G to G' can be done in linear time.

In Algorithm 1, let $w(u,v)$ be the weight of the edge (u,v). The algorithm, for each vertex $u \in V(G')$ such that $d_G(u) \geq 2$, calls the function `LongestInducedPathFrom`, which do a Depth-First Search to find the longest induced path in G' from u such that the last vertex is the only vertex in the path that either has degree ≤ 2, besides perhaps the vertex u, or is in the neighborhood of a vertex already in the path, and, then, it subtracts the length of the found path by one. This is necessary because a longest induced path from some vertex u in G can end in a vertex v internal to a ladder, but internal vertices of

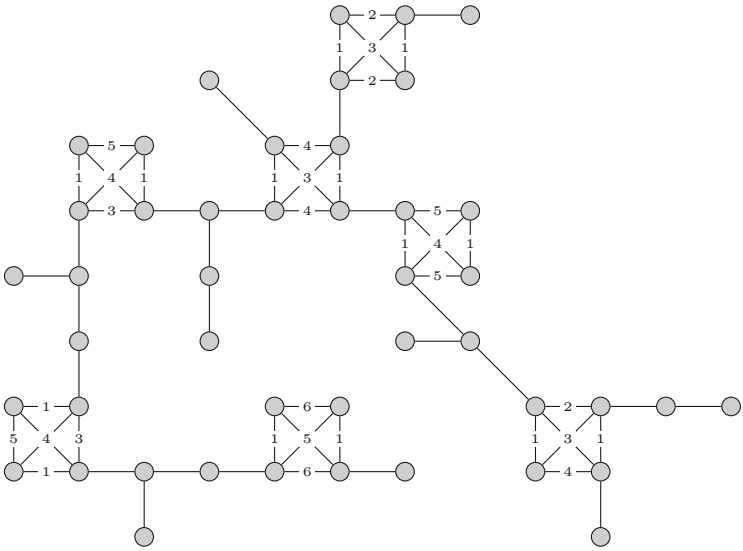

Fig. 5. The resulting graph of the transformation applied to the graph in the Fig. 4.

a ladder are not represented in G'. However, if that happens, since all vertices internal to a ladder have degree 3, then v must be adjacent to some vertex at the extremity of the ladder that has degree 2.

In any case, the resulting length corresponds to the length of the longest induced path in G beginning in u, which last vertex has degree 3 and is not in the neighborhood of any vertex already in the path. Then, it compares all these values, according to the Lemma 1, to find $t(G)$.

Since there is only one induced path between any two vertices in G', we have that the recursive function `LongestInducedPathFrom` takes the same time as any Depth-First Search algorithm. Thus, since $m = O(n)$, the Function `LongestInducedPathFrom` takes $O(n)$ time. Therefore, the Algorithm 1 takes $O(n^2)$ time. ■

4 Percolation Time Problem in Graphs with Bounded Max Degree $\Delta \geq 4$

In Sect. 2 (Theorem 1), we proved that the Percolation Time Problem is polynomial time solvable in grid graphs with $\Delta(G) \leq 3$ for $k = O(\log n)$. In this section, we prove that this not happen for general graphs with fixed maximum degree $\Delta \geq 4$, unless P = NP.

Theorem 4. *Let $\Delta \geq 4$ be fixed. Deciding whether $t(G) \geq k$ is NP-complete for graphs with bounded maximum degree Δ and any $k \geq \log_{\Delta-2} n$.*

Algorithm 1. Algorithm that finds $t(G)$ for any solid grid graph G with $\Delta = 3$

```
Algorithm MaximumTimeSolidGridΔ3(G)
    G' = Transform(G)
    maxPercTime = 0
    forall the u ∈ V(G') such that d_G(u) ≥ 2 do
        if d_G(u) = 2 then
            percTimeU = LongestInducedPathFrom(G', u)+1
        else
            percTimeU = ⌊(LongestInducedPathFrom(G', u)+2)/2⌋
        if maxPercTime < percTimeU then
            maxPercTime = percTimeU
    return maxPercTime
```

Proof (sketch of the proof). We obtain a reduction from the variation of the **SAT** problem where each clause has exactly three literals, each variable appears in at most four clauses [29].

Given M clauses $\mathcal{C} = \{C_1, \ldots, C_M\}$ on N variables $X = \{x_1, \ldots, x_N\}$ as an instance of **SAT**, we denote the three literals of C_i by $\ell_{i,1}$, $\ell_{i,2}$ and $\ell_{i,3}$. Note, since any variable can only appear in at most 4 clauses, that $N/3 \leq M \leq 4N/3$. So, first, let us show how to construct a graph G with maximum degree Δ. For each clause C_i of $\mathcal{C}$, add to G a gadget as the one in Fig. 6. Then, for each pair of literals $\ell_{i,a}, \ell_{j,b}$ such that one is the negation of the other, add a vertex $y_{(i,a),(j,b)}$ and link it to either $w^A_{i,a}$ or $w^B_{i,a}$ and either $w^A_{j,b}$ or $w^B_{j,b}$, but always respecting the restriction where each one of the vertices $w^A_{i,a}$, $w^B_{i,a}$, $w^A_{j,b}$ and $w^B_{j,b}$ can only have degree at most 4. Since each variable can appear in at most 4 clauses, it is always possible to do that. Let Y be the set of all vertices $y_{(i,a),(j,b)}$ created this way. Notice that $y = |Y| \leq 4N$.

Then, add the maximum full $(\Delta - 2)$-ary tree with root z such that the number of leaves is less than y and, then, add a new vertex adjacent vertex of degree one to each vertex in the tree. Let T be the tree that we have just added and t be the set of vertices that we just added. After that, link each leaf to at least one and at most $\Delta - 2$ vertices in Y. Thus, each vertex in the tree has degree Δ, except for the leaves, which have degree at most Δ, and z, which has degree $\Delta - 1$. Note that $t = 2 \cdot |V(T)| \leq 2 \cdot \frac{(\Delta-2)y-1}{\Delta-3} \leq 16N$ and $height(T) = \lceil \log_{\Delta-2} y \rceil$.

Let $c = (\Delta-2)^{-8}$, $\alpha = xc$, where $x = \max(41 + \lceil c \rceil, \lceil 1/c \rceil)$, $r = \lceil \log_{\Delta-2}(4N\alpha) \rceil - \lceil \log_{\Delta-2} y \rceil$ and $\beta = 4N\alpha/c - (39M + y + t + 2 + 2r)$. With some work, one can prove that both r and β are non-negative integers.

If $r > 0$, add a path with r vertices, link one end to z and let q be the other end. Also, add a new neighbor of degree one to each vertex that belongs to the path. Let P' be the set of vertices in this path and his neighbors of degree one. If $r = 0$, let $q = z$.

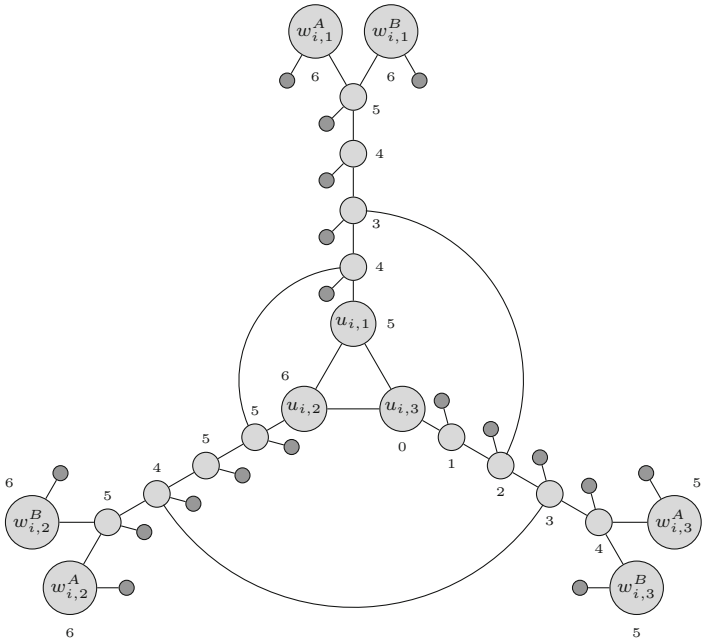

Fig. 6. Gadget with infection times for each clause C_i.

Finally, add a path with $\beta + 2$ vertices and link one end to q. Let P be the set of vertices in this path and let x be the vertex in P that is adjacent to q. By our construction, since $\Delta \geq 4$, we have that G is a graph in which every vertex has degree at most Δ.

Notice that any percolating set must contain a vertex of $\{u_{i,1}, u_{i,2}, u_{i,3}\}$ for each clause C_i of $\mathcal{C}$ and all vertices that have degree 1. Thus, following similar arguments presented in [24], it is possible to prove that the maximum percolation time of the vertex z is $height(T) + 7$ if and only if $\mathcal{C}$ is satisfiable, which implies that the maximum percolation time of the vertex x is $\lceil \log_{\Delta-2}(4N\alpha) \rceil + 8$ if and only if $\mathcal{C}$ is satisfiable.

Then, we have that $\mathcal{C}$ is satisfiable if and only if $t(G) \geq \lceil \log_{\Delta-2}(4N\alpha) \rceil + 8$, but, since $n = |V(G)| = 39M + y + t + 2 + 2r + \beta$, then $4N\alpha = c \cdot n$. Therefore, since $c = (\Delta - 2)^{-8}$, $\mathcal{C}$ is satisfiable if and only if $t(G) \geq \lceil \log_{\Delta-2} n \rceil$. ■

5 Fixed Parameter Tractability of the Percolation Time Problem

We say that a decision problem is *fixed parameter tractable* (or just *fpt*) on some parameter Ψ if there exists an algorithm (called *fpt-algorithm*) that solves the problem in time $f(\Psi) \cdot n^{O(1)}$, where n is the size of the input and f is an arbitrary function depending only on the parameter Ψ.

In this section, we show that the Percolation Time Problem is fixed parameter tractable for the parameter $tw(G) + k$, for the parameter $\Delta(G) + k$ and for the parameter $q(G)$, where $tw(G)$ is the treewidth of the graph and $q(G)$ is the minimum $q \geq 4$ such that G is a $(q, q-4)$-graph, which is a graph such that every subset of at most q vertices induces at most $q-4$ P_4's (cographs have $q(G) = 4$ and P_4-sparse graphs have $q(G) = 5$). These theorems will imply that, if k is fixed, then deciding if $t(G) \geq k$ is linear time solvable for graphs with bounded treewidth or bounded maximum degree. Moreover, they will imply that determining the maximum percolation time is polynomial time solvable for $(q, q-4)$-graphs with fixed q.

Theorem 5. *Percolation Time Problem is fixed parameter tractable with parameter $tw(G) + k$.*

Proof (sketch of the proof). A consequence of the Courcelle's theorem [18,21] states that, if a decision problem on graphs can be expressed in a Monadic Second Order (MSO) sentence φ, then this problem is fixed parameter tractable in the parameter $tw(G) + |\varphi|$. Moreover, the running time is linear on the size of the input. The Percolation Time Problem can be expressed by the following MSO-sentence:

$$maxtime_k := \exists w, X_0, X_1, \ldots, X_k \; \forall x \left(x \in X_k\right) \wedge \left(\bigwedge_{0 \leq i < k} (x \in X_i \rightarrow x \in X_{i+1})\right) \wedge$$

$$\wedge \left(\bigwedge_{0 \leq i < k} (x \in X_{i+1} \backslash X_i) \rightarrow \exists y, z (Exy \wedge Exz \wedge (y \in X_i) \wedge (z \in X_i))\right) \wedge \left(w \in X_k \backslash X_{k-1}\right),$$

where X_i represents the set of vertices infected at time i, Exy is true if xy is an edge (and false, otherwise) and $\wedge$ is the *and* operator. This MSO sentence asserts that all vertices are infected in time k, that a vertex infected in time i remains infected in time $i+1$, that a vertex infected in time $i+1$, but not infected in time i, has two neighbors infected in time i, and that there exists a vertex w infected in time k but not infected in time $k-1$. ■

Theorem 6. *Percolation Time Problem is fixed parameter tractable with parameter $\Delta(G) + k$. Moreover, for fixed Δ, the Percolation Time Problem is polynomial time solvable in graphs with bounded maximum degree Δ for $k = \log_\Delta O(\log n)$, if $\Delta \geq 4$, and for $k = O(\log n)$, if $\Delta = 3$.*

Proof (sketch of the proof). Let $\Delta = \Delta(G)$ and let $u \in V(G)$. Then $|N_{\leq k}(u)| \leq \Delta^k$ and, consequently, the power set $2^{N_{\leq k}(u)}$ has $2^{|N_{\leq k}(u)|} \leq 2^{\Delta^k}$ sets. We claim that $t(G) \geq k$ if and only if there is a vertex u and a percolating set $S \supseteq N_{\geq k}(u)$ such that $t(G, S, u) = k$.

If $t(G) \geq k$, then there is a percolating set S' that infects some vertex u at time k. In [24], it was proved that, given a graph G, a set $Q \subseteq V(G)$ and a vertex $z \in V(G) \backslash S$, if $t(G, Q, w) \geq k$, then $t(G, Q, w) \geq t(G, Q \cup \{z\}, w) \geq k$, for any k and any $w \in N_{\geq k}(z)$. Then, applying this result once for each vertex in $N_{\geq k}(u)$, the percolating set $S = S' \cup N_{\geq k}(u)$ infects u also at time k.

On the other hand, if there is a percolating set $S \supseteq N_{\geq k}(u)$ such that $t(G, S, u) = k$, for some vertex u, then, trivially, $t(G) \geq k$. Then the claim is true.

Therefore, since for each vertex u and set $S' \subseteq N_{\leq k-1}(u)$, it takes $O(km)$ time to know whether the set $S' \cup N_{\geq k}(u)$ infects u at time k, this equivalence gives us an algorithm that decides whether $t(G) \geq k$ in time $n \cdot O(m + km \cdot 2^{\Delta^k}) = O(2^{\Delta^k} k\Delta \cdot n^2)$, since $m = O(\Delta n)$. Notice that, if $k = \log_\Delta O(\log n)$, then the time is polynomial in n. Moreover, if $\Delta = 3$, by Theorem 1, we are done. ■

Finally, we prove the fixed parameter tractability for the parameter $q(G)$. In 2014, Campos et al. [14] proved that determining the minimum percolating set is fixed parameter tractable on the parameter $q(G)$. Here, we prove the following.

Theorem 7. *Percolation Time Problem is fixed parameter tractable on parameter $q(G)$. Moreover, $t(G) \leq q(G) + 3$ for every graph G.*

To prove this theorem, we use a graph decomposition, called *primeval decomposition*, which is based on some graph operations: *union*, *join*, *spider* and *p-component*. Below we define these operations and present the lemmas used to obtain the maximum percolation time. Because of space restrictions, we omit the proofs.

The *union* $G = G_1 \cup G_2$ of two graph G_1 and G_2 is the graph such that $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$. The *join* $G = G_1 \vee G_2$ is the graph obtained from $G_1 \cup G_2$ by joining every vertex of G_1 to every vertex of G_2. A *spider* (R, K, S) is a graph $G = (R \cup K \cup S,\ E)$ such that $K = \{k_1, \ldots, k_p\}$ and $S = \{s_1, \ldots, s_p\}$, for $p \geq 2$, induce a clique and a stable set, respectively; either s_i is adjacent to k_j if and only if $i = j$ (a thin spider), or s_i is adjacent to k_j if and only if $i \neq j$ (a thick spider); and every vertex of R is adjacent to each vertex of K and non-adjacent to each vertex of S.

Lemma 2 (union, join and spider). *Let G, G_1 and G_2 be graphs. If $G = G_1 \cup G_2$, then $t(G) = \max\{t(G_1), t(G_2)\}$. If $G = G_1 \vee G_2$ and G_1 and G_2 have at least two vertices each, then $t(G) \leq 3$. If $G = G_1 \vee G_2$ and G_2 has exactly one vertex, then $t(G) = diameter(G_1) + 1$, if either G_1 is disconnected or contains three vertices u, v, w such that $dist(u, w) = dist(v, w) = diameter(G_1)$ and there is no neighbor of u and v in a minimum path from u to w; otherwise, $t(G) = diameter(G_1)$. If G is a spider, then $t(G) \leq 3$. In all the three last cases, $t(G)$ can be found in the worst case in $O(mn^2)$ time.*

A graph is *p-connected* if, for every partition of the vertex set into two parts A and B, there is a crossing P_4 (with vertices of A and B). A p-connected graph is *separable* if it has a particular bipartition (H_1, H_2) such that every crossing P_4 $wxyz$ satisfies $x, y \in H_1$ and $w, z \in H_2$ (such a bipartition is unique [3]). A *p-component* of a graph is a maximal p-connected subgraph.

Given an arbitrary graph G' and a separable p-connected graph H with separation (H_1, H_2), let $G' \uplus H$ be the graph obtained from $G' \cup H$ by joining

every vertex of G' to every vertex of H_1 and and to no vertex of H_2. Note that every spider $(\emptyset, K, S)$ is a separable p-connected graph with separation (K, S).

In [3], it was proved an important structural result for $(q, q-4)$-graphs. If G is a $(q, q-4)$-graph, then either $G = G' \cup G''$, or $G = G' \vee G''$, or $G = G' \uplus H$, or G has less than q vertices, where G' and G'' are $(q, q-4)$-graphs and H is either a spider (R, K, S) with $R = \emptyset$, or a separable p-connected $(q, q-4)$-graph with less than q vertices. This characterization leads to a graph decomposition, called *primeval decomposition*, which can be obtained in linear time $O(m+n)$ [4,9].

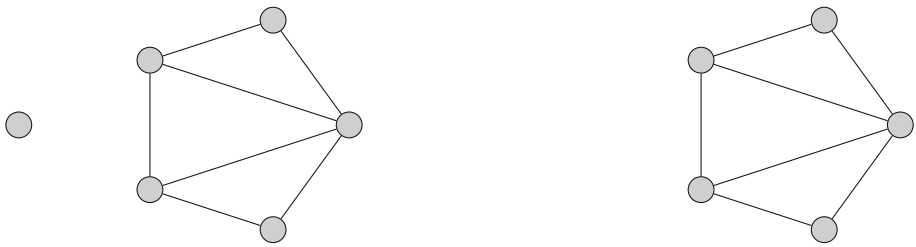

Fig. 7. The two possibilities for the graph $H^+[C]$.

Lemma 3 (p-component). *Let G' be a $(q, q-4)$-graph and let H be a separable p-connected $(q, q-4)$-graph with separation (H_1, H_2). If $G = G' \uplus H$, then $t(G) = t(H^+)$, where H^+ is the graph obtained from H by adding a set C of $x = \min\{|V(G')|, 6\}$ new vertices linked to all vertices of H_1, not linked to any vertices in H_2, and, if $x \leq 6$, let $H^+[C]$ and G' be isomorphic graphs. If $x \geq 7$:*

- *if G' is a clique then let $H^+[C]$ be a clique K_x;*
- *if G' is a stable set then let $H^+[C]$ also be a stable set;*
- *Otherwise, let $H^+[C]$ be isomorphic to the graph in Fig. 7(a), if G' is not connected, and let $H^+[C]$ be isomorphic to the graph in Fig. 7(b), if G' is connected.*

As a consequence, if H has less than q vertices (fixed $q \geq 4$), since $|V(H^+)| \leq q+5$, then $t(G) \leq q+3$ and $t(G)$ can be obtained in constant time $\leq 2^q q$ (by checking all subsets of vertices of H^+). The two lemmas above, together with the primeval decomposition of $(q, q-4)$-graphs, imply a polynomial time algorithm to determine the maximum percolation time of a $(q, q-4)$-graph, for fixed q, in $O(mn^2)$ time. This also implies that the Percolation Time Problem is fixed parameter tractable for the parameter $q(G)$.

References

1. Amini, H.: Bootstrap percolation in living neural networks. J. Stat. Phys. **141**(3), 459–475 (2010)
2. Araújo, R., Sampaio, R., Santos, V., Szwarcfiter, J.: The convexity of induced paths of order three and applications: complexity aspects. Discrete Appl. Math. (2015, to appear)
3. Babel, L., Olariu, S.: On the structure of graphs with few P_4's. Discrete Appl. Math. **84**, 1–13 (1998)
4. Babel, L., Kloks, T., Kratochvíl, J., Kratsch, D., Müller, H., Olariu, S.: Efficient algorithms for graphs with few P_4's. Discrete Math. **235**, 29–51 (2001)
5. Balogh, J., Bollobás, B., Duminil-Copin, H., Morris, R.: The sharp threshold for bootstrap percolation in all dimensions. Trans. Amer. Math. Soc. **364**(5), 2667–2701 (2012)
6. Balogh, J., Pete, G.: Random disease on the square grid. Random Struct. Algorithms **13**, 409–422 (1998)
7. Balogh, J., Bollobás, B., Morris, R.: Bootstrap percolation in three dimensions. Ann. Probab. **37**(4), 1329–1380 (2009)
8. Balogh, J., Bollobás, B., Morris, R.: Bootstrap percolation in high dimensions. Combin. Probab. Comput. **19**(5–6), 643–692 (2010)
9. Baumann, S.: A linear algorithm for the homogeneous decomposition of graphs. Report No. M-9615, Zentrum Mathematik. Technische Universität München (1996)
10. Benevides, F., Campos, V., Dourado, M.C., Sampaio, R.M., Silva, A.: The maximum time of 2-neighbour bootstrap percolation: algorithmic aspects. In: The Seventh European Conference on Combinatorics, Graph Theory and Applications, Series CRM, vol. 16, pp. 135–139. Scuola Normale Superiore (2013)
11. Benevides, F., Przykucki, M.: Maximum percolation time in two-dimensional bootstrap percolation. Submitted (2014). http://arxiv.org/abs/1310.4457v1
12. Benevides, F., Przykucki, M.: On slowly percolating sets of minimal size in bootstrap percolation. Electron. J. Comb. **20**(2), 46 (2013)
13. Bollobás, B., Holmgren, C., Smith, P.J., Uzzell, A.J.: The time of bootstrap percolation with dense initial sets. Ann. Probab. **42**(4), 1337–1373 (2014)
14. Campos, V., Sampaio, R., Silva, A., Szwarcfiter, J.: Graphs with few P4s under the convexity of paths of order three. Discrete Appl. Math. (2015, to appear). http://dx.doi.org/10.1016/j.dam.2014.05.005
15. Chalupa, J., Leath, P.L., Reich, G.R.: Bootstrap percolation on a Bethe lattice. J. Phys. C **12**(1), 31–35 (1979)
16. Chen, N.: On the approximability of influence in social networks. SIAM J. Discrete Math. **23**(3), 1400–1415 (2009)
17. Coelho, E.M.M., Dourado, M.C., Sampaio, R.M.: Inapproximability results for graph convexity parameters. In: Kaklamanis, C., Pruhs, K. (eds.) WAOA 2013. LNCS, vol. 8447, pp. 97–107. Springer, Heidelberg (2014)
18. Courcelle, B., Makowsky, J., Rotics, U.: On the fixed parameter complexity of graph enumeration problems definable in monadic second order logic. Discrete Appl. Math. **108**, 23–52 (2001)
19. Dreyer, P.A., Roberts, F.S.: Irreversible k-threshold processes: graph-theoretical threshold models of the spread of disease and of opinion. Discrete Appl. Math. **157**(7), 1615–1627 (2009)
20. Fey, A., Levine, L., Peres, Y.: Growth rates and explosions in sandpiles. J. Stat. Phys. **138**, 143–159 (2010)

21. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2010)
22. Holroyd, A.E.: Sharp metastability threshold for two-dimensional bootstrap percolation. Probab. Theor. Relat. Fields **125**(2), 195–224 (2003)
23. Itai, A., Papadimitriou, C.H., Szwarcfiter, J.L.: Paths in grid graphs. SIAM J. Comput. **11**, 676–686 (1982)
24. Marcilon, T., Nascimento, S., Sampaio, R.: The maximum time of 2-neighbour bootstrap percolation: complexity results. In: Kratsch, D., Todinca, I. (eds.) WG 2014. LNCS, vol. 8747, pp. 372–383. Springer, Heidelberg (2014)
25. Morris, R.: Minimal percolating sets in bootstrap percolation. Electron. J. Comb. **16**(1), 20 (2009)
26. Papadimitriou, C.H., Vazirani, U.V.: On two geometric problems related to the travelling salesman problem. J. Algorithms **5**(2), 231–246 (1984)
27. Przykucki, M.: Maximal percolation time in hypercubes under 2-bootstrap percolation. Electron. J. Comb. **19**(2), 41 (2012)
28. Riedl, E.: Largest minimal percolating sets in hypercubes under 2-bootstrap percolation. Electron. J. Comb. **17**(1), 13 (2010)
29. Tovey, C.A.: A simplified NP-complete satisfiability problem. Discrete Appl. Math. **8**(1), 85–89 (1984)
30. Umans, C., Lenhart, W.: Hamiltonian cycles in solid grid graphs. In: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS), 1997, Miami, USA, pp. 496–505. IEEE Computer Society, Washington, DC (1997)