Theoretical Computer Science 557 (2014) 76–86

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Brushing with additional cleaning restrictions

Piotr Borowiecki [a,1], Dariusz Dereniowski [a,2], Paweł Prałat [b,*,3]

[a] Department of Algorithms and System Modeling, Gdańsk University of Technology, Gdańsk, Poland
[b] Department of Mathematics, Ryerson University, Toronto, ON, Canada, M5B 2K3

ARTICLE INFO

Article history:
Received 17 December 2013
Received in revised form 28 August 2014
Accepted 8 September 2014
Available online 16 September 2014
Communicated by G. Ausiello

Keywords:
Graph searching
Brush number
Chip firing

ABSTRACT

In graph cleaning problems, brushes clean a graph by traversing it subject to certain rules. We consider the process where at each time step, a vertex that has at least as many brushes as incident, contaminated edges, sends brushes down these edges to clean them. Various problems arise, such as determining the minimum number of brushes (called the brush number) that are required to clean the entire graph. Here, we study a new variant of the problem in which no more than k brushes can be sent at any time step.

1. Introduction

Imagine a network of pipes that must be periodically cleaned of a regenerating contaminant, say algae. In *cleaning* such a network (see [11] for a paper that introduced this process), there is an initial configuration of brushes on vertices; every vertex and edge is initially regarded as *dirty*. A vertex is ready to be cleaned if it has at least as many brushes as incident dirty edges. When a vertex is cleaned, it sends one brush along each incident dirty edge; these edges are now said to be *clean*. (No brush ever traverses a clean edge.) The vertex is also deemed clean. Excess brushes remain on the clean vertex and take no further part in the process. (In fact, for our purpose in this paper, we may think about clean vertices as if they were removed from the graph.) Fig. 1 illustrates the cleaning process for a graph G where there are initially two brushes at vertex b and one brush at vertex c. The solid edges indicate dirty edges while the dashed edges indicate clean edges. For example, the process starts with vertex b being cleaned, sending a brush to each of vertices a and d.

This model, perhaps surprisingly, corresponds to the *minimum total imbalance* of the graph which is used in graph drawing theory [5] and is well-studied, in particular, for random graphs [1,15]. (See also [9] for algorithmic aspects, [12,14] for a related model of cleaning with brooms, [4] for a variant with no edge capacity restrictions, [7] for a combinatorial game, and [13] for a relation to more general family of perfect vertex elimination schemes leading to upper-locally distributed lattices.) Having been inspired by chip firing processes [3,10], the manner in which brushes disperse from an individual vertex is such that they do so in unison, provided that their vertex meets the criteria to be cleaned. Models in which multiple vertices may be cleaned simultaneously are called *parallel cleaning models* (see [6] for more details). In contrast,

* Corresponding author.
E-mail addresses: pborowie@eti.pg.gda.pl (P. Borowiecki), deren@eti.pg.gda.pl (D. Dereniowski), pralat@ryerson.ca (P. Prałat).
[1] The first author was partially supported by National Science Centre grant DEC-2011/02/A/ST6/00201.
[2] The second author was partially supported by National Science Centre grant DEC-2011/02/A/ST6/00201 and a scholarship for outstanding young researchers founded by the Polish Ministry of Science and Higher Education.
[3] The third author was supported by NSERC and Ryerson University.

http://dx.doi.org/10.1016/j.tcs.2014.09.005

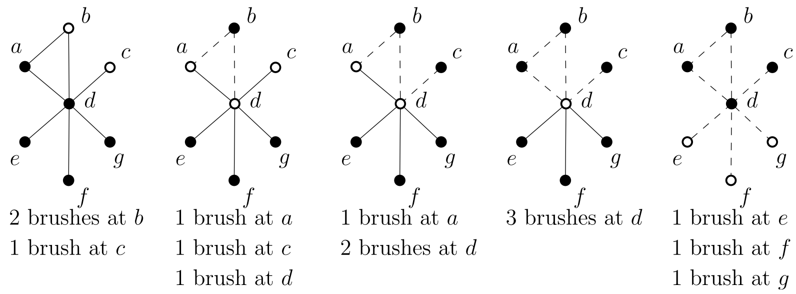

Fig. 1. An example of the cleaning process for graph G.

sequential cleaning models mandate that vertices are cleaned one at a time. The variant considered in [11] and the one we consider in this paper are sequential in nature.

In this paper, we consider the variant of the process in which there is one additional restriction, namely, it is not allowed to clean vertices with more than k dirty neighbours (k is a fixed parameter). For example, 3 brushes were enough to clean the graph presented on Fig. 1. However, if at most $k = 2$ brushes can be sent in each time step, vertex d cannot be cleaned; one additional brush has to be introduced (say, at vertex e) and our task can be accomplished with 4 brushes in total.

The formal definition of the k-restricted process and th k-brush number, $b_k(G)$, is introduced in Section 2 ($b(G)$ denotes the original brush number introduced in [11] and also defined in Section 2). In Section 3, we characterize graphs that can be cleaned in this process. Moreover, we show that for any $k \in \mathbb{N}$, $x \in [1, 2]$, and $\varepsilon > 0$, there exists a tree T such that $|b_k(T)/b(T) - x| < \varepsilon$. This result is sharp, that is, this property does not hold for other values of x. Similarly, it is shown that for any $k \in \mathbb{N} \setminus \{1\}$, $x \in [1, \infty)$, and $\varepsilon > 0$, there exists a graph G such that $|b_k(G)/b(G) - x| < \varepsilon$. In Section 4, we study a polynomial-time algorithm that, given any tree T and any $k \in \mathbb{N}$, computes the k-brush number of T. The paper is concluded with a few open problems (see Section 5).

Throughout, we consider only finite, simple, undirected graphs in the paper. For background on graph theory, the reader is directed to [16].

2. Definitions

As already noted, the graph cleaning model we consider here differs from the one presented in [11] in that one is not allowed to clean vertices with more than k dirty neighbours. Now, we formally define the cleaning process we are considering in this paper. Let $k \in \mathbb{N}$ and let $G = (V, E)$ be any graph. The initial configuration of brushes is given by the function $\omega_0 : V \to \mathbb{N} \cup \{0\}$, where $\omega_0(v)$ is the number of brushes initially at vertex v, and all vertices and edges of the graph are initially dirty. At the end of each step t of the process, $\omega_t(v)$ denotes the number of brushes at vertex $v \in V$, and $D_t \subseteq V$ denotes the set of dirty vertices. An edge $uv \in E$ is dirty if and only if both u and v are dirty; that is, $\{u, v\} \subseteq D_t$. Finally, let $D_t(v)$ denote the number of dirty edges incident to v at the end of step t; that is,

$$D_t(v) = \begin{cases} |N(v) \cap D_t| & \text{if } v \in D_t \\ 0 & \text{otherwise} \end{cases}$$

(where $N(v)$ denotes, as usual, the neighbourhood of v).

Definition 1. Let $k \in \mathbb{N}$. The ***k*-restricted cleaning process** $\mathfrak{P}(G, k, \omega_0) = \{(\omega_t, D_t)\}_{t=0}^{L}$ of an undirected graph $G = (V, E)$ with an **initial configuration of brushes** ω_0 is as follows:

0. Initially, all vertices are dirty: $D_0 = V$; set $t := 0$.
1. Let α_{t+1} be any vertex in D_t such that $\omega_t(\alpha_{t+1}) \geq D_t(\alpha_{t+1})$ and $D_t(\alpha_{t+1}) \leq k$. If no such vertex exists, then stop the process (set $L := t$), return the **cleaning sequence** $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_L)$, the **final set of dirty vertices** D_L, and the **final configuration of brushes** ω_L.
2. Clean α_{t+1} and all dirty incident edges by moving a brush from α_{t+1} to each dirty neighbour. More precisely, $D_{t+1} = D_t \setminus \{\alpha_{t+1}\}$, $\omega_{t+1}(\alpha_{t+1}) = \omega_t(\alpha_{t+1}) - D_t(\alpha_{t+1})$, and for every $v \in N(\alpha_{t+1}) \cap D_t$, $\omega_{t+1}(v) = \omega_t(v) + 1$, the other values of ω_{t+1} remain the same as in ω_t.
3. Set $t := t + 1$ and go back to 1.

In the model considered in [11], there is no restriction that $D_t(\alpha_{t+1}) \leq k$. All other rules remain the same. It was shown in [11] that, given a graph G and an initial configuration ω_0, the cleaning process of [11] returns a unique final set of dirty vertices. Consequently, if for a given ω_0, there exists a cleaning sequence that cleans G, we know that every cleaning sequence will clean G. This is also the case here and easily follows from the simple observation that once a vertex v is ready to be cleaned it stays in this state until it is actually cleaned.