Parallel Cleaning of a Network with Brushes *

Serge Gaspers[†] Margaret-Ellen Messinger[‡] Richard J. Nowakowski[‡]
Paweł Prałat[§]
[†]LIRMM – University of Montpellier 2, CNRS, 34392 Montpellier, France
[‡]Department of Mathematics and Statistics, Dalhousie University,
Halifax, NS, B3H 3J5, Canada
[§]Department of Mathematics, West Virginia University,
Morgantown, WV 26506-6310, USA

Abstract

We consider the process of cleaning a network where at each time step, all vertices that have at least as many brushes as incident, contaminated edges, send brushes down these edges and remove them from the network. An added condition is that, because of the contamination model used, the final configuration must be the initial configuration of another cleaning of the network. We find the minimum number of brushes required for trees, cycles, complete bipartite networks; and for all networks when all edges must be cleaned on each step. Finally, we give bounds on the number of brushes required for complete networks.

1 Introduction

Imagine a network of pipes with a biological contaminant that is: (1) relatively mobile so that contamination can spread from one area to another; and (2) that regenerates. For example, water pipes that have algae or zebra mussels contamination [6, 7]. To prevent the network from clogging, the pipes must be cleaned on a regular basis and steps must be taken to prevent recently cleaned pipes from being recontaminated. Because of the regeneration of the contaminant, the final configuration is the basis for the start of the next round of network cleaning. There is much work on models of contamination of networks where condition (1) is paramount, see [1, 9] for examples. Condition (2), regeneration, is new in this context. See [14, 15] for a situation where the contaminant regenerates but is immobile.

The *(sequential) brush cleaning* model was introduced in [8, 11]. Initially, every edge and every vertex of a network is dirty and a fixed number of brushes start on a set of vertices. A vertex may be 'cleaned' if it contains as many brushes as dirty incident edges. At each step, one vertex v is cleaned and each incident, dirty edge is traversed by a brush from v. Once a brush has traversed a dirty edge, that edge has been cleaned and for this round of cleaning, may be regarded as being deleted from the network. Thus, a network has been cleaned once every edge has been cleaned. The brush number of network G, is the minimum number of

*Partially supported by grants from the ACEnet, NFR, NSERC, MITACS, and SHARCNET.
gaspers@lirmm.fr, messnger@mathstat.dal.ca, rjn@mathstat.dal.ca, pralat@math.wvu.edu

brushes needed to clean G and is denoted $b(G)$. The model requires that the cleaning process be *continual*: once the network has been cleaned, the network is regarded as contaminated again and the brushes re-clean the network. If the vertices are cleaned sequentially, it was shown in [11] that the process is always continual, or more specifically 'reversible'. That is, let ω_0 be an initial configuration of brushes that will clean a network G, which leaves a final configuration ω_n of brushes. Using the initial configuration $\tau_0 = \omega_n$ of brushes, G can be cleaned leaving the final configuration $\tau_n = \omega_0$. However, this is not to say that sequential cleaning is easy. On the contrary in general, it is difficult to find $b(G)$. In [5], it was shown that determining the value of $b(G)$ is NP-complete and the problem remains NP-complete for bipartite graphs of maximum degree 6, planar graphs of maximum degree 4, and 5-regular graphs.

The sequential brush number has also been studied on random regular networks [2, 13] and on random networks [17], showing that the brush number is almost surely close to $dn/4$ for networks of (large average) degree d. It is also important to investigate the worst case scenario, the Broom number, $B(G)$, which is the maximum number of brushes that can be used to clean the graph where every brush has to clean at least one edge. (Note that the restriction is necessary; else 'infinitely' many brushes can be used.) The Broom number was recently studied in [12, 16].

In this paper, we consider a variant of the cleaning model, introduced in [8] where at each step, every vertex which may be cleaned, is cleaned simultaneously. In this variant, the **parallel cleaning model**, the minimum number of brushes needed to clean a network G, with parallel firing of vertices, is called the *parallel brush number* and is denoted $pb(G)$. See Figure 1 for an example. It is also clear that the parallel cleaning process for the network in Figure 1 is always reversible. This is not the case for the network G' in Figure 2, with

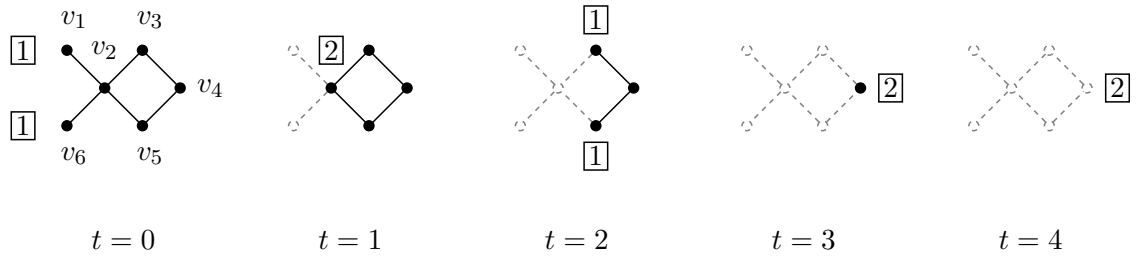


Figure 1: The parallel cleaning process for a network G with one brush initially at each of $v_1,\ v_6$.

one brush initially at each of $v_1,\ v_7$. It can only be cleaned once. The final configuration of brushes (one at each of $v_4,\ v_5$) is not a viable initial configuration of brushes if the edges of the network were to become dirty again. This provides motivation for the results of this paper. We wish to determine the minimum number of brushes needed to ensure a network can be parallel cleaned *continually*. Note that, although the practical situation being modeled only requires every edge to be cleaned for the network to be deemed clean, the assumption taken in [11] and taken here, is that a network has been cleaned once every vertex (and hence every edge) has been cleaned. Although this viewpoint may seem unnatural, it simplified much of the analysis in [11]. Therefore, in Figure 1, at time $t = 3$, vertex v_4 is cleaned (although it

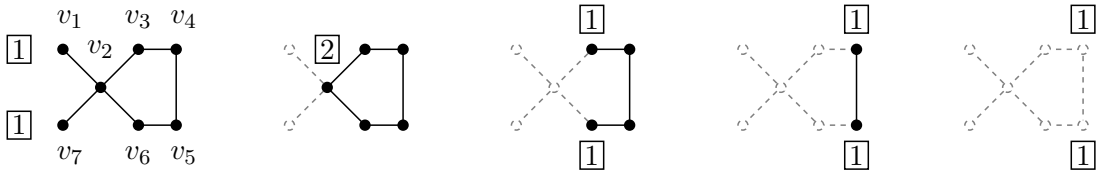


Figure 2: The parallel cleaning process for a network G' with one brush initially at each of v_1, v_7.

has no dirty incident edges).

Section 2 formally defines the parallel cleaning process and the notion of continually parallel cleaning a network. In Section 3, we determine the exact number of brushes required to continually one-step clean a network (in each cleaning, all edges of the network are cleaned in the first step). Section 4 focuses on the minimum number of brushes to continually parallel clean various networks, finding exact results for cycles, trees and complete bipartite networks. Finally, Section 4.4 bounds the number of brushes needed to continually parallel clean a complete network K_n between $\frac{5}{16}n^2 + O(n)$ and $\frac{4}{9}n^2 + O(n)$, a table of exact values is also given.

2 Definitions

Following the terminology used for the (sequential) cleaning model in [11], at each time step t, $\omega_t(v)$ denotes the number of brushes at vertex v and D_t denotes the set of dirty vertices. An edge $uv \in E$ is dirty if and only if both u and v are dirty: $\{u, v\} \subseteq D_t$. Finally, let $D_t(v)$ denote the number of dirty edges incident to v at step t:

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

Definition 2.1 *The **parallel cleaning process** $\mathfrak{C} = \{(\omega_t, D_t)\}_{t=0}^{K}$ of an undirected network $G = (V, E)$ 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 $\rho_{t+1} \subseteq D_t$ be the set of vertices such that $\omega_t(v) \geq D_t(v)$ for $v \in \rho_{t+1}$. If $\rho_{t+1} = \emptyset$, then stop the process ($K = t$), return the **parallel cleaning sequence** $\rho = (\rho_1, \rho_2, \ldots, \rho_K)$, the **final set of dirty vertices** D_K, and the **final configuration of brushes** ω_K*

(2) *Clean each vertex $v \in \rho_{t+1}$ and all incident, dirty edges by traversing a brush from v to each dirty neighbour. More precisely, $D_{t+1} = D_t \setminus \rho_{t+1}$; for every $v \in \rho_{t+1}$, $\omega_{t+1}(v) = \omega_t(v) - D_t(v) + |N(v) \cap \rho_{t+1}|$; for every $u \in D_{t+1}$, $\omega_{t+1}(u) = \omega_t(u) + |N(u) \cap \rho_{t+1}|$; and $\omega_{t+1}(v) = \omega_t(v)$ for all other vertices.*

(3) *$t := t + 1$ and go back to* **(1)**.

One condition the cleaning model has, like the chip-firing game, but not the edge-searching problem, is that the cleaning process is to be automatic, continuing on for the lifetime of the network. That is, a final configuration of brushes (after a network has been cleaned) is to be a viable initial configuration of brushes (to clean the network again).

Definition 2.2 *The **parallel brush number**, $pb(G)$, is the minimum number of brushes needed to clean G using a parallel cleaning process.*

It was shown in [11] that $b(G) = pb(G)$ for any network G. However, as shown in Figure 2, a final configuration of brushes using the parallel cleaning process may not be a viable initial configuration. Consequently, an obvious question and the focus of this paper, is "how many extra brushes are required to continually clean a network?"

Let δ_A denote the Kronecker delta where

$$\delta_A = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

Definition 2.3 *Let G be a network with initial configuration $\omega_0^0 = \omega_0$. Then G can be **continually cleaned** using the parallel cleaning process beginning from configuration ω_0 if for each $s \in \mathbb{N} \cup \{0\}$, G can be cleaned in parallel using initial configuration ω_0^s, yielding the final configuration $\omega_{K_s}^s$ where $\omega_0^{s+1} = \omega_{K_s}^s$.*

Given initial configuration ω_0^s, let D_t^s denote the set of dirty vertices at step t; let $D_t^s(v) = |N(v) \cap D_t^s| \delta_{v \in D_t^s}$ denote the number of dirty edges incident to v at step t; and let $\rho_{t+1}^s = \{v \in D_t^s \ : \ \omega_t^s(v) \geq D_t^s(v)\}$ denote the set of vertices that may be cleaned at step t.

Definition 2.4 *The **continual parallel brush number**, $cpb(G)$, of a network G is the minimum number of brushes needed to continually clean G using a parallel cleaning process.*

3 Continual One-Step Cleaning

In this section, we address the problem of continually cleaning all the edges of a network in one step.

Definition 3.1 *Let ω_0 be an initial configuration that will continually clean network G. For each $s \in \mathbb{N} \cup \{0\}$, if $D_1^s(v) = 0$ for each $v \in V(G)$, then we say that G can be **continually one-step cleaned**.*

Note that the condition "$D_1^s(v) = 0$ for each $v \in V(G)$" in the previous definition requires only that each edge is cleaned in one step, not each vertex (which would require the condition "$D_1^s = \emptyset$").

Given configuration ω_0, suppose network G can be continually one-step cleaned. As each edge must be cleaned after the first step in each cleaning, the set of vertices ρ_1^s for $s \in \mathbb{N} \cup \{0\}$ must be a vertex cover.

Theorem 3.2 *Given initial configuration ω_0, G can be continually one-step cleaned if and only if ρ_1^s is a vertex cover for every $s \in \mathbb{N} \cup \{0\}$.*

For every network $G = (V, E)$ there is an initial configuration ω_0 such that the network can be continually one-step cleaned. Simply set $\omega_0(v) = \deg(v)$ for every $v \in V$. Then at $t = 0$, every vertex is cleaned (using a total of $2|E|$ brushes). Thus, the following definition is natural.

Definition 3.3 *The **continual one-step brush number** of network G, denoted by $cpb_1(G)$, is the minimum number of brushes needed to continually one-step clean G.*

It is also not difficult to see that $cpb_1(G) \geq |E|$ as every edge of G must be traversed by at least one brush in one step. It will be shown in Theorem 3.4 that for a connected network $G = (V, E)$, $|E|$ and $2|E|$ are the only two possible values for $cpb_1(G)$.

The chip firing game (see [4] for example), begins after each vertex is assigned a (finite) number of chips. At each step, one vertex, with at least as many chips as its degree, is fired (whereupon it sends one chip to each neighbour). In the parallel chip firing game, introduced in [3], at each step, every vertex with at least as many chips as its degree is fired. Note that the process of one-step cleaning a network is simply a case of the parallel chip-firing game. It was observed in [3] that for the parallel chip-firing game, any chip configuration converges in a finite time T to a limit cycle of period p. As the one-step cleaning process is an instance of the parallel chip-firing game, the observation can be applied here also and in the proof of Theorem 3.4, the focus is on the steps from T to $T + p - 1$.

Theorem 3.4 *For any connected network $G = (V, E)$,*

$$cpb_1(G) = \begin{cases} |E| & \text{if } G \text{ is bipartite} \\ 2|E| & \text{otherwise.} \end{cases}$$

Proof: Suppose first that G is bipartite with partite sets: V_0 and V_1. For every $v \in V$, let

$$\omega_0^0(v) = \deg(v)\delta_{v \in V_0}.$$

Then for every $v \in V$,

$$\omega_1^0(v) = \deg(v)\delta_{v \in V_1}$$

and $\omega_2^0(v) = \omega_1^0(v)$ with $K_0 = 2$. It is easy to see that $\omega_t^s = \omega_t^{s \ mod \ 2}$ for all $t \in \{0, 1, 2\}$ and all $s \in \mathbb{N} \cup \{0\}$. Thus $cpb_1(G) \leq |E|$ and, since $cpb_1(G) \geq |E|$ for every network G, the assertion follows.

Suppose now that G is not bipartite. Let $\omega_0^0 = \omega_0$ be an initial configuration that will continually one-step clean G.

Recall that as the continual one-step cleaning of a network G is an instance of the parallel chip-firing game, it must eventually become cyclic with period p (since there are finitely many possible configurations of brushes); that is, $\omega_0^T = \omega_0^{T+p}$ for some $T, p \in \mathbb{N}$. Thus, $\omega_0^{T+t} = \omega_0^{T+(t \ mod \ p)}$ for $t \geq 0$ where p is the length of the period.

During a period of length p, suppose a vertex v is cleaned k_v times in the first step. Then in one period, $k_v \deg(v)$ brushes traverse from v to its neighbours and $\sum_{u \in N(v)} k_u$ brushes traverse from neighbours of v to v. So

$$k_v \deg(v) = \sum_{u \in N(v)} k_u \tag{1}$$

for every $v \in V$. It will be shown that all vertices are cleaned in the first step the same number of times during the period of length p.

For a contradiction, suppose that there is a vertex v such that $k_v \geq k_u$ for $u \in N(v)$ and there is a vertex $w \in N(v)$ such that $k_v > k_w$. But, using (1), we get the contradiction

$$\deg(v) = \sum_{u \in N(v)} \frac{k_u}{k_v} < \deg(v)\,.$$

Thus, all vertices must be cleaned in the first step exactly k times during each period.

It is clear that $k \leq p$. Note also that if a vertex is not cleaned at the first step of the i^{th} cleaning process, it will be cleaned at the first step of the $i+1^{th}$ cleaning process. So $p/2 \leq k \leq p$. In fact, k is greater than $p/2$. For a contradiction, suppose that $k = p/2$: every vertex is cleaned in the first step during the i^{th} process for $i = 2, 4, 6, \dots$ (or for $i = 1, 3, 5, \dots$) only and the vertex set can be decomposed into two sets ρ_0^i and ρ_1^i. Note that no two neighbours are cleaned in the first step of the i^{th} process (for any i); if they are, then they both are not cleaned in the first step of the $i+1^{th}$ process and the edge joining them is not cleaned in the first step of the $i+1^{th}$ process. However, this implies there is no edge between sets ρ_0^i and ρ_1^i. This is a contradiction because G is not bipartite.

Now, let us fix an edge $uv \in E$. Since both u and v are cleaned more than $p/2$ times during the period of length p, there must be some step $T_{u,v}$ at which both vertices are cleaned. Then a brush b_u is sent from u to v and a brush b_v is sent from v to u. We will show that we can force these two brushes to stay in vertices u or v from that step of the process on.

Let $A(t)$ denote the event that exactly one of b_u, b_v is at each of u, v; $A(T_{u,v})$ trivially holds. If both u and v are cleaned at step t and $A(t)$ holds, then brushes exchange each other and $A(t+1)$ holds as well. If only one vertex, say u, is cleaned at step t and $A(t)$ holds, then v contains at least $\deg(v)+1$ brushes at time $t+1$, including brushes b_u and b_v; v is cleaned at time $t+1$, and since we have enough brushes we can leave b_u at v and move b_v to u, thus $A(t+2)$ holds as well. So brushes b_u, b_v stay at u or v at every time $t \geq T_{u,v}$ and every edge is associated with two brushes for $t \geq \max\{T_{u,v} : uv \in E\}$. Finally, the total number of brushes is at least $2|E|$, and since $cpb_1(G) \leq 2|E|$ for every network G, the proof is complete. ∎

The concept can easily be extended to continual k-step cleaning of a network G where $cpb_k(G)$ is the minimum number of brushes needed to continually k-step clean a network G.

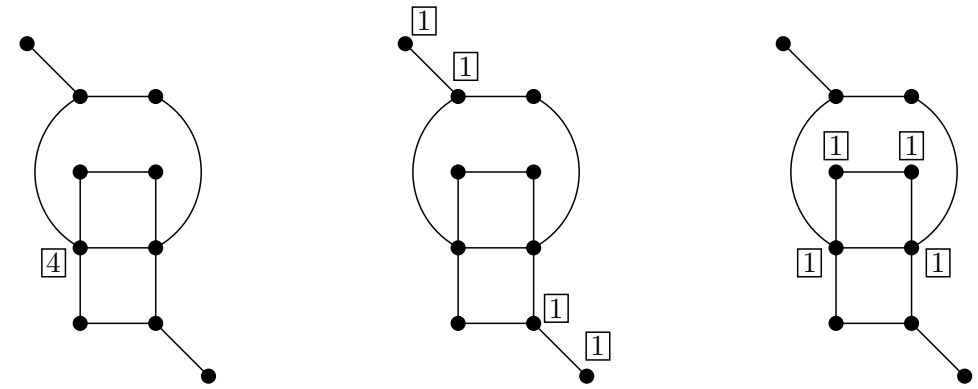
4 Continual Parallel Cleaning

In this section, the focus shifts from examining the number of brushes needed to continually clean a network in one step, to the more general problem of determining the minimum number of brushes, $cpb(G)$, needed to continually clean a network G (regardless of the number of steps). Section 4.1 determines the continual brush number for cycles and Section 4.2 determines the continual brush number for trees. Although the continual brush number for complete bipartite networks is determined in Section 4.3, only upper and lower bounds will be determined for complete networks. Theorem 4.7 and Corollary 4.11 bound the continual brush number for a complete network K_n between $5/16n^2 + O(n)$ and $4/9n^2 + O(n)$.

Given initial configuration $\omega_0 = \omega_0^0$, suppose that a network G can be continually cleaned. As G is finite, there are finitely many possible configurations of brushes. Thus, the cleanings

must eventually become cyclic with period p, that is, settle into some periodic sequence of initial conditions. That is, $\omega_0^T = \omega_0^{T+p}$ for some $T, p \in \mathbb{N}$. Thus, $\omega_0^{T+t} = \omega_0^{T+(t \bmod p)}$ for $t \geq 0$ where p is the length of the period.

Let ω_0 be an initial configuration for a network G. The problem of determining $cpb(G)$ is also made difficult by the fact that G may be cleaned k times (for some value of k), but not $k+1$ times. For example, given the network G with initial configuration ω_0^0 shown in Figure 3 (a), the network can be cleaned, leaving final configuration ω_7^0. Using $\omega_7^0 = \omega_0^1$ as the initial configuration for the second cleaning as in Figure 3 (b), G is cleaned leaving final configuration ω_5^1. Using $\omega_5^1 = \omega_0^2$ as the initial configuration for the third cleaning as in Figure 3 (c), G cannot be cleaned a third time.



(a) ω_0^0 (b) $\omega_7^0 = \omega_0^1$ (c) $\omega_5^1 = \omega_0^2$

Figure 3: A network G with several initial configurations.

4.1 Continual Parallel Cleaning: Cycles

Given a cycle C_3 with vertices a, b, c, it is easy to see that $b(C_3) = 2$ and $cpb(C_3) = 3$. Suppose two brushes are initially placed at vertex a. Then in the first step, a is cleaned, sending one brush to each of $b,\ c$. In the second step, both b and c are cleaned, each sending a brush to the other. In the final configuration, each of $b,\ c$ has one brush. Certainly, this is not a viable initial configuration to clean C_3 again. Thus, at least 3 brushes are needed to clean C_3 and it is easy to see this is an exact value.

Theorem 4.1 *For any cycle C_n with $n \geq 2$,*

$$cpb(C_n) = \begin{cases} 2 & \text{if } n \text{ is even} \\ 3 & \text{if } n = 3 \\ 4 & \text{otherwise.} \end{cases}$$

Proof: Let $C_n = (V, E)$ be a cycle on n vertices and label the vertices of C_n such that $v_i v_{i+1} \in E$ for all $i \in \{0, 1, \ldots, n-2\}$ and $v_0 v_{n-1} \in E$. It is clear that at least two brushes are needed to clean a cycle. If $n = 3$, then by the argument above, $cpb(C_3) = 3$. Suppose n is even. If two brushes are placed at v_0 in the initial configuration, then after the network has been cleaned (in parallel), the two brushes must be located at $v_{n/2}$. As the network is

symmetric, the final configuration is equivalent to the initial configuration. Thus, $cpb(C_n) = 2$ if n is even.

Suppose n is odd and $n > 3$. Two brushes are placed at v_0 in the initial configuration. Then in the final configuration, there is one brush located at $v_{(n-1)/2}$ and one at $v_{(n+1)/2}$. As C_n cannot be cleaned a second time, $cpb(C_n) > 2$.

Now we try to clean C_n using 3 brushes. Initially place two brushes at v_0. The third brush must be located at either $v_{(n-1)/2}$ or $v_{(n+1)/2}$ in order to be able to clean the network again. By symmetry, suppose it is located at $v_{(n-1)/2}$. Once C_n has been cleaned, there are two brushes at $v_{(n-1)/2}$ and one brush at $v_{(n+1)/2}$. The network can be cleaned a second time, but note that the brush located at $v_{(n+1)/2}$ has no impact on the second cleaning process. That is, the final configuration (of the second cleaning) yields one brush at each of $v_{(n+1)/2}$, v_0 and v_{n-1}. As C_n cannot be cleaned a third time, $cpb(C_n) > 3$.

It is easy to see that $cpb(C_n) \leq 4$ by initially placing two brushes at each of v_0, v_1. After the first cleaning there are two brushes at $v_{(n+1)/2}$ and one at each of v_0, v_1. After the second cleaning, there are two brushes at each of v_0, v_1, which is where they began originally. ∎

4.2 Continual Parallel Cleaning: Trees

Theorem 4.2 *Let $G = (V, E)$ be a network that contains a bridge $ab \in E$. Suppose G is cleaned by a parallel cleaning process using $b(G) = pb(G)$ brushes. Then the vertices a and b are not cleaned at the same time step.*

Proof: Let $\mathfrak{P} = \{(\rho_t, D_t)\}_{t=0}^{K}$ be a parallel cleaning process cleaning G with $pb(G)$ brushes and suppose that a and b are both cleaned at time step k, $1 \leq k \leq K$. We will show that $b(G) < pb(G)$.

Let (A, B) be a partition of V such that $a \in A$, $b \in B$ and ab is the only edge adjacent to a vertex of A and a vertex of B in G. Let $G_A = G[A \cup \{b\}]$ and $G_B = G[B]$ where $G[S]$ denotes the subnetwork of G induced by the set $S \subseteq V$. Note that since $\mathfrak{P}$ is a parallel cleaning sequence, no brush that started from a vertex in A cleans an edge in B and no brush that started from a vertex in B cleans an edge in A.

A parallel cleaning sequence $\rho_1, \rho_2, \ldots, \rho_k$ can be turned into a sequential cleaning sequence by cleaning all the vertices of ρ_1 in any order (instead of all at once), then the vertices of ρ_2, and so on. In this way, form a sequential cleaning sequence $\mathfrak{P}^*$ for G. By restricting to just the vertices of G_A we have a sequential cleaning sequence $\mathfrak{Q}_A$ for G_A except since a and b are cleaned at the same time in $\mathfrak{P}$, we clean a then clean b and since it has no dirty incident edges in G_A no brushes move. In the same way, we obtain a cleaning sequence $\mathfrak{Q}_B$ for G_B except the brush that would go to a from b in $\mathfrak{P}$ is left at b and therefore at the end of $\mathfrak{Q}_B$, b has at least one brush.

In [11], it was shown that the reverse of a sequential cleaning sequence is also a cleaning sequence, thus the reverse of $\mathfrak{Q}_B$, denoted $\mathfrak{Q}_B^*$ is also a sequential cleaning sequence. Let the final configuration of brushes in G_B after $\mathfrak{Q}_B$ be denoted by τ^* and the initial configuration in $G[A]$ be τ. Now consider the configuration of brushes

$$\sigma_0(v) = \begin{cases} \tau(v) & \text{if } v \in A \\ \tau^*(v) - 1 & \text{if } v = b \\ \tau^*(v) & \text{otherwise,} \end{cases}$$

and the sequential cleaning sequence for G formed by implementing $\mathfrak{Q}_A$ then $\mathfrak{Q}_B^*$. Firstly, all the vertices of A are cleaned with the result that the edge ab is cleaned and there is an extra brush at b, that is, b now has $\tau^*(b)$ brushes. Therefore $\mathfrak{Q}_B^*$ now cleans B. Thus all of G has been cleaned using one brush less than the parallel cleaning sequence required by $\mathfrak{P} = \{(\omega_t, D_t)\}_{t=0}^K$.

■

As every edge of a tree is a bridge, the next corollary is an easy consequence of the previous result.

Corollary 4.3 *Consider a parallel cleaning sequence cleaning a tree T using $b(T) = pb(T)$ brushes. The set of vertices cleaned at each time step is an independent set.*

Theorem 4.4 *For any tree T, $cpb(T) = b(T) = pb(T)$.*

Proof: Consider a parallel cleaning process $\mathfrak{P}$ cleaning tree T with $b(T)$ brushes. It is clear that we can also clean T using the following sequential cleaning $\mathfrak{C}$: clean (in any order) the set of vertices cleaned at the first step of $\mathfrak{P}$, then the set of vertices cleaned at the second step, and so on. Note also that it follows from Corollary 4.3 that the final configuration of $\mathfrak{C}$ is the same as in $\mathfrak{P}$.

It was shown in [11] that the *sequential* cleaning process is reversible. Given an initial configuration ω_0, let ω_n be the final configuration of brushes after a network G has been sequentially cleaned. It was shown in [11] that using initial configuration $\tau_0 = \omega_n$, G can be sequentially cleaned, yielding final configuration $\tau_n = \omega_0$.

Consequently $\mathfrak{C}$ is reversible which implies that $\mathfrak{P}$ is reversible. ■

4.3 Continual Parallel Cleaning: Complete Bipartite networks

Theorem 4.5 *For any complete bipartite network $K_{m,n}$, $cpb(K_{m,n}) = \lceil mn/2 \rceil$.*

Proof: It was shown in [10] that $b(K_{m,n}) \geq \lceil mn/2 \rceil$; consequently $cpb(K_{m,n}) \geq pb(K_{m,n}) \geq b(K_{m,n}) \geq \lceil mn/2 \rceil$.

Let M and N be the partite sets of $K_{m,n}$, where $M = \{u_1, u_2, \ldots, u_m\}$ and $N = \{v_1, v_2, \ldots, v_n\}$. The proof is broken into two cases: first, assuming m is even and second, assuming both m, n are odd.

Suppose m is even and set

$$\omega_0^0(v) = \begin{cases} n & \text{if } v = u_i \text{ for } i = 1, 2, \ldots, m/2 \\ 0 & \text{otherwise.} \end{cases}$$

In step 1, the first $m/2$ of the u_i's are cleaned and at step 2, all v_j's are cleaned as $\omega_1^0(v_j) = m/2$ for all j. The final configuration is

$$\omega_3^0(v) = \omega_2^0(v) = \begin{cases} n & \text{if } v = u_i \text{ for } i = m/2+1, m/2+2, \ldots, m \\ 0 & \text{otherwise} \end{cases}$$

and as the final configuration is equivalent to the initial one, the process is continual.

Suppose now that both m, n are odd and set

$$\omega_0^0(v) = \begin{cases} 1 & \text{if } v = u_i \text{ for } i = 1, 2, \ldots, (m+1)/2 \\ m & \text{if } v = v_j \text{ for } j = 1, 2, \ldots, (n-1)/2 \\ 0 & \text{otherwise.} \end{cases}$$

At step 1, the first $(n-1)/2$ of the v_j's are cleaned as $\omega_0^0(v_j) = \deg(v_j)$.

$$\omega_1^0(v) = \begin{cases} (n+1)/2 & \text{if } v = u_i \text{ for } i = 1, 2, \ldots, (m+1)/2 \\ (n-1)/2 & \text{if } v = u_i \text{ for } i = (m+1)/2+1, \ldots, m \\ 0 & \text{otherwise.} \end{cases}$$

At step 2, there are $(n+1)/2$ dirty vertices in N, so all vertices in M with $(n+1)/2$ brushes are cleaned.

$$\omega_2^0(v) = \begin{cases} (n-1)/2 & \text{if } v = u_i \text{ for } i = (m+1)/2+1, \ldots, m \\ (m+1)/2 & \text{if } v = v_j \text{ for } j = (n+1)/2, \ldots, n \\ 0 & \text{otherwise.} \end{cases}$$

At step 3, there are $(m-1)/2$ dirty vertices in M, so all vertices in N with $(m+1)/2$ brushes are cleaned. Every edge has now been cleaned and

$$\omega_4^0(v) = \omega_3^0(v) = \begin{cases} n & \text{if } v = u_i \text{ for } i = (m+1)/2+1, \ldots, m \\ 1 & \text{if } v = v_j \text{ for } j = (n+1)/2, \ldots, n \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to see that the network can be cleaned a second time: Let $\omega_0^1 = \omega_4^0$. Then $(m-1)/2$ vertices of M are cleaned in the first step, each sending a brush to each vertex in N. Then $(n+1)/2$ vertices in N each have $(m+1)/2$ brushes while the other vertices of N each have $(m-1)/2$. In the second step, the $(n+1)/2$ vertices of N with $(m+1)/2$ brushes are cleaned. Then the $(m+1)/2$ dirty vertices of M each have $(n+1)/2$ brushes and the $(n-1)/2$ dirty vertices of N each have $(m-1)/2$ brushes. In the third step, each dirty vertex of M is cleaned, leaving a brush at each, and each dirty vertex of N now has m brushes. Finally, $\omega_3^1 = \omega_4^1$ and as $\omega_4^1 = \omega_0^0$, the process must be continual.

In both cases, the network is cleaned with $\lceil mn/2 \rceil$ brushes. ∎

Let $G = (V, E)$ be a bipartite network with initial configuration ω_0 that will parallel clean G using $pb(G) = b(G)$ brushes (note that we are considering G to be cleaned *once*). One might be tempted to suggest that if the vertices cleaned at each step form an independent set, then $cpb(G) = pb(G) = b(G)$. This is actually not the case. If the vertices cleaned at each step form an independent set, then G may be cleaned a second time, but it does not guarantee that G can be continually cleaned using the initial configuration ω_0. This is easily demonstrated by the example in Figure 4. Given the initial configuration in Figure 4 (a), network G can be cleaned such that the vertices cleaned at each step in the process form an independent set. The vertices cleaned at each time step *mod* 2 are indicated in Figure 4 (b). However, note from the explanation of Figure 3, that the network can be cleaned a second time, but not a third time.

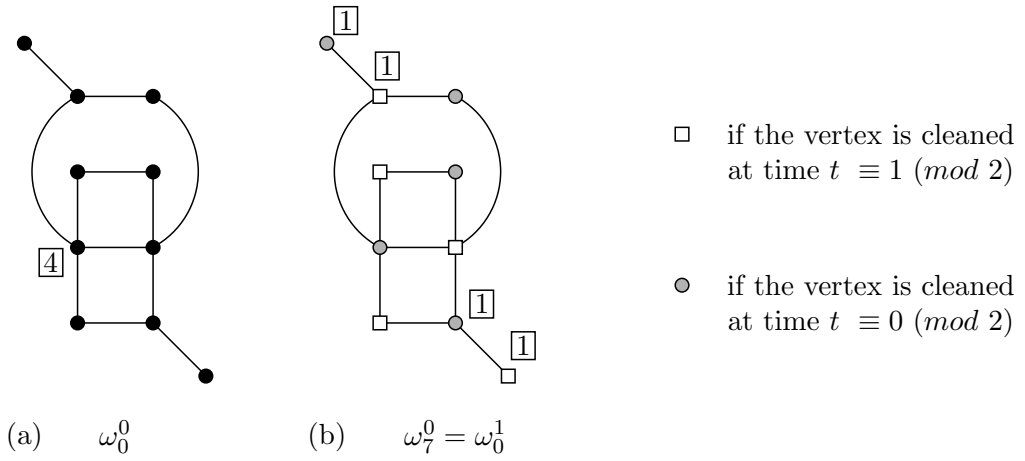


Figure 4: A network G with several initial configurations.

It is interesting to note, however, that given the initial configuration shown in Figure 5 (a), it is easy to see that G can be continually cleaned using $cpb(G) = b(G)$ brushes.

The final configuration for the first cleaning can be used as an initial configuration for the second cleaning, whose final configuration is the same as the initial configuration of the first round.

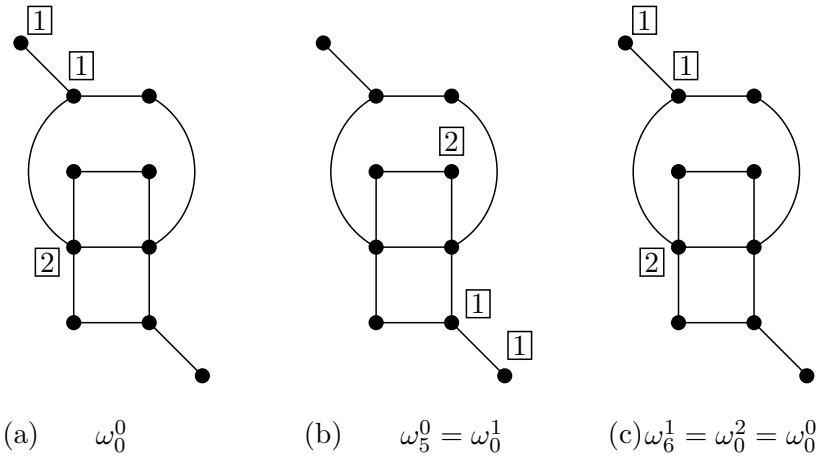


Figure 5: A network G with an initial configuration such that $cpb(G) = b(G)$.

This leads to the following question: does there always exist an initial configuration that yields $cpb(G) = b(G)$ for a bipartite network G? The answer to this question is a resounding 'no' and an example of a bipartite network G for which $cpb(G) > b(G)$ is given in Figure 6. One can easily determine that $b(G) \leq 4$ (in fact $b(G) = 4$), by placing three brushes at x_2 and one brush at y_2 in the initial configuration. We now try to clean G in parallel using four brushes. There are only five initial configurations that can be used to clean network G from Figure 6 using four brushes:

(1) $$\omega_0(v) = \begin{cases} 3 & if\ v = x_2 \\ 1 & if\ v = y_2 \\ 0 & otherwise; \end{cases}$$

(2) $$\omega_0(v) = \begin{cases} 3 & if\ v = y_2 \\ 1 & if\ v = x_2 \\ 0 & otherwise; \end{cases}$$

(3) $$\omega_0(v) = \begin{cases} 2 & if\ v = x_1 \\ 1 & if\ v = x_2 \\ 1 & if\ v = y_2 \\ 0 & otherwise; \end{cases}$$

(4) $$\omega_0(v) = \begin{cases} 2 & if\ v = y_1 \\ 1 & if\ v = y_2 \\ 1 & if\ v = x_2 \\ 0 & otherwise; \end{cases}$$

(5) $$\omega_0(v) = \begin{cases} 1 & if\ v = x_4 \\ 1 & if\ v = x_6 \\ 1 & if\ v = y_4 \\ 1 & if\ v = y_6 \\ 0 & otherwise; \end{cases}$$

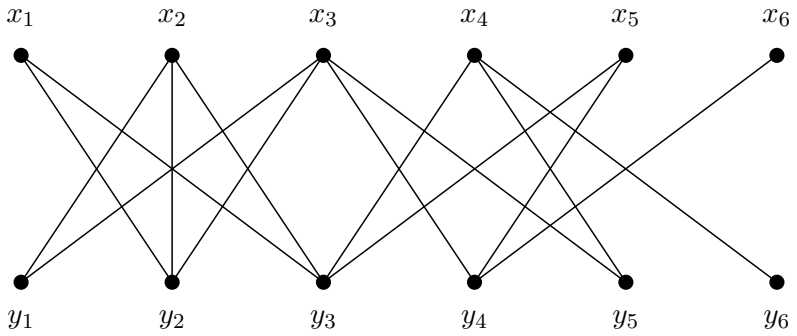


Figure 6: A bipartite network G for which $cpb(G) > b(G)$.

If G is parallel cleaned using initial configuration (1), (2), (3), or (4), the final configuration is (5). However, if G is cleaned using initial configuration (5), the final configuration, shown in Figure 7, cannot be used to clean the network a second time. Consequently $cpb(G) > pb(G) = b(G)$ for the network shown in Figures 6 and 7. Given the initial configuration with two brushes at each of u_1, v_1, one can easily determine that $cpb(G) = 5$. See Figure 8 for a configuration of 5 brushes that continually clean.

This leads us to the following question.

Question 4.6 *For a bipartite network G, can the difference between $cpb(G)$ and $b(G)$ be arbitrarily large?*

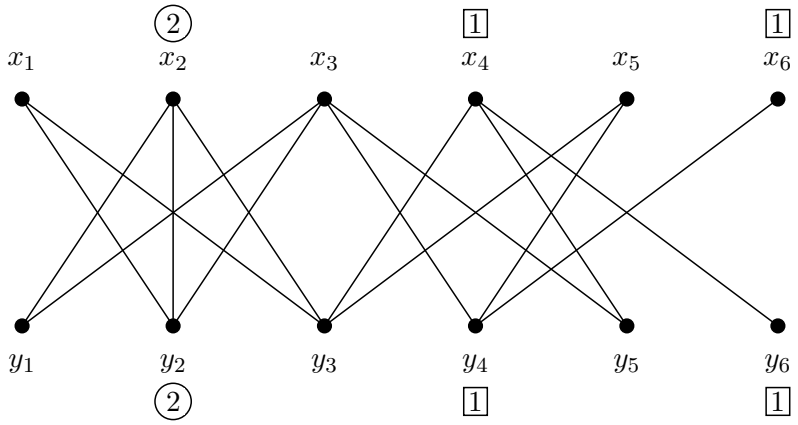


Figure 7: A bipartite network G with initial configuration (5) and final configuration indicated with circles.

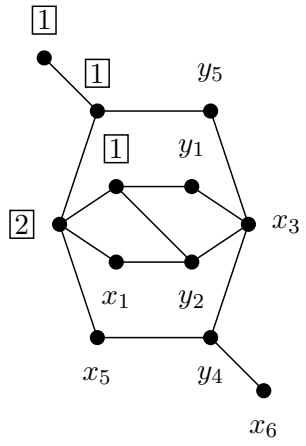


Figure 8: Alternative drawing of the network in Figure 6 with 5 brushes that continually clean.

Since there are a lot of non-isomorphic (connected) bipartite networks on n vertices, it is impossible to check all possible graphs without computer support. We implemented and ran programs written in C/C++ to determine that $cpb(G) = b(G)$ for every bipartite network $G = (V, E)$ with $|V| \leq 11$; Figure 6 illustrates the only network on 12 vertices for which $cpb(G) \neq b(G)$ (the programs can be downloaded from [18]).

4.4 Continual Parallel Cleaning: Complete networks

In this section, we show that the continual parallel brush number for complete networks K_n is bounded between $5/16n^2 + O(n)$ and $4/9n^2 + O(n)$ by Theorems 4.7 and Corollary 4.11. Initial configurations that will continually parallel clean K_n using $4/9n^2 + O(n)$ brushes are also given in Theorems 4.8, 4.9, and 4.10.

As determined in Section 4.1, $cpb(K_3) = 3$. Similarly, one can easily determine that $cpb(K_4) = 5$. However, determining $cpb(K_n)$ for larger values of n is certainly not trivial and this section concludes with a table of calculated values of $cpb(K_n)$.

Theorem 4.7 *For any complete network K_n, $cpb(K_n) \geq \frac{5}{16}n^2 + O(n)$.*

Proof: Label the vertices of K_n as $v_0, v_1, \ldots, v_{n-1}$ and note that

$$cpb(K_n) \geq pb(K_n) = b(K_n) = \lfloor n^2/4 \rfloor.$$

It was determined in [11] that to clean K_n with $\lfloor n^2/4 \rfloor$ brushes, we must use the following assignment:

$$\omega_0^0(v_i) = \begin{cases} n - 2i - 1 & \text{if } i \leq \lfloor \frac{n-1}{2} \rfloor \\ 0 & \text{otherwise.} \end{cases}$$

Note that up to a relabelling of the vertices, none of the vertices can have less brushes than given in the assignment. It is easy to see that after K_n has been parallel cleaned once, the final configuration is

$$\omega_T^0(v_i) = \begin{cases} 0 & \text{if } i \leq \lfloor \frac{n-1}{2} \rfloor \\ \lfloor \frac{n}{2} \rfloor & \text{otherwise.} \end{cases}$$

After the first cleaning, there are $\lceil n/2 \rceil$ vertices, each with $\lfloor n/2 \rfloor$ brushes. However, if we wish to continually clean K_n, then in the final configuration (of the first cleaning), there must be at least one vertex which has at least $n - 1$ brushes, at least one additional vertex with at least $n - 3$ brushes, and so on.

More precisely, if $\lceil n/2 \rceil$ is even, then to continually parallel clean K_n, at least an additional

$$1 + 3 + 5 + \cdots + \lceil n/2 \rceil - 1 = \lceil n/2 \rceil^2/4$$

brushes are required. If $\lceil n/2 \rceil$ is odd, then to continually parallel clean K_n, at least an additional

$$2 + 4 + 6 + \cdots + \lceil n/2 \rceil - 1 = (\lceil n/2 \rceil^2 - 1)/4$$

brushes are required.

$$cpb(K_n) \geq \begin{cases} \frac{5}{16}n^2 & \text{if } n \text{ is even and } \lceil n/2 \rceil \text{ is even} \\ \frac{5}{16}n^2 - \frac{1}{4} & \text{if } n \text{ is even and } \lceil n/2 \rceil \text{ is odd} \\ \frac{5}{16}n^2 + \frac{1}{8}n - \frac{3}{16} & \text{if } n \text{ is odd and } \lceil n/2 \rceil \text{ is even} \\ \frac{5}{16}n^2 + \frac{1}{8}n - \frac{7}{16} & \text{if } n \text{ is odd and } \lceil n/2 \rceil \text{ is odd.} \end{cases}$$

■

We now determine upper bounds for the continual parallel brush number of K_n. Note that it is broken into three results: $n = 3k$ in Theorem 4.9; $n = 3k + 1$ in Theorem 4.10; and $n = 3k + 2$ in Theorem 4.8. We only give the proof of Theorem 4.8 since all the initial configurations, and hence the proofs, are very similar. To help work through the construction we first present an overview.

The vertices are cleaned in three phases:

1) Initially, a set A of approximately k vertices is cleaned by starting with the only primed vertex (the only vertex ready to be cleaned), then the next two, then the next four, and so on, although the cardinality of the last subset of vertices cleaned need not be a power of 2.

2) The remaining approximately $2k$ vertices partition into two almost equal sized sets, B and C. In phase 2, all of B's vertices are cleaned.

3) Finally all of C's vertices are cleaned in phase 3, but being a clique, the number of brushes at each vertex of C remains the same.

For example with $k = 5$, $n = 3 \cdot 5 + 1$, we give the number of brushes and illustrate the cleaning of the vertices, underlined vertices are cleaned at the next time step:

	C					B						A				
	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}
Phase 1	0	1	2	3	4	5	5	5	5	5	5	11	12	13	14	<u>15</u>
	1	2	3	4	5	6	6	6	6	6	6	12	13	<u>14</u>	<u>15</u>	0
	3	4	5	6	7	8	8	8	8	8	8	<u>14</u>	<u>15</u>	1	2	0
Phase 2	5	6	7	8	9	<u>10</u>	<u>10</u>	<u>10</u>	<u>10</u>	<u>10</u>	<u>10</u>	3	4	1	2	0
Phase 3	<u>11</u>	<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	5	5	5	5	5	5	3	4	1	2	0
Final Config.	11	12	13	14	15	5	5	5	5	5	5	3	4	1	2	0

Theorem 4.8 *For any complete network on $n = 3k + 2$ vertices,*

$$cpb(K_{3k+2}) \leq 4k^2 + 4k + 2.$$

Proof: Let $n = 3k + 2$, label the vertices of K_n as $v_0, v_1, \ldots, v_{3k+1}$, and set

$$\omega_0(v_i) = \begin{cases} k+1 & \text{if } i = k, k+1, \ldots, 2k+1 \\ i & \text{otherwise.} \end{cases}$$

Consider the vertices v_i, $2k + 2 \leq i \leq 3k + 1$, that is, those that we claim are cleaned in phase 1. At time 1, only v_{3k+1} is cleaned. By induction, 2^{t-1} vertices are cleaned at time

t and if v_i has not been cleaned, then $\omega_{t-1}(v_i) < D_{t-1}(v_i)$ where $\omega_t(v_i) = i + (2^t - 1)$ and $D_t(v_i) = 3k + 1 - (2^t - 1)$. Vertex v_i is cleaned therefore at time t which satisfies

$$\omega_{t-1}(v_i) \geq D_{t-1}(v_i) \text{ and } \omega_{t-2}(v_i) < D_{t-2}(v_i),$$

which implies that

$$3k + 3 - 2^{t-1} > i \geq 3k + 3 - 2^t.$$

It follows that blocks of powers of 2 will be cleaned until the last block which will consist of the remaining vertices. Note that after cleaning at step t, v_i will have

$$\omega_t(v_i) = \omega_{t-1}(v_i) - D_{t-1}(v_i) + 2^{t-1} - 1 = i + 3 \cdot 2^{t-1} - 3k - 4$$

brushes.

For a vertex v_i that is cleaned at step $l = \lceil \log_2(k+1) \rceil$ (the last step of phase 1), since there are a total of $k - (2^{l-1} - 1)$ vertices cleaned at step l,

$$\omega_l(v_i) = \omega_{l-1}(v_i) - D_{l-1}(v_i) + k - (2^{l-1} - 1) - 1 = i + 2^{l-1} - 2k - 3.$$

At step $l + 1$ (phase 2), $D_l(v_i) = 2k + 2$ and $\omega_l(v_i) = \omega_0(v_i) + k$ for all $v_i \in D_l$. Thus, we clean $v_k, v_{k+1}, \ldots, v_{2k+1}$ at step $l + 1$ and

$$\omega_{l+1}(v_i) = \omega_l(v_i) - D_l(v_i) + k + 1 = k$$

for a vertex v_i that is cleaned at step $l + 1$.

At step $l + 2$ (phase 3), $\omega_{l+1}(v_i) = i + 2k + 2$ for all $v_i \in D_{l+1}$ so the remaining k vertices are cleaned, yielding final configuration of

$$\omega_{l+2}(v_i) = \begin{cases} i + 3 \cdot 2^{t_i - 1} - 3k - 4 & \text{for } i = 3k - 2^{l-1} + 3, \ldots, 3k + 1 \\ i + 2^{l-1} - 2k - 3 & \text{for } i = 2k + 2, \ldots, 3k - 2^{l-1} + 2 \\ k & \text{for } i = k, k + 1, \ldots, 2k + 1 \\ i + 2k + 2 & \text{for } i = 0, 1, \ldots, k - 1 \end{cases}$$

where $t_i = \lceil \log_2(3k - i + 3) \rceil$ is the step at which v_i was cleaned.

By renaming the sets and variables, we have an initial configuration $\omega_0^1 = \omega_{l+2}$ equivalent to $\omega_0 = \omega_0^0$. ∎

Theorem 4.9 *Let $n = 3k$ and label the vertices of K_n as $v_0, v_1, \ldots, v_{3k-1}$. If*

$$\omega_0(v_i) = \begin{cases} k & \text{if } i = k - 1, k, \ldots, 2k - 1 \\ i & \text{otherwise,} \end{cases}$$

then K_{3k} can be continual parallel cleaned using this initial configuration of $4k^2 - k + 1$ brushes.

Theorem 4.10 *Let $n = 3k + 1$ and label the vertices of K_n as $v_0, v_1, \ldots, v_{3k}$. If*

$$\omega_0(v_i) = \begin{cases} k & \text{if } i = k, k + 1, \ldots, 2k \\ i & \text{otherwise,} \end{cases}$$

then K_{3k+1} can be continual parallel cleaned using this initial configuration of $4k^2 + k$ brushes.

From Theorem 4.9, Theorem 4.10, and Theorem 4.8:

Corollary 4.11

$$cpb(K_n) \leq \begin{cases} 4/9n^2 - 1/3n + 1 & \text{if } n = 3k \\ 4/9n^2 - 5/9n + 1/9 & \text{if } n = 3k+1 \\ 4/9n^2 - 4/9n + 10/9 & \text{if } n = 3k+2. \end{cases}$$

So in general, $cpb(K_n) \leq 4/9n^2 + O(n)$.

We implemented and ran programs written in C/C++ to determine the values of $cpb(K_n)$ for $n = 3, 4, \ldots, 20$ — see Table 1 (the programs and results can be downloaded from [18]). It seems that there should be at least one initial configuration with the maximum number of $n-1$ brushes on any vertex that can be used to start the process of cleaning K_n that continues forever, but no proof of this fact is known. However, by checking all such configurations we were able to find an upper bound for $cpb(K_n)$ for $n = 21, 22, \ldots, 26$. We also conjecture that those are, in fact, the exact values.

n	$b(K_n)$	$cpb(K_n)$	n	$b(K_n)$	$cpb(K_n)$	n	$b(K_n)$	$cpb(K_n)$
3	2	3	11	30	50	19	90	150
4	4	5	12	36	60	20	100	170
5	6	10	13	42	66	21	110	≤ 173
6	9	15	14	49	76	22	121	≤ 197
7	12	18	15	56	95	23	132	≤ 214
8	16	25	16	64	105	24	144	≤ 243
9	20	33	17	72	122	25	156	≤ 258
10	25	39	18	81	135	26	169	≤ 279

Table 1: $b(G)$ and $cpb(K_n)$ for $n = 3, 4, \ldots, 26$.

Figure 9 plots $b(K_n)/cpb(K_n)$ for $n = 3, 4, \ldots, 20$ and it remains an open question to determine the value of $\lim_{n\to\infty} b(K_n)/cpb(K_n)$ if it exists. From Table 1, the upper bound of Theorems 4.10, 4.8 and 4.9 sometimes give the minimum number of brushes. Thus, we conjecture that:

$$\lim_{n\to\infty} b(K_n)/cpb(K_n) = (1/4)/(4/9) = 9/16.$$

5 Acknowledgement

This work was made possible by the facilities of

- the Shared Hierarchical Academic Research Computing Network SHARCNET, Ontario, Canada (www.sharcnet.ca): 8,082 CPUs,
- the Atlantic Computational Excellence Network ACEnet, Memorial University of Newfoundland, St. John's, NL, Canada (www.ace-net.ca): 412 CPUs.

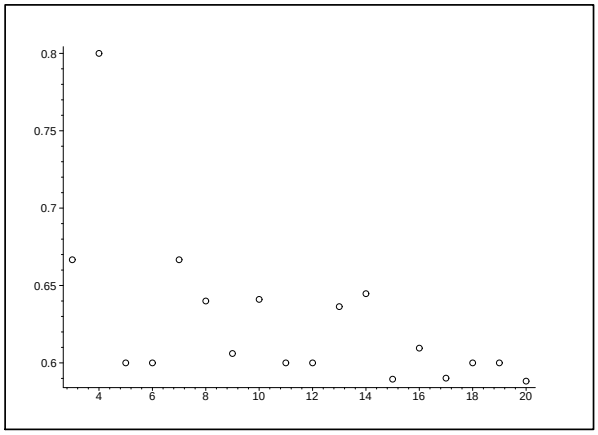


Figure 9: A network of $b(K_n)/cpb(K_n)$ versus n (from 3 to 20).

References

[1] B. Alspach, Searching and Sweeping Graphs: A Brief Survey, *Le Matematiche*, 2004, **59**, 5-37.

[2] N. Alon, P. Prałat, and N. Wormald, Cleaning d-regular graphs with brushes, *SIAM Journal on Discrete Mathematics* **23** (2008), 233–250.

[3] J. Bitar, E. Goles, Parallel chip firing games on graphs, *Theoretical Computer Science* **92** (1992), 291–300.

[4] A. Björner, L. Lovász, W. Shor, Chip-firing games on graphs, *European Journal of Combinatorics* **12** (1991) 283–291.

[5] S. Gaspers, M. E. Messinger, R. Nowakowski, and P. Prałat, Clean the graph before you draw it!, *Information Processing Letters* **109** (2009), 463–467.

[6] B. Hobbs and J. Kahabka, 1995, Underwater Cleaning Technique Used for Removal of Zebra Mussels at the Fitzpatrick Nuclear Power Plant. *Proceedings of The Fifth International Zebra Mussel and Other Aquatic Nuisance Organisms Conference*, Toronto, Canada, February 1995.

[7] S. R. Kotler, E. C. Mallen, and K. M. Tamms, Robotic Removal of Zebra Mussel Accumulations in a Nuclear Power Plant Screenhouse, *Proceedings of The Fifth International Zebra Mussel and Other Aquatic Nuisance Organisms Conference*, Toronto, Canada, February 1995.

[8] S. McKeil, Chip Firing Cleaning Processes, MSc Thesis, Dalhousie University, 2007.

[9] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou, The complexity of searching a graph, *Journal of the Association for Computing Machinery,* 1988, **35**, 18–44.

[10] M. E. Messinger, Methods of Decontaminating a Network, PhD Thesis, Dalhousie University, 2008.

[11] M. E. Messinger, R. J. Nowakowski, and P. Prałat, Cleaning a Network with Brushes, *Theoretical Computer Science* **399** (2008) 191–205.

[12] M. E. Messinger, R. J. Nowakowski, and P. Prałat, Cleaning with Brooms, *Graphs and Combinatorics*, submitted, 14pp.

[13] M. E. Messinger, R. J. Nowakowski, P. Prałat, and N. Wormald, Cleaning random d-regular graphs with brushes using a degree-greedy algorithm, *Proceedings of the 4th Workshop on Combinatorial and Algorithmic Aspects of Networking (CAAN2007)*, Lecture Notes in Computer Science, Springer, 2007, 13–26.

[14] M. E. Messinger, R. J. Nowakowski, The Robot Cleans Up, *Proceedings of the 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA'08)*, Lecture Notes in Computer Science, Springer, 2008, 309–318.

[15] M. E. Messinger, R. J. Nowakowski, The Robot Cleans Up, *Journal of Combinatorial Optimization*, to appear.

[16] P. Prałat, Cleaning random d-regular graphs with Brooms, *Graphs and Combinatorics*, submitted, 22pp.

[17] P. Prałat, Cleaning random graphs with brushes, *Australasian Journal of Combinatorics* **43** (2009), 237–251.

[18] P. Prałat, programs written in C/C++, `http://www.math.wvu.ca/~pralat/index.php?page=publications`.