Cleaning a Network with Brushes

M. E. Messinger, R. J. Nowakowski, and P. Prałat *
Department of Math. & Stats.
Dalhousie University,
Halifax, NS, B3H 3J5, Canada.

Abstract

Following the decontamination metaphor for searching a graph, we introduce a cleaning process, which is related to both the chip-firing game and edge searching. Brushes (instead of chips) are placed on some vertices and, initially, all the edges are dirty. When a vertex is 'fired', each dirty incident edge is traversed by only one brush, cleaning it, but a brush is not allowed to traverse an already cleaned edge; consequently, a vertex may not need degree-many brushes to fire. The model presented is one where the edges are continually recontaminated, say by algae, so that cleaning is regarded as an on-going process. Ideally, the final configuration of the brushes, after all the edges have been cleaned, should be a viable starting configuration to clean the graph again. We show that this is possible with the least number of brushes if the vertices are fired sequentially but not if fired in parallel. We also present bounds for the least number of brushes required to clean graphs in general and some specific families of graphs.

Key words: searching, chip-firing, cleaning sequence, brush number.

AMS subject classification: 05C38, 05C78, 05C85

1 Introduction

In [19, 20], Parsons introduced the problem of searchers looking for a lost spelunker in a network of caves (see [1] for a recent survey of the literature). One condition was that the lost spelunker, or intruder in later literature, was infinitely fast. A new metaphor was introduced to accommodate this infinite speed, that of chemical or biological contamination of the graph – any break in the line of searchers would allow contamination behind them and therefore those vertices or edges would have to be considered recontaminated. In the standard searching models, a searcher can leave any vertex at any time.

In *chip firing* (see [2, 3] for example) there is an initial configuration of chips on vertices and a vertex is 'primed' if it has at least as many chips as its degree. A primed vertex may 'fire' whereupon it sends one chip along each incident edge. The main questions considered have been variants of "does this process stop or can it continue forever?"; "how many chips are needed to produce a cycle?" and "how long before a cycle?".

*Partially supported by grants from the NSERC and MITACS. messnger@mathstat.dal.ca, rjn@mathstat.dal.ca, pralat@mathstat.dal.ca

The *cleaning* model, introduced in [15], is a combination of chip firing and searching. We envision a network of pipes that have to be periodically cleaned of a contaminant that regenerates, say algae. This is accomplished by having cleaning agents, colloquially, 'brushes', assigned to some vertices. To reduce the recontamination, when a vertex is 'cleaned', a brush must travel down each contaminated edge. Once a brush has traversed an edge, that edge has been *cleaned*. A graph G has been *cleaned* once every edge of G has been cleaned. McKeil [15] considered the model where more than one brush can travel down an edge and brushes can travel down cleaned edges. The particular version in this paper allows only one brush to travel along an edge and a brush is not allowed to travel down an edge that has already been cleaned. One condition that this model has, like chip-firing but not searching, is that the cleaning process is to be automatic, i.e. a union of 'vertex firing' sequences where each sequence cleans the graph, continuing on for the lifetime of the network. Therefore, the problems to solve are: firstly, a brush configuration and corresponding vertex firing sequence that cleans the graph; and secondly, having the final configuration of brushes be a starting configuration for another vertex firing sequence that also cleans the graph; and so on.

The model is similar to the *mutating* chip firing game [7, 9, 14] where when a vertex fires the edges traversed by a chip may be removed but others may also be added. The model used in [7, 9, 14] considered directed graphs obtained by replacing every undirected edge by a pair of directed edges.

In a graph G, $|E(G)|$ many chips are required for a configuration to give an infinite (repeating) chip firing game. Finding a configuration is easy [17]: start with any configuration where each vertex has at least as many chips as its degree and identify each chip with the edge that it is first fired down. When the new vertex is fired, the same chip goes back along the same edge. When the configuration repeats, any chip that has not been identified with an edge is removed. This gives a recurrent configuration with $|E(G)|$ many chips.

However, for the cleaning game, *What is the minimum number of brushes required to clean G?* and *What is the complexity of finding it?* are open questions. As the edges cleaned at each step are all incident to a vertex $v \in G$, for the purposes of this paper, it is more convenient to define the cleaning process in terms of the vertices. Initially, all vertices are *dirty* and we say a vertex is *cleaned* when its associated brushes are fired down the incident dirty edges. Note that with these definitions, a vertex may be dirty but be incident with only clean edges. For example, given a path with three vertices a, b, c, put two brushes on b and clean b. The incident edges ab and bc are now both cleaned but the vertices a and c are still dirty even though their incident edges are clean. In this paper, we will insist on cleaning the vertices a and b despite the fact that no edges will be cleaned. (Note that in this example, starting with just one brush on a is sufficient to be able to clean the graph.)

Figure 1 illustrates the cleaning process for a graph G where there are initially 2 brushes at vertex a. The solid edges indicate dirty edges while the dotted edges indicate clean edges. First, vertex a is cleaned, sending a brush to each of vertices b, c. Second, vertex b is cleaned, sending a brush to c. Vertex c now has 2 brushes and 1 dirty edge; it is cleaned and sends one brush to vertex d. At this step, both c, d have one brush and although G contains no dirty edges, we still clean vertex d, by sending a brush down each dirty edge (of which there are none). Thus, G has been cleaned.

It is important to note that we are 'cleaning' each vertex of G: G may contain no dirty edges after step t, but we still must 'clean' the remaining vertices (as described in Figure 1).

To recap, in our model every edge in a graph G is initially dirty and a fixed number of

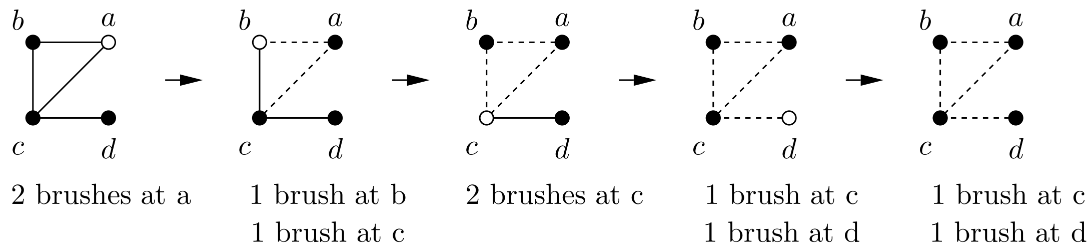


Figure 1: An example of the cleaning process for graph G.

brushes begin on a set of vertices. At each step of the process, vertex v may be *cleaned* (instead of fired) if there are at least as many brushes on v as there are dirty incident edges. When it is cleaned, every dirty edge must be traversed by one (and only one) brush, moreover, brushes cannot traverse a clean edge. Other cleaning rules are considered in [15]. Our approach of focusing on cleaning vertices instead of edges makes the proofs more transparent for Theorem 4.1, an upper bound on the brushes required for the Cartesian Product; Theorem 2.3, which shows that a cleaning sequence can be run in reverse which addresses the hoped-for automatic nature of the cleaning process; Theorem 2.1, the final dirty set of vertices depends only on the initial configuration; and Theorem 2.2, where we show that with an initial configuration of brushes, the graph can be cleaned by sequential cleaning if and only if it can be cleaned with parallel cleaning of vertices. In this paper, we concentrate on the sequential cleaning mode. In Section 2 we present the important basic results for cleaning; in Section 3 we give several lower bounds on the least number of brushes required; in Section 4 we give upper bounds for the Cartesian product and particularly for hypercubes; in Section 5 we apply some of the earlier results to obtain exact numbers for or bounds on the number of brushes. In Section 6 we consider the graphs for which given an initial configuration, there is a unique cleaning sequence that cleans the graph, in particular we give a constructive proof of the maximum number of edges such a graph can contain.

Formally, at each step t, $\omega_t(v)$ denotes the number of brushes at vertex v ($\omega_t : V \to \mathbb{N} \cup \{0\}$) 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 1.1 *The* ***cleaning process*** $\mathfrak{P}(G, \omega_0) = \{(\omega_t, D_t)\}_{t=0}^{T}$ *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* $\omega_t(\alpha_{t+1}) \geq D_t(\alpha_{t+1})$*. If no such vertex exists, then stop the process (*$T = t$*), return the* ***cleaning sequence*** $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_T)$*, the* ***final set of dirty vertices*** D_T*, and the* ***final configuration of brushes*** ω_T

(2) *Clean* α_{t+1} *and all dirty incident edges by traversing 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) *$t := t + 1$ and go back to* **(1)**

Note that for a graph G and initial configuration ω_0, the cleaning process can return different cleaning sequences and final configurations of brushes; consider, for example, an isolated edge uv and $\omega_0(u) = \omega_0(v) = 1$. We will show in Theorem 2.1, however, that the final set of dirty vertices is determined by G and ω_0. Thus, the following definition is natural.

Definition 1.2 *A graph $G = (V, E)$* ***can be cleaned*** *by the initial configuration of brushes ω_0 if the cleaning process $\mathfrak{P}(G, \omega_0)$ returns an empty final set of dirty vertices ($D_T = \emptyset$).*

Let the brush number, $b(G)$, be the minimum number of brushes needed to clean G, that is,

$$b(G) = \min_{\omega_0 : V \to \mathbb{N} \cup \{0\}} \Big\{ \sum_{v \in V} \omega_0(v) : G \text{ can be cleaned by } \omega_0 \Big\}.$$

Similarly, $b_\alpha(G)$ is defined as the minimum number of brushes needed to clean G using the cleaning sequence α.

It is clear that for every cleaning sequence α, $b_\alpha(G) \geq b(G)$ and $b(G) = \min_\alpha b_\alpha(G)$. (The last relation can be used as an alternative definition of $b(G)$.) In general, it is difficult to find $b(G)$, but $b_\alpha(G)$ can be easily computed. To do this, it seems better not to choose the function ω_0 in advance, but to run the cleaning process in some order, and compute the initial number of brushes needed to clean a vertex. We can adjust ω_0 along the way, letting

$$\omega_0(\alpha_{t+1}) = \max\{2D_t(\alpha_{t+1}) - \deg(\alpha_{t+1}), 0\}, \qquad \text{for } t = 0, 1, \ldots, |V| - 1, \tag{1}$$

since that is how many brushes we have to add over and above what we get for free.

When a graph G is cleaned using the cleaning process, each edge of G is traversed exactly once and by exactly one brush which gives rise to the following definition.

Definition 1.3 *Given some initial configuration ω_0 of brushes, suppose $G = (V, E)$ admits a cleaning sequence $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_T)$ which cleans G. As each edge in G is traversed exactly once and by exactly one brush, an orientation of the edges of G is permitted such that for every $\alpha_i \alpha_j \in E(G)$, $\alpha_i \to \alpha_j$ if and only if $i < j$.*

The ***brush path*** *of a brush b is the oriented path formed by the set of edges cleaned by b (note that a vertex may not be repeated in a brush path). Then G can be decomposed into $b_\alpha(G)$ oriented brush paths (note that no brush can stay at its initial vertex in the minimal brush configuration).*

Alternately, we can consider following variation of the above process: at each step, instead of cleaning just one vertex, we clean all vertices which are ready to be cleaned. In general, therefore, cleaning in parallel will terminate before cleaning one vertex at a time.

Definition 1.4 *The* ***parallel cleaning process*** *$\mathfrak{C} = \{(\omega_t, D_t)\}_{t=0}^{K}$ 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 $\rho_{t+1} \subseteq D_t$ 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 dirty incident edges by traversing a brush from v to each dirty neighbour. More precisely, $D_{t+1} = D_t \setminus \rho_{t+1}$, for every $v \in \rho_{t+1}$, $\omega_{t+1}(v) = \omega_t(v) - D_t(v) + |N(v) \cap \rho_{t+1}|$, and for every $u \in D_{t+1}$, $\omega_{t+1}(u) = \omega_t(u) + |N(u) \cap \rho_{t+1}|$ the other values of ω_{t+1} remain the same as in ω_t*

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

Let the parallel brush number, $pb(G)$, be the minimum number of brushes needed to clean G.

Note that with parallel cleaning, two adjacent vertices can be cleaned at the same time and the common edge will have two brushes traverse it in opposite directions. The brushes, therefore, may not decompose the graph into oriented (brush) paths.

2 General Results

Consider the cleaning process $\mathfrak{P}(G, \omega_0) = \{(\omega_t, D_t)\}_{t=0}^{T}$. Note that if v is dirty at step t, then $\omega_t(v)$ is a function of G, ω_0, and D_t, namely,

$$\begin{aligned} \omega_t(v) &= \omega_0(v) + \deg(v) - D_t(v) \\ &= \omega_0(v) + \deg(v) - |N(v) \cap D_t|, \end{aligned} \tag{2}$$

since $\omega_t(v)$ cannot be decreased during that period of time and each edge incident to v which was cleaned before time t increased the number of brushes at v by 1.

Theorem 2.1 *Given a graph G and the initial configuration of brushes ω_0, the cleaning algorithm returns a unique final set of dirty vertices.*

Proof: Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_T)$ and $\beta = (\beta_1, \beta_2, \ldots, \beta_U)$ be two cleaning sequences of the cleaning processes $\mathfrak{P}_\alpha = \{(\omega_t, D_t)\}_{t=0}^{T}$ and $\mathfrak{P}_\beta = \{(\tau_t, C_t)\}_{t=0}^{U}$, respectively ($\omega_0 = \tau_0$). Note that it is enough to prove that $\{\alpha_1, \alpha_2, \ldots, \alpha_T\} = \{\beta_1, \beta_2, \ldots, \beta_U\}$.

Suppose that there is a vertex in β which is not in α. Let β_l, $1 \leq l \leq U$, be the first such vertex. Consider now $\mathfrak{P}_\alpha$ at the final step T and $\mathfrak{P}_\beta$ at step $l-1$. Clearly $\omega_T(\beta_l) < D_T(\beta_l)$ and, since α contains vertices $\beta_1, \beta_2, \ldots, \beta_{l-1}$, $D_T(\beta_l) \leq C_{l-1}(\beta_l)$. Using (2) we get

$$\begin{aligned} \omega_T(\beta_l) &= \omega_0(\beta_l) + \deg(\beta_l) - D_T(\beta_l) \\ &\geq \tau_0(\beta_l) + \deg(\beta_l) - C_{l-1}(\beta_l) \\ &= \tau_{l-1}(\beta_l)\,. \end{aligned}$$

Since β_l was cleaned at step l of the process $\mathfrak{P}_\beta$, $\tau_{l-1}(\beta_l) \geq C_{l-1}(\beta_l)$. Thus,

$$\omega_T(\beta_l) \geq \tau_{l-1}(\beta_l) \geq C_{l-1}(\beta_l) \geq D_T(\beta_l)$$

which gives us a contradiction.

A symmetric argument can be used to show that β contains all vertices of α. So α is a permutation of β and the assertion holds. ■

Actually, a more general theorem is true. Take any vertex deletion algorithm where, once a vertex can be deleted, further deletions of other vertices do not change that fact (the cleaning algorithm is of this type). Then the result of the algorithm will always be the same. This is easy to see: take one run of the algorithm and let S_i be the set of vertices deleted after i steps. By induction on i, all runs of the algorithm must eventually remove all vertices in S_i.

Theorem 2.2 *For any graph G, $b(G) = pb(G)$.*

Proof: It is clear that $b(G) \leq pb(G)$: if (G, ω_0) can be cleaned using a parallel cleaning sequence $\rho = (\rho_1, \rho_2, \ldots, \rho_K)$, then (G, ω_0) can also be cleaned using, as a (sequential) cleaning sequence, any permutation of ρ_1, then any permutation of ρ_2, and so on. Thus, it is enough to show that $pb(G) \leq b(G)$.

Let $n = |V(G)|$ and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ be a cleaning sequence of the process $\mathfrak{P}(G, \omega_0) = \{(\omega_t, D_t)\}_{t=0}^{n}$ such that $b(G)$ brushes are used to clean G. For a contradiction, suppose that $pb(G) > b(G)$, that is, the parallel process $\mathfrak{C}(G, \omega_0) = \{(\tau_t, C_t)\}_{t=0}^{K}$ $(\tau_0 = \omega_0)$ returns a nonempty set of dirty vertices C_K. Let $i_0 = \min\{i \in [n] : \alpha_i \in C_K\}$. Using a similar argument as in Theorem 2.1, we can show that α_{i_0} can be cleaned at step $K+1$ of $\mathfrak{C}(G, \omega_0)$. This contradiction finishes the proof. ■

Theorem 2.3 ***The Reversibility Theorem***
Given the initial configuration ω_0, suppose G can be cleaned yielding final configuration ω_n, $n = |V(G)|$. Then, given initial configuration $\tau_0 = \omega_n$, G can be cleaned yielding the final configuration $\tau_n = \omega_0$.

Proof: Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ be a cleaning sequence of the cleaning process $\mathfrak{P}_+ = \{(\omega_t, D_t)\}_{t=0}^{n}$ which will clean graph G. Let $N^-(\alpha_t) = |\{\alpha_t\alpha_i \in E(G) \ : \ i < t\}|$ and similarly $N^+(\alpha_t) = |\{\alpha_t\alpha_i \in E(G) : i > t\}|$, clearly $\deg(\alpha_t) = N^-(\alpha_t) + N^+(\alpha_t)$. Vertex α_t is dirty at time $t-1$, so using (2) we have

$$\begin{aligned}
\omega_n(\alpha_t) &= \omega_t(\alpha_t) = \omega_{t-1}(\alpha_t) - D_{t-1}(\alpha_t) \\
&= \omega_0(\alpha_t) + \deg(\alpha_t) - D_{t-1}(\alpha_t) - D_{t-1}(\alpha_t) \\
&= \omega_0(\alpha_t) + \deg(\alpha_t) - N^+(\alpha_t) - N^+(\alpha_t) \\
&= \omega_0(\alpha_t) + N^-(\alpha_t) - N^+(\alpha_t).
\end{aligned} \tag{3}$$

We show now that the cleaning process $\mathfrak{P}_- = \{(\tau_t, C_t)\}_{t=0}^{U}$, $\tau_0 = \omega_n$, can be used to clean G using a cleaning sequence $(\alpha_n, \alpha_{n-1}, \ldots, \alpha_1)$, that is, vertex α_{n-t+1} is cleaned at time t. We use induction on t. Since

$$\tau_0(\alpha_n) = \omega_n(\alpha_n) = \omega_0(\alpha_n) + N^-(\alpha_n) - N^+(\alpha_n) \geq N^-(\alpha_n) = C_0(\alpha_n),$$

vertex α_n can be cleaned at the first step and the basis step is verified. For the induction step, assume that vertices $\alpha_n, \alpha_{n-1}, \ldots, \alpha_k$ $(n \leq k < 1)$ are clean at time $n-k+1$ of the process $\mathfrak{P}_-$. It is not difficult to check, again using (2) and (3), that α_{k-1} can be cleaned in a very next step. Indeed,

$$\begin{aligned}\tau_{n-k+1}(\alpha_{k-1}) &= \tau_0(\alpha_{k-1}) + \deg(\alpha_{k-1}) - C_{n-k+1}(\alpha_{k-1}) \\ &= \omega_n(\alpha_{k-1}) + N^+(\alpha_{k-1}) \quad = \quad \omega_0(\alpha_{k-1}) + N^-(\alpha_{k-1}) \\ &\geq N^-(\alpha_{k-1}) \quad = \quad C_{n-k+1}(\alpha_{k-1}).\end{aligned}$$

To finish the proof it is enough to show that $\tau_n = \omega_0$. Using a similar calculation as in (3) we get $\tau_n(\alpha_t) = \tau_0(\alpha_t) + N^+(\alpha_t) - N^-(\alpha_t)$. Now, replacing $\tau_0(\alpha_t)$ by $\omega_n(\alpha_t)$ and using (3), one can check that the assertion follows. ∎

The concept of reversibility, however, does not extend to the parallel cleaning process. For example, consider cleaning K_3 using the parallel cleaning process: initially one vertex contains two brushes and is cleaned at step 1. At step 2 the remaining two vertices are cleaned, but in the final configuration, each contains one brush. Clearly this process cannot be reversed.

As a final result, there is a trivial upper bound on the number of brushes needed. We use a cleaning sequence that starts with a path that forms a diameter of the graph. One brush then travels the length of path yielding the following result (which is sharp for paths).

Theorem 2.4 *Let G be a connected graph. Then $b(G) \leq |E(G)| - diam(G) + 1$.*

3 Lower Bounds

Erdös asked what the minimum number of paths into which every connected graph can be decomposed [6]. Gallai conjectured [11] that this number is $\lceil \frac{|V(G)|}{2} \rceil$. If this is correct, it yields a lower bound for $b(G)$; only a lower bound because some path decompositions would not be valid in the cleaning process. For example, K_4 can be decomposed into two edge-disjoint paths, but $b(K_4) = 4$.

Following Definitions 1.1 and 1.3, every vertex of odd degree in a graph G will be the endpoint of (at least) one brush path. This leads to a natural lower bound for $b(G)$ since any graph with d_o odd vertices, can be decomposed into a minimum of $\frac{d_o}{2}$ paths.

Theorem 3.1 *Given initial configuration ω_0, suppose G can be cleaned yielding final configuration ω_T. Then for every vertex v in G with odd degree, either $\omega_0(v) > 0$ or $\omega_T(v) > 0$. In particular, $b(G) \geq \frac{d_o(G)}{2}$ where $d_o(G)$ denotes a number of vertices of odd degree.*

Proof: Suppose a graph $G = (V, E)$ is cleaned by process $\mathfrak{P}(G, \omega_0) = \{(\omega_t, D_t)\}_{t=0}^T$ and let $v \in V$ be a vertex of odd degree that is cleaned at step t. Using (3) we have that

$$\omega_T(v) - \omega_0(v) = \deg(v) - 2D_{t-1}(v).$$

As $\deg(v)$ is odd, the right side of the equality is also odd and it is not possible that both $\omega_T(v)$ and $\omega_0(v)$ are equal to zero. This finishes the first part of the proof.

For the second part, note that, by pigeonhole principle, there are at least $\frac{d_o(G)}{2}$ odd vertices with brushes at the initial configuration or at least $\frac{d_o(G)}{2}$ ones at the final configuration. Thus,

$$b(G) = \min_{\omega_0 : V \to \mathbb{N} \cup \{0\}} \left\{ \sum_{v \in V} \omega_0(v) = \sum_{v \in V} \omega_T(v) : G \text{ can be cleaned by } \omega_0 \right\} \geq \frac{d_o(G)}{2},$$

and the assertion follows. ■

Note that the lower bound given by Theorem 3.1 is sharp since $b(T) = \frac{d_o(T)}{2}$ for any tree T (see Theorem 5.1).

We can also create a lower bound for $b(G)$ dependent on the girth of G, see Corollary 3.3. But first we introduce a more general theorem. Let S be a subset of the vertices of a graph G, we denote by $G[S]$ the subgraph induced by S.

Theorem 3.2 *Let $G = (V, E)$ be any graph on n vertices, and for any $k \in [n]$,*

$$b_k = \min_{S \subseteq V, |S|=k} \left\{ \sum_{v \in S} \deg_G(v) - 2|E(G[S])| \right\} .$$

Then $b(G) \geq \max_k b_k$.

Proof: Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ be the cleaning sequence that cleans G using the optimal number of $b(G)$ brushes and fix $k \in [n]$. Using (1) we get that

$$\begin{aligned} b(G) &= \sum_{i=1}^{n} \max \left\{ 2D_{i-1}(\alpha_i) - \deg_G(\alpha_i), 0 \right\} \\ &\geq \sum_{i=1}^{k} \left(2D_{i-1}(\alpha_i) - \deg_G(\alpha_i) \right) \\ &= \sum_{i=1}^{k} \left(\deg_G(\alpha_i) - 2(\deg_G(\alpha_i) - D_{i-1}(\alpha_i)) \right) \\ &= \sum_{i=1}^{k} \deg_G(\alpha_i) - 2|E(G[\{\alpha_1, \alpha_2, \ldots, \alpha_k\}])| \geq b_k , \end{aligned}$$

since each edge in the induced subgraph $E(G[\{\alpha_1, \alpha_2, \ldots, \alpha_k\}])$ appears exactly once in the sum as a clean edge. ■

Note that in Theorem 3.2, $\sum_{v \in S} \deg_G(v) - 2|E(G[S])|$ is the number of edges from the subset S to its complement in G, that is, the 'boundary' edges.

Let $\delta(G)$ be the minimum degree of graph G. The next result is a simple corollary of Theorem 3.2.

Corollary 3.3 *For any graph G with girth $g < \infty$, $b(G) \geq (\delta(G) - 2)g$.*

Proof: Take any $S \subseteq V$ of order g, $v \in S$. Since G has no cycle of length less than g, $G[S \setminus \{v\}]$ induces a forest with $g - 1 - l$ edges (l denotes the number of components). If $l = 1$, then v can have at most two neighbours among vertices from $S \setminus \{v\}$; otherwise at most l vertices can be adjacent to v. Thus, $|E(G[S])| \leq g$ and we can use Theorem 3.2 with

$$\begin{aligned} b_g &= \min_{S \subseteq V, |S|=g} \left\{ \sum_{v \in S} \deg_G(v) - 2|E(G[S])| \right\} \\ &\geq \delta(G)g - 2 \max_{S \subseteq V, |S|=g} \{|E(G[S])|\} \quad \geq \quad (\delta(G) - 2)g . \end{aligned}$$

■

Definition 3.4 *Let $G = (V, E)$ be a graph and $f : V \to \{1, 2, \ldots n\}$ be a linear layout of G. The* ***cutwidth*** *of f is*

$$cw_f(G) = \max_{1 \le i \le n} |\{(u, v) \in E \ : \ f(u) \le i < f(v)\}|.$$

The cutwidth denoted $cw(G)$, is the minimum cutwidth over all possible linear layouts of G.

Theorem 3.5 *For any graph G, $cw(G) \le b(G)$.*

Proof: Let $G = (V, E)$ be a graph with $|V(G)| = n$. Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ be a cleaning sequence of the cleaning process $\mathfrak{P}(G, \omega_0)$ that will clean G using $b(G)$ brushes. As any cleaning sequence which yields a clean graph G is a linear layout of the vertices of G, let $f(\alpha_i) = i$ for all $i \in [n]$. Let $A_i = |\{(u, v) \in E : f(u) \le i < f(v)\}|$ and note it represents the number of brushes which are no longer at their initial vertices and, at step i, are at dirty vertices. That is, the number of brushes which are at vertices v_j where $v_j \in D_i$ and $\omega_i(v_j) > \omega_0(v_j)$. Clearly $cw_f(G) = \max_{1 \le i \le n} A_i \le b(G)$ and finally $cw(G) \le cw_f(G) \le b(G)$. ■

Definition 3.6 *In the discrete edge-searching process of $G = (V, E)$, an* ***edge-search strategy*** *is a sequence of actions such that the final action leaves all edges of G uncontaminated. (See [1, 8, 16] for more on searching.)*

Initially, all edges E are contaminated and a fixed number of searchers are placed on vertices of G. An edge $uv \in E$ becomes ***decontaminated*** *when a searcher traverses edge uv from u to v while there is a second searcher on u or while all other edges incident with u are already decontaminated. If edge e is decontaminated and an action results in a path (with no searchers) from a contaminated edge to edge e, then e has become recontaminated.*

A vertex has been ***decontaminated*** *if all incident edges are decontaminated. A* ***graph*** *G* ***is decontaminated*** *when all $v \in V$ have been decontaminated (or, equivalently, when all edges E have been decontaminated). The minimum number of searchers needed to decontaminate G is the edge-search number $es(G)$.*

It is clear that when a vertex is cleaned, sending the brushes one at a time, is an edge search which proves the next inequality.

Theorem 3.7 *For any graph G, $es(G) \le b(G)$.*

4 Cartesian Products of Graphs

The graph $G \square H$ is the *Cartesian product* of graphs G and H. It contains vertex set $V(G) \times V(H)$ where $(u, v) \in V(G \square H)$ is adjacent to $(u', v') \in V(G \square H)$ when either $u = u'$ and $vv' \in E(H)$ or $v = v'$ and $uu' \in E(G)$. It can easily be seen that $G \square H$ decomposes into $|V(G)|$ copies of H and also into $|V(H)|$ copies of G. This idea is used in creating an upper bound for $G \square H$ in Theorem 4.1. As the bound of Theorem 4.1 can be hard to compute, Corollary 4.2 gives an easier (but weaker) upper bound to compute.

Theorem 4.1 *Given cleaning processes $\mathfrak{P}(G,\omega_0)$, $\mathfrak{C}(H,\tau_0)$ that clean graphs G and H, respectively,*

$$b(G\Box H) \le \sum_{\alpha\in V(G)} \sum_{\beta\in V(H)} \max\{0, \omega_0(\alpha)+\tau_0(\beta)-\omega_{|V(G)|}(\alpha)-\tau_{|V(H)|}(\beta)\}.$$

Proof: For graphs G,H with $|V(G)| = g$, $|V(H)| = h$, let $\alpha = (\alpha_1,\alpha_2,\dots,\alpha_g)$, $\beta = (\beta_1,\beta_2,\dots,\beta_h)$ be cleaning sequences of the respective processes $\mathfrak{P}(G,\omega_0) = \{(\omega_t, D_t)\}_{t=0}^{g}$, $\mathfrak{C}(H,\tau_0) = \{(\tau_t, C_t)\}_{t=0}^{h}$. Applying (3) to α and β respectively, we get that

$$\begin{aligned} \deg_G(\alpha_i) - D_{i-1}(\alpha_i) &= D_{i-1}(\alpha_i) + \omega_g(\alpha_i) - \omega_0(\alpha_i) \\ \deg_H(\beta_j) - C_{j-1}(\beta_j) &= C_{j-1}(\beta_j) + \tau_h(\beta_j) - \tau_0(\beta_j) \end{aligned} \quad (4)$$

for $i\in[g]$, $j\in[h]$.

Label the vertices of $G\Box H$ as (α_i,β_j) for $i\in[g]$ and $j\in[h]$. Set

$$\psi_0((\alpha_i,\beta_j)) = \max\{0, \omega_0(\alpha_i)+\tau_0(\beta_j)-\omega_g(\alpha_i)-\tau_h(\beta_j)\} \quad (5)$$

and $\gamma = ((\alpha_1,\beta_1),(\alpha_1,\beta_2),\dots,(\alpha_1,\beta_h),\dots,(\alpha_g,\beta_1),(\alpha_g,\beta_2),\dots,(\alpha_g,\beta_h))$. Then, to finish the proof it is enough to show that given initial configuration ψ_0, $G\Box H$ can be cleaned by a cleaning process $\mathfrak{B}(G\Box H,\psi_0) = \{(\psi_t, B_t)\}_{t=0}^{gh}$ using sequence γ: we use induction on t.

From the Cartesian product definition, $\deg_{G\Box H}((\alpha_i,\beta_j)) = \deg_G(\alpha_i)+\deg_H(\beta_j)$. Since $\psi_0(\gamma_1) = \omega_0(\alpha_1)+\tau_0(\beta_1) = \deg_G(\alpha_1)+\deg_H(\beta_1) = B_0(\gamma_1)$, γ_1 can be cleaned at the first step and the basis step is verified. For the induction step, we assume that $(\gamma_1,\gamma_2,\dots,\gamma_t)$, $t=(i-1)h+j-1$, cleans the first t vertices of $G\Box H$.

We next show $\gamma_{t+1} = (\alpha_i,\beta_j)$ can be cleaned at step $t+1$. Note that $B_t((\alpha_i,\beta_j)) = D_{i-1}(\alpha_i)+C_{j-1}(\beta_j)$. Combining this with (4) and (5), we have

$$\begin{aligned} \psi_t(\gamma_{t+1}) &= \psi_0((\alpha_i,\beta_j)) + \deg_{G\Box H}((\alpha_i,\beta_j)) - B_t((\alpha_i,\beta_j)) \\ &= \max\{0, \omega_0(\alpha_i)+\tau_0(\beta_j)-\omega_g(\alpha_i)-\tau_h(\beta_j)\} \\ &\quad + \deg_G(\alpha_i)+\deg_H(\beta_j)-D_{i-1}(\alpha_i)-C_{j-1}(\beta_j) \\ &= \max\{\omega_g(\alpha_i)-\omega_0(\alpha_i)+\tau_h(\beta_j)-\tau_0(\beta_j),0\} + D_{i-1}(\alpha_i)+C_{j-1}(\beta_j) \\ &\ge B_t(\gamma_{t+1}). \end{aligned}$$

This implies that γ_{t+1} can be cleaned at step $t+1$ and the assertion follows. ∎

Corollary 4.2 *Given cleaning processes $\mathfrak{P}(G,\omega_0)$, $\mathfrak{C}(H,\tau_0)$ that clean graphs G and H, respectively,*

$$b(G\Box H) \le |V(H)|b(G) + |V(G)|b(H).$$

Proof: For graphs G,H with $|V(G)| = g$, $|V(H)| = h$, let cleaning processes $\mathfrak{P}(G,\omega_0) = \{(\omega_t,D_t)\}_{t=0}^{g}$, $\mathfrak{C}(H,\tau_0) = \{(\tau_t,C_t)\}_{t=0}^{h}$ clean G,H, with $b(G), b(H)$ brushes, respectively.

By Theorem 4.1

$$
\begin{aligned}
b(G\Box H) &\leq \sum_{\alpha\in G}\sum_{\beta\in H}\max\{0,\omega_0(\alpha)+\tau_0(\beta)-\omega_g(\alpha)-\tau_h(\beta)\}\\
&\leq \sum_{\alpha\in G}\sum_{\beta\in H}(\omega_0(\alpha)+\tau_0(\beta))\\
&= \sum_{\alpha\in G}(h\omega_0(\alpha)+b(H))\\
&= hb(G)+gb(H)\,.
\end{aligned}
$$

■

Theorem 4.1 and Corollary 4.2 can easily be extended to the general case to give an upper bound for $b(G_1\Box G_2\Box\ldots\Box G_m)$.

Note that these bounds depend on the original cleaning sequences, moreover, different sequences (even if all use the minimum number of brushes) could give different number of brushes for the product graph. For example, Figure 2 presents a graph G with four different initial configurations which are minimum, but when Theorem 4.1 is applied to $G\Box K_2$, they give different upper bounds. Specifically, that of Figure 2a gives an upper bound of 7 while that of Figure 2b gives 6 (the other two symmetric initial configurations also give upper bounds of 6 and 7). Note that in Figures 2a and 2b, the boxed numbers indicate the vertices with $\omega_n > 0$. As it happens, neither is the correct number: $G\Box K_2$ can be cleaned with 5 brushes. For this, the initial configuration is $w_0((1,a)) = 3$ and $w_0((1,b)) = 2$. The table shows the cleaning sequence (an 'x' indicates the vertex that was cleaned) and the configuration at each step.

Vertex	(1,a)	(1,b)	(1,c)	(1,d)	(1,e)	(2,a)	(2,b)	(2,c)	(2,d)	(2,e)
w_0	3	2								
w_1	x	3	1			1				
w_2		x	2	1		1	1			
w_3			x	2		1	1	1		
w_4				x	1	1	1	1	1	
w_5					x	1	1	1	1	1

At this stage there is one copy of G remaining with a brush at every vertex which can cleaned by the cleaning sequence $(2,e),(2,d),(2,c),(2,b),(2,a)$.

Combining the lower bound from Theorem 3.1 with the upper bound of Theorem 4.1, we can determine the brush number for the product of two finite paths.

Theorem 4.3 *For $m,n>1$, $b(P_m\Box P_n)=m+n-2$.*

Proof: Let $G = P_m\Box P_n$. From Theorem 3.1 and the Reversibility Theorem, we assume there are at least $\frac{d_0}{2} = \frac{2(m-2)+2(n-2)}{2} = m+n-4$ odd vertices with brushes at the initial configuration. Suppose G is cleaned with $b(G)$ brushes and v was the first vertex cleaned; the initial number of brushes at v is equal to $\deg_G(v)\in\{2,3,4\}$. Then, there must be at least two extra brushes initially at v and thus $b(G)\geq m+n-2$.

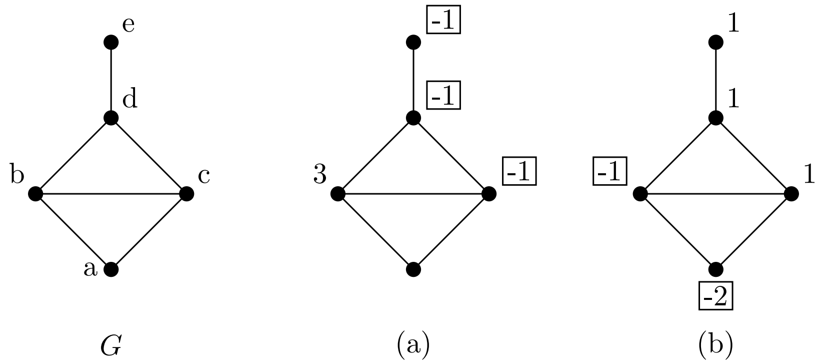


Figure 2: Two initial/final configuration of brushes which can clean G.

To show an upper bound we use Theorem 4.1 with an initial configurations of brushes ω_0 and τ_0 of paths $P_m = \{u_1, u_2, \dots, u_m\}$ and $P_n = \{v_1, v_2, \dots, v_n\}$, respectively; $\omega_0(u_1) = 1$, $\omega_0(u_i) = 0$ for $1 < i \le m$; $\tau_0(v_1) = 1$, $\tau_0(v_j) = 0$ for $1 < j \le n$.

$$\begin{aligned} b(G) &\le \sum_{i=1}^{m}\sum_{j=1}^{n} \max\{0, \omega_0(u_i) + \tau_0(v_j) - \omega_m(u_i) - \tau_n(v_j)\} \\ &= \sum_{j=1}^{n} \max\{0, 1 + \tau_0(v_j) - \tau_n(v_j)\} + (m-2)\sum_{j=1}^{n} \max\{0, \tau_0(v_j) - \tau_n(v_j)\} \\ &\quad + \sum_{j=1}^{n} \max\{0, \tau_0(v_j) - \tau_n(v_j) - 1\} \\ &= n + (m-2)\,. \end{aligned}$$

■

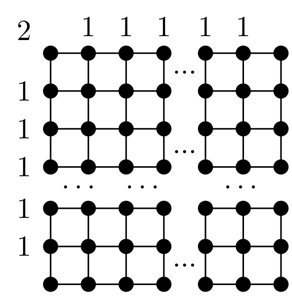


Figure 3: An initial configuration of brushes which will clean $P_m \Box P_n$.

5 Families of Graphs

Theorem 5.1 *For any tree T with $d_o(T)$ vertices of odd degree, $b(T) = \frac{d_o(T)}{2}$.*

Proof: We use induction on $|V(T)|$. The basis step is trivial: $b(K_1) = 0 = \frac{d_o(K_1)}{2}$. For the induction step we assume $b(T) = \frac{d_o(T)}{2}$ for all trees T on k ($k \geq 1$) vertices. Let $T' = (V, E)$ be a tree with $|V(T')| = k + 1$, v be any leaf of T', and w be the only neighbour of v. As $|V(T' - v)| = k$, the inductive hypothesis implies $b(T' - v) = \frac{d_o(T'-v)}{2}$. Let $\alpha = (\alpha_1, \ldots, \alpha_k)$ be a cleaning sequence returned by the process $\mathfrak{P}(T' - v, \omega_0)$ which yields $b(T' - v) = \frac{d_o(T'-v)}{2}$. By Theorem 3.1, we simply need to show that $b(T') \leq \frac{d_o(T')}{2}$.

If $\deg_{T'-v}(w)$ is even, then $d_o(T') = d_o(T' - v) + 2$ and $b(T' - v) = \frac{d_o(T')}{2} - 1$. Set $\tau_0(v) = 1, \tau_0(\alpha_i) = \omega_0(\alpha_i)$ for $i \in [k]$; then $\mathfrak{P}(T', \tau_0)$ cleans T' using cleaning sequence $\alpha' = (v, \alpha_1, \ldots, \alpha_k)$. Thus, $b(T') \leq b(T' - v) + 1 = \frac{d_o(T')}{2}$.

If $\deg_{T'-v}(w)$ is odd, then $d_o(T') = d_o(T' - v)$ and $b(T' - v) = \frac{d_o(T')}{2}$. Using Theorem 3.1 and Theorem 2.3, we can, without loss of generality, assume that $\omega_k(w) > 0$. Set $\tau_0(v) = 0, \tau_0(\alpha_i) = \omega_0(\alpha_i)$ for $i \in [k]$; then $\mathfrak{P}(T', \tau_0)$ cleans T' using cleaning sequence $\alpha' = (\alpha_1, \ldots, \alpha_k, v)$. Thus, $b(T') \leq b(T' - v) = \frac{d_0(T')}{2}$. ■

Theorem 5.2 *For a complete graph K_n,*

$$b(K_n) = \begin{cases} \frac{n^2}{4} & \text{if } n \text{ is even} \\ \frac{n^2-1}{4} & \text{if } n \text{ is odd.} \end{cases}$$

Proof: Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ denote the cleaning sequence used to clean K_n with $b(K_n)$ brushes. The symmetry of K_n implies that all cleaning sequences of K_n are equivalent.

Note that $\deg(\alpha_i) = n - 1$ for all $i \in [n]$ and $D_t(\alpha_{t+1}) = n - (t + 1)$. Then, using (1), we get

$$\omega_0(\alpha_{t+1}) = \max\{2D_t(\alpha_{t+1}) - \deg(\alpha_{t+1}), 0\} = \begin{cases} n - 2t - 1 & \text{if } t \leq \lfloor \frac{n-1}{2} \rfloor \\ 0 & \text{otherwise.} \end{cases}$$

Thus

$$b(K_n) = \sum_{i=1}^{n} \omega_0(\alpha_i) = \sum_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} (n - 2i - 1) = \begin{cases} \frac{n^2}{4} & \text{if } n \text{ is even} \\ \frac{n^2-1}{4} & \text{if } n \text{ is odd.} \end{cases}$$

■

Theorem 5.3 *Let $K_{(n,m)}$ be the complete multipartite graph with m colour classes each of size n. Then $b(K_{(n,m)}) = \frac{m^2n^2}{4} + O(mn^2)$.*

Proof: Let $V(K_{(n,m)}) = \{(i, j) \ : \ 1 \leq i \leq n, 1 \leq j \leq m\}$ where $V_j = \{(i, j) \ : \ 1 \leq i \leq n\}$ is the jth colour class.

For an upper bound, consider cleaning the graph with the cleaning sequence $(1, j)$, $1 \leq j \leq m$ followed by $(2, j)$, $1 \leq j \leq m$, etc. Each vertex $(1, j)$ requires $n(m - 1) - 2(j - 1)$ brushes. Vertex $(2, 1)$ requires $n(m - 1) - 2(m - 1)$, the same as $(1, m)$. In general, when it is their turn to clean, (i, j) has received $(i - 1)(m - 1) + (j - 1)$ brushes and is adjacent to the same number of clean edges and so requires

$$\max\{n(m - 1) - 2(i - 1)(m - 1) - 2(j - 1), 0\}$$

brushes in the original configuration. The initial configuration then needs

$$2\sum_{i=1}^{\frac{n(m-1)}{2}} i + O(mn^2) = \frac{m^2n^2}{4} + O(mn^2)$$

brushes.

Suppose n is even and consider a subgraph S of order $\frac{nm}{2}$. It is easy to verify that the S has the least number of edges to $G-S$ if it is isomorphic to the subgraph induced by the vertex set $\{(i,j)\ :\ 1\le i\le \frac{n}{2}, 1\le j\le m\}$. With this subgraph, from Theorem 3.2, we have that $b(K_{(n,m)}) \ge \frac{nm}{2}\cdot\frac{nm}{2} = \frac{n^2m^2}{4}$. The case where n is odd is similar and left to the reader. ∎

Recall that the hypercube Q_n is the Cartesian product of an edge with itself n times. Alternatively, given a set S of cardinality n, it is the graph whose vertices are the subsets of S and two vertices x and y are adjacent if $|x\setminus y| = 1$ or $|y\setminus x| = 1$.

Theorem 5.4 *For the hypercube Q_n,*

$$\frac{2}{3}(2^n-1) \le b(Q_n) \le \binom{n}{0}n + \binom{n}{1}(n-2) + \binom{n}{2}(n-4) + \ldots + \binom{n}{\lfloor\frac{n}{2}\rfloor}(n-2\lfloor\frac{n}{2}\rfloor).$$

Proof: From Theorem 4.1, we can obtain an upper bound for hypercubes. However, an easier way to get the same bound—that it is the same, we leave to the reader—is to use the representation of Q_n where the vertices are subsets of $\{1,2,3\ldots,n\}$. Every vertex has degree n and if a vertex corresponds to a cardinality k subset then it has k edges incident to vertices with subsets of cardinality $k-1$ and $n-k$ edges to those with cardinality $k+1$. The appropriate cleaning sequence is to go in order of the cardinalities: first the vertex corresponding to the empty set starts with n brushes, the vertices with a cardinality 1 next but need $n-2$ initial brushes; the vertices of cardinality 2 need $n-4$ brushes, etc. Once the vertices of cardinality $\lceil\frac{n}{2}\rceil$ have been reached, no new initial brushes are needed.

In [18] (see also [5, 10, 12]) it is shown that $cw(Q_n) = \frac{2}{3}(2^n-a)$ where $a=1$ if n is even and $\frac{1}{2}$ otherwise. From Theorem 3.5 we have that $b(Q_n)\ge cw(G)$ giving the lower bound. ∎

If $n=2m$ then the sum in the theorem is $(m+1)\binom{2m}{m+1}$.

For the Cartesian product $K_m\Box K_n$, by symmetry, we may suppose that $m\ge n$. Assume that both n and m are even and consider the subgraphs of order $\frac{mn}{2}$. We wish to find the subgraph G of order $\frac{mn}{2}$ that has the fewest boundary edges, or, equivalently, that subgraph G, which with its complement G^c, together they have the maximum number of interior edges. This occurs when $G = K_m\Box K_{\frac{n}{2}}$. We outline a proof: let $V(K_m) = \{1,2,\ldots,m\}$, $V(K_n) = \{1,2,\ldots,n\}$, $G^i = G\cap(\{i\}\times K_n)$ and $G_j = G\cap(K_m\times\{j\})$. We may suppose that vertices of G are arranged so that $|V(G^i)|\ge|V(G^{i+1})|$ and $|V(G_j)|\ge|V(G_{j+1})|$. With this arrangement, it can be easily shown that the number of edges in G and G^c is greatest is $G^i = \{i\}\times\{1,2,\ldots,k_i\}$ and $G_j = \{1,2,\ldots,l_j\}\times\{j\}$. Suppose G is not isomorphic to $K_m\Box K_{\frac{n}{2}}$ then there are j and k , $j<k$ such that $m > |V(G_j)| \ge |V(G_k)| > 0$. The number of interior edges of G plus G^c can now be increased by deleting part (or all) of $V(G_k)$ and adding that number of vertices to $V(G_j)$. The process continues until the final graph is $K_m\Box K_{\frac{n}{2}}$.

Now G has $\frac{mn}{2}\cdot\frac{n}{2}$ edges to its complement so that by Theorem 3.2, $b(K_m\Box K_{\frac{n}{2}}) \ge \frac{mn}{2}\cdot\frac{n}{2}$. By Theorem 5.2, there is essentially only one cleaning sequence for a complete graph. Take the cleaning sequence that cleans copies of K_m first, that is, (a_i,b_j), $j=1,2,\ldots,m$ for

$i = 1, 2, \ldots, n$. Following Theorem 4.1, the subgraph with $i = 1$ requires $(n-1+m-1)+(n-1+m-3)+\ldots$ brushes; with $i = 2$ requires $(n-3+m-1)+(n-3+m-3)+\ldots$ brushes; etc. finishing with $i = n$ which requires $(-(n-1)+m-1)+(-(n-1)+m-3)+\ldots$ brushes. Since $m+n$ is even then the summation is

$$\sum_{j=\frac{m-n}{2}}^{\frac{m+n-2}{2}} 2\sum_{i=0}^{j} i = \frac{3m^2n+n^3-4n}{12}.$$

If one or both of n and m are odd, at most nm further brushes are required. This proves the following result.

Theorem 5.5 *If $m \geq n$, then*

$$\frac{mn^2}{4} + O(mn) \leq b(K_m \square K_n) \leq \frac{3m^2n+n^3-4n}{12} + mn.$$

Note that if $n = m$ and both are even, then careful calculation gives

$$\frac{m^3}{4} \leq b(K_m \square K_m) \leq \frac{m^3-m}{3}.$$

6 Unique cleaning sequence

Before we move to the main problem of this section, let us mention a problem of a similar flavour. Is there a graph G that has a unique initial configuration yielding a minimum number of brushes? The answer is simple: by the Reversibility Theorem, the only graphs that satisfy this property are the empty graphs. Thus, it seems natural to try to characterize the family of graphs having exactly two minimum cleaning configurations (each configuration yields the other as a final configuration) or having all cleaning configurations equivalent (up to isomorphism). This is still an open question.

In this section we would like to characterize graphs on n vertices that, together with some initial configurations of brushes, yield a unique cleaning sequence. In other words, at each step there is only one vertex than can be cleaned. Note then the sequential and parallel cleaning processes are would be identical. The main result gives an upper bound for the number of edges of any graph in this family.

Suppose that $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ is a unique cleaning sequence of the cleaning process $\mathfrak{P} = \{(\omega_t, D_t)\}_{t=0}^n$ which cleans a graph $G = (V, E)$. We use the notation introduced before: $N^-(\alpha_t) = |\{\alpha_t\alpha_i \in E(G) \ : \ i < t\}|$ and $N^+(\alpha_t) = |\{\alpha_t\alpha_i \in E(G) \ : \ i > t\}|$ (clearly $\deg(\alpha_t) = N^-(\alpha_t) + N^+(\alpha_t)$ and $D_t(\alpha_t) = N^+(\alpha_t)$).

From the fact that vertex α_{t+1} cannot be cleaned at time t and must be ready to be cleaned at time $t+1$, it follows that $\alpha_t\alpha_{t+1} \in E$ for any $t \in [n-1]$. This necessary condition gives a lower bound for the number of edges, namely, $|E(G)| \geq n-1$ (the result is sharp since a path P_n belongs to the family we consider).

Since α_t cannot be cleaned at time $t-1$ and path $P = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ is a subgraph of G, $\omega_0(\alpha_t) + N^-(\alpha_t) - 1 < N^+(\alpha_t) + 1$. From this, we can obtain a sufficient and necessary condition for a graph to have a unique sequence $(\alpha_1, \alpha_2, \ldots, \alpha_n)$. Note that $\omega_0(\alpha_t)$ can be adjusted to ensure α_t can be cleaned at time t, namely, set

$$\omega_0(\alpha_t) = \max\{N^+(\alpha_t) - N^-(\alpha_t), 0\}.$$

Theorem 6.1 *Let $\mathfrak{P} = \{(\omega_t, D_t)\}_{t=0}^n$ be a cleaning process which cleans a graph $G = (V, E)$. $\mathfrak{P}$ returns a unique cleaning sequence $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ if and only if*

(P1) *Path $P = (\alpha_1, \alpha_2, \dots, \alpha_n)$ is a subgraph of G,*

(P2) $\omega_0(\alpha_t) = \max\{N^+(\alpha_t) - N^-(\alpha_t), 0\}$ *where* $N^-(\alpha_t) \le N^+(\alpha_t) + 1$ *for* $t \in [n]$.

Moreover,

$$n - 1 \;\le\; |E(G)| \;\le\; n\lfloor \sqrt{2n} - 1/2 \rfloor - \binom{\lfloor \sqrt{2n} + 3/2 \rfloor}{3} \;\sim\; \frac{2\sqrt{2}}{3} n^{3/2}\,.$$

Proof: We have already discussed the necessary and sufficient conditions and lower bound for the number of edges in a graph G. It remains to be shown that the upper bound holds.

Consider first two graphs $F = F_1$ and $H = H_2$ constructed by deterministic processes described below. Both processes ensure that final graphs satisfy desired conditions.

Let H_{n+1} be an empty graph on vertex set $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$. We construct a final graph H_2 by saturating the vertices one by one, maximizing $N^-(\alpha_i)$. Formally, given a graph H_{i+1} $(2 \le i \le n)$ we construct a graph H_i by adding $h_i = \min\{N^+(\alpha_i) + 1, i - 1\}$ edges $\alpha_j \alpha_i$ for $\max\{i - N^+(\alpha_i) - 1, 1\} = i - h_i \le j \le i - 1$.

Let F_n be an empty graph on a vertex set $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$. We construct a final graph F_1 by saturating a vertices one by one, maximizing $N^+(\alpha_i)$. Formally, given a graph F_{i+1} $(1 \le i \le n-1)$ we construct a graph F_i by adding f_i edges $\alpha_i \alpha_j$ if $j > i$ and $N^-(\alpha_j) < N^+(\alpha_j) + 1$.

It is not hard to see that F and H are exactly the same graphs (Figure 4 presents the history of both processes run on graphs with 7 vertices). We introduce two algorithms for generating the same graph since we need a property following from the construction of H_2 but we cannot find a number of edges in terms of $\sum_{i=2}^n h_i$; fortunately $\sum_{i=1}^{n-1} f_i$ is relatively easy to compute. In order to find the number f_i of edges added to F_{i+1} consider a vector $(N^+_{F_{i+1}}(\alpha_j) - N^-_{F_{i+1}}(\alpha_j) + 1)_{j=i+1}^n$; f_i is equal to the number of positive coordinates. The first vectors generated during the process are: (1), $(2,0)$, $(2,1,0)$, $(3,1,0,0)$, $(3,2,0,0,0)$, $(3,2,1,0,0,0)$, $(4,2,1,0,0,0,0)$, etc. (see also Figure 4).

Noting the pattern we get that

$$\begin{aligned} f(n) &= |E(F)| \;=\; 0 + \sum_{i=1}^{n-1} f_i \\ &= 0 + 1 + 1 + 2 + 2 + 2 + 3 + 3 + 3 + 3 + \cdots + f_1 \qquad (6) \\ &= (n-1) + (n-1-2) + \cdots + (n-1-2-\cdots-f_1) \\ &= nf_1 - \sum_{i=1}^{f_1} \sum_{j=1}^{i} j \;=\; nf_1 - \sum_{i=1}^{f_1} \binom{i+1}{2} \;=\; nf_1 - \binom{f_1+2}{3}. \end{aligned}$$

Moreover, $f_1 = k$ if $\sum_{i=1}^k i < n \le \sum_{i=1}^{k+1} i$ (note that (6) contains n terms). Since n is an integer, this is equivalent to

$$\begin{aligned} \frac{k(k+1)}{2} + \frac{1}{8} &< \quad n \quad < \frac{(k+1)(k+2)}{2} + \frac{1}{8} \\ \left(k + \frac{1}{2}\right)^2 &< \quad 2n \quad < \left(k + \frac{3}{2}\right)^2 \\ k &< \sqrt{2n} - \tfrac{1}{2} < k + 1 \end{aligned}$$

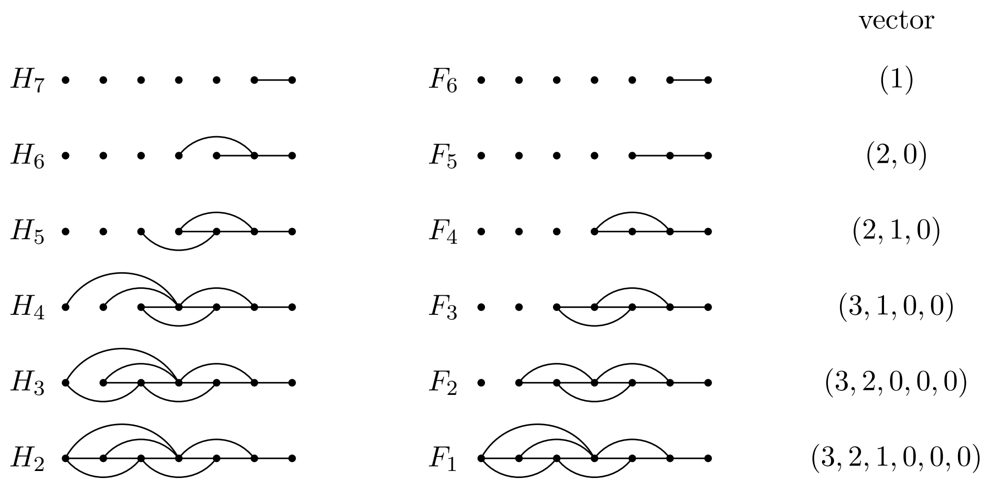


Figure 4: The history of both processes run on graphs with 7 vertices.

and thus $f_1 = \lfloor \sqrt{2n} - 1/2 \rfloor$. This implies that the upper bound we claim is achieved by the graph F. We will show that F contains a maximum possible number of edges, that is, $|E(G)| \leq f(n)$. This will finish the proof of the theorem.

Having a graph G that satisfies properties (P1) and (P2), we consider the operation of moving 'left endpoints' of edges 'to right' while maintaining these properties. Assume that $\alpha_i \alpha_j \in E(G)$ and $i < j$. Then the operation is defined as follows:

$MoveToRight(\alpha_i, \alpha_j)$**:**

(1) $k := \max(\{x : i < x < j \text{ and } \alpha_x\alpha_j \notin \mathrm{E(G)}\} \cup \{\mathrm{i}\})$,

(2) $Z := \{x < i : \alpha_x \alpha_i \in E(G)\}$,

(3) If $Z = \emptyset$, then put $E(G) := (E(G) \setminus \{\alpha_i\alpha_j\}) \cup \{\alpha_k\alpha_j\}$;

otherwise put $E(G) := (E(G) \setminus \{\alpha_{\min Z}\alpha_i, \alpha_i\alpha_j\}) \cup \{\alpha_{\min Z}\alpha_k, \alpha_k\alpha_j\}$.

Finally, we apply the following operation φ on graph G.

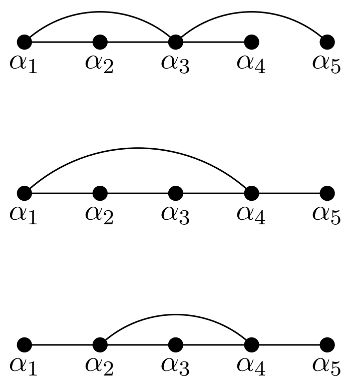
$\varphi(G)$**:** for $j := n$ down to 2

for $i := j - 1$ down to 1

if $\alpha_i\alpha_j \in E(G)$, then $MoveToRight(\alpha_i, \alpha_j)$.

An example of 'MoveToRight' can be seen in Figure 5. It is easily seen that $\varphi(G)$ is a subgraph of $H = F$: suppose $\varphi(G)$ is not a subgraph of $H = F$. There must exist some $\alpha_u\alpha_w$ ($u < w$) which is an edge in $\varphi(G)$ but not in $H = F$. By construction of H, the number of 'left neighbours' of α_w in $\varphi(G)$ is at most the number of 'left neighbours' of α_w in $H = F$, so there must exist some $\alpha_v\alpha_w$ ($v < w$) which is an edge in $H = F$, but not in $\varphi(G)$. If $u < v < w$, then in applying φ to G, 'MoveToRight' is used and $\alpha_u\alpha_w$ must be deleted. If $v < u < w$, then in the construction of F, F_u would have added the edge $\alpha_u\alpha_w$ (before considering the edge $\alpha_v\alpha_w$). Thus, $\varphi(G)$ must be a subgraph of $H = F$.

Finally, since the number of edges does not change after applying φ, $|E(G)| = |E(\varphi(G))| \leq |E(F)| = f(n)$. ■



a) MoveToRight(α_3, α_5): $k = 4, Z = \{1, 2\}$
$E(G) := (E(G)\backslash\{\alpha_1\alpha_3, \alpha_3\alpha_5\}) \cup \{\alpha_1\alpha_4, \alpha_4\alpha_5\}$

b) MoveToRight(α_3, α_4): $k = 3, Z = \{2\}$, $E(G) := E(G)$
MoveToRight(α_1, α_4): $k = 2, Z = \emptyset$
$E(G) := (E(G)\backslash\{\alpha_1\alpha_4\}) \cup \{\alpha_2\alpha_4\}$

c) MoveToRight(α_2, α_3): $k = 2, Z = \{1\}$, $E(G) := E(G)$
MoveToRight(α_1, α_2): $k = 1, Z = \emptyset$, $E(G) := E(G)$

Figure 5: An example of MoveToRight for a graph on 5 vertices.

7 Conjectures

Finding the number of chips and a configuration that gives an infinite (recurrent) chip firing game is very easy. Is this true for a cleaning sequence? Because of the closer relationship to searching we conjecture the following.

Conjecture 7.1 *It is an NP-complete problem to determine whether k brushes will clean a graph.*

The sequence from Theorem 4.1 cleans a copy of one of the factors before moving on to the next, the next being determined by the cleaning sequence of the other factor. In Theorems 5.4 and 5.5 the cleaning sequence obtained from Theorem 4.1 was very close to optimal. Even the graph in Figure 2, which shows the bound of Theorem 4.1 is not necessarily the best, still has the optimal cleaning sequence where one cleans the copies of one factor in order.

Conjecture 7.2 *Every cleaning sequence of $G\square H$ using the least number of brushes, consists of using a cleaning sequence of one factor and the copies of the other factor are cleaned in that order.*

This would imply that $b(Q_n)$ is closer if not equal to the upper bound given in Theorem 5.4.

References

[1] B. Alspach, Searching and sweeping graphs: a brief survey, *International Conference in Combinatorics*, Le Matematiche, Vol LIX (2004) - Fasc. I-II, pp. 5–37.

[2] A. Björner, L. Lovasz, and P. Shor, Chip firing games on graphs, *European Journal of Combinatorics* **12** (1991), 283–291.

[3] P. Bak, C. Tang, and K. Wiesenfeld, Physics Review Letters **59** (1987) 381.

[4] H. Bodlaender, A partial k-arboretum of graphs with bounded treewidth, *Theoretical Computer Science*. **209** (1998) 1–45.

[5] B. Bollobás and I. Leader, Edge-isoperimetric inequalities in the grid, *Combinatorica*, **11** (1991), 299–314.

[6] F. R. K. Chung, Open Problems of Paul Erdős in Graph Theory, *Journal of Graph Theory* **25** (1997) 3–36.

[7] D. Dhar, P. Ruelle, S. Sen, and D. Verma, Algebraic aspects of sandpile models, *Journal of Physics A* **28** (1995) 805–831.

[8] D. Dyer, *Sweeping graphs and digraphs*, Ph.D. thesis, Simon Fraser University, 2004.

[9] K. Eriksson, Chip firing games on mutating graphs, *SIAM Journal of Discrete Mathematics*, **9** (1996) 118–128.

[10] S. Even and R. Kupershtok, Layout area of the hypercube, *Journal of Interconnection Networks* **4** (2003) 395-417.

[11] G. Fan, Path decompositions and Gallai's conjecture, *Journal of Combinatorial Theory*. **B93** (2005) 117–125.

[12] L. H. Harper, Optimal assignments of numbers to vertices, *SIAM Journal of Applied Mathematics* **12** (1964), 131–135.

[13] L. M. Kirousis and C.H. Papadimitriou, Interval graphs and searching, *Discrete Math* **55** (1985) 181–184.

[14] C. Magnien, Classes of lattices induced by chip firing (and sandpile) dynamics, *European Journal of Combinatorics*, **24** (2003) pp. 665–683.

[15] S. McKeil, *Chip Firing Cleaning Processes*, M.Sc. Thesis, Dalhousie University (2007).

[16] N. Megiddo, S. L. Hakimi, M. Garey, D. Johnson, C. H. Papadimitriou The complexity of searching a graph. *Journal of the ACM* **35**(1988) pp. 1844.

[17] C. Merino, The Chip Firing Game and Matroid Complexes, in Discrete Models: Combinatorics, Computation, and Geometry, DM-CCG 2001, *Discrete Mathematics and Theoretical Computer Science Proceedings AA*, (2001) pp. 245–256.

[18] K. Nakano, Linear layout of generalized hypercubes, *International Journal of Foundations of Computer Science,* **14** (2003) 137-156.

[19] T. D. Parsons, Pursuit-evasion in a graph, *Theory and Applications of Graphs*, Y. Alavi and D. R. Lick, eds. Springer, Berlin, (1976), pp. 426–441.

[20] T. D. Parsons, The search number of a connected graph, *Proc. Ninth Southeastern Conf. Combinatorics*, Graph Theory and Computing, Congressus Numerantium XXI, Winnipeg, 1978, pp. 549–554.