Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Zacatenco
Departamento de Matemáticas

El grupo de pilas de arena de una multigráfica

Tesis que presenta

Carlos Alejandro Alfaro Montúfar

para obtener el grado de
Maestro en Ciencias
en la Especialidad en Matemáticas

Director de tesis: Dr. Carlos Enrique Valencia Oleta

México, D.F. Febrero del 2010.

Center for Research and Advanced Studies
of the National Polytechnic Institute
Campus Zacatenco
Department of Mathematics

The Sandpile group of a multigraph

A Dissertation presented by:

Carlos Alejandro Alfaro Montúfar

to obtain the degree of
Master in Science
in the speciality of Mathematics

Thesis Advisor: Professor Carlos Enrique Valencia Oleta

México, D.F. February, 2010.

Resumen

Consideremos una multigráfica G y un vértice fijo $s \in V(G)$. Denotaremos por $SP(G,s)$ al grupo de pilas de arena de G, el cual es un subconjunto de vectores en $\mathbb{N}^{V(G\setminus s)}$, llamado las configuraciones recurrentes de G. La operación en el grupo de pilas de arena está dada por la suma entrada por entrada entre vectores. El grupo crítico de G, denotado por $K(G)$, se define como el cokernel de la transpuesta de la matriz Laplaciana reducida de una multigrafica. Esto es

$$K(G) = \mathbb{Z}^n / \mathrm{Im} L(G,s)^t.$$

Se sabe que el grupo de pilas de arena de una multigráfica G es isomorfo al grupo crítico de G.

En esta tesis estamos interesados en dar una descripción algebraica del grupo de pilas de arena de una multigráfica. Comenzamos con una descripción de la estructura algebraica del grupo de pilas de arena de $\mathcal{C}_n$ y una descripción combinatoria explicita de los generadores de $SP(\mathcal{C}_n, s)$.

Introducimos el concepto nuevo de homomorfismo uniforme, $f : G \to H$, entre dos multigráficas G y H. Con este homomorfismo obtenemos un homomorfismo inyectivo, $\widetilde{f}$, entre los grupos $SP(c(G),s)$ y $SP(c(H),s)$. El concepto de homomorfismo uniforme está entre un homomorfismo completo y un isomorfismo de multigráficas.

Sea $G\square H$ el producto cartesiano de dos multigráficas G y H. El producto cartesiano de dos configuraciones $\mathbf{a}$ de $c(G)$ y $\mathbf{b}$ de $c(H)$ está dado por

$$(\mathbf{a}\square\mathbf{b})_{(u,v)} = \mathbf{a}_u + \mathbf{b}_v \text{ para todos } u \in V(G) \text{ y } v \in V(H).$$

Probamos que si $\mathbf{a}$ y $\mathbf{b}$ son configuraciones recurrentes, entonces $\mathbf{a}\square\mathbf{b}$ es una configuración recurrente.

Más aún, probamos que la función $\widetilde{i_G} : SP(c(G),s) \to SP(c(G\square H),t)$ dada por

$$\widetilde{i_G}(\mathbf{a}) = \mathbf{a}\square\mathbf{e},$$

es un homomorfismo injectivo canónico entre $SP(c(G),s)$ y $SP(c(G\square H),t)$.

Este hecho nos lleva a establecer una conjetura la cual nos proporciona una descripción algebraica y combinatoria del grupo de pilas de arena del cono del hipercubo Q_d de dimensión d.

Más precisamente,

$$K(c(Q_d)) \cong \bigoplus_{i=1}^{d} \mathbb{Z}_{2i+1}^{\binom{d}{i}} = \mathbb{Z}_3^d \oplus \mathbb{Z}_5^{\binom{d}{2}} \oplus \cdots \oplus \mathbb{Z}_{2d-1}^d \oplus \mathbb{Z}_{2d+1}.$$

y

$$SP(c(Q_d),s) = \bigoplus_{\{\beta\in\{0,1\}^d\}} \widetilde{K}_\beta \cong \bigoplus_{\{\beta\in\{0,1\}^d\}} \mathbb{Z}_{2|\beta|+1} \cong K(c(Q_d)),$$

donde $\widetilde{K}_\beta = \{g_\beta(r,t) + (d-|\beta|)\mathbf{1} \mid 0 \le r,t \le d \text{ and } r = |\beta| \text{ or } t = |\beta|\} \cong \mathbb{Z}_{2|\beta|+1}$ for all $\beta \in \{0,1\}^d$ y $g_\beta(r,t) \in \mathbb{N}^{V(Q_d)}$ está dado por

$$g_\beta(r,t)_{\mathrm{a}} = \begin{cases} r & \text{si } \beta\cdot\mathbf{a} \text{ es par,} \\ t & \text{si } \beta\cdot\mathbf{a} \text{ es impar.} \end{cases}$$

Finalmente, damos una forma alternativa de calcular las configuraciones recurrentes de una multigráfica usando programación lineal entera. Esta técnica puede ser usada para calcular la identidad del grupo de pilas de arena y el orden de una configuración recurrente. Además desarrollamos un programa en C++ que cálcula la estructura combinatoria del grupo de pilas de arena de una multigráfica.

Abstract

Given a multigraph G and a fixed vertex $s \in V(G)$, the abelian sandpile group of G, denoted by $SP(G,s)$, consists of a subset of vectors in $\mathbb{N}^{V(G\setminus s)}$, called the recurrent configurations of G and the stabilization of the sum entry by entry of vectors as the group operation. It is known that the sandpile group of a multigraph is isomorphic to its critical group. The critical group of G, denoted by $K(G)$, is defined as the cokernel of the transpose of the reduced Laplacian matrix of G

$$K(G) = \mathbb{Z}^n / \mathrm{Im} L(G,s)^t.$$

In this thesis we are interested in studying the combinatorial and algebraic structure of the sandpile group of a multigraph. We begin with the description of the algebraic structure of the sandpile of C_n and an explicit combinatorial description of the generators of $SP(C_n,s)$. After that, we introduce a new concept, called uniform homomorphism of multigraphs. If $f : G \to H$ is a sobrejective uniform homomorphism, then we get an injective homomorphism of groups

$$\widetilde{f} : SP(c(G),s) \to SP(c(H),s).$$

The concept of uniform homomorphism is between the concepts of a full homomorphism and the isomorphism of multigraphs.

If $G\Box H$ is the cartesian product of the two multigraphs G and H, we define the cartesian product of an $\mathbf{a}$ configuration of $c(G)$ and a configuration $\mathbf{b}$ of $c(H)$ by

$$(\mathbf{a}\Box\mathbf{b})_{(u,v)} = \mathbf{a}_u + \mathbf{b}_v \text{ for all } u \in V(G) \text{ and } v \in V(H).$$

We prove that, if $\mathbf{a}$ and $\mathbf{b}$ are recurrent configurations, then $\mathbf{a}\Box\mathbf{b}$ is a recurrent configuration. Moreover, we prove that the mapping $\widetilde{i_G} : SP(c(G),s) \to SP(c(G\Box H),t)$ given by

$$\widetilde{i_G}(\mathbf{a}) = \mathbf{a}\Box\mathbf{e},$$

is a canonical injective homomorphism between $SP(c(G),s)$ and $SP(c(G\Box H),t)$.

This results lead us to establish a conjecture that gives us an algebraic and a combinatorial description of the sandpile group of the cone of the hypercube Q_d of dimension d. More precisely,

$$K(c(Q_d)) \cong \bigoplus_{i=1}^{d} \mathbb{Z}_{2i+1}^{\binom{d}{i}} = \mathbb{Z}_3^d \oplus \mathbb{Z}_5^{\binom{d}{2}} \oplus \cdots \oplus \mathbb{Z}_{2d-1}^d \oplus \mathbb{Z}_{2d+1}.$$

and

$$SP(c(Q_d),s) = \bigoplus_{\{\beta\in\{0,1\}^d\}} \widetilde{K}_\beta \cong \bigoplus_{\{\beta\in\{0,1\}^d\}} \mathbb{Z}_{2|\beta|+1} = \bigoplus_{i=1}^{d} \mathbb{Z}_{2i+1}^{\binom{d}{i}},$$

where $\widetilde{K}_\beta = \{g_\beta(r,t) + (d-|\beta|)\mathbf{1} \mid 0 \le r,t \le d \text{ and } r = |\beta| \text{ or } t = |\beta|\} \cong \mathbb{Z}_{2|\beta|+1}$ for all $\beta \in \{0,1\}^d$ and $g_\beta(r,t) \in \mathbb{N}^{V(Q_d)}$ is given by

$$g_\beta(r,t)_{\mathrm{a}} = \begin{cases} r & \text{if } \beta\cdot\mathbf{a} \text{ is even,} \\ t & \text{if } \beta\cdot\mathbf{a} \text{ is odd.} \end{cases}$$

Finally, we also give an alternative way to compute the recurrent configurations of a multigraph using integer linear programming. This technique can be used in order to calculate the identity of the sandpile group and the order of a recurrent configuration. The last part of this thesis consist of the development of a C++ program that computes the combinatorial structure of the sandpile group of any multigraph.

Reconocimientos

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por otorgarme la beca durante estos dos años, y a mi asesor por los consejos y el apoyo que recibí.

Dedicatoria

A Paul Erdös (1913-1996) y a México.

○

Contents

Introduction

The main goal of this master thesis is the study of the sandpile group of a graph, in particular we are interested in the combinatorial structure of the generators of the sandpile group of the cone of the cartesian product of some graphs.

This thesis represent the first time that the combinatorial structure of the generators of the sandpile group is studied, all the previous results in the literature only describe the abstract structure of the sandpile group of a graph.

The concept of sandpile group was introduced by Bak et. al. [**3, 4**] in 1987 and Dhar [**24, 23**] in 1990. The sandpile group is also known as the critical group [**8**], the Jacobian group [**7**] and the Picard group [**34, 35**]. The sandpile automaton is also know as the chip-firing game, see [**9, 10, 7, 8**].

In the last twenty years, the sandpile group has been studied by several authors, see for instance [**35, 37, 8, 30, 22, 16, 28, 40, 15, 29, 33, 32, 2, 18, 19, 20**].

There exist several ways to define the sandpile group of a graph, maybe the most simple way is the following: Given a graph $G = (V, E)$ the sandpile group of G, denoted by $K(G)$, is defined as $\mathbb{Z}$-module given by

$$\mathbb{Z} \oplus K(G) = \mathbb{Z}^{|V|}/\mathrm{Im}L(G),$$

where $L(G)$ is the Laplacian matrix of the graph G

$$L(G)_{i,j} = \begin{cases} d_i & \text{if } i = j, \\ -\mu_{i,j} & \text{in other case.} \end{cases}$$

In section one of chapter two we present a combinatorial alternative way to define the sand pile group of a graph through recurrent configurations. The combinatorial version of the sand pile group of a graph, is denoted by $SP(G, s)$ where $s \in V(G)$. Is not difficult to prove that $K(G) \cong SP(G, s)$ for all $s \in V(G)$. The main difference between this two notations for the sandpile group of a graph is that when we use $SP(G, s)$ we are thinking mainly in their combinatorial structure and not only in their group structure.

In general is very difficult to calculate the sandpile group of a graph. There are very few families of graphs for which the critical group structure has been completely determined, such as:

(1) Complete graphs, in 1991 [**35**];
(2) Complete multipartite graphs, in 1991 [**35**];
(3) Cycles, in 1992 [**37**];
(4) Wheels, in 1999 [**8**];
(5) The cartesian product of complete graphs, in 2003 [**30**];
(6) The dihedral group graph, in 2003 [**22**];
(7) The Möbius ladder, in 2006, [**16**];
(8) The square of a cycle C_n^2, in 2006, [**28**];
(9) Regular trees, in 2007, [**40**];

(10) The cartesian product $P_4 \square C_n$, in 2008 [**15**];
(11) The cartesian product $\mathcal{K}_3 \square C_n$, in 2008 [**29**];
(12) The cartesian product $\mathcal{K}_m \square P_n$, in 2008 [**33**];
(13) Trees, in 2009 [**32**]; etc.

This thesis is divided in two chapters and one appendix:

i) Preliminaries,
ii) The sandpile group of a multigraph,
iii) CSandPile.

In chapter 1 we present the material necessary in order to introduce the results in the rest of the thesis.

Chapter 2 contains the main result of the thesis and is divided in seven sections. In the first section you can find the combinatorial definition of the sandpile group of a graph through stable and recurrent configurations. In the rest of the sections we study the sandpile group of a graph with connectivity one, the sandpile group of a thick cycle $\mathcal{C}_n$, the generators of a thick $\mathcal{C}_3$, the relation between the sandpile group and the integer linear programming, the group homomorphism that is induced by a uniform homomorphism of graphs, the group homomorphism that is induced by the inclusion of a graph G in a cartesian product of graphs $G \square H$ and finally the combinatorial structure of the sand pile group of the cone of the hypercube Q_d.

More precisely, section two contain a description of the sand pile group of a graph with connectivity one in function of theirs blocks.

Theorem 2.23 Let G a multigraph and G_1, G_2, ..., G_l be its blocks, then

$$K(G) = K(G_1) \oplus K(G_2) \oplus \cdots \oplus K(G_l).$$

Section three contains the description of the sandpile group of the thick cycle $\mathcal{C}_n$ and the combinatorial description of the powers of a generator of some thick $K(\mathcal{C}_3)$.

Theorem 2.28 Let $\mathcal{C}_n$ be the thick cycle with m_i the multiplicity between v_i and v_{i+1}, then

$$\Delta_i = \begin{cases} \gcd\left\{\prod_{1\leq j_1<\cdots<j_i\leq n} m_{j_1}\cdots m_{j_i}\right\} & \text{if } 1 \leq i \leq n-1, \\ (-1)^n \sum_{i=1}^{n} m_1 m_2 \cdots \hat{m_i} \cdots m_n & \text{if } i = n. \end{cases}$$

Theorem 2.33 Let $\alpha \in \mathbb{Z}_+$, if $\gcd(m_1, m_2) = 1$, $\gcd(m_2, m_3) = 1$ or $\gcd(m_3, m_1) = 1$ then

$$K(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3)) = \mathbb{Z}_\alpha \oplus \mathbb{Z}_{\alpha(m_1m_2+m_2m_3+m_3m_1)}$$

and $e = \left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right)$ is the identity of $SP(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3), v_1)$.

Theorem 2.34 Let $m \in \mathbb{Z}_+$, then $(m, 0)$ is a generator of $SP(\mathcal{C}_3(m, 1, 1), v_1)$ and

$$k(m,0) = \begin{cases} (m-j, m) & \text{if } k = 2j \leq 2m, \\ (m, j) & \text{if } k = 2j+1 \leq 2m+1, \end{cases}$$

in $SP(\mathcal{C}_3(m, 1, 1), v_1)$.

Theorem 2.35 Let $m \in \mathbb{Z}_+$ such that $\gcd(m, 2) = 1$, then $(m+1, m)$ is a generator of $SP(\mathcal{C}_3(m, 2, 2), v_1)$ and

$$k(m+1,m) = \begin{cases} (m+1, m-k+1) & \text{if } 1 \leq k \leq m+1, \\ (k-m-1-\frac{1+(-1)^k}{2}, m+\frac{1+(-1)^{k+1}}{2}) & \text{if } m+2 \leq k \leq 2m+2, \\ (m, 3m+2-k) & \text{if } 2m+3 \leq k \leq 3m+2, \\ (k-3(m+1)-\frac{1+(-1)^{k+1}}{2}, m+1-\frac{1+(-1)^{k+1}}{2}) & \text{if } 3m+3 \leq k \leq 4m+4. \end{cases}$$

Section four contains a result that give an alternative way to calculate the recurrent representative of a given configuration using integer linear programming. As corollaries we get the identity of the sandpile group of a graph, the degree of a recurrent configuration.

Theorem 2.36 Let G be a multigraph, $s \in V(G)$, $\mathbf{0} \leq \mathbf{c} \leq \mathbf{d}_{(G,s)} - \mathbf{1}$ a stable configuration of (G,s), where $\mathbf{d}_{(G,s)} = (d_G(v_1), ..., d_G(v_n))$, and $\mathbf{x}^*$ an optimal solution of the following integer linear problem:

$$\begin{array}{ll} \text{maximize} & |\mathbf{x}| \\ \text{subject to} & \mathbf{0} \leq L(G,s)^t\mathbf{x} + \mathbf{c} \leq \mathbf{d}_{(G,s)} - \mathbf{1} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{1}$$

then $L(G,s)^t\mathbf{x}^* + \mathbf{c} \in SP(G,s)$ and $[\mathbf{c}] = [L(G,s)^t\mathbf{x}^* + \mathbf{c}]$.

Corollary 2.38 Let G be a multigraph, $s \in V(G)$ and $\mathbf{x}^*$ an optimal solution of the following integer linear problem:

$$\begin{array}{ll} \text{maximize} & |\mathbf{x}| \\ \text{subject to} & \mathbf{0} \leq L(G,s)^t\mathbf{x} \leq \mathbf{d}_{(G,s)} - \mathbf{1} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{2}$$

then $L(G,s)^t\mathbf{x}^* \in SP(G,s)$ is the identity of $K(G)$.

Corollary 2.41 Let G be a multigraph, $s \in V(G)$, $\mathbf{c}$ a recurrent configuration of (G,s), and $(d,\mathbf{x})^*$ an optimal solution of the following integer linear problem:

$$\begin{array}{ll} \text{minimize} & d \\ \text{subject to} & d\mathbf{c} - L(G,s)^t\mathbf{x} = \mathbf{0} \\ & d \geq 1, \mathbf{x} \geq \mathbf{0}, \end{array} \tag{3}$$

then d is the degree of $\mathbf{c}$ in $K(G)$.

In section five we define the new concept of a uniform homomorphism of graphs and we prove that every surjective uniform homomorphism $f : G \to H$ induce an injective homomorphism of groups $\widetilde{f} : K(c(H)) \triangleleft K(c(G))$

Theorem 2.48 Let G and H be multigraphs and $f : G \to H$ be a surjective uniform homomorphism, then the induced mapping $\widetilde{f} : SP(c(H),s) \to SP(c(G),s)$ given by

$$\widetilde{f}(u)_v = u_x \in \mathbb{N}^{V(G)} \text{ for all } v \in S_x = f^{-1}(x),$$

is an injective homomorphism of groups, that is, $K(c(H)) \triangleleft K(c(G))$.

In section six we use the inclusion homomorphism of graphs between G and $G\square H$ in order to induce an injective homomorphism of groups $\widetilde{i_G} : K(c(G)) \to K(c(G\square H))$.

Theorem 2.52 Let G and H be two multigraphs, $s \in V(c(G)) \setminus V(G)$ and $t \in V(c(G\square H)) \setminus V(G\square H)$ and $i_G : \mathbb{Z}^{|V(G)|} \to \mathbb{Z}^{|V(G\square H)|}$ given by

$$i_G(\mathbf{a})_{(u,v)} = a_u + e_v \text{ for all } u \in V(G) \text{ and } v \in V(H),$$

where $\mathbf{a} = (a_v)_{v\in V(G)} \in \mathbb{Z}^{|V(G)|}$ and $\mathbf{e} = (e_v)_{v\in V(H)} \in SP(c(H),s)$ the identity of the sandpile group of $c(H)$. Then the induced mapping

$$\widetilde{i_G} : SP(c(G),s) \to SP(c(G\square H),t)$$

is well defined, that is, $\mathbf{a}$ is a recurrent configuration of $c(G)$ if and only if $\widetilde{i_G}(\mathbf{a})$ is a recurrent configuration of $c(G\square H)$. Moreover,

$$\widetilde{i_G} : K(c(G)) \to K(c(G\square H))$$

is an injective homomorphism of groups.

Section seven contains the main result of this thesis. The two previous results are used in order to prove the main theorem of this thesis. This theorem give us a combinatorial description of the powers of the generators of the sand pile group of the cone of the hypercube Q_d. The corollary give us a nice formula of the sand pile group of the cone of the hypercube Q_d.

Theorem 2.56 Let $\widetilde{K}_\beta = \{g_\beta(r,t) + (d - |\beta|)\mathbf{1} \mid 0 \leq r,t \leq d \text{ and } r = |\beta| \text{ or } t = |\beta|\}$ for all $\beta \in \{0,1\}^d$. Then

(*i*) $\mathbb{Z}_{2|\beta|+1} \cong \widetilde{K}_\beta \lhd K(c(Q_d))$ for all $\beta \in \{0,1\}^d$,
(*ii*) $d\mathbf{1} = g_1(d,d)$ is the identity of $\widetilde{K}_\beta$ for all $\beta \in \{0,1\}^d$,
(*iii*) $\widetilde{K}_\beta \cap \widetilde{K}_{\beta'} = d\mathbf{1}$ for all $\beta \neq \beta' \in \{0,1\}^d$,
(*iv*) $\widetilde{K}_\beta \cap \bigoplus_{\{\beta' \mid \mathrm{supp}(\beta') \subsetneq \mathrm{supp}(\beta)\}} \widetilde{K}_{\beta'} = d\mathbf{1}$ for all $\beta \in \{0,1\}^d$,
(*v*) $\widetilde{i}_{\beta,1}(SP(c(Q_\beta),s)) = \bigoplus_{\{\beta' \mid \mathrm{supp}(\beta') \subseteq \mathrm{supp}(\beta)\}} \widetilde{K}_{\beta'}$ for all $\beta \in \{0,1\}^d$,
(*vi*) $\widetilde{i}_{\beta,1}(SP(c(Q_\beta),s)) \cap \widetilde{i}_{\beta,1}(SP(c(Q_{\beta'}),s)) = \widetilde{i}_{\beta,1}(SP(c(Q_{\beta \odot \beta'}),s))$ for all $\beta', \beta \in \{0,1\}^d$,
where $(\mathbf{a} \odot \mathbf{b})_i = \mathbf{a}_i \cdot \mathbf{b}_i$ for all i.

Corollary 2.57 Let d be a natural number, then the sandpile group of the cone of the hypercube $c(Q_d)$ is given by:

$$K(c(Q_d)) \cong \bigoplus_{i=1}^{d} \mathbb{Z}_{2i+1}^{\binom{d}{i}} = \mathbb{Z}_3^d \oplus \mathbb{Z}_5^{\binom{d}{2}} \oplus \cdots \oplus \mathbb{Z}_{2d-1}^d \oplus \mathbb{Z}_{2d+1}.$$

Finally, the appendix contains the code of a program developed in C++, called CSandPile. This program was created in order to calculate the stable and recurrent configurations of a given graph, the powers of a recurrent configuration, the inverse of a recurrent configuration, the identity of the sandpile group, the order of the sandpile group, etc.

Almost all the results presented in this thesis are original, see for instance 2.23, 2.28, 2.33, 2.34, 2.35, 2.36, 2.38, 2.41, 2.48, 2.55, 2.59, and 2.60, and are collected in the article [**1**].

CHAPTER 1

Preliminaries

In this chapter we shall see the necessary tools that will be needed to develop the theory of sandpile group. Mainly, we give a glimpse in graph theory and $\mathbb{Z}$-modules.

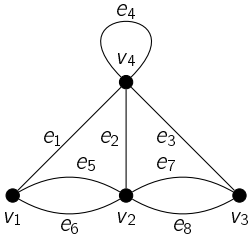
1 Graph Theory

In this section we shall see topics needed in the development of the theory of sandpile group, such as graphs and its subgraphs, connectedness, graph homomorphisms, the Laplacian matrix, some operations on graphs: the cone and the cartesian product.

1.1 Graphs and Subgraphs

Definition 1.1. *Let V be a finite nonempty set. A graph G is a pair (V, E), where E is a subset of the set of unordered pair of elements of V.*

The elements of V and E are called *vertices* and *edges*, respectively. If $G = (V, E)$ is a graph, the vertices of G will be denoted by $V(G)$, and the edges by $E(G)$. The number of vertices is the *order* of the graph and is denoted by $|G|$. An edge $e = \{x, y\}$ will be denoted by xy, then xy and yx are the same edge. Also, $e = xy$ is said to be *incident* to x and y. Furthermore, x and y are said to be *incident* to $e = xy$, and x and y are called the *ends* of e.



$$G = (V, E)$$
where
$$V = \{v_1, v_2, v_3, v_4\} \text{ and } E = \{e_1, e_2, e_3, e_4\}$$
with
$$e_1 = v_1v_4,\ e_2 = v_2v_4,\ e_3 = v_3v_4 \text{ and } e_4 = v_4v_4$$
$$e_5 = v_1v_2,\ e_6 = v_1v_2,\ e_7 = v_2v_3,\ e_8 = v_2v_3,$$

Figure 1.1. A graph.

We always can represent a graph by mean of a figure, where the vertices are represented by a point and an edge by a line joining its ends, see figure 1.1.

A *loop* is an edge incident to a unique vertex. A set of two or more edges with the same ends are called *multiple edges*. The number of multiple edges with ends u and v is called the *multiplicity* and denoted by $\mu_{u,v}$. For instance in figure 1.1, e_4 is a loop and, e_5 and e_6 are multiple edges with $\mu_{v_1,v_2} = 2$.

Two different vertices are *independent* if they are not adjacent. A set of vertices of G is *independent* if no two of its vertices are adjacent.

Definition 1.2. *A simple graph is a graph without loops and multiple edges.*

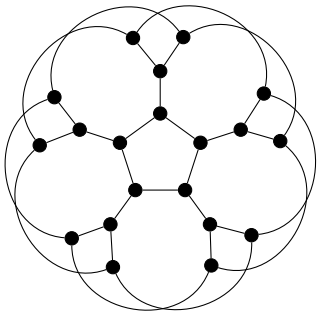


Figure 1.2. A simple graph.

If both multiple edges and loops are allowed, the graph is called *pseudograph*. However, we are more interested in graphs with no loops and multiple edges.

Definition 1.3. *A multigraph is a graph with multiple edges and no loops.*

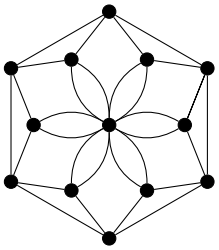


Figure 1.3. A multigraph.

The *underlying graph* of a multigraph is obtained by reducing all nonzero edge multiplicities to 1.

Definition 1.4. *A graph $G' = (V', E')$ is a subgraph of the graph $G = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$ and is denoted by $G' \subseteq G$.*

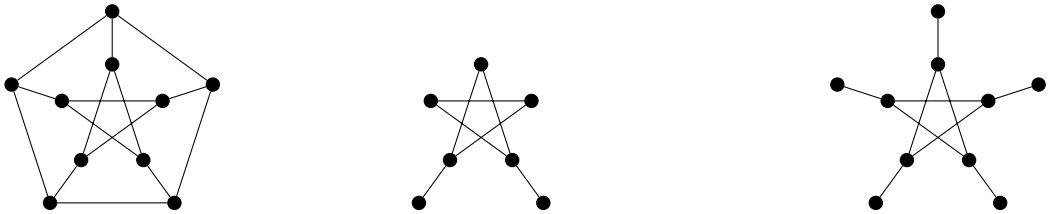
An *induced* subgraph $G[V'] = (V', E')$ is a subgraph of $G = (V, E)$ such that every edge $e \in E$ having its ends in V' is in E'. A *spanning* subgraph is a subgraph $G' \subseteq G$ with the same vertex set than G. In figure 1.4 we illustrate the induced and spanning subgraphs of the Petersen graph.

Definition 1.5. *The degree of a vertex $v \in G$ is the number of incident edges to v and is denoted by $d_G(v) = d(v)$.*

A graph is *k-regular* if every vertex has degree equal k. The number $\min\{d(v)|v \in V(G)\}$ is the *minimum degree* and it is denoted by $\delta(G)$. Similarly, the number $\max\{d(v)|v \in V(G)\}$ is the *maximum degree* and it is denoted by $\Delta(G)$. If we count all vertex's degree, we count the edges twice, then we have

$$\sum_{v \in G} d(v) = 2|E(G)|.$$

Another type of graph is when the edges have a direction.



(a) G (b) An induced subgraph of G (c) A spanning subgraph of G

Figure 1.4. Induced and spanning subgraph.

Definition 1.6. *A directed graph or digraph $D = (V, A)$ consists of a finite nonempty set V and a subset of the set of ordered pair of elements of A.*

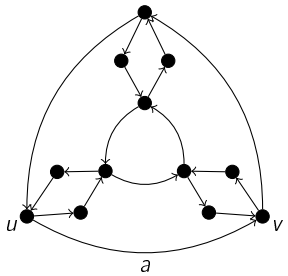


Figure 1.5. A digraph.

The elements of A are called *arcs*. An arc $a = (u, v)$ will be denoted by uv. In contrast to edges, uv and vu are different arcs. If $a = uv$ is an arc from u to v, u is called the *tail* and v is called the *head*, see figure 1.5. In digraph case, we will define *outdegree* $d_D^+(v)$ of a vertex v as the number of arcs with tail v and the *indegree* $d_D^-(v)$ of a vertex v as the number of arcs with head v.

Definition 1.7. *A pseudo-symmetric digraph D is a digraph such that $d_D^+(v) = d_D^-(v)$ for all $v \in V(D)$.*

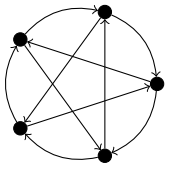


Figure 1.6. A pseudosymmetric digraph.

1.2 Connectedness

A *path* $P = v_0v_2v_3...v_k$ is a sequence of different vertices such that v_i and v_{i+1} are adjacent. Also, we say that P is a path from v_0 to v_k is a (v_0, v_k)-path, and k is the *length* of P. The *distance* $d_G(u, v)$ of two vertices $u, v \in V(G)$ is the shortest length of the (u, v)-paths.

Definition 1.8. *A graph is connected if for every different pair of vertices u, v there is a (u, v)-path.*

A *cut-vertex* is a vertex which removed increases the number of components.

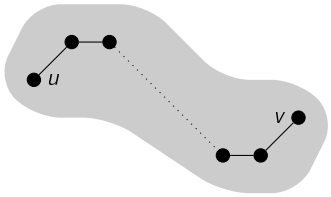


Figure 1.7. A path.

Definition 1.9. *A block is a maximal connected subgraph without a cut-vertex.*

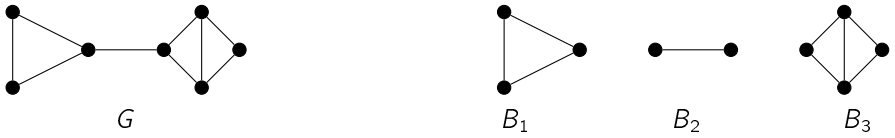


Figure 1.8. A graph and its blocks.

In the following of the subsection we will see some kinds of connected graphs and some of its properties. One of the simpler graphs a cycle.

Definition 1.10. *A cycle C_n is a 2-regular connected graph of n vertices, see figure 1.9.*

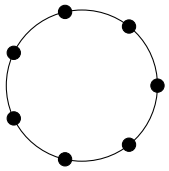


Figure 1.9. The cylce C_7.

Observe that cycles are simple graphs when vertices are more or equal than 3. Cycles helps us here to define another kind of graph.

Definition 1.11. *A tree T is a connected graph without cycles as subgraph, see figure 1.10.*

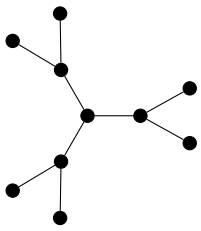


Figure 1.10. A tree.

The *spanning tree* is the graph that is a tree and a spanning subgraph at the same time. The number of spanning trees of G is denoted by $\tau(G)$.

An edge $e \in G$ is *deleted* by taking $E(G)$ without e as the edge set. The graph with the edge e deleted is denoted by $G \backslash e$. An edge $e \in G$ is *contracted* by deleting e and identifying its ends. We denote the graph with e contracted by G/e. See figure 1.11.

Now, we will see an equation that it serves to calculate the number of spanning trees in a recursively way.

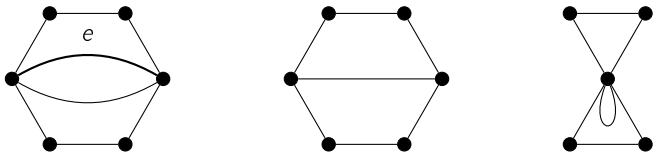


Figure 1.11. Deleting and contracting an edge.

Proposition 1.12. *If e is an edge, then* $\tau(G) = \tau(G\backslash e) + \tau(G/e)$.

Proof. Consider the following statements.

- Due to the fact that T is a spanning tree having e as an edge in G if and only if T/e is a spanning tree in G/e, the number of spanning trees that have e as an edge is equal to the number of spanning trees in G/e.
- It is clear that, the number of spanning trees that does not have e as an edge is equal to the number of spanning trees in $G\backslash e$.

On the other hand, since the number of spanning trees is equal to the sum of the number of spanning trees having e as an edge and the number of spanning trees that does not have e as an edge, then the formula follows. □

Definition 1.13. *The complete graph of n vertices* K_n *is the simple graph such that every pair of vertices are adjacent.*

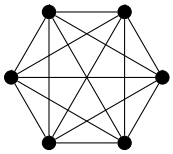


Figure 1.12. The complete graph of 6 vertices.

Definition 1.14. *The n-cube* Q_n *is the graph whose vertices are binary n-tuples with two vertices being adjacent if they differ in precisely one coordinate position.*

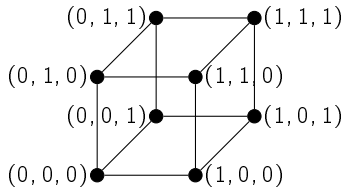


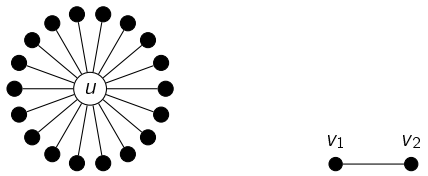
Figure 1.13. 3-cube Q_3.

1.3 Homomorphisms

Definition 1.15. *Let G and H be graphs. A homomorphism from G to H is a mapping f from* $V(G)$ *to* $V(H)$ *such that* $f(u)f(v) \in E(H)$ *whenever* $uv \in E(G)$.

For instance, if G and H are as in figure 1.14, then the following is a homomorphism from G to H

$$f(x) = \begin{cases} v_1 & \text{if } x = u \\ v_2 & \text{in other case} \end{cases}$$



(a) G (b) H

Figure 1.14. Homomorphism of graph.

Theorem 1.16. **[27]** *If $f : G \to H$ is a homomorphism, then*

$$d_H(f(u), f(v)) \leq d_G(u, v) \text{ for all } u, v \in V(G).$$

where $d_G(u, v)$ and $d_H(u, v)$ are the distance between u and v in G and H, respectively.

If G and H are digraphs, we can change, in the definition of homomorphism, edges by arcs.

Definition 1.17. *Let G and H be graphs. A isomorphism from G to H, written $G \cong H$, is a bijective mapping f from $V(G)$ to $V(H)$ such that $f(u)f(v) \in E(H)$ if and only if $uv \in E(G)$.*

Graphs in figure 1.15 are isomorphic, by mean of the isomorphisms $v_i \longmapsto u_i$ and $u_i \longmapsto w_i$.

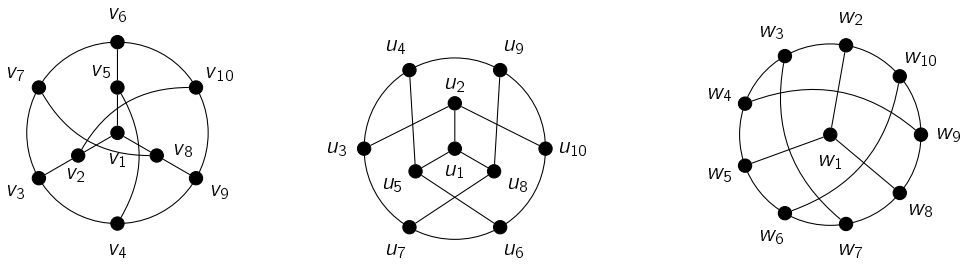


Figure 1.15. Isomorphic graphs.

1.4 The Laplacian Matrix of a graph

First of all, we will see two usual ways to associate a matrix to any graph. These are the incidence and adjacency matrices.

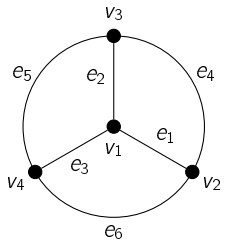
Definition 1.18. *The incidence matrix $B(G)$ is the matrix such that the element $B(G)_{i,j}$ is 1 if the j-th edge is incident to the i-th vertex, and 0 in other case.*

Definition 1.19. *The adjacency matrix $A(G)$ is the matrix such that the element $A(G)_{i,j}$ is the multiplicity of the multiple edges between the vertices v_i and v_j.*

Definition 1.20. *The Laplacian matrix $L(G)$ of the graph G is defined by*

$$L(G)_{i,j} = \begin{cases} d_i & \text{if } i = j, \\ -\mu_{i,j} & \text{in other case,} \end{cases}$$

where $\mu_{i,j}$ is the multiplicity of the edges between the vertex i and j.



Adjacency

	v_1	v_2	v_3	v_4
v_1	0	1	1	1
v_2	1	0	1	1
v_3	1	1	0	1
v_4	1	1	1	0

Incidency

	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	1	0	0	0
v_2	1	0	0	1	0	1
v_3	0	1	0	1	1	0
v_4	0	0	1	0	1	1

Figure 1.16. Adjacency and incidence matrix.

For instance, the following matrix is the laplacian matrix of the graph in figure 1.16.

$$L(G) = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix}$$

If $D(G)$ is the diagonal matrix such that $d_{i,i} = d_G(v_i)$, then $L(G) = D(G) - A(G)$.

We will denote by $M[u, v]$, the matrix obtained by removing of the matrix M the row corresponding to the vertex u and the column corresponding to the vertex v. And, $M[u]$ will denote the matrix obtained by removing of M the row and column corresponding to the vertex u.

Lemma 1.21. *Let G be a graph. If u is a vertex of G and e an edge incident to u, then*

$$\det L(G)[u] = \det L(G\backslash e)[u] + \det L(G/e)[u].$$

Proof. We can number the vertices in order that $u = v_n$. Suppose $e = uv_j$. Now, consider the following

$$\begin{aligned} \det L(G)[u] &= (-1)^{j+1} L(G)_{j,1} \det L(G)[u][v_j, v_1] + \cdots + (-1)^{j+j} L(G)_{j,j} \det L(G)[u][v_j, v_j] \\ &\quad + \cdots + (-1)^{j+n-1} L(G)_{j,n-1} \det L(G)[u][v_j, v_{n-1}] \\ &= ((-1)^{j+1} L(G)_{j,1} \det L(G)[u][v_j, v_1] + \cdots + (-1)^{j+j} (L(G)_{j,j} - 1) \det L(G)[u][v_j] \\ &\quad + \cdots + (-1)^{j+n-1} L(G)_{j,n-1} \det L(G)[u][v_j, v_{n-1}]) + \det L(G)[u][v_j] \\ &= \det L(G\backslash e)[u] + \det L(G)[u][v_j] \end{aligned}$$

Since, the vertex set of G/e is equal to $V(G)\backslash u$ and the incidence in vertices different of u and v_j does not change, we have that $L(G/e)[u] = L(G)[u][v_j]$. Thus, it turns out that

$$\det L(G)[u] = \det L(G\backslash e)[u] + \det L(G/e)[u].$$

□

Theorem 1.22. *Let G be a graph. If $u \in V(G)$, then $\det L(G)[u]$ is the number of spanning trees of G.*

Proof. Let $e = uv$ be an incident edge to u. We will proof that

$$\det L(G\backslash e)[u] = \tau(G\backslash e) \text{ and } \det L(G/e)[u] = \tau(G/e)$$

by induction on the number of edges. Let us denote n the number of vertices and m the number of edges. For $m = 1$,

- if $n = 2$, then $G =$  . Thus, $G\backslash e =$  and $G/e =$. Because of that, we have
$$L(G\backslash e) = \begin{bmatrix} 0 \end{bmatrix} \quad \text{and} \quad L(G/e) = \begin{bmatrix} 1 \end{bmatrix}$$
then $\det L(G\backslash e)[u] = \tau(G\backslash e) = 0$ and $\det L(G/e)[u] = \tau(G/e) = 1$.
- if $n \geq 3$, then $\det L(G\backslash e)[u] = \tau(G\backslash e) = 0$ and $\det L(G/e)[u] = \tau(G/e) = 0$.

Now assume that $m \geq 2$. There exist an edge e' different of e. Then, by induction hypothesis, proposition 1.12 and lemma 1.21 we have

$$\begin{aligned} \det L(G\backslash e)[u] &= \det L(G\backslash e\backslash e')[u] + \det L(G\backslash e/e')[u] \\ &= \tau(G\backslash e\backslash e') + \tau(G\backslash e/e') \\ &= \tau(G\backslash e), \end{aligned}$$

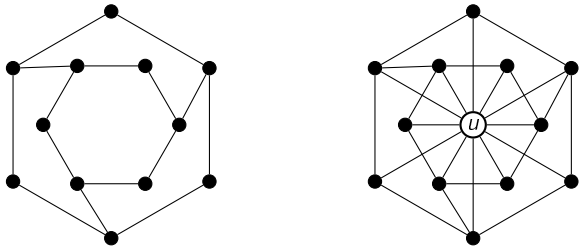
and

$$\begin{aligned} \det L(G/e)[u] &= \det L(G/e\backslash e')[u] + \det L(G/e/e')[u] \\ &= \tau(G/e\backslash e') + \tau(G/e/e') \\ &= \tau(G/e). \end{aligned}$$

Thus, the formula follows. □

1.5 Operations on Graphs

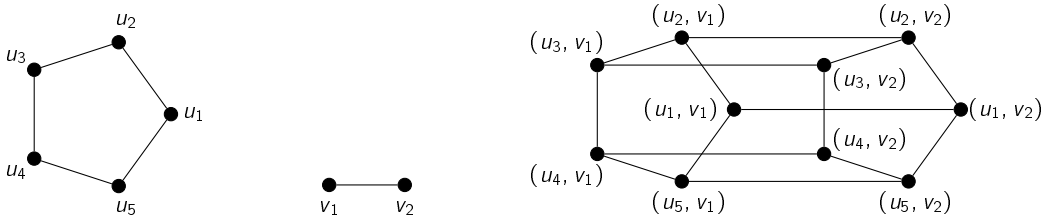
Definition 1.23. *The cone $c(G)$ of a graph G is defined as the graph that has $V(G)$ adding a new vertex u as its vertex set, and $E(G) \cup \{uv \mid v \in V(G)\}$ as its edge set.*



(a) G (b) $c(G)$

Figure 1.17. Cone of a graph.

Definition 1.24. *The cartesian product $G_1 \square G_2$ of two graphs G_1 and G_2 is the simple graph with $V(G_1) \times V(G_2)$ as its vertex set and two vertices u_1v_1 and u_2v_2 are adjacent in $G_1 \square G_2$ if and only if either $u_1 = u_2$ and $v_1v_2 \in E(G_2)$, or $v_1 = v_2$ and $u_1u_2 \in E(G_1)$.*



(a) C_5 (b) K_2 (c) $C_5 \square K_2$

Figure 1.18. Cartesian product.

Lemma 1.25. $G_1 \square (G_2 \square G_3) \cong (G_1 \square G_2) \square G_3$

Proof. Let us define $f : V((G_1 \square G_2)\square G_3) \to V(G_1 \square (G_2 \square G_3))$ such that $f((x,y),z) = (x,(y,z))$ for all $x \in G_1$, $y \in G_2$ and $z \in G_3$. So, we need to proof that $((x_1,y_1),z_1)((x_2,y_2),z_2) \in E((G_1 \square G_2)\square G_3)$ if and only if $(x_1,(y_1,z_1))(x_2,(y_2,z_2)) \in (G_1 \square (G_2 \square G_3))$.

$\Rightarrow$) Let $((x_1,y_1),z_1)((x_2,y_2),z_2) \in E((G_1 \square G_2)\square G_3)$, thus
(1) $(x_1,y_1) = (x_2,y_2)$ and $z_1z_2 \in E(G_3)$, or
(2) $z_1 = z_2$ and $(x_1,y_1)(x_2,y_2) \in E(G_1 \square G_2)$.
If (1) happens, we have $x_1 = x_2$, $y_1 = y_2$ and $z_1z_2 \in E(G_3)$. Hence, $x_1 = x_2$ and $(y_1,z_1)(y_2,z_2) \in E(G_2 \square G_3)$. Then, $(x_1,(y_1,z_1))(x_2,(y_2,z_2)) \in (G_1 \square (G_2 \square G_3))$.
If (2) happens, we have either $z_1 = z_2$, $x_1 = x_2$ and $y_1y_2 \in E(G_2)$, or $z_1 = z_2$, $y_1 = y_2$ and $x_1x_2 \in E(G_1)$. Hence, either $x_1 = x_2$ and $(y_1,z_1)(y_2,z_2) \in E(G_2 \square G_3)$, or $x_1x_2 \in G_1$ and $(y_1,z_1) = (y_2,z_2)$. In both cases, it turns out that $(x_1,(y_1,z_1))(x_2,(y_2,z_2)) \in (G_1 \square (G_2 \square G_3))$.

$\Leftarrow$) Let $(x_1,(y_1,z_1))(x_2,(y_2,z_2)) \in E(G_1 \square (G_2 \square G_3))$, thus
(1) $x_1 = x_2$ and $(y_1,z_1)(y_2,z_2) \in E(G_2 \square G_3)$, or
(2) $x_1x_2 \in G_1$ and $(y_1,z_1) = (y_2,z_2)$.
If (1) happens, we have either $x_1 = x_2$, $y_1 = y_2$ and $z_1z_2 \in E(G_3)$, or $x_1 = x_2$, $z_1 = z_2$ and $y_1y_2 \in E(G_2)$. Hence, either $z_1z_2 \in E(G_3)$ and $(x_1,y_1) = (x_2,y_2)$ or, $z_1 = z_2$ and $(x_1,y_1)(x_2,y_2) \in E(G_1 \square G_2)$. Then, in both cases $((x_1,y_1),z_1)((x_2,y_2),z_2) \in E((G_1 \square G_2)\square G_3)$.
If (2) happens, we have $x_1x_2 \in E(G_1)$ and $z_1 = z_2$, $y_1 = y_2$. Hence, $z_1 = z_2$ and $(x_1,y_1)(x_2,y_2) \in E(G_1 \square G_2)$. Then, we have $((x_1,y_1),z_1)((x_2,y_2),z_2) \in E((G_1 \square G_2)\square G_3)$.

□

Then, inductively, we can forget the parentheses for products of more than 2 graphs. Also, we have that the cartesian product

$$G = G_1 \square \cdots \square G_n$$

can be defined by $V(G) = \{(v_1,...,v_n) \mid v_i \in G_i \text{ for all } i = 1,...,n\}$, and two edges $(v_1,...,v_n)$ and $(v_1',...,v_n')$ are adjacent if and only if there exist a unique $j \in \{1,...,n\}$ such that $v_j = v_j'$, and then $v_i \neq v_i'$ for $i \neq j$.

If we take $K_2 = (\{0,1\},\{01\})$, it is easy to see the following corollary.

Corollary 1.26. $Q_n \cong \underbrace{K_2 \square \cdots \square K_2}_{n\text{-times}}$.

Lemma 1.27. $G_1 \square G_2 \cong G_2 \square G_1$

Proof. Let $f : V(G_1 \square G_2) \to V(G_2 \square G_1)$ such that $(x,y) \mapsto (y,x)$. It is clear that f is bijective. Now, $(u_1v_1)(u_2v_2) \in E(G_1 \square G_2) \Leftrightarrow$ either $u_1 = u_2$ and $v_1v_2 \in E(G_2)$, or $v_1 = v_2$ and $u_1u_2 \in E(G_1)$. If and only if, either $v_1 = v_2$ and $u_1u_2 \in E(G_1)$, or $u_1 = u_2$ and $v_1v_2 \in E(G_2) \Leftrightarrow (v_1u_1)(v_2u_2) \in E(G_2 \square G_1)$. □

Since, $G \square K_1 \cong G$ and by lemmas 1.27 and 1.25, we have the following proposition:

Proposition 1.28. *Let $\mathcal{G}$ be set of graphs. Then $\mathcal{G}(\square, K_1)$ is monoid.*

2 $\mathbb{Z}$-modules and the Smith Normal Form

In this section we shall focus in the properties of $\mathbb{Z}$ and the Smith normal form.

2.1 $\mathbb{Z}$-modules

Definition 1.29. *A group is a set G together with a binary operation $\cdot : G \times G \to G$ that satisfy the following four conditions:*

- *$a, b \in G$, $a \cdot b \in G$ (closure),*
- *$a, b, c \in G$ implies that $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ (associativity),*

- *G has a identity element e such that $a \cdot e = e \cdot a = a$ for every $a \in G$ (identity),*
- *For all $a \in G$ there exist an element a^{-1} such that $a \cdot a^{-1} = a^{-1} \cdot a = e$ (inverse).*

For instance, let S_n be the set of all bijective mappings of a set of n elements onto itself. The set S_n is closed under the composition. It is because the composition of injective mappings is injective and the composition of surjective mappings is again surjective. Also, S_n is associative under composition, since $\alpha \circ (\beta \circ \gamma)(x) = \alpha(\beta(\gamma(x))) = (\alpha \circ \beta) \circ \gamma(x)$ for all $\alpha, \beta, \gamma \in S_n$ and $x \in [n] := \{1, ..., n\}$. In addition, $1_{[n]}$, such that $1_{[n]}(x) := x$ for all $x \in [n]$, is in S_n, and $1_{[n]} \circ \alpha = \alpha \circ 1_{[n]}(x) = \alpha$. Additionally, If $\alpha \in S_n$, then define α^{-1} such that $\alpha(x) \mapsto x$ for all $x \in [n]$, therefore $\alpha \circ \alpha^{-1} = \alpha^{-1} \circ \alpha = 1_{[n]}$. Then S_n with the composition of maps is a group, called the *symmetric group*.

Definition 1.30. *A group G is called abelian if for every $a, b \in G$, $a \cdot b = b \cdot a$.*

For example, the set $\mathbb{Z}$ of all integers under the ordinary addition is an abelian group. Actually, $\mathbb{Z}$ is a commutative ring.

Definition 1.31. *A commutative ring is a set R together with two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ such that R with $+$ operation is an abelian group, and it satisfy the following conditions:*

(1) $a, b \in R \Rightarrow a \cdot b \in R$,
(2) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all $a, b, c \in R$,
(3) $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(a + b) \cdot c = a \cdot c + b \cdot c$ for all $a, b, c \in R$,
(4) There exist an element $1 \in R$ such that $1 \cdot a = a \cdot 1 = a$ for $a \in R$,
(5) $a \cdot b = b \cdot a$ for all $a, b \in R$.

Another common examples are $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$.

Definition 1.32. *A subset S of a group G is called subgroup if $s \in S$ implies $s^{-1} \in S$ and $s, t \in S$ imply $s \cdot t \in S$.*

The set $2\mathbb{Z} = \{2i \mid i \in \mathbb{Z}\}$ is an additive subgroup of $\mathbb{Z}$. Let G be a group, If $a \in G$ we define $\langle a \rangle = \{a^i \mid i \in \mathbb{Z}\}$. $\langle a \rangle$ is a subgroup called the *cyclic subgroup generated by a*.

Definition 1.33. *A group G is called cyclic if there exist $a \in G$ such that $\langle a \rangle = G$.*

The *congruence class* $\{b \mid b \equiv a \mod n\}$ of an integer $a \mod n$ will be denoted by $\overline{a}$. The set $\mathbb{Z}_n$ of all congruence classes $\mod n$ is an abelian group with the operation $\overline{a} + \overline{b} = \overline{a + b}$, $\overline{0}$ as identity and $\overline{1}$ as generator.

Definition 1.34. *A $\mathbb{Z}$-module is an abelian group M together with an operation $\cdot : \mathbb{Z} \times M \to M$ such that*

(1) $(a + b) \cdot x = a \cdot x + b \cdot x$,
(2) $a \cdot (x + y) = a \cdot x + a \cdot y$,
(3) $a \cdot (b \cdot x) = (ab) \cdot x$,
(4) $1 \cdot x = x$.

for all $a, b \in \mathbb{Z}$ and $x, y \in M$.

Since there is a unique way to express $n \cdot x$ as $x + \cdots + x$ n-times for every $n \in \mathbb{Z}$, we have that every $\mathbb{Z}$-module is an abelian group.

Definition 1.35. *Let M and N be $\mathbb{Z}$-modules. A map $\varphi : M \to N$ is called $\mathbb{Z}$-module homomorphism (or simply homomorphism) if, for all $a \in \mathbb{Z}$ and $m, n \in M$,*

(1) $\varphi(a \cdot m) = a \cdot \varphi(m)$,
(2) $\varphi(m + n) = \varphi(m) + \varphi(n)$.

Definition 1.36. *A submodule N of M is an additive subgroup of M that if $a \in \mathbb{Z}$ y $x \in N$, then $a \cdot x \in N$.*

Definition 1.37. *Let M be a $\mathbb{Z}$-module and $N \subset M$ be a submodule. We define the quotient module M/N by*

$$M/N = \{m + N \mid m \in M\}$$

Thus M/N is the set of equivalence classes of elements of M, where $m, n \in M$ are equivalent if $m - n \in N$.

Definition 1.38. *Let M_i, $i \in I$, be $\mathbb{Z}$-modules. The direct sum $\bigoplus_{i\in I}$ is defined by*

$$\bigoplus_{i\in I} M_i = \{(m_i)_{i\in I} \mid m_i \in M_i, m_i \neq 0 \text{ for only finitely many } i\}.$$

Proposition 1.39. *Let $\varphi : M \to N$ be an homomorphism, then*

$$Im(\varphi) \cong M/Ker(\varphi).$$

Proof. Let us define the map $\psi : M/\mathrm{Ker}(\varphi) \to \mathrm{Im}(\varphi)$ by $\psi(m + \mathrm{Ker}(\varphi)) := \varphi(m)$. ψ is well defined because if $x \sim y$, therefore $x - \mathrm{Ker}(\varphi) = y - \mathrm{Ker}(\varphi)$ and thus $\varphi(x) = \psi(x + \mathrm{Ker}(\varphi)) = \psi(y + \mathrm{Ker}(\varphi)) = \varphi(y)$. ψ is a homomorphism because $\psi(x + y + \mathrm{Ker}(\varphi)) = \varphi(x + y) = \varphi(x) + \varphi(y) = \psi(x + \mathrm{Ker}(\varphi)) + \psi(y + \mathrm{Ker}(\varphi))$ and $\psi(a \cdot x + \mathrm{Ker}(\varphi)) = \varphi(a \cdot x) = a \cdot \varphi(x) = a \cdot \psi(x + \mathrm{Ker}(\varphi))$. ψ is surjective by definition. Now, let $\psi(x + \mathrm{Ker}(\varphi)) = \varphi(x) = 0$, therefore $x \in \mathrm{Ker}(\varphi)$. Then $x + \mathrm{Ker}(\varphi) = \mathrm{Ker}(\varphi)$ which is the 0-element in $M/\mathrm{Ker}(\varphi)$. □

Theorem 1.40. *Let M_1, M_2 be $\mathbb{Z}$-modules and N_1 submodule of M_1 and N_2 submodule of M_2. If $M = M_1 \oplus M_2$ and $N = N_1 \oplus N_1$, then*

$$M/N \cong M_1/N_1 \oplus M_2/N_2.$$

Proof. Consider the $\mathbb{Z}$-module homomorphism $\varphi : M_1 \oplus M_2 \to M_1/N1 \oplus M_2/N_2$ such that $(x, y) \mapsto (x + N_1, y + N_2)$. Clearly, φ is surjective, thus $\mathrm{Im}(\varphi) = M_1/N_1 \oplus M_2/N_2$. We will proof that $\mathrm{Ker}(\varphi) = N_1 \oplus N_2$. If $(x, y) \in N_1 \oplus N_2$, then $\varphi(x, y) = (x + N_1, y + N_2) = (N_1, N_2)$ which is 0-element in $M_1/N_1 \oplus M_2/N_2$. On the other hand, if $(x, y) \in \mathrm{Ker}(\varphi)$, then $\varphi(x, y) = (N_1, N_2)$, thus $x \in N_1$ and $y \in N_2$. it turns out in $(x, y) \in N_1 \oplus N_2$. Finally, by Proposition 1.39, the equation $M/N \cong M_1/N_1 \oplus M_2/N_2$ is evident. □

Definition 1.41. *A $\mathbb{Z}$-module M is called finitely generated if $M = \sum_{i=1}^{n} \mathbb{Z} \cdot m_i$ for suitable $m_1, m_2, ..., m_n \in M$. Then, we write $M = \langle m_1, m_2, ..., m_n \rangle$, and $m_1, m_2, ..., m_n$ are called generators of M.*

Theorem 1.42. *Let K be a submodule of $\mathbb{Z}^n$ is finitely generated by at most $m \leq n$ elements.*

Proof. We will proof it by induction. Let K a submodule of $\mathbb{Z}^n$. If $n = 1$, let m the least positive integer in K. All elements in K are multiple of m. Because if there is an element p that it is not divisible by m, then by the division algorithm there exist q, and r such that $p = mq + r$ with $0 \leq r < m$. That is, r is linear combination of m and p least than m which is a contradiction. Thus, $K = m\mathbb{Z}$.

Now, let $n \geq 2$ and suppose that every submodule of $\mathbb{Z}^{n-1}$ is finitely generated. Let K a submodule of $\mathbb{Z}^n$. Let m the least positive integer of the first components of the elements of K. Next, let $k = (k_1, k_2, ..., k_n) \in K$. By the same argument in the case $n = 1$, we have that the first component of the elements of K is a multiple of m. Hence $k_1 = qm$. Choose an element $(m, x_2, ..., x_n) \in K$. Thus $k = q(m, x_2, ..., x_n) + (0, k_2 - qx_2, ..., k_n - qx_n)$. The set of elements of the form $(k_2 - qx_2, ..., k_n - qx_n)$ are a submodule in $\mathbb{Z}^{n-1}$. Using the induction hypothesis, the submodule form $(0, k_2 - qx_2, ..., k_n - qx_n)$ is finitely generated. And adding $(m, x_2, ..., x_n)$ to the generators set, it turns out that K is finitely generated. □

Let $M \cong \mathbb{Z}^n/K$ be $\mathbb{Z}$-module. By theorem 1.42, we conclude that K is finitely generated. Hence, we can assume that the set $(f_1, f_2, ..., f_m)$ is a set of generators of the submodule K and let $(e_1, e_2, ..., e_n)$ be a base of $\mathbb{Z}^n$. Writing $f_i = a_{i,1}e_1 + a_{i,2}e_2 + \cdots + a_{i,n}e_n$, we have a matrix $m \times n$ matrix $A = (a_{i,j})$ that is known as the *relations matrix* of the ordered set of generators $(f_1, ..., f_n)$ in terms of the ordered base $(e_1, ..., e_n)$. We can choose any other base $(e'_1, ..., e'_n)$ for $\mathbb{Z}^n$ such that $e'_i = \sum_{j=1}^{n} p_{i,j}e_j$ where $P = (p_{i,j})$ is an invertible matrix in $M_n(\mathbb{Z})$ with inverse $P^{-1} = (p'_{i,j})$. On the other hand, if $Q = (q_{i,j})$ is an invertible matrix in $M_n(\mathbb{Z})$ with inverse $Q^{-1} = (q'_{i,j})$, then $f'_1, ..., f'_n$, where $f'_k = \sum_{i=1}^{n} q_{k,i}f_i$, is another set of generators for K. Furthermore,

$$f'_l = \sum_{i=1}^{n} q_{l,i}f_i = \sum_{i,j} q_{l,i}a_{i,j}e_j = \sum_{i,j,k} q_{l,i}a_{i,j}p'_{j,k}e'_k.$$

Hence the relation matrix of the f''s relative to the e''s is $A' = QAP^{-1}$.

Definition 1.43. *Two $m \times n$ matrices A, B with entries in $\mathbb{Z}$ are said to be equivalent if there exist an invertible matrices $P \in M_m(\mathbb{Z})$ and $Q \in M_n(\mathbb{Z})$ such that $B = PAQ$.*

We Summarize the before information in the following lemma.

Lemma 1.44. *If $A, B \in M_n(\mathbb{Z})$ are equivalent, then $\mathbb{Z}^n/A \cong \mathbb{Z}^n/B$.*

Proposition 1.45. *Let A be a matrix in $M_n(\mathbb{Z})$ and $M = \mathbb{Z}^n/A$. Then, $M \cong \mathbb{Z}_{d_1} \oplus \cdots \oplus \mathbb{Z}_{d_n}$ with $d_i \mid d_{i+1}$ if and only if there exist invertible matrices P and Q such that*

$$PAQ = \begin{bmatrix} d_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_n \end{bmatrix}$$

Proof. $\Leftarrow$) Applying recursively proposition 1.39 we have the result.
$\Rightarrow$) It follows from the proof of theorem 1.50, (that, is in the next section). □

Remark 1.46. *Since P and Q are invertible we have that the number of elements in $M \cong \mathbb{Z}^n/A$ is $\det(A) = d_1 d_2 \cdots d_n$.*

Definition 1.47. *Let $\varphi : M \to N$ be an homomorphism, then*

$$Coker(\varphi) := N/Im(\varphi)$$

is called the cokernel of φ.

Remark 1.48. *Let G be a connected graph and $L(G)$ be the Laplacian matrix. Considering $L(G)$ as a linear map from $\mathbb{Z}^n$ to itself, its cokernel has the form*

$$\mathbb{Z}^n/Im(L(G)) \cong \mathbb{Z} \oplus K(G),$$

where $K(G)$ is defined to be the critical group (also called the Picard group [**2**]*, Jacobian group* [**4**]*, or sandpile group* [**7**]*).*

2.2 The Smith Normal Form

In the following we will consider the problem of selecting among the matrices equivalent to a given matrix A one that has a particular simple form. We will obtain this matrix by means of the following operations:

i. Interchanging two rows or two columns,
ii. Multiplying a row or a column by -1,
iii. Adding a multiple of a row to another row, or adding a multiple of a column to another column,

which can be described by mean of product of matrices:

I. Interchanging two rows (column) i, j is performed by a left (right) product of A by the identity matrix which the rows (column) i and j had been interchanged,
II. Multiplying the *i-th* row (column) is performed by a left (right) product of A by the identity matrix which the *i-th* row (column) has been multiplied by -1,
III. Adding a multiple of the *i-th* row (column) to the *j-th* row (column) is performed by the left (right) product of A by a matrix of the following form

$$\begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & m & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & \cdots & m & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}.$$

The above matrices are called *elementary matrices of types I, II and III*. It is clear that elementary matrices of types I, II and III have $\det = \pm 1$ and they are invertible.

Definition 1.49. *A matrix $A \in M_n(\mathbb{Z})$ is in Smith normal form if it is diagonal, $a_{i,i} \geq 0$ for all $0 \leq i \leq n$ and $a_{i,i} \mid a_{i+1,i+1}$ when $a_{i,i} \neq 0$ and $a_{i+1,i+1} \neq 0$.*

For example, the matrix

$$\begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 12 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

is in Smith normal form.

Theorem 1.50. *If $A \in M_{m,n}(\mathbb{Z})$, then A is equivalent to the diagonal matrix with*

$$diag\{d_1, d_2, ..., d_n, 0, ..., 0\} = \begin{bmatrix} d_1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & d_n & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix}$$

where the $d_i \neq 0$ and $d_i \mid d_j$ if $i \leq j$.

Proof. If $A = 0$ there is nothing to do. Interchanging rows and columns we can bring the integer with least absolute value to the $(1,1)$ position. So we can suppose it is there initially. Let $A = (a_{i,j})$ and d_1 be the greatest common divisor of $a_{1,1}$ and $a_{2,1}$. There exist $p, q \in \mathbb{Z}$ such that $pa_{1,1} + qa_{2,1} = d_1$, and since d_1 divides $a_{1,1}$ and $a_{2,1}$ there exist α and β such that $a_{1,1} = d_1\alpha$ and $a_{2,1} = d_1\beta$. Hence $p\alpha + q\beta = 1$. Thus the matrix

$$P_{1,2} = \begin{bmatrix} p & q & \mathbf{0} \\ -\beta & \alpha & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{n-2} \end{bmatrix}$$

is invertible. But when $P_{1,2}$ is multiplied by the left to A we obtain

$$\begin{bmatrix} d_1 & a'_{1,2} & \cdots \\ 0 & a'_{2,2} & \cdots \\ a_{3,1} & a_{3,2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Following the before idea, we can make zero all the entries of first column and first row, except by the position $(1,1)$, to obtain a matrix equivalent to A of the form

$$\begin{bmatrix} b_{1,1} & 0 & \cdots & 0 \\ 0 & b_{2,2} & \cdots & b_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & b_{n,2} & \cdots & b_{n,n} \end{bmatrix}$$

Now, if there exist a $b_{i,j}$ with $2 \leq i, j \leq n$ such that $b_{1,1} \nmid b_{i,j}$, we add the *j-th* column to the first column. And we eliminate the first row in the same way. Finishing this process we obtain a new matrix, that, the position $(1,1)$ divides the position (i,j) because in the $(1,1)$ position is the g.c.d. of the first row. Thus we can eliminate $b_{i,j}$'s that are not divisible by $b_{1,1}$, and we get a matrix that the position $(1,1)$ divides each position in (i,j) with $2 \leq i, j \leq n$. Now, we repeat recursively the before idea to the sub-matrix

$$\begin{bmatrix} b_{k,k} & \cdots & b_{k,n} \\ \vdots & \ddots & \vdots \\ b_{n,k} & \cdots & b_{n,n} \end{bmatrix}$$

for $2 \leq k \leq n$. It turns out the result. □

Example 1.51. *Let us compute the Smith normal form of the matrix*

$$\begin{bmatrix} 21 & 105 \\ 14 & 147 \end{bmatrix}$$

using the method described in the proof of the theorem. First of all, we interchange the first and the second row.

$$\begin{bmatrix} 14 & 147 \\ 21 & 105 \end{bmatrix}$$

Thus, the least element is in the $(0,0)$ *position. Now, the greatest common divisor of* 14 *and* 147 *is* 7*, and since*

$$\begin{aligned} 147 &= 10 \cdot 4 + 7, \\ 14 &= 2 \cdot 7 + 0, \end{aligned}$$

we have that $1 \cdot 147 - 10 \cdot 14 = 7$ *and then* $1 \cdot 21 - 10 \cdot 2 = 1$*. Hence*

$$\begin{bmatrix} 14 & 147 \\ 21 & 105 \end{bmatrix} \begin{bmatrix} -10 & 21 \\ 1 & -2 \end{bmatrix} = \begin{bmatrix} 7 & 0 \\ 105 & 231 \end{bmatrix}.$$

Since 105 *is divisible by* 7*, it turns out that it is equivalent to*

$$\begin{bmatrix} 7 & 0 \\ 0 & 231 \end{bmatrix}.$$

The diagonal elements in the Smith normal form are called *invariant factors*. The following theorem will give us formulas to obtain the invariant factors.

Theorem 1.52. *Let A be a matrix in $M_{m,n}(\mathbb{Z})$ and suppose the rank of A is r. For each $i \leq r$, let Δ_i be the g.c.d. of the $i \times i$ minors of A. Then any set of invariant factors for A differ by unit multipliers from the elements*

$$d_1 = \Delta_1, d_2 = \Delta_2 \Delta_1^{-1}, ..., d_r = \Delta_r \Delta_{r-1}^{-1}.$$

Proof. Let $Q = (q_{i,j})$ be an $m \times m$ matrix in $M_n(\mathbb{Z})$. Then the (i,j)-entry of QA is $\sum_k q_{i,k} a_{k,j}$. Thus the rows of QA are linear combinations of the rows of A. Hence the $i \times i$ minors of QA are linear combinations of the $i \times i$ minors of A and so the g.c.d. of the $i \times i$ minors of A is divisor of the g.c.d. of the $i \times i$ minors of QA. Similarly, if $P \in M_n(\mathbb{Z})$, the columns of AP are linear combinations of the columns of A, therefore the g.c.d. of the $i \times i$ minors of A is divisor of the g.c.d. of the $i \times i$ minors of AP. Using these two facts and symmetry of the equivalence, we have that if A and B are equivalent, the g.c.d. of the $i \times i$ minors of A and B are the same. Now, let $B = diag\{d_1, ..., d_r, 0, ..., 0\}$ be the Smith normal form of A. Since if $i \leq j$, then $d_i \mid d_j$, we have that the g.c.d. of the $i \times i$ minors of B Δ_i is $d_1 d_2 \cdots d_i$. Thus, $\Delta_i / \Delta_{i-1} = d_i$. □

CHAPTER 2

The sandpile group of a multigraph

The concept of sandpile group was introduced by Bak et. al. [**3, 4**] in 1987 and Dhar [**24, 23**] in 1990. The sandpile group is also know as the critical group [**8**], the Jacobian group [**7**] and the Picard group [**34, 35**]. The sandpile automaton is also know as the chip-firing game, see [**9, 10, 7, 8**].

In the last twenty years, the sandpile group has been studied by several authors, see for instance [**2, 8, 30, 19, 28, 20, 22, 29, 33, 32, 35, 37, 16, 15, 40, 18**].

This chapter contains the main result of the thesis and is divided in seven sections. In the first section you can find the combinatorial definition of the sand pile group of a graph through stable and recurrent configurations. In the rest of the sections we study the sandpile group of a graph with connectivity one, the sandpile group of a thick cycle $\mathcal{C}_n$, the generators of a thick $\mathcal{C}_3$, the relation between the sandpile group and the integer linear programming, the group homomorphism that is induced by a uniform homomorphism of graphs, the group homomorphism that is induced by the inclusion of a graph G in a cartesian product of graphs $G\square H$ and finally the combinatorial structure of the sand pile group of the cone of the hypercube Q_d.

1 Stable and recurrent configurations

In the following, every multigraph will be connected. Also, every multigraph G will have a distinguished vertex $s \in V(G)$, called *sink*. The non-sink vertices set will be denoted by $\tilde{V}$.

1.1 Stable configurations

Definition 2.1. *Let $G = (V, E)$ be a multigraph with $V = \{v_1, ..., v_n\}$. A configuration of (G, s) is an element $u \in \mathbb{N}^n$.*

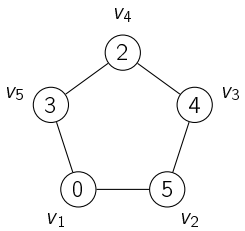


Figure 2.1. The configuration $(0, 5, 4, 2, 3)$ on C_5.

Definition 2.2. *Let $G = (V, E)$ be a multigraph and u a configuration. A non-sink vertex v is called stable if $deg_G(v) < u_v$.*

For instance, in figure 2.1 the vertex v_1 is stable and the others are unstable.

Definition 2.3. *Let $G = (V, E)$ be a multigraph and u a configuration. A configuration is called stable if every $v \in \tilde{V}$ is stable.*

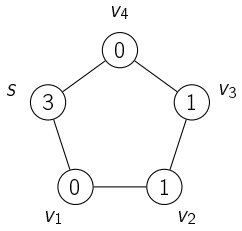


Figure 2.2. A stable configuration.

Toppling the vertex v_i is performed by decreasing u_i by his degree d_i, and adding the multiplicity $\mu_{i\,j}$ to each adjacent vertex to v_j. For instance, taking the unstable configuration $u = (1, 0, 2, 2, 2)$ in the graph C_5 with v_5 as sink, the stable configuration $v = (1, 1, 1, 1, 3)$ will be reached after a finite sequence of topplings. See figure 2.3.

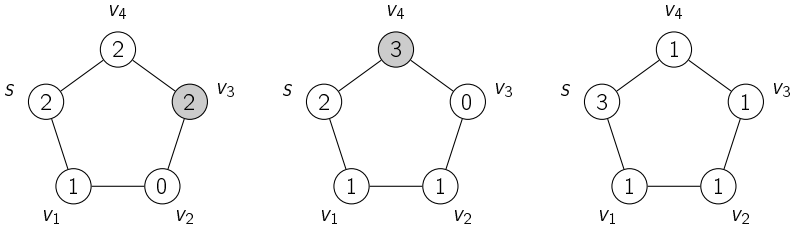


Figure 2.3. Toppling an unstable configuration of C_5 until obtain a stable configuration.

Now, let Δ_i be equal to the ith-row of the Laplacian matrix $L(G)$ for $i = 1, ..., n$. Thus, toppling v_i means subtract Δ_i to u.

Example 2.4. *The Laplacian matrix of the graph in figure 2.3 is*

$$L(C_5) = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}$$

First, we have the configuration $u = (1, 0, 2, 2, 2)$ subtracting $\Delta_3 = (0, -1, 2, -1, 0)$ we obtain the configuration $(1, 1, 0, 3, 2)$. Finally, we subtract $\Delta_4 = (0, 0, -1, 2, -1)$ and we get the configuration $s(u) = (1, 1, 1, 1, 3)$, that it is stable.

Remark 2.5. *Note in the example 2.4 that $|u| = \sum_{i=1}^{n} u_i$ and $|v| = \sum_{i=1}^{n} v_i$ are equal, it is because each $|\Delta_i| = 0$. This fact will be used in the proof of Proposition 2.6*

If all unstable vertices are toppled, a stable configuration is reached, the stable configuration associated to u will be denoted by $s(u)$. Then, $s(u) = u - L(G)^t b$ for a $b \in \mathbb{N}^n$. Furthermore, it always exists and is unique.

Proposition 2.6. *For any positive configuration u there exist a unique stable configuration v obtained by toppling the unstable vertices.*

Proof. Existence. Suppose $\{v_1, ..., v_n\}$ is the set of vertex of the graph G and v_n is the sink. Let $X_k = \{v \in V(G) \mid d_G(v_n, v) = k\}$, where $d_G(u, v)$ denotes the distance between u and v. Let d be the greatest distance to the sink v_n, to any configuration u we associate the $(d+1)$-tuple $\mu(u) = (\mu_0(u), \mu_1(u), ..., \mu_d(u))$ given by

$$\mu_i(u) = \sum_{v \in X_i} u_v.$$

We consider the following lexicographic order $\prec$ on these d-tuples:

$$\mu(u) \prec \mu(v) \Leftrightarrow \text{ exist } 0 \leq k \leq d \text{ such that } \mu_0(u) = \mu_0(v), ..., \mu_{k-1}(u) = \mu_{k-1}(v), \mu_k(u) < \mu_k(v).$$

Since $|\Delta_i| = 0$, then if $u, v \in \mathbb{N}^n$ with $u = v + L(G)^t b$ we have that $|u| = \sum_{i=0}^{d} \mu(u)_i$ is equal to $|v| = \sum_{i=0}^{d} \mu(v)_i$. Hence, there exist a finite number of $u \in \mathbb{N}^n$ with the same sum in coordinates. Thus, there exist a finite ascending chain for $\prec$ with the same sum in coordinates.

Uniqueness. It is consequence of the commutativity of the toppling operator; $(u - \Delta_i) - \Delta_j = (u - \Delta_j) - \Delta_i$. □

We will define the following operation in configurations:

Definition 2.7. *Let u, v configurations in $SP(G, s)$.*

$$u + v := (u_1 + v_1, ..., u_n + v_n).$$

Proposition 2.8. *If $s(u) = u'$ and $s(v) = v'$, then $s(u + v) = s(u' + v')$.*

Proof. Since $s(u) = u'$ and $s(v) = v'$, there exist $c_u, c_v \in \mathbb{N}^n$ such that $u' = u - c_u L(G)$ and $v' = v - c_v L(G)$ are stable configurations. Therefore, $u' + v' = u + v - (c_u + c_v)L(G)$. Since $u' + v'$ is not necessary an stable configuration, we have that $s(u + v) = s(u' + v')$. □

Since we can not topple the sink v_n, we will say that two configurations c and c' are the same if $c_v = c'_v$ for all $v \in \tilde{V}$.

Definition 2.9. *Two configurations c and c' are equal, if $c_v = c'_v$ for all $v \in \tilde{V}$.*

Then some times we will refer to c as if it becomes to $\mathbb{N}^{V(G)\setminus s}$. On the other hand, let $c \in \mathbb{N}^{V(G)\setminus s}$ a configuration, the *support* of c is the set $supp(c) = \{v \in \widetilde{V} \mid c_v \neq 0\}$.

1.2 Recurrent configurations

We will define some special configurations which have the property of being an abelian group.

Definition 2.10. *The configuration u is recurrent if there exist a non-zero configuration v such that $s(u+v) = u$.*

Example 2.11. *Consider the configuration $(1, 1, 1, 1)$ in C_5 with v_5 as sink. Here we omit the sink. Now, when have $(1, 1, 1, 1) + (1, 0, 0, 1) = (2, 1, 1, 2)$. Toppling v_1, the configuration $(0, 2, 1, 2)$ is obtained. Then we topple v_4, and we get $(0, 2, 2, 0)$. This time, we topple v_2 and we obtain $(1, 0, 3, 0)$. Finally, we topple v_3 to obtain $(1, 1, 1, 1)$. And thus $(1, 1, 1, 1)$ is a recurrent configuration in C_5 with v_5 as sink.*

Definition 2.12. *A configuration $\beta \in \mathbb{N}^{V(G)\setminus s} \geq \mathbf{1}$ of (G, s) is called a burning configuration if*

- *$\beta = \mathbf{z}^t L(G, s)$ for some $\mathbf{z} \in \mathbb{Z}^{V(G)\setminus s}$,*
- *for all $v \in V(G) \setminus s$, there exists a path to v from some vertex of* $\text{supp}(\beta)$.

Theorem 2.13. [**39**] *Let β be a burning configuration of (G, s), then a configuration $\mathbf{c} \in \mathbb{N}^{V(G)\setminus s}$ of (G, s) is recurrent if and only if*

$$s(\mathbf{c} + \beta) = \mathbf{c} \text{ with firing vector equal to } \beta^t L(G, s)^{-1}.$$

Theorem 2.14. [**39**] *There exist a unique burning configuration $\beta_{\min}$ such that*

$$\beta_{\min}^t L(G,s)^{-1} \leq \beta'^t L(G,s)^{-1} \text{ for all } \beta' \text{ a burning configuration.}$$

Moreover $\beta_{\min}^t L(G,s)^{-1} \geq \mathbf{1}$ with equality if and only if G has no a vertex $v \in V(G)\setminus s$ with $\deg_G^+(v) < \deg_G^-(v)$.

Definition 2.15. *Let $G = (V,E)$ be a multigraph with $s \in V$. The sandpile group of G is the set of recurrent configurations and is denoted by $SP(G,s)$.*

For instance, in the next proposition, we shall describe the sandpile group of the cycle. To avoid any confuse, in the next computations $L(G,v_n)_i$ will denote the ith-row of the reduced Laplacian $L(G,v_n)$.

Proposition 2.16. *The sandpile group of the cycle is composed of the following elements in $\mathbb{N}^{n-1}$:*

- *the vector of ones, and,*
- *the vectors of ones except for a unique zero.*

Proof. It is clear that the stable configurations are of the form $(x_1,...,x_{n-1})$ with $0 \leq x_i \leq 1$. We have that

$$\begin{array}{rcccccccc} L(G,v_n)_1 & = & (2, & -1, & 0, & \cdots & 0, & 0) \\ L(G,v_n)_2 & = & (-1, & 2, & -1, & \cdots & 0, & 0) \\ \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L(G,v_n)_{n-1} & = & (0, & 0, & 0, & \cdots & -1, & 2) \end{array}$$

Therefore, toppling the ith-vertex means subtract $L(G,v_n)_i$ to the configuration. Hence, Toppling every vertex means to subtract $b = (1,0,...,0,1)$ to the configuration. Then, if c is a configuration as in the sentence we have that $s(c+b) = c$. And finally we conclude that c is recurrent.

As we shall proof later, corollary 2.31, the number of elements in the sandpile group of the cycle is n. Then, we found the complete description of the elements of $SP(C_n,s)$. □

In the following, v_n will be the sink. Let us define $u \oplus v := s(u+v)$. So, we will see that sandpile group with the operation $\oplus$ is effectively a group.

Remark 2.17. *Let $\sigma_{max} = (d_G(v_1)) - 1,...,d_G(v_n) - 1)$, and $\delta = \sigma + 1$. Then, it is easy to see that*

i. *σ_{max} is stable.*
ii. *δ is unstable.*
iii. *$(\delta - s(\delta))_i > 0$ for all $1 \leq i \leq n-1$.*
iv. *$\delta - s(\delta) \in \langle \Delta_1, \Delta_2, ..., \Delta_n \rangle$*

Lemma 2.18. *Let $\epsilon = 2\sigma - 2s(\sigma)$. If u is recurrent, then $s(u+\epsilon) = u$.*

Proof. Let u recurrent. Hence, there exist a configuration v such that $s(u+v) = u$. Then, applying several times the remark 2.8, we have

$$\begin{array}{rcl} s(u+v+\epsilon) & = & s(u+v+2\sigma-2s(\sigma)) = s(u+v+2s(\sigma)-2s(\sigma)) \\ & = & s(u+v) = u. \end{array}$$

On the other hand,

$$s(u+v+\epsilon) \quad = \quad s(u+\epsilon).$$

□

Proposition 2.19. *For every configuration u there exist a unique recurrent configuration v such that $u - v \in \langle \Delta_1, ..., \Delta_{n-1} \rangle$.*

Proof. Existence. Let u a configuration. Since $(\delta - s(\delta))_i > 0$, we can find $k > 0$ such that $u + k(\delta - s(\delta)) > \sigma_{max}$. Now, let $v = s(u + k(\delta - s(\delta)))$. Hence, v is stable. And, since $u + k(\delta - s(\delta)) > \sigma_{max}$, there exist a configuration c such that $u + k(\delta - s(\delta)) > \sigma_{max} = v + c$. Then, v is recurrent.

Uniqueness. Let u and v two recurrent configurations such that $u - v \in \langle \Delta_1, ..., \Delta_{n-1} \rangle$. Then, $u - v = \sum_{i=1}^{n-1} c_1 \Delta_i$. Take $I = \{i \mid c_i < 0\}$ and $J = \{i \mid c_i \geq 0\}$, define

$$\beta = u + \sum_{i \in I} (-c_i)\Delta_i = v + \sum_{i \in J} c_i \Delta_i.$$

Let $k = \max_{v \in \tilde{V}} \{| \, c_v \mid d_v\}$, and $\tau = \beta + k\epsilon$. Hence

$$\tau = \beta + k\epsilon = u + \sum_{i \in I} (-c_i)\Delta_i + k\epsilon$$

Since $\tau_v \geq -c_v d_v$, we can topple $-c_v$ times each vertex of τ. Then,

$$s(\tau) = s(u + k\epsilon) = s(u + \epsilon) = u$$

On the other hand,

$$\tau = \beta + k\epsilon = v + \sum_{i \in J} (c_i)\Delta_i + k\epsilon$$

Since $\tau_v \geq c_v d_v$, we can topple c_v times each vertex of τ. Then,

$$s(\tau) = s(v + k\epsilon) = s(v + \epsilon) = v$$

Thus, by proposition 2.6, we have that $u = v$. □

Thus, each element in $\mathbb{Z}^{n-1} / \langle \Delta_1, ..., \Delta_{n-1} \rangle$ has associated a unique recurrent configuration. Since $\mathbb{Z}^{n-1} / \langle \Delta_1, ..., \Delta_{n-1} \rangle$ is an abelian group, it remains to proof that $SP(G, s)$ is closed under $\oplus$.

Theorem 2.20. *Let $G = (V, E)$ be a multigraph with $s \in V$. The sandpile group is a group.*

Proof. We will proof that $SP(G, s)$ is closed under $\oplus$. Let $u, v \in SP(G, s)$, then u, v are recurrent. Thus, there exist u' and v' such that $s(u + u') = u$ and $s(v + v') = v$. Now, $s(u + u' + v + v') = s(u + v)$ and $s(u + v + u' + v') = s(s(u + v) + u' + v')$. Hence, $s(u + v) = u \oplus v$ is recurrent. And, it turns out that $u \oplus v \in SP(G, s)$. □

In fact, the theorem 2.20 proves that $SP(G, s) \cong \mathbb{Z}^{n-1} / L(G, s)$, where $L(G, s)$ is the reduced Laplacian with respect to s. Thus, the Sandpile group is isomorphic to the Critical group $K(G)$ (Remark 1.48).

Another main result is that $SP(G, u) \cong SP(G, v)$ for $u \neq v \in V(G)$.

Theorem 2.21. *Let $G = (V, E)$ be a multigraph. The sandpile group of G is independent of the sink.*

Proof. Let P be the identity matrix such that the k-th row is interchanged to the n-th row. Since $P^{-1} = P$, we know that

$$L(G, v_n) = \begin{bmatrix} d_1 & -\mu_{1,2} & \cdots & \mu_{1,n-1} & -\mu_{1,n} \\ -\mu_{2,1} & d_2 & \cdots & \mu_{2,n-1} & -\mu_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\mu_{n-1,1} & -\mu_{n-1,2} & \cdots & d_{n-1} & -\mu_{n-1,n} \\ -\mu_{n,1} & -\mu_{n,2} & \cdots & -\mu_{n,n-1} & d_n \end{bmatrix}.$$

Now, we have the following product

$$P \cdot L(G, v_n) \cdot P = \begin{bmatrix} d_1 & -\mu_{1,2} & \cdots & \mu_{1,n} & \cdots & -\mu_{1,k} \\ -\mu_{2,1} & d_2 & \cdots & \mu_{2,n} & \cdots & -\mu_{2,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ -\mu_{n,1} & -\mu_{n,2} & \cdots & d_n & \cdots & -\mu_{n,k} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\mu_{k,1} & -\mu_{k,2} & \cdots & -\mu_{k,n-1} & \cdots & d_k \end{bmatrix} = L(G, v_k).$$

Thus, by Lemma 1.44 we have the result. □

2 The sandpile group of a graph with connectivity one

Proposition 2.22. *Let $G = (V, E)$ be a connected graph, u be a cut vertex and $B_1, B_2, ..., B_k$, with $k \geq 2$, be the connected components that remain after we remove u from G. Furthermore, let $G_1 = G[V(B_1) \cup u]$ and $G_2 = G[V(B_2) \cup \cdots \cup V(B_k) \cup u]$. Then*

$$K(G) = K(G_1) \oplus K(G_2).$$

Proof. Without loss of generality, we can suppose that $V(G_1) = \{v_1, ..., v_j = u\}$ and $V(G_2) = \{v_j = u, ..., v_n\}$. Since u is a cut vertex, there is no edges between v_a and v_b with $1 \leq a \leq j-1$, $j+1 \leq b \leq n$. Then the Laplacian matrix has the following form

$$L(G) = \begin{bmatrix} d_1 & -\mu_{1,2} & \cdots & \mu_{1,j-1} & -\mu_{1,j} & 0 & \cdots & 0 \\ -\mu_{2,1} & d_2 & \cdots & \mu_{2,j-1} & -\mu_{2,j} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -\mu_{j-1,1} & -\mu_{j-1,2} & \cdots & d_{j-1} & -\mu_{j-1,j} & 0 & \cdots & 0 \\ -\mu_{j,1} & -\mu_{j,2} & \cdots & -\mu_{j,j-1} & d_j & -\mu_{j,j+1} & \cdots & -\mu_{j,n} \\ 0 & 0 & \cdots & 0 & -\mu_{j+1,j} & d_{j+1} & \cdots & -\mu_{j+1,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & -\mu_{n,j} & -\mu_{n,j+1} & \cdots & d_n \end{bmatrix}$$

Taking $u = v_j$ as the sink, by Theorem 1.40 we have

$$K(G) = \mathbb{Z}^{n-1}/L(G\backslash u) \cong \mathbb{Z}^{j-1}/L(G_1\backslash u) \oplus \mathbb{Z}^{n-j}/L(G_2\backslash u) \cong K(G_1) \oplus K(G_2).$$

□

Applying inductively the Proposition 2.22 we have the following theorem:

Theorem 2.23. *Let G a multigraph and G_1, G_2, ..., G_l be its blocks, then*

$$K(G) = K(G_1) \oplus K(G_2) \oplus \cdots \oplus K(G_l).$$

A *thick tree* is a multigraph T such that its underlying graph T' is a tree.

Corollary 2.24. [**17**] *Let T be a thick tree, then*

$$K(G) = \bigoplus_{i=1}^{n-1} \mathbb{Z}_{m_i},$$

where m_i is the multiplicity of the edge $e_i \in E(T')$.

Proof. Since the blocks of a tree T are its edges, then the result it follows by the Theorem 2.23. □

3 The sandpile group of the thick cycle $\mathcal{C}_n$

Definition 2.25. *The thick cycle $\mathcal{C}_n$ is the graph that its underlying graph is a cycle.*

To simplify the notation, we will take $m_1 = \mu_{v_1,v_2}$, $m_2 = \mu_{v_2,v_3}$ and so on. Therefore, the Laplacian matrix of $\mathcal{C}_n$ is

$$L(\mathcal{C}_n) = \begin{bmatrix} m_1 + m_n & -m_1 & \cdots & 0 & -m_n \\ -m_1 & m_1 + m_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & m_{n-2} + m_{n-1} & -m_{n-1} \\ -m_n & 0 & \cdots & -m_{n-1} & m_{n-1} + m_n \end{bmatrix}$$

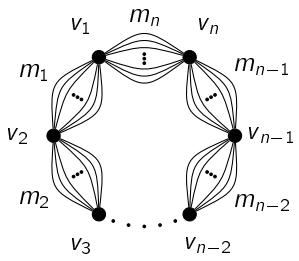


Figure 2.4. Thick cycle $\mathcal{C}_n$.

Taking v_n as the sink, the reduced Laplacian matrix has form

$$L(\mathcal{C}_n, v_n) = \begin{bmatrix} m_1+m_n & -m_1 & \cdots & 0 & 0 \\ -m_1 & m_1+m_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & m_{n-3}+m_{n-2} & -m_{n-2} \\ 0 & 0 & \cdots & -m_{n-2} & m_{n-2}+m_{n-1} \end{bmatrix}$$

Lemma 2.26. *The reduced Laplacian matrix $L(\mathcal{C}_n, v_n)$ is equivalent to the matrix*

$$R = \begin{bmatrix} -m_1 & 0 & \cdots & 0 & m_{n-1} \\ 0 & -m_2 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & -m_{n-2} & m_{n-1} \\ m_n & m_n & \cdots & m_n & m_{n-1}+m_n \end{bmatrix}$$

Proof. The matrix $L(\mathcal{C}_n, v_n)$ is equivalent to the matrix which the first row is in the last one.

$$\begin{bmatrix} -m_1 & m_1+m_2 & -m_2 & \cdots & 0 & 0 & 0 \\ 0 & -m_2 & m_2+m_3 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -m_3 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -m_{n-3} & m_{n-3}+m_{n-2} & -m_{n-2} \\ 0 & 0 & 0 & \cdots & 0 & -m_{n-2} & m_{n-2}+m_{n-1} \\ m_1+m_n & -m_1 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}$$

The last matrix is equivalent to the matrix that was added the first column to de second, the second to the third, and so on until the last column.

$$\begin{bmatrix} -m_1 & m_2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -m_2 & m_3 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -m_3 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -m_{n-3} & m_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & 0 & -m_{n-2} & m_{n-1} \\ m_1+m_n & m_n & m_n & \cdots & m_n & m_n & m_n \end{bmatrix}$$

Therefore, we add the $(n-1)$-th row to the $(n-2)$-th row, thereupon, add the $(n-2)$-th row to the $(n-3)$-th row, and we follow until we add the second to the first row.

$$\begin{bmatrix} -m_1 & 0 & 0 & \cdots & 0 & 0 & m_{n-1} \\ 0 & -m_2 & 0 & \cdots & 0 & 0 & m_{n-1} \\ 0 & 0 & -m_3 & \cdots & 0 & 0 & m_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -m_{n-3} & 0 & m_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & -m_{n-2} & m_{n-1} \\ m_1+m_n & m_n & m_n & \cdots & m_n & m_n & m_n \end{bmatrix}$$

Finally, we add the first row to the last one.

$$\begin{bmatrix} -m_1 & 0 & 0 & \cdots & 0 & 0 & m_{n-1} \\ 0 & -m_2 & 0 & \cdots & 0 & 0 & m_{n-1} \\ 0 & 0 & -m_3 & \cdots & 0 & 0 & m_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -m_{n-3} & 0 & m_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & -m_{n-2} & m_{n-1} \\ m_n & m_n & m_n & \cdots & m_n & m_n & m_{n-1}+m_n \end{bmatrix}$$

□

Lemma 2.27. *The determinant of the matrix R, defined in 2.26, is equal to* $(-1)^n \sum_{i=1}^{n} m_1 m_2 \cdots \hat{m_i} \cdots m_n$

Proof. The determinant of the matrix R is equal to

$$(-1)^n m_n \det \begin{bmatrix} 0 & 0 & \cdots & 0 & m_{n-1} \\ -m_2 & 0 & \cdots & 0 & m_{n-1} \\ 0 & -m_3 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & -m_{n-2} & m_{n-1} \end{bmatrix}$$

$$+\cdots+(-1)^{n+i-1} m_n \det \begin{bmatrix} -m_1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & -m_2 & \cdots & 0 & 0 & 0 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & \cdots & -m_{i-1} & 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & 0 & \cdots & 0 & -m_{i+1} & 0 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & -m_{n-2} & m_{n-1} \end{bmatrix} +\cdots+$$

$$+(-1)^{2n-2}(m_{n-1}+m_n) \det \begin{bmatrix} -m_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -m_2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -m_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -m_{n-3} & 0 \\ 0 & 0 & 0 & \cdots & 0 & -m_{n-2} \end{bmatrix}$$

On the other hand, we recall the following properties of the determinants:

(i) $\det \begin{bmatrix} x & \mathbf{a} \\ \mathbf{0} & \mathbf{B} \end{bmatrix} = x \det \begin{bmatrix} \mathbf{B} \end{bmatrix}$.

(ii) $\det \begin{bmatrix} 0 & \mathbf{a} \\ x & \mathbf{b} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} = -x \det \begin{bmatrix} \mathbf{a} \\ \mathbf{C} \end{bmatrix}$.

Thus, applying (i) and (ii) to the before equation, we obtain that it is equal to

$$(-1)^n m_n m_2 \cdots m_{n-3} \det\begin{bmatrix} 0 & m_{n-1} \\ -m_{n-2} & m_{n-1} \end{bmatrix} + (-1)^n m_n m_1 \det\begin{bmatrix} 0 & 0 & \cdots & 0 & m_{n-1} \\ -m_3 & 0 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & 0 & \cdots & -m_{n-2} & m_{n-1} \end{bmatrix}$$

$$+ \cdots + (-1)^{n+2i-2} m_n m_1 \cdots m_{i-1} \begin{bmatrix} 0 & 0 & \cdots & 0 & m_{n-1} \\ -m_{i+1} & 0 & \cdots & 0 & m_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & m_{n-1} \\ 0 & 0 & \cdots & -m_{n-2} & m_{n-1} \end{bmatrix} + \cdots +$$

$$(-1)^{n+2n-4}(m_{n-1} + m_n) m_1 \cdots m_{n-2}.$$

Finally, we get that the determinant of R is

$$(-1)^n m_2 \cdots m_n + (-1)^n m_1 m_3 \cdots m_n + \cdots + (-1)^n m_1 \cdots \hat{m}_i \cdots m_n + \cdots + (-1)^n m_1 \cdots m_{n-1}.$$

□

Theorem 2.28. *Let $\mathcal{C}_n$ be the thick cycle with m_i the multiplicity between v_i and v_{i+1}, then*

$$\Delta_i = \begin{cases} \gcd\left\{\prod_{1 \le j_1 < \cdots < j_i \le n} m_{j_1} \cdots m_{j_i}\right\} & \text{if } 1 \le i \le n-1, \\ (-1)^n \sum_{i=1}^{n} m_1 m_2 \cdots \hat{m}_i \cdots m_n & \text{if } i = n. \end{cases}$$

Proof. Since Δ_n that is equal to the determinant of the the matrix R, it remains to compute the Δ_i of the matrix R when $1 \le i \le n-1$. Note that the $i \times i$ minor is a linear combinations of multiplications of i different multiplicities. Due to $\gcd(a_1 x_1 + \cdots + a_n x_n, x_1, ..., x_n) = \gcd(x_1, ..., x_n)$, we just need to check that each multiplications of i different multiplicities is a $i \times i$ minor. In this sense, it is clear that there exist a $i \times i$ minor that it is equal to a multiplication of the multiplicities m_i, $0 \le i \le n-2$. If the multiplication has m_{n-1} and has not m_n, then the minor, equal to it, is which is obtained from the rows, that contain the multiplicities in the multiplication an another different of the last row, and the columns that contain the multiplicities and the last column. Analogously, if the multiplication has m_n and has not m_{n-1}, then the minor, equal to it, is which is obtained from the rows, that contain the multiplicities in the multiplication and the last row, and the columns that contain the multiplicities and another that different to the last column. If the multiplication has both m_{n-1} and m_n, the minor is which is obtained from the rows and columns that contain the multiplicities with index minor or equal ti $n-2$ in the multiplication and the last row and the last column. Then we use $\gcd(a_1 x_1 + \cdots + a_n x_n, x_1, ..., x_n) = \gcd(x_1, ..., x_n)$ to obtain the equality. □

Thus, by proposition 1.45, theorem 1.52 and theorem 2.28 we have that

Theorem 2.29. *Let $\mathcal{C}_n$ be the thick cycle with m_i the multiplicity between v_i and v_{i+1}, then*

$$K(\mathcal{C}_n) = \mathbb{Z}_{\Delta_1} \oplus \left(\bigoplus_{i=2}^{n} \mathbb{Z}_{\Delta_i/\Delta_{i-1}}\right).$$

Now, we will show some special cases.

Corollary 2.30. *Let $\mathcal{C}_n$ be the thick cycle with $m_i = m$, the multiplicity between v_i and v_{i+1}, then*

$$K(\mathcal{C}_n) = \left(\bigoplus_{i=1}^{n-1} \mathbb{Z}_m\right) \oplus \mathbb{Z}_n.$$

Proof. By theorem 2.28 we have that

- $\Delta_1 = m$
- $\Delta_i/\Delta_{i-1} = m^i/m^{i-1} = m$ for $2 \le i \le n-1$.

and since $(-1)^n \sum_{i=1}^{n} m^{n-1} = (-1)^n n m^{n-1}$, then except for a product by -1, we have $\Delta_n/\Delta_{n-1} = nm^{n-1}/m^{n-1} = n$. Thus, by theorem 2.29

$$K(\mathcal{C}_n) = \left(\bigoplus_{i=1}^{n-1} \mathbb{Z}_m\right) \oplus \mathbb{Z}_n.$$

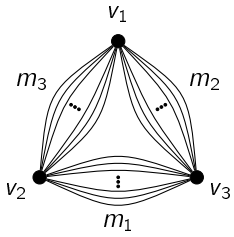
□

An special case of the corollary 2.30 is when $m = 1$, then it follows trivially the following corollary:

Corollary 2.31. *The sandpile group of the cycle of n vertices is* $\mathbb{Z}_n$.

3.1 The generators of the sandpile group of thick $\mathcal{C}_3$

Proposition 2.32. *Let* $m_1, m_2, m_3 \in \mathbb{Z}_+$, $\mathcal{C}_3(m_1, m_2, m_3)$ *be the multigraph with* $V = \{v_1, v_2, v_3\}$ *as vertex set,* $\mu_{v_1,v_2} = m_3$, $\mu_{v_1,v_3} = m_2$, *and* $\mu_{v_2,v_3} = m_1$.



Then $SP(\mathcal{C}_3(m_1, m_2, m_3), v_1) = C_{v_2} \cup C_{v_3}$, *where*

$$C_{v_2} = \{(a, b) \mid m_1 \leq a \leq m_1 + m_3 - 1 \text{ and } 0 \leq b \leq m_1 + m_2 - 1\}$$

and

$$C_{v_3} = \{(a, b) \mid 0 \leq a \leq m_1 + m_3 - 1 \text{ and } m_1 \leq b \leq m_1 + m_2 - 1\}.$$

Proof. Let $C = (a, b) \in C_{v_2} \cup C_{v_3}$, clearly C is a stable configuration and $C' = C + (m_3, m_2)$ is an unstable configuration. Toppling both vertices on the configuration C', we have $s(C') = C' - (m_3, m_2) = C$. Therefore, C is a recurrent configuration and $C_{v_2} \cup C_{v_3} \subseteq SP(\mathcal{C}(m_1, m_2, m_3), v_1)$.

On the other hand,

$$\begin{aligned} |C_{v_2} \cup C_{v_3}| &= |C_{v_2}| + |C_{v_3}| - |C_{v_2} \cap C_{v_3}| \\ &= |C_{v_2}| + |C_{v_3}| - |\{(c, d) \mid m_1 \leq c \leq m_1 + m_3 - 1 \text{ and } m_1 \leq d \leq m_1 + m_2 - 1\}| \\ &= m_3(m_2 + m_1) + (m_1 + m_3)m_2 - m_3 m_2 = m_2 m_3 + m_1 m_3 + m_1 m_2. \end{aligned}$$

By theorem [**6**, Chapter 6] or theorem [**26**, 13.2.1], the number of recurrent configurations of a graph G is $\det(L(G, s))$. Since $|C_{v_2} \cup C_{v_3}|$ is equal to $\det(L(\mathcal{C}(m_1, m_2, m_3), v_1))$, then $C_{v_2} \cup C_{v_3}$ is the set of recurrent configurations of $\mathcal{C}(m_1, m_2, m_3)$. □

Theorem 2.33. *Let* $\alpha \in \mathbb{Z}_+$, *if* $\gcd(m_1, m_2) = 1$, $\gcd(m_2, m_3) = 1$ *or* $\gcd(m_3, m_1) = 1$ *then*

$$K(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3)) = \mathbb{Z}_\alpha \oplus \mathbb{Z}_{\alpha(m_1 m_2 + m_2 m_3 + m_3 m_1)}$$

and $e = \left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right)$ *is the identity of* $SP(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3), v_1)$.

Proof. If $\gcd(m_1, m_2) = 1$, $\gcd(m_2, m_3) = 1$ or $\gcd(m_3, m_1) = 1$, then $\gcd(m_1, m_2, m_3) = 1$. Therefore $\Delta_1 = \alpha$. And by theorem 2.28, $\Delta_2 = \alpha(m_1 m_2 + m_2 m_3 + m_3 m_1)$. Finally, by theorem 2.29,

$$K(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3)) = \mathbb{Z}_\alpha \oplus \mathbb{Z}_{\alpha(m_1 m_2 + m_2 m_3 + m_3 m_1)}.$$

On the other hand, since $\alpha m_1 \leq \left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3 \leq \left\lfloor \frac{m_1 + m_3}{m_3} \right\rfloor \alpha m_3 \leq \alpha(m_1 + m_3) - 1$ and $\alpha m_1 \leq \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2 \leq \left\lfloor \frac{m_1 + m_2}{m_2} \right\rfloor \alpha m_2 \leq \alpha(m_1 + m_2) - 1$, then by proposition 2.32 we have that $\left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right)$ is a recurrent configuration. Now, since

$$s\left(2\left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right)\right) = \left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right),$$

then $\left(\left\lceil \frac{m_1}{m_3} \right\rceil \alpha m_3, \left\lceil \frac{m_1}{m_2} \right\rceil \alpha m_2\right)$ is the identity of $SP(\mathcal{C}_3(\alpha m_1, \alpha m_2, \alpha m_3), v_1)$.

□

Theorem 2.34. *Let $m \in \mathbb{Z}_+$, then $(m,0)$ is a generator of $SP(\mathcal{C}_3(m,1,1), v_1)$ and*

$$k(m,0) = \begin{cases} (m-j, m) & \text{if } k = 2j \leq 2m, \\ (m, j) & \text{if } k = 2j+1 \leq 2m+1, \end{cases}$$

in $SP(\mathcal{C}_3(m,1,1), v_1)$.

Proof. We will use induction on k, for $k = 1$ the result is trivial. We will assume that the result is true for all $k \leq i$ and we will prove the formula for $i = k+1$. We need to consider to cases:

Case ($k+1 = 2j$):
Using induction hypothesis, $2j(m,0) = (2j-1)(m,0)+(m,0) = (m, j-1)+(m,0) = (2m, j-1)$. Toppling both vertices $j-1$ times on configuration $(2m, j-1)$ we get $2j(m,0) = (2m-j+1, 0)$ and toppling the vertex v_2 on configuration $(2m-j+1, 0)$ we get $2j(m,0) = (m-j, m)$.

Case ($k+1 = 2j+1$):
Using induction hypothesis, $(2j+1)(m,0) = 2j(m,0)+(m,0) = (m-j, m)+(m,0) = (2m-j, m)$. Toppling both vertices $m-j$ times on configuration $(2m-j, m)$ we get $(2j+1)(m,0) = (m, j)$.

Since $k(m,0) \neq (m,m)$ for all $1 \leq k \leq 2m$, then $(m,0)$ is a generator of $SP(\mathcal{K}_3(m,1,1), v_1)$. □

Theorem 2.35. *Let $m \in \mathbb{Z}_+$ such that $\gcd(m,2) = 1$, then $(m+1, m)$ is a generator of $SP(\mathcal{C}_3(m,2,2), v_1)$ and*

$$k(m+1,m) = \begin{cases} (m+1, m-k+1) & \text{if } 1 \leq k \leq m+1, \\ (k-m-1-\frac{1+(-1)^k}{2}, m+\frac{1+(-1)^{k+1}}{2}) & \text{if } m+2 \leq k \leq 2m+2, \\ (m, 3m+2-k) & \text{if } 2m+3 \leq k \leq 3m+2, \\ (k-3(m+1)-\frac{1+(-1)^{k+1}}{2}, m+1-\frac{1+(-1)^{k+1}}{2}) & \text{if } 3m+3 \leq k \leq 4m+4. \end{cases}$$

Proof. We will use induction to proof that $(m+1, m)$ is a generator. For $k = 1$ the result is trivial. Thus we have the following cases:

Case ($1 \leq k \leq m+1$):
Using induction hypothesis, we have $k(m+1,m) = (m+1, m-k+2)+(m+1,m) = (2m+2, 2m-k+2)$, $s(k(m+1,m)) = (m+1, m-k+1)$ by toppling both vertices $\frac{m+1}{2}$ times.

Case ($k = m+2$):
Using induction hypothesis, we have $(m+2)(m+1,m) = (m+1,0)+(m+1,m) = (2m+2, m)$, we obtain $(m+3, 1)$ by toppling both vertices $\frac{m-1}{2}$ times, and then $s((m+2)(m+1,m)) = (1, m+1)$ by toppling v_2,

Case ($m+3 \leq k \leq 2m+2$):
Using induction hypothesis, we have $k(m+1,m) = (k-1-m-1-\frac{1+(-1)^{k-1}}{2}, m+\frac{1+(-1)^k}{2})+(m+1,m) = (k-1-\frac{1+(-1)^{k-1}}{2}, 2m+\frac{1+(-1)^k}{2})$, $s(k(m+1,m)) = (k-m-1-\frac{1+(-1)^k}{2}, m+\frac{1+(-1)^{k+1}}{2})$ by toppling both vertices $\frac{m+(-1)^k}{2}$ times.

Case ($2m+3 \leq k \leq 3m+2$):
Using induction hypothesis, we have $k(m+1,m) = (m, 3m-k+3)+(m+1,m) = (2m+1, 4m-k+3)$, $s(k(m+1,m)) = (m, 3m-k+2)$ by toppling both vertices $\frac{m+1}{2}$ times.

Case ($k = 3m+3$):
Using induction hypothesis, we have $(3m+3)(m+1,m) = (m,0)+(m+1,m) = (2m+1, m)$, we have $(m+2, 1)$ by toppling both vertices $\frac{m-1}{2}$ times, then $s((3m+3)(m+1,m)) = (0, m+1)$ by toppling v_2.

Case ($3m+4 \leq k \leq 4m+4$):
Using induction hypothesis, we have $i(m+1,m) = (k-1-3(m-1)-\frac{1+(-1)^k}{2}, m+1-\frac{1+(-1)^k}{2})+(m+1,m) = (k-1-2(m+1)-\frac{1+(-1)^k}{2}, 2m+1-\frac{1+(-1)^k}{2})$, $s(k(m+1,m)) = (k-3(m-1)-\frac{1+(-1)^{k+1}}{2}, m+1-\frac{1+(-1)^{k+1}}{2})$ by toppling both vertices $\frac{m+(-1)^{k+1}}{2}$ times.

Since $k(m+1,m) \neq (m,m)$ for all $1 \leq k \leq 4m+4$, then $(m+1,m)$ is a generator of $SP(\mathcal{K}_3(m,2,2),v_1)$. □

4 Integer linear programming and the sandpile group

In this section we will see how to use integer linear programming in order to compute the recurrent configuration of a stable configuration, the identity of the sand pile group of a graph and the degree of a recurrent configuration.

Let G be a multigraph, $V(G)=\{v_1,\ldots,v_n,s\}$ and $\mathbf{d}_{(G,s)}=(d_G(v_1),...,d_G(v_n))$.

Theorem 2.36. *Let G be a multigraph, $s \in V(G)$, $\mathbf{0} \leq \mathbf{c} \leq \mathbf{d}_{(G,s)} - \mathbf{1}$ a stable configuration of (G,s) and $\mathbf{x}^*$ an optimal solution of the following integer linear problem:*

$$\begin{array}{ll} \text{maximize} & |\mathbf{x}| \\ \text{subject to} & \mathbf{0} \leq L(G,s)^t\mathbf{x}+\mathbf{c} \leq \mathbf{d}_{(G,s)}-\mathbf{1} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{4}$$

then $L(G,s)^t\mathbf{x}^*+\mathbf{c} \in SP(G,s)$ and $[\mathbf{c}]=[L(G,s)^t\mathbf{x}^*+\mathbf{c}]$.

Proof. Let x^* be an optimal solution of the integer linear program (1). Clearly, $\mathbf{r}=L(G,s)^t\mathbf{x}^*+\mathbf{c}$ is an stable configuration of (G,s) and $[\mathbf{c}]=[\mathbf{r}]$ in $K(G)$. Therefore, only remains to prove that $\mathbf{r}$ is a recurrent configuration of (G,s). Let $\beta_{\min}$ the burning configuration as in Theorem 2.14, then by Theorem 2.13 $\mathbf{r}$ is a recurrent configuration of (G,s) if and only if $s(\mathbf{r}+\beta_{\min})=\mathbf{r}$ with firing vector equal to $\beta_{\min}L(G,s)^{-1}$. If we assume that $\mathbf{r}$ is not a recurrent configuration of (G,s), then there exist $\mathbf{b} \in \mathbb{N}^{V(G)\setminus s}$ such that $\mathbf{0} \leq \mathbf{b} < \beta_{\min}L(G,s)^{-1}$ and $\mathbf{r}+\beta_{\min}-\mathbf{b}^tL(G,s)$ is an stable configuration of (G,s). Since $\mathbf{r}+\beta_{\min}-\mathbf{b}^tL(G,s)=\mathbf{c}+(\mathbf{x}^*+\mathbf{z}-\mathbf{b})^tL(G,s)$, where $\mathbf{z}=\beta_{\min}L(G,s)^{-1}$ and $\mathbf{z}-\mathbf{b}>\mathbf{0}$, then $\mathbf{x}'=\mathbf{x}^*+\mathbf{z}-\mathbf{b}$ is a feasible solution of integer linear program (1) with $\mathbf{x}^*<\mathbf{x}'$; a contradiction to the optimality of $\mathbf{x}^*$. □

Example 2.37. *Let $c=(0,0,1,0)$ a configuration in (C_5,v_5). The corresponding integer linear program is:*

$$\begin{array}{ll} \text{maximize} & x_1+x_2+x_3+x_4 \\ \text{subject to} & \\ & \begin{bmatrix} 0\\0\\0\\0 \end{bmatrix} \leq \begin{bmatrix} 2 & -1 & 0 & 0\\ -1 & 2 & -1 & 0\\ 0 & -1 & 2 & -1\\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1\\x_2\\x_3\\x_4 \end{bmatrix} + \begin{bmatrix} 0\\0\\1\\0 \end{bmatrix} \leq \begin{bmatrix} 1\\1\\1\\1 \end{bmatrix} \\ & \mathbf{x} \geq \mathbf{0} \end{array}$$

We use the following code in Maple:

```
with(Optimization):
LPSolve( x[1]+x[2]+x[3]+x[4],
{
-x[3]+2*x[4] <= 1,
-x[1]+2*x[2]-x[3] <= 1,
-x[2]+2*x[3]-x[4]+1 <= 1,
2*x[1]-x[2] <= 1
},
assume = {integer, nonnegative}, maximize)
```

and we obtain $\mathbf{x}^*=(1,1,1,1)$. And, $L(G,s)^t\mathbf{x}^*+\mathbf{c}=(1,0,1,1)$ is a recurrent configuration.

Corollary 2.38. *Let G be a multigraph, $s \in V(G)$ and $\mathbf{x}^*$ an optimal solution of the following integer linear problem:*

$$\begin{array}{ll} \text{maximize} & |\mathbf{x}| \\ \text{subject to} & \mathbf{0} \leq L(G,s)^t\mathbf{x} \leq \mathbf{d}_{(G,s)}-\mathbf{1} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{5}$$

then $L(G,s)^t\mathbf{x}^* \in SP(G,s)$ is the identity of $K(G)$.

Proof. It follows directly from Theorem 2.36 taking $\mathbf{c} = \mathbf{0}$. □

Example 2.39. *We shall compute the identity configuration of* $SP(\mathcal{C}_3(3,1,1), v_1)$. *The corresponding integer linear program is:*

$$\begin{array}{ll} \text{maximize} & x_1 + x_2 \\ \text{subject to} & \\ & \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} 4 & -3 \\ -3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 3 \\ 3 \end{bmatrix} \\ & \mathbf{x} \geq \mathbf{0} \end{array}$$

We use the following code in Maple:

```
with(Optimization):
LPSolve( x[1]+x[2],
{
4*x[1]-3*x[2] <= 3,
-3*x[1]+4*x[2] <= 3,
},
assume = {integer, nonnegative}, maximize)
```

and we obtain $\mathbf{x}^* = (3,3)$. *And,* $L(G,s)^t\mathbf{x}^* = (3,3)$ *is the identity.*

Corollary 2.40. *Let* G *be a connected,* r*-regular multigraph, then* $r\mathbf{1}$ *is the identity of the sandpile group,* $SP(c(G), s)$.

Proof. The dual linear problem of (2) is given by:

$$\begin{array}{ll} \text{minimize} & \mathbf{y}^t(\mathbf{d}_G, \mathbf{0}) \\ \text{subject to} & \big(L(c(G),s), -L(c(G),s)^t\big)\mathbf{y} \geq \mathbf{1} \\ & \mathbf{y} \geq \mathbf{0}. \end{array}$$

Since G is a r-regular multigraph, then the vectors $\mathbf{x}^t = r\mathbf{1}$ and $\mathbf{y}^t = (\mathbf{1}, \mathbf{0})$ are feasible integral solutions of the primal and the dual linear problems, respectively, with cost equal to $r\mathbf{1} \cdot \mathbf{1} = r|V(G)| = \mathbf{1} \cdot \mathbf{d}_G$. By the weak duality theorem [**5**, Corollary 4.2], $\mathbf{x}$ is an optimal solution of the integer linear problem (2). Therefore, by Corollary 2.38, $r\mathbf{1} = rL(G,s)^t\mathbf{1} = L(G,s)^t r\mathbf{1}$ is is the identity of the sandpile group of $(c(G), s)$. □

Corollary 2.41. *Let* G *be a multigraph,* $s \in V(G)$, $\mathbf{c}$ *a recurrent configuration of* $SP(G,s)$, *and* $(d, \mathbf{x})^*$ *an optimal solution of the following integer linear problem:*

$$\begin{array}{ll} \text{minimize} & d \\ \text{subject to} & d\mathbf{c} - L(G,s)^t\mathbf{x} = \mathbf{0} \\ & d \geq 1, \mathbf{x} \geq \mathbf{0}, \end{array} \tag{6}$$

then d *is the degree of* $\mathbf{c}$ *in* $K(G)$.

Proof. Since $\mathbf{c}$ is recurrent, then $\mathbf{c} = [\mathbf{c}]$ and therefore $\deg_{K(G)}([\mathbf{c}]) = \min\{d \mid d[\mathbf{c}] = [\mathbf{0}]\}$. Thus

$$\deg_{K(G)}([\mathbf{c}]) = \min\{d \mid d[\mathbf{c}] = [d\mathbf{c}] = [L(G,s)^t\mathbf{x}] = [\mathbf{0}]\} = \min\{d \mid d\mathbf{c} - L(G,s)^t\mathbf{x} = \mathbf{0}, d \geq 1, \mathbf{x} \geq \mathbf{0}\}.$$

□

Example 2.42. *We shall compute the degree of the configuration* $(1,0,1,1)$ *of* $SP(C_5, v_5)$. *The corresponding integer linear program is:*

$$\begin{aligned} &\text{minimize} && d \\ &\text{subject to} && \\ & && \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} d \\ 0 \\ d \\ d \end{bmatrix} \\ & && \mathbf{x} \geq \mathbf{0} \\ & && d \geq 0 \end{aligned}$$

We use the following code in Maple:

```
with(Optimization):
LPSolve( d,
{
2*x[1]-x[2] = d,
-x[1]+2*x[2]-x[3] = 0,
-x[2]+2*x[3]-x[4] = d,
-x[3]+2*x[4] = d,
d >= 1,
},
assume = {integer, nonnegative})
```

and we obtain that $\mathbf{x}^* = (7,9,11,8)$ *and the degree* $d = 5$. *Actually,* $(1,0,1,1)$ *is a generator of* $SP(C_5, v_5)$.

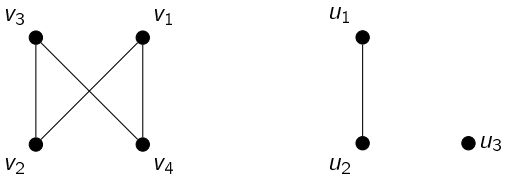
5 Graph Homomorphism and the sandpile group

Let G and H be multigraphs. A *full homomorphism* of G to H, denoted by $f : G \to H$, is a mapping

$$f : V(G) \to V(H)$$

such that $f(u)f(v) \in E(H)$ if and only if whenever $uv \in E(G)$.

The definitions of full homomorphism and isomorphism are similar, however, the difference is that an isomorphism is bijective. For example, let C_4 and P_3 be graphs as in figure 2.5. The map $f : V(C_4) \to V(P_3)$ such that $v_1, v_3 \mapsto u_1$ and $v_2, v_4 \mapsto u_2$ is a full homomorphism.



(a) C_4 (b) P_3

Figure 2.5. Full homomorphism.

The following proposition give us an equivalent way to define a full homomorphism of graphs:

Proposition 2.43. **[27]** *Let* G *and* H *be multigraphs without loops. Then* $f : G \to H$ *is an (full) homomorphism if and only if*

i. $S_x = f^{-1}(x)$ *is an independent set of* G *for all* $x \in V(H)$,
ii. if $xy \notin (\in)\, E(H)$, *then* $uv \notin (\in)\, E(G)$ *for all* $u \in S_x$ *and* $v \in S_y$.

Now, we will introduce the new concept of uniform homomorphism, this concept is between the concepts of homomorphism and full homomorphism.

Definition 2.44. *Let G and H be multigraphs without loops. An uniform homomorphism of G to H, denoted by $f : G \to H$, is a mapping $f : V(G) \to V(H)$ such that for all $x, y \in V(H)$ we have that*

$$d_{G[S_x \cup S_y]}(u) = m_{x,y} \text{ for all } u \in S_x \cup S_y,$$

where $S_x = f^{-1}(x)$ and $m_{x,y} = d_{H[x,y]}(x) = d_{H[x,y]}(y)$.

Example 2.45. *Consider the graphs C_5 and C_3, with $V(C_5) = \{v_1, v_2, v_3, v_4, v_5\}$, $E(C_5) = \{v_1v_2, v_2v_3, v_3v_4, v_4v_5, v_5v_1\}$, $V(C_3) = \{u_1, u_2, u_3\}$ and $E(C_5) = \{u_1u_2, u_2u_3, u_3u_1\}$. And, let $f : V(C_5) \to V(C_3)$ such that*

$$v_1 \mapsto u_1, v_2 \mapsto u_2, v_3 \mapsto u_3, v_4 \mapsto u_2, v_5 \mapsto u_3,$$

It is clear that it is an homomorphism. but it is not a full or uniform homomorphism. However, taking $C_5 + v_2v_5$

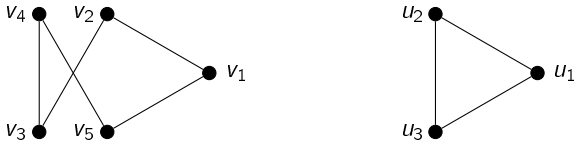


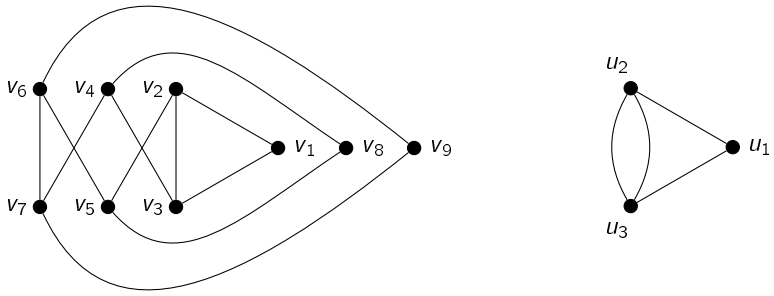
Figure 2.6. Homomorphism between C_5 and C_3.

with the same mapping, we obtain a full homomorphism that is not uniform. Finally, considering the same mapping from $C_5 + v_2v_5$ to $C_3 + v_2v_3$, we get a full and uniform homomorphism.

In the following example, we shall show an example of a uniform homomorphism that is not full.

Example 2.46. *Let G and H be graphs as in figure 2.7, with $V(G) = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$ and $V(H) = \{u_1, u_2, u_3\}$. Moreover, let $f : V(G) \to V(H)$ be the mapping given by*

$$v_1 \mapsto u_1, v_2 \mapsto u_2, v_3 \mapsto u_3, v_4 \mapsto u_2, v_5 \mapsto u_3, v_6 \mapsto u_2, v_7 \mapsto u_3.v_8 \mapsto u_1, v_9 \mapsto u_1,$$



(a) G *(b) H*

Figure 2.7. Uniform homomorphism.

Then f is a uniform homomorphism. Furthermore, we have that it is not a full homomorphism, because v_2v_5 is not an edge of C_5 as required by theorem 2.43.ii.

Proposition 2.47. [**27**] *Let G and H be multigraphs without loops. Then $f : G \to H$ is an uniform homomorphism if and only if*

- *$S_x = f^{-1}(x)$ is an independent set of G for all $x \in V(H)$,*

- $G[S_x \cup S_y]$ *is a* $m_{x,y}$*-regular bipartite graph.*

Proof. It follows directly from the definition of uniform homomorphism. □

The concept of uniform homomorphism of graphs is relevant in the study of the sandpile group of graphs as shows the following result:

Theorem 2.48. *Let G and H be multigraphs and $f : G \to H$ be a surjective uniform homomorphism, then the induced mapping $\widetilde{f} : SP(c(H)) \to SP(c(G))$ given by*

$$\widetilde{f}(u)_v = u_x \in \mathbb{N}^{V(G)} \text{ for all } v \in S_x = f^{-1}(x),$$

is an injective homomorphism of groups, that is, $K(c(H)) \triangleleft K(c(G))$.

Proof. It is clear that f induces an injective homomorphism from $\widehat{f} : \mathbb{Z}^{|V(c(H))|} \to \mathbb{Z}^{|V(c(G))|}$. If this homomorphism send recurrent configuration of $SP(c(H))$ to recurrent configuration of $SP(c(G))$, and if $\widetilde{f}(1_{SP(c(H))}) = 1_{SP(c(G))}$, then $\widehat{f}$ induces

$$\widetilde{f} : SP(c(H)) \to SP(c(G))$$

In order to see this, we shall show the following lemma:

Lemma 2.49. *Let c_1, c_2 be two configurations of $SP(H, s)$ such that $c_2 = c_1 - bL(H, s)$ with $b \in \mathbb{N}^{|SP(c(H),s)|}$, that is, the classes are the same. And, let $\widetilde{f}$ as in theorem 2.48. If $\widetilde{c_1}, \widetilde{c_2} \in SP(G, s)$ are the induced configurations of c and c_2, respectively, then $\widetilde{c_2} = \widetilde{c_1} - \widetilde{b}L(G, s)$ with $\widetilde{f}(b) = \widetilde{b}$.*

Proof. It is clear that vertex $v \in H$ is toppled if and only if vertices $S_v \in G$ are toppled. And $G[S_x \cup S_y]$ is a $m_{x,y}$-regular bipartite graph, then when the vertices S_x are toppled, each vertex in S_x loss $m_{x,y}$, and each vertex in S_y gains $m_{x,y}$. Thus, considering all neighbors S_y of S_x, we have the result. □

Then, if c is recurrent, then there exist u such that $s(c + u) = c$, and by the lemma we have that $s(\widetilde{c} + \widetilde{u}) = \widetilde{c}$. Proving that $\widetilde{c}$ is recurrent. Similarly, we have that $s(1_{SP(c(H))} + 1_{SP(c(H))}) = 1_{SP(c(H))}$ implies $s(1_{SP(c(G))} + 1_{SP(c(G))}) = 1_{SP(c(G))}$. □

Example 2.50. *Consider C_4 with $V(C_4) = \{v_1, v_2, v_3, v_4\}$, $E(C_4) = \{v_1v_2, v_2v_3, v_3v_4, v_4v_1\}$ and $\mathcal{K}_2(2)$ with $V(\mathcal{K}_2(2)) = \{x_1, x_2\}$, $E(\mathcal{K}_2(2)) = \{x_1x_2, x_1x_2\}$. The mapping $f : V(C_4) \to V(\mathcal{K}_2)$ given by*

$$f(v_1) = x_1, f(v_2) = x_2, f(v_3) = x_1, f(v_4) = x_2,$$

is an uniform homomorphism and the induced map $\widetilde{f} : SP(c(C_4)) \to SP(c(\mathcal{K}_2(2)))$ is given by

$$\widetilde{f}((2,0)) = (2,0,2,0), \widetilde{f}((1,2)) = (1,2,1,2), \widetilde{f}((2,1)) = (2,1,2,1),$$
$$\widetilde{f}((0,2)) = (0,2,0,2), \text{ and } \widetilde{f}((2,2)) = (2,2,2,2).$$

Corollary 2.51. *Let G be a r-regular bipartite graph with bipartition (V_1, V_2), then*

$$\mathbb{Z}_{2r+1} \triangleleft K(c(G)).$$

Proof. Let $\mathcal{K}_2(r)$ be the multigraph with $V = \{v_1, v_2\}$ as set of vertices and $m_{1,2} = r$. Since G is a bipartite r-regular graph, then the mapping $f : G \to \mathcal{K}_2(r)$, given by

$$f(v) = \begin{cases} v_1 & \text{if } v \in V_1 \\ v_2 & \text{if } v \in V_2 \end{cases}$$

is a surjective uniform homeomorphism. Therefore, by theorems 2.48 and 2.33 we have that

$$\mathbb{Z}_{2r+1} = K(c(\mathcal{K}_2(r))) \triangleleft K(c(G)).$$

□

6 The sandpile group of the cartesian product of graphs

Let G and H be multigraphs, $G\Box H$ be the cartesian product of G and H and let $i_G : G \to G\Box H$ the canonical inclusion map given by

$$i_G(v) = (v, u) \text{ for all } v \in V(G),$$

for some fixed vertex u of H. Let $i_H : H \to G\Box H$ the inclusion map defined in a similar way.

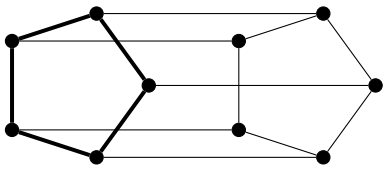


Figure 2.8. Cartesian product $C_5\Box K_2$ and the image of i_{C_5}.

Thus, is not difficult to see that the mappings i_G and i_H are injective homomorphism of graphs, that is, the image of i_G is a subgraph of $G\Box H$ isomorphic to G. For the rest of this section let $s_G \in V(c(G)) \setminus V(G)$, $s_H \in V(c(H)) \setminus V(H)$ and $s_{G\Box H} \in V(c(G\Box H)) \setminus V(G\Box H)$.

Now, let $\mathbf{a} \in \mathbb{N}^{V(G)}$ be a configuration of $c(G)$, $\mathbf{b} \in \mathbb{N}^{V(H)}$ be configurations of $c(H)$ and let $\mathbf{a}\Box\mathbf{b} \in \mathbb{N}^{V(G\Box H)}$, the configuration of $c(G\Box H)$ given by

$$(\mathbf{a}\Box\mathbf{b})_{(u,v)} = \mathbf{a}_u + \mathbf{b}_v \text{ for all } u \in V(G) \text{ and } v \in V(H).$$

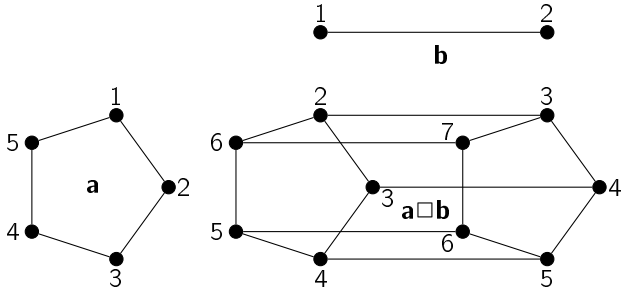


Figure 2.9. Cartesian product of configurations.

As shows the following lemma the cartesian product of configurations of $c(G)$ and $c(H)$ is compatible with the toppling operators of $c(G)$, $c(H)$ and $c(G\Box H)$.

Lemma 2.52. *Let G and H be multigraphs, $\mathbf{a} \in \mathbb{N}^{V(G)}$ be a configuration of $c(G)$ and $\mathbf{b} \in \mathbb{N}^{V(H)}$ be configurations of $c(H)$. Then*

(i): If $\mathbf{a}$ and $\mathbf{b}$ are stable configurations, then $\mathbf{a}\Box\mathbf{b}$ is a stable configuration of $c(G\Box H)$,

(ii): If $\mathbf{a}$ and $\mathbf{b}$ are recurrent configurations, then $\mathbf{a}\Box\mathbf{b}$ is a recurrent configuration of $c(G\Box H)$.

Proof. (i) If $\mathbf{a}$ and $\mathbf{b}$ are stable configurations of $c(G)$ and $c(H)$ respectively, then

$$\mathbf{a}_u \leq deg_{c(G)}(u) - 1 \text{ for all } u \in V(G) \text{ and } \mathbf{b}_v \leq deg_{c(H)}(v) - 1 \text{ for all } v \in V(H).$$

Hence $\mathbf{a}\Box\mathbf{b}_{(u,v)} = \mathbf{a}_u + \mathbf{b}_v \leq deg_{c(G)}(u) + deg_{c(H)}(v) - 2 = deg_{c(G\Box H)}((u, v)) - 1$, that is, $\mathbf{a}\Box\mathbf{b}$ is a stable configuration of $c(G\Box H)$.

(ii) In order to prove the second part of this lemma we need the following characterization of the recurrent configurations of the cone of a multigraph.

Since $\mathbf{1} = L(c(G), s)^t\mathbf{1}$ is a burning configuration of $c(G)$ for any multigraph G (definition 2.12), then by theorem 2.13, $\mathbf{a}$ is a recurrent configuration of $c(G)$ if and only if $\mathbf{a}$ is stable and there exist an order $u_1, \ldots, u_n$ of the vertices of $V(G)$ such that if $\mathbf{a}_1 = \mathbf{a} + \mathbf{1}$ and

$$\mathbf{a}_i = \mathbf{a}_{i-1} - \Delta_{u_{i-1}} \text{ for all } i = 2, \ldots, n,$$

then u_i is a not stable vertex of $\mathbf{a}_i$ for all $i = 1, \ldots, n$. Note that $\mathbf{a} = \mathbf{a}_n - \Delta_{u_n}$.

Claim 2.53. *Let* $\mathbf{a}$ *be a recurrent configuration of* $c(G)$, $\mathbf{b}$ *be a recurrent configuration* $c(H)$, $\mathbf{a}_1 = \mathbf{a} + \mathbf{1}$, $\mathbf{b}_1 = \mathbf{b} + \mathbf{1}$, *and*

$$\mathbf{a}_i = \mathbf{a}_{i-1} - \Delta_{u_{i-1}} \text{ for all } i = 2, \ldots, n, \text{ and } \mathbf{b}_i = \mathbf{b}_{i-1} - \Delta_{v_{i-1}} \text{ for all } j = 2, \ldots, m,$$

such that the vertex u_i *is a not stable in* $\mathbf{a}_i$ *for all* $i = 1, \ldots, n$ *and the vertex* v_j *is a not stable in* $\mathbf{b}_j$ *for all* $j = 1, \ldots, m$. *If* $\mathbf{c} = \mathbf{a}\Box\mathbf{b}$, $\mathbf{c}_{(1,1)} = \mathbf{a}\Box\mathbf{b} + \mathbf{1} = \mathbf{a}_1\Box\mathbf{b} = \mathbf{a}\Box\mathbf{b}_1$, *and*

$$\mathbf{c}_{(i,j)} = \begin{cases} \mathbf{c}_{(i-1,m)} - \Delta_{(u_{i-1},v_m)} & \text{if } i = 2, \ldots, n \text{ and } j = 1, \\ \mathbf{c}_{(i,j-1)} - \Delta_{(u_i,v_{j-1})} & \text{otherwise,} \end{cases}$$

then the vertex (u_i, v_j) *is a not stable in* $\mathbf{c}_{(i,j)}$ *for all* $i = 1, \ldots, n$ *and* $j = 1, \ldots, m$.

Proof. Since the vertex u_i is a not stable in $\mathbf{a}_i$ for all $i = 1, \ldots, n$ and the vertex v_j is a not stable in $\mathbf{b}_j$ for all $j = 1, \ldots, m$, then $(\mathbf{a}_i)_{u_i} \geq \deg_{c(G)}(u_i)$ for all $i = 1, \ldots, n$ and $(\mathbf{b}_j)_{v_j} \geq \deg_{c(H)}(v_j)$ for all $j = 1, \ldots, m$.

Now, $\mathbf{c}_{(i,1)} = \mathbf{c}_{(1,1)} - \sum_{1 \leq k \leq i-1} \sum_{1 \leq l \leq m} \Delta_{(u_k,v_l)} = (\mathbf{a}_1 - \sum_{1 \leq k \leq i-1} \Delta_{u_k})\Box\mathbf{b} = \mathbf{a}_i\Box\mathbf{b}$ for all $i = 1, \ldots, n$. Thus, $(\mathbf{c}_{(i,1)})_{(u_i,v_1)} = (\mathbf{a}_i\Box\mathbf{b})_{(u_i,v_1)} = (\mathbf{a}_i)_{u_i} + \mathbf{b}_{v_1} = (\mathbf{a}_i)_{u_i} + (\mathbf{b}_1)_{v_1} - 1 \geq \deg_{c(G)}(u_i) + \deg_{c(H)}(v_1) - 1 = \deg_{c(G\Box H)}((u_i, v_1))$ for all $i = 1, \ldots, n$.

Moreover, since $\mathbf{c}_{(i,j)} = \mathbf{a}_i\Box\mathbf{b} - \sum_{1 \leq l \leq j} \Delta_{(u_i,v_l)} = (\mathbf{a}_i - \mathbf{1})\Box\mathbf{b}_1 - \sum_{1 \leq l \leq j} \Delta_{(u_i,v_l)}$, then $(\mathbf{c}_{(i,j)})_{(u_i,v_j)} = (\mathbf{a}_i)_{u_i} + (\mathbf{b}_j)_{v_j} - 1$. Thus,

$$(\mathbf{c}_{(i,j)})_{(u_i,v_j)} = (\mathbf{a}_i)_{u_i} + (\mathbf{b}_j)_{v_j} - 1 \geq \deg_{c(G)}(u_i) + \deg_{c(H)}(v_j) - 1 = \deg_{c(G\Box H)}((u_i, v_j))$$

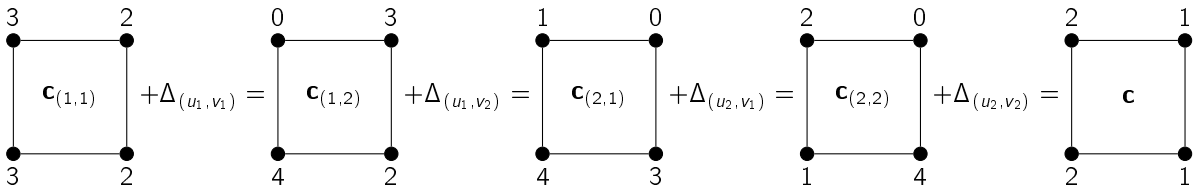
for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

Therefore (u_i, v_j) is a not stable vertex of $\mathbf{c}_{(i,j)}$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$. □

Finally, using the part (i) and the previous claim we obtain that $\mathbf{a}\Box\mathbf{b}$ is recurrent. □

Example 2.54. *Let* $G \cong H \cong \mathcal{K}_2$ *with* $V(G) = \{u_1, u_2\}$ *and* $V(H) = \{v_1, v_2\}$ *as vertex sets,* $\mathbf{a} = (1, 1)$ *be a recurrent configurations of* $c(G)$ *and* $\mathbf{b} = (1, 0)$ *be a recurrent configurations of* $c(H)$. *The configuration* $\mathbf{a} = (1, 1)$ *is recurrent because* $\mathbf{a}_1 = (2, 2)$ *is not stable,* $\mathbf{a}_2 = (0, 3) = \mathbf{a}_1 + \Delta_{u_1}$ *is not stable, and* $\mathbf{a} = \mathbf{a}_2 + \Delta_{u_2}$. *In a similar way, the configuration* $\mathbf{b} = (1, 0)$ *is recurrent because* $\mathbf{b}_1 = (2, 1)$ *is not stable,* $\mathbf{b}_2 = (0, 2) = \mathbf{b}_1 + \Delta_{v_1}$ *is nos stable, and* $\mathbf{b} = \mathbf{b}_2 + \Delta_{v_2}$.

Hence $\mathbf{c} = (2, 1, 2, 1) = (1, 1)\Box(1, 0)$ *is a recurrent configuration of* $c(G\Box H)$ *because*



As shows the next theorem, the mappings i_G and i_H induces homomorphism of groups between the sandpile groups of the cones of G and $G\Box H$, and H and $G\Box H$; respectively.

Theorem 2.55. *Let* G *and* H *be two multigraphs and* $\mathbf{e} \in SP(c(H), s_H)$ *the identity of the sandpile group of the cone of* H. *Then the mapping* $\widetilde{i_G} : SP(c(G), s_G) \to SP(c(G\Box H), s_{G\Box H})$ *given by*

$$\widetilde{i_G}(\mathbf{a}) = \mathbf{a}\Box\mathbf{e},$$

is an injective homomorphism of groups.

Proof. Since $\mathbf{e}$ is recurrent, then using lemma 2.52 (ii) we have that $\widetilde{i_G}(\mathbf{a}) = \mathbf{a}\Box\mathbf{e}$ is a recurrent configuration of $c(G\Box H)$ for all $\mathbf{a} \in SP(c(G), s_G)$, that is, the mapping $\widetilde{i_G}$ is well defined.

Now, we will prove that $\widetilde{i_G}$ is an homomorphism of groups. Let $\mathbf{a}, \mathbf{b} \in SP(c(G), s_G)$, then

$$\begin{aligned}\widetilde{i_G}(\mathbf{a} \oplus \mathbf{b}) &= (\mathbf{a} \oplus \mathbf{b})\Box\mathbf{e} = s(\mathbf{a}+\mathbf{b})\Box\mathbf{e} =_{L(c(G\Box H),t)} (\mathbf{a}+\mathbf{b})\Box\mathbf{e} = \mathbf{a}\Box\mathbf{e} + \mathbf{b}\Box\mathbf{e} =_{L(c(G\Box H),t)} s(\mathbf{a}\Box\mathbf{e} + \mathbf{b}\Box\mathbf{e}) \\ &= \mathbf{a}\Box\mathbf{e} \oplus \mathbf{b}\Box\mathbf{e} = \widetilde{i_G}(\mathbf{a}) \oplus \widetilde{i_G}(\mathbf{b}),\end{aligned}$$

and therefore $\widetilde{i_G}$ is an homomorphism of groups.

Finally, since for all $\mathbf{a}, \mathbf{b} \in SP(c(G), s_G)$ the configurations $\widetilde{i_G}(\mathbf{a})$ and $\widetilde{i_G}(\mathbf{b})$ are recurrent, then $\widetilde{i_G}(\mathbf{a}) = \widetilde{i_G}(\mathbf{b})$ if and only if $\mathbf{a}\Box\mathbf{e} = \mathbf{b}\Box\mathbf{e}$ if and only if $\mathbf{a} = \mathbf{b}$ and therefore $\widetilde{i_G}$ is an injective homomorphism of groups. □

Example 2.56. *Using the CSandPile program with the following instructions:*

```
csandpile c3 -cycle 3
csandpile c3 -group
```

we obtain the `c3.csp` *file containing the following*

```
 Generator 1: 1 0 S
Checking configuration: 1 0 S

Powers
1 - 1 0 S
2 - 0 1 S
3 - 1 1 S

Generator 2: 0 1 S
Checking configuration: 0 1 S

Powers
1 - 0 1 S
2 - 1 0 S
3 - 1 1 S
```

Thus, $SP(c(\mathcal{K}_2)) = \mathbb{Z}_3$ *is generated by* $(1,0)$ *with identity* $(1,1)$.

On the other hand, using the following instruction in CSandPile:

```
csandpile cC5 -group
```

with the file `cC5.gph` *containing the Laplacian matrix of the cone of* C_5.

```
6
3 -1 0 0 -1 -1
-1 3 -1 0 0 -1
0 -1 3 -1 0 -1
0 0 -1 3 -1 -1
-1 0 0 -1 3 -1
-1 -1 -1 -1 -1 5
6
```

we obtain the file `cC5.csp`

```
 Generator 1: 1 0 0 0 0 S
Checking configuration: 2 1 1 1 1 S

Powers
1 - 2 1 1 1 1 S
2 - 0 2 1 1 2 S
3 - 1 2 1 1 2 S
4 - 2 2 1 1 2 S
5 - 2 0 2 2 0 S
6 - 0 1 2 2 1 S
7 - 1 1 2 2 1 S
8 - 2 1 2 2 1 S
9 - 0 2 2 2 2 S
10 - 1 2 2 2 2 S
11 - 2 2 2 2 2 S

Generator 2: 0 1 0 0 0 S
Checking configuration: 1 2 1 1 1 S

Powers
1 - 1 2 1 1 1 S
2 - 2 0 2 1 1 S
3 - 2 1 2 1 1 S
4 - 2 2 2 1 1 S
5 - 0 2 0 2 2 S
6 - 1 0 1 2 2 S
7 - 1 1 1 2 2 S
8 - 1 2 1 2 2 S
9 - 2 0 2 2 2 S
10 - 2 1 2 2 2 S
11 - 2 2 2 2 2 S
...
```

That is, $K(c(C_5)) = \mathbb{Z}_{11}^2$ is generated by $(2,1,1,1,1)$ and $(1,2,1,1,1)$ with identity $e = (2,2,2,2,2)$ (see [**19**]*).*

Finally, using CSandPile we can see that $K(c(C_5 \square \mathcal{K}_2)) = \mathbb{Z}_{11\cdot 29} \oplus \mathbb{Z}_{3\cdot 11\cdot 29}$. Furtermore, using the mapping $i_{\mathcal{K}_2}$ we have that

$$i_{\mathcal{K}_2}(1,0) = (3,3,3,3,3,2,2,2,2,2)$$

is a generator of a subgroup of $K(c(C_5 \square \mathcal{K}_2))$ isomorphic to $\mathbb{Z}_3$, and using the mapping i_{C_5} we have that

$$i_{C_5}(2,1,1,1,1) = (3,2,2,2,2,3,2,2,2,2) \text{ and } i_{C_5}(1,2,1,1,1) = (2,3,2,2,2,2,3,2,2,2)$$

are generators of subgroups of $K(c(C_5 \square \mathcal{K}_2))$ isomorphic to $\mathbb{Z}_{11}$,

The last result of this section will be useful in order to calculate in the next section the sand pile group of the cone of the hypercube.

Lemma 2.57. *Let G be a graph with n vertices, then*

$$L(c(\mathcal{K}_2 \square G), s) = \begin{pmatrix} L(c(G),s) + I_n & -I_n \\ -I_n & L(c(G),s) + I_n \end{pmatrix} \sim \begin{pmatrix} I_n & 0 \\ 0 & L(c(G),s)(L(c(G),s) + 2I_n) \end{pmatrix}$$

Furthermore, $|L(c(\mathcal{K}_2 \square G), s) + kI_n| = |L(c(G),s) + kI_n| \cdot |L(c(G),s) + (k+2)I_n|$ for all $k \in \mathbb{N}$.

Proof. First of all, let us simplify $L(c(\mathcal{K}_2 \square G), s) + kI_n$.

$$L(c(\mathcal{K}_2 \square G), s) + kI_n = \begin{bmatrix} L(c(G),s) + (k+1)I_n & -I_n \\ -I_n & L(c(G),s) + (k+1)I_n \end{bmatrix}$$

$$\sim \begin{bmatrix} L(c(G),s) & L(c(G),s) \\ -I_n & L(c(G),s) + (k+1)I_n \end{bmatrix} \sim \begin{bmatrix} I_n & L(c(G),s) + (k+1)I_n \\ -L(c(G),s) & L(c(G),s) \end{bmatrix}$$

$$\sim \begin{bmatrix} I_n & 0 \\ 0 & L(c(G),s)(L(c(G),s)+(k+1)I_n)+L(c(G),s) \end{bmatrix}$$
$$= \begin{bmatrix} I_n & 0 \\ 0 & L(c(G),s)(L(c(G),s)+(k+2)I_n) \end{bmatrix}$$

At this point, we can conclude the result taking $k=0$ and computing the determinant of the last matrix. □

Example 2.58. *For instance, the reduced Laplacian matrix of $c(\mathcal{K}_2\square C_5)$ is:*

$$\begin{bmatrix} 4&-1&0&0&-1&-1&0&0&0&0\\ -1&4&-1&0&0&0&-1&0&0&0\\ 0&-1&4&-1&0&0&0&-1&0&0\\ 0&0&-1&4&-1&0&0&0&-1&0\\ -1&0&0&-1&4&0&0&0&0&-1\\ -1&0&0&0&0&4&-1&0&0&-1\\ 0&-1&0&0&0&-1&4&-1&0&-1\\ 0&0&-1&0&0&0&-1&4&-1&0\\ 0&0&0&-1&0&0&0&-1&4&-1\\ 0&0&0&0&-1&-1&0&0&-1&4 \end{bmatrix} \sim \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&26&-10&1&1&-10\\ 0&0&0&0&0&-10&26&-10&1&1\\ 0&0&0&0&0&1&-10&26&-10&1\\ 0&0&0&0&0&1&1&-10&26&-10\\ 0&0&0&0&0&-10&1&1&-10&26 \end{bmatrix}$$

7 The sandpile group of $c(Q_d)$

In this section we present a combinatorial description of the generators of some subgroups of of the sandpile group of the cone of the hypercube of dimension d.

Let $Q_d = \square_{i=1}^{d}\mathcal{K}_2 = \underbrace{\mathcal{K}_2\square\cdots\square\mathcal{K}_2}_{d-\text{copies of } \mathcal{K}_2}$ with vertex set $V(Q_d)=\{v_{\mathbf{a}} \mid \mathbf{a}\in\{0,1\}^d\}$ and edge set

$$E(Q_d)=\{\{v_{\mathbf{a}},v_{\mathbf{a}'}\} \mid \mathbf{a},\mathbf{a}'\in\{0,1\}^d \text{ and } \mathbf{a}-\mathbf{a}'=\pm e_i \text{ for some } 1\le i\le d\}.$$

Now, for all $\beta\in\{0,1\}^d$, let $Q_\beta = Q_d[\{v_{\mathbf{a}} \mid \mathrm{supp}(\mathbf{a})\subseteq\mathrm{supp}(\beta)\}]$ an induced subgraph of Q_d over the vertices $\{v_{\mathbf{a}} \mid \mathrm{supp}(\mathbf{a})\subseteq\mathrm{supp}(\beta)\}$. Is not difficult note that.

- $Q_\beta \cong \square_{i=1}^{|\beta|}\mathcal{K}_2 = Q_{|\beta|}$, where $|\beta|=\sum_{i=1}^{d}\beta_i$,
- $Q_{\beta'} = Q_\beta[\{v_a \mid \mathrm{supp}(a)\subseteq\mathrm{supp}(\beta')\}]$ for all $\mathrm{supp}(\beta')\subseteq\mathrm{supp}(\beta)$,
- $Q_\beta = Q_{\beta'}\square Q_{\beta-\beta'}$ for all $\mathrm{supp}(\beta')\subsetneq\mathrm{supp}(\beta)$,
- $Q_d = Q_{(1,\dots,1)} = Q_\beta\square Q_{1-\beta}$ for all $\beta\in\{0,1\}^d$.

If $0\le r,t\le d$ and $r=|\beta|$ or $t=|\beta|$, then $g_\beta(r,t)+(d-|\beta|)\mathbf{1}\in\mathbb{N}^{V(Q_d)}$ is a recurrent configurations of $c(Q_d)$, where $g_\beta(r,t)\in\mathbb{N}^{V(Q_d)}$ is given by

$$g_\beta(r,t)_{\mathbf{a}} = \begin{cases} r & \text{if } \beta\cdot\mathbf{a} \text{ is even,}\\ t & \text{if } \beta\cdot\mathbf{a} \text{ is odd.}\end{cases}$$

Let $\beta'\neq\beta\in\{0,1\}^d$ such that $\beta'\le\beta$ and $\widetilde{i}_{\beta',\beta}: SP(c(Q_{\beta'}),s)\to SP(c(Q_\beta),s)$ the induced map by the inclusion $i_{\beta',\beta}: Q_{\beta'}\to Q_{\beta'}\square Q_{\beta-\beta'}$.

Theorem 2.59. *Let $\widetilde{K}_\beta=\{g_\beta(r,t)+(d-|\beta|)\mathbf{1} \mid 0\le r,t\le d \text{ and } r=|\beta| \text{ or } t=|\beta|\}$ for all $\beta\in\{0,1\}^d$. Then*

(i): $\mathbb{Z}_{2|\beta|+1}\cong\widetilde{K}_\beta\lhd K(c(Q_d))$ for all $\beta\in\{0,1\}^d$,
(ii): $d\mathbf{1}=g_1(d,d)$ is the identity of $\widetilde{K}_\beta$ for all $\beta\in\{0,1\}^d$,
(iii): $\widetilde{K}_\beta\cap\widetilde{K}_{\beta'}=d\mathbf{1}$ for all $\beta\neq\beta'\in\{0,1\}^d$,
(iv): $|SP(c(Q_{|\beta|}),s)|=\prod_{i=1}^{|\beta|}(2i+1)^{\binom{|\beta|}{i}}$,
(vi): $\widetilde{i}_{\beta,1}(SP(c(Q_\beta),s))\cap\widetilde{i}_{\beta,1}(SP(c(Q_{\beta'}),s))=\widetilde{i}_{\beta,1}(SP(c(Q_{\beta\odot\beta'}),s))$ for all $\beta',\beta\in\{0,1\}^d$,

where $(\mathbf{a}\odot\mathbf{b})_i=\mathbf{a}_i\cdot\mathbf{b}_i$ for all i.

Proof. Let $\beta \in \{0,1\}^d$ and $f_\beta : V(Q_\beta) \to V(\mathcal{K}_2(|\beta|))$ the graph homomorphism given by

$$f_\beta(\mathbf{a}) = \begin{cases} v_1 & \text{if } \beta \cdot \mathbf{a} \text{ is even,} \\ v_2 & \text{if } \beta \cdot \mathbf{a} \text{ is odd.} \end{cases}$$

Since $\widetilde{K}_\beta = \mathrm{Im}(\widetilde{i}_{\beta,1} \circ \widetilde{f}_\beta)$, then using theorems 2.48, 2.55, 2.33 and 2.34 we have that

$$\mathbb{Z}_{2|\beta|+1} \cong \widetilde{K}_\beta \triangleleft SP(c(Q_d), s) \text{ for all } \beta \in \{0,1\}^d$$

and $d\mathbf{1} = g_1(d,d)$ is the identity of $\widetilde{K}_\beta$. The part (iii) of the theorem it follows by inspection.

Applying Lemma 2.57 (ii)

$$\begin{aligned} |SP(c(Q_{|\beta|}), s)| &= |L(c(Q_{|\beta|}), s)| \overset{\text{(ii)}}{=} |L(c(Q_{|\beta|-1}), s)| \cdot |L(c(Q_{|\beta|-1}), s) + 2I_{2^{|\beta|-1}}| \\ &\overset{\text{(ii)}}{=} \prod_{i=0}^{|\beta|-1} |L(c(Q_1), s) + 2iI_2|^{\binom{|\beta|-1}{i}} = \prod_{i=0}^{|\beta|-1} [(2i+3)(2i+1)]^{\binom{|\beta|-1}{i}} \\ &= \prod_{i=1}^{|\beta|} (2i+1)^{\binom{|\beta|}{i}}. \end{aligned}$$

□

Conjecture 2.60. *Let d be a natural number, then the sandpile group of the cone of the hypercube $c(Q_d)$ is given by:*

$$K(c(Q_d)) \cong \bigoplus_{i=1}^{d} \mathbb{Z}_{2i+1}^{\binom{d}{i}} = \mathbb{Z}_3^d \oplus \mathbb{Z}_5^{\binom{d}{2}} \oplus \cdots \oplus \mathbb{Z}_{2d-1}^d \oplus \mathbb{Z}_{2d+1}.$$

Furthermore,

$$SP(c(Q_d), s)) = \bigoplus_{\beta \in \{0,1\}^d} \widetilde{K}_\beta.$$

Example 2.61. *If $d = 1$, we have that $K(c(Q_1)) = \mathbb{Z}_3$, $SP(c(Q_1), s) = \{(1,0),(0,1),(1,1)\}$, $(1,0)$ and $(0,1)$ are generators $K(c(Q_1))$ and $(1,1)$ is the identity of $K(c(Q_1))$.*

If $d = 2$, we have that $K(c(Q_2)) = \mathbb{Z}_3^2 \oplus \mathbb{Z}_5$, $SP(c(Q_2), s)$ is generated by the recurrent configurations $\{(2,2,1,1),(2,1,2,1),(2,0,2,0)\}$, and $(2,2,2,2)$ is the recurrent configuration that plays the role of the identity in $K(c(Q_2))$. Furthermore,

$$\widetilde{K}_{(1,0)} = \{(2,2,1,1),(1,1,2,2),(2,2,2,2)\} = \{(1,1,0,0),(0,0,1,1),(1,1,1,1)\} + (1,1,1,1)$$

and

$$\widetilde{K}_{(0,1)} = \{(2,1,2,1),(1,2,1,2),(2,2,2,2)\} = \{(1,0,1,0),(0,1,0,1),(1,1,1,1)\} + (1,1,1,1)$$

form two subgroups $K(c(Q_2))$ isomorphics to $\mathbb{Z}_3$,

$$\widetilde{K}_{(1,1)} = \{(2,0,2,0),(1,2,1,2),(2,1,2,1),(0,2,0,2),(2,2,2,2)\}$$

form one subgroups isomorphic to $\mathbb{Z}_5$ and $K(c(Q_2)) = \widetilde{K}_{(1,0)} \oplus \widetilde{K}_{(0,1)} \oplus \widetilde{K}_{(1,1)}$.

APPENDIX A

CSandPile

This appendix describes the version 0.5 of the program *CSandPile*, a program that allow to compute the sandpile group of a multigraph G. The *CSandPile* program was programmed in C++ language using the GNU Compiler Collection and had been used over Windows XP, MAC OS 10.5.8 and Ubuntu Linux 9.04.

We developed a computer program called *CSandPile*, because we want to understand the combinatorial structure of the recurrent configurations of the sandpile group of a multigraph G, an important part of this thesis. The *CSandPile* program version 0.5 is mainly a tool for compute the group operations of the recurrent representatives of non-negative configurations of G, that is, is the first effort to find the combinatorial structure of the group operations of the recurrent configurations that generate the sandpile group of G.

If c be a non-negative configuration, then using *CSandPile*, one may compute the following:

- the stabilization of c,
- the recurrent representative of c,
- the powers of a recurrent configuration,
- the representative of the inverse of a recurrent configuration c,
- the recurrent representative of the identity of the sandpile group of G,
- the determinant of the Laplacian matrix of G,
- the powers of the representative of the canonical base.

1 The structure of *CSandPile*

Now, we will explain how to use the program *CSandPile*, it consist of two files:

- `csandpile.cpp` the file with the source code, and
- `csandpile.exe` the executable file if you are using a Windows environment, or `csandpile.out` the executable file if you are using a UNIX environment.

Also, you need to have an input file `<my project>.gph` with the input data.

1.1 The input file

The input file `<my project>.gph` is structured as follows: The first line contains the order of the Laplacian matrix of G, the next lines contain the rows of the Laplacian matrix of G, the next line contain the vertex of G that will play the role of the sink, and finally the last line contain some configuration.

Example A.1. *Let G be the cycle C_4 with four vertices, the Laplacian matrix of C_4 is given by:*

$$L(C_4) = \begin{bmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

The file called "c4.gph" has the order, the Laplacian matrix of C_4, the vertex 4 *as a sink and the vector* $(2, 2, 1, 0)$ *as the configuration.*

```
4
2 -1 0 -1
-1 2 -1 0
0 -1 2 -1
-1 0 -1 2
4
2 2 1 0
```

If we do not write a configuration, the configuration $\sigma_{MAX} = (\deg(1) - 1, \deg(2) - 1, ..., \deg(n) - 1)$ *will be taken by default.*

1.2 Running *CSandPile*

The files `csandpile.exe` and `csandpile.out` are the executables files in Windows and UNIX, respectively. You must run it from the console of your operating system. The syntax for calling *CSandPile* is

`csandpile <my project> [-option]`

where -option can be one of the following options:

```
-s           to obtain the stable configuration
-p           to obtain the powers of the recurrent configuration
-i           to obtain the identity
-r           to obtain the recurrent configuration
-ri          to obtain the inverse recurrent configuration
-det         to obtain the determinant of the reduced Laplacian matrix
-group       to obtain the powers of the standard base
-complete n  to create the Laplacian matrix of the complete graph of n
             vertices
-path n       to create the Laplacian matrix of the path of n vertices
-cycle n      to create the Laplacian matrix of the cycle of n vertices
```

When you type and execute `csandpile` or `./csandpile.out` if you are working in a UNIX environment on the console, the program creates a file called `<my project>.csp`.

For instance, if you wan to obtain the stable configuration, you need to type

`csandpile <my project> -s` or `./csandpile.out <my project> -s`.

and *CSandPile* will be create a file called `<my project>.csp`. Thus, using "c4.gph" as an input file, the file `c4.csp` contains the following information:

```
The stable configuration of
2 2 1 S
is
0 1 1 S
```

Note that we put a `S` in the coordinate of the sink.

To obtain the powers of a recurrent configuration, you need to type `-p` as an option. Using "c4.gph" as an input file, the `c4.csp` file contains

```
Checking configuration: 0 1 1 S
Powers
1 - 0 1 1 S
2 - 1 1 1 S
3 - 1 1 0 S
4 - 1 0 1 S
```

To obtain the identity configuration, you need to type `-i` as an option. Using "c4.gph" as an input file, the `c4.csp` file contains

```
Identity: 1 0 1 S
```

To obtain the recurrent configuration of the configuration given in the `<my project>.gph`, you need to type `-r` as an option. Using "c4.gph" as an input file, the `c4.csp` file contains

```
The recurrent configuration of 2 2 1 S
is 0 1 1 S
```

To obtain the recurrent inverse configuration of the configuration given in the `<my project>.gph`, you need to type `-ri` as an option. Using "c4.gph" as an input file, the file `c4.csp` contains

```
Inverse recurrent configuration: 1 1 0 S
```

To obtain the determinant of the reduced Laplacian matrix, you need to type `-det` as an option. Using "c4.gph" as an input file, the `c4.csp` file contains

```
Determinant: 4
```

To obtain the powers of the canonical base, you need to type `-group` as an option. Using "c4.gph" as an input file, the `c4.csp` file contains

```
Generator 1: 1 0 0 S
Checking configuration: 0 1 1 S
Powers
1 - 0 1 1 S
2 - 1 1 1 S
3 - 1 1 0 S
4 - 1 0 1 S

Generator 2: 0 1 0 S
Checking configuration: 1 1 1 S
Powers
1 - 1 1 1 S
2 - 1 0 1 S

Generator 3: 0 0 1 S
Checking configuration: 1 1 0 S
Powers
1 - 1 1 0 S
2 - 1 1 1 S
3 - 0 1 1 S
4 - 1 0 1 S
```

1.3 Some special graphs

Also, *CSandPile* can generate the Laplacian matrix of the complete graph of n vertices, the path of n vertices and the cycle of n vertices and write it in the `<my project>.gph` file. This can be done by typing `-complete` n, `-cycle` n or `-path` n as an option, respectively. For instance, if do you write

```
csandpile k4 -complete 4
```

you will obtain the Laplacian matrix of the complete graph of 4 vertices in the `k4.gph` file.

```
4
3 -1 -1 -1
-1 3 -1 -1
-1 -1 3 -1
-1 -1 -1 3
4
```

Note that by default, *CSandPile* will define the vertex *n* as the sink.

2 The source code

The program was coded using C++. We used the `libraries`

```
#include <iostream>
#include <fstream>
#include <string.h>
#include <math.h>
#include <stdlib.h>
using namespace std;
```

In order to write the file `<my project>.csp` from any part of the program, we declare the following global variable:

```
ofstream fout;
```

The program has a `class` called csandpile. The sandpile `class` has `private members`:

```
int *degree;
int *configuration;
int *config;
int **rest;
int **laplacian;
float **inverse;
int m;
```

and `public members`:

```
csandpile(char *);
int wini(int *);
void sum(int, int *);
void top(int *);
void print();
void print(int *);
void print(float *);
void printmatrix(int **, int);
void printmatrix(float **,int);
void stable();
bool areequals(int *, int *);
void recurrent(int *);
```

```
void powerof(int *);
void powerof();
void identity();
void reprec();
void reprecinv();
int cofactor(int **, int);
void determinant();
void group();
```

2.1 Functions

Now, we will explain how each function works. The `csandpile(char *) constructor` reads from the `<my project>.gph` file, the order of the Laplacian matrix, the Laplacian matrix, the sink and a configuration. Then, the reduced Laplacian matrix is built and the degree of the non sink vertices are stored in the `degree` variable. If the file `<my project>.gph` does not contain a configuration, then the program creates one.

```
csandpile::csandpile(char *name){
char *input_file_name = new char [(strlen(name)+3)];
strcpy(input_file_name,name);
strcat(input_file_name,".gph");
ifstream fin(input_file_name);
int dim;
fin>>dim;
laplacian = new int*[dim];
for(int i=0; i<dim; i++)
  laplacian[i] = new int [dim];
for(int i=0; i<dim; i++)
  for(int j=0; j<dim; j++)
    fin>>laplacian[i][j];
fin>>sink;
m = dim - 1;
rest = new int*[m];
for(int i=0; i<m; i++)
  rest[i] = new int [m];
int flagi=0, flagj=0;
for(int i=0; i<dim; i++)
  for(int j=0; j<dim; j++){
    if(i!=(sink-1))
      if(j!=(sink-1))
        rest[i-flagi][j-flagj]=laplacian[i][j];
      else
        flagj=1;
    else
      flagi=1;
      if(j==(dim-1))
        flagj=0;
  }
```

```
 degree = new int [m];
for(int i=0; i<m; i++)
  degree[i] = rest[i][i];
config = new int[dim];
configuration = new int[m];
if(!fin.eof()){
  for(int i=0; i<dim; i++)
    fin>>config[i];
  flagi=0;
  for(int i=0; i<dim; i++)
    if(i!=sink)
      configuration[i-flagi] = config[i];
    else
      flagi=1;
}
else
  for(int i=0; i<m; i++)
    configuration[i]=degree[i]-1;
fin.close();
}
```

2.1.1 The "wini" function

The "wini" function finds out an index of the vector "tocheck" that is non-stable. Moreover, if the configuration is stable, then the function returns -1.

```
int csandpile::wini(int *tocheck){
for(int i=0; i<m; i++)
  if(tocheck[i]>=degree[i])
    return i;
return -1;
}
```

2.1.2 The "sum" function

The "sum" function adds the Δ_k toppling operator to the "tocheck" vector.

```
void csandpile::sum(int k, int *tocheck){
for(int i=0; i<m; i++)
  tocheck[i] -= rest[k][i];
}
```

2.1.3 The "top" function

The "top" function checks when the "tocheck" vector is stable. Moreover, if "tocheck" is not stable in the vertex k, then the "top" function adds the k-restriction. The function repeats this process until the "tocheck" vector is stable.

```
void csandpile::top(int *tocheck){
int aux=wini(tocheck);
while(aux!=-1){
  sum(aux, tocheck);
  aux=wini(tocheck);
}
}
```

2.1.4 The "print" function

The "print" function prints the "configuration" vector.

```
void csandpile::print(){
fout<<" ";
for(int i=0; i<m; i++){
  if(i==(sink-1))
    fout<<"S ";
  fout<<configuration[i]<<" ";
}
if(m==(sink-1))
  fout<<"S";
fout<<endl;
}
```

2.1.5 The "print" function

The "print" function prints the "tocheck" vector when it has integer entries.

```
void csandpile::print(int *tocheck){
fout<<" ";
for(int i=0; i<m; i++){
  if(i==(sink-1))
    fout<<"S ";
  fout<<tocheck[i]<<" ";
}
if(m==(sink-1))
  fout<<"S";
fout<<endl;
}
```

2.1.6 The "print" function

The "print" function prints the "tocheck" vector when it has float entries.

```
void csandpile::print(float *tocheck){
fout<<" ";
for(int i=0; i<m; i++){
  if(i==(sink-1))
    fout<<"S ";
  fout<<tocheck[i]<<" ";
}
if(m==(sink-1))
  fout<<"S";
fout<<endl;
}
```

2.1.7 The "printmatrix" function

The "printmatrix" function prints the "matrix" matrix when it has integer entries.

```
void csandpile::printmatrix(int **matrix, int n){
fout<<endl;
for(int i=0; i<n; i++){
  for(int j=0; j<n; j++)
    fout<<" "<<matrix[i][j];
  fout<<endl;
 }
}
```

2.1.8 The "printmatrix" function

The "printmatrix" function prints the "matrix" matrix when it has float entries.

```
void csandpile::printmatrix(float **matrix, int n){
fout<<endl;
for(int i=0; i<n; i++){
  for(int j=0; j<n; j++)
    fout<<" "<<matrix[i][j];
  fout<<endl;
 }
}
```

2.1.9 The "stable" function

The "stable" function obtains and prints the stable configuration of a vector using the "top" function.

```
void csandpile::stable(){
int *aux;
aux = new int [m];
for(int i=0; i<m; i++)
  aux[i]=configuration[i];
fout<<" The stable configuration of"<<endl;
print(aux);
top(aux);
fout<<endl;
fout<<" is"<<endl;
print(aux);
}
```

2.1.10 The "areequals" function

The "areequals" function check when the vectors "aux" and "tocheck" are equals.

```
bool csandpile::areequals(int *aux, int * tocheck){
for(int i=0; i<m; i++)
  if(aux[i]!=tocheck[i])
    return false;
return true;
}
```

2.1.11 The "recurrent" function

The "recurrent" function computes the recurrent configuration of the "tocheck" vector using the following result:

Proposition A.2. *If c is a non-negative configuration, then the configuration*

$$s(2\sigma_{MAX} - s(2\sigma_{MAX}) + c)$$

is recurrent and is in the same equivalence class of c.

Proof. Since $2\sigma_{MAX} - s(2\sigma_{MAX}) \in \langle \Delta_1, ..., \Delta_n, x_n \rangle$, we have that $2\sigma_{MAX} - s(2\sigma_{MAX}) + c$ and c are in the same equivalence class. Hence, $s(2\sigma_{MAX} - s(2\sigma_{MAX}) + c)$ is in the equivalence class of c. On the other hand,

$$\begin{aligned} s(2\sigma_{MAX} - s(2\sigma_{MAX}) + c) &= s(\sigma_{MAX} + \sigma_{MAX} - s(2\sigma_{MAX}) + c) \\ &= s(\sigma_{MAX} + pos) \end{aligned}$$

where pos is a non-negative configuration. By theorem 2.13 we have that $s(\sigma_{MAX} + pos)$ is recurrent. Thus, $s(2\sigma_{MAX} - s(2\sigma_{MAX}) + c)$ is recurrent. □

```
void csandpile::recurrent(int *tocheck){
int *aux = new int [m];
for(int i=0; i<m; i++)
  aux[i] = 2*(degree[i]-1);
top(aux);
for(int i=0; i<m; i++)
  aux[i] = 2*(degree[i]-1) - aux[i] + tocheck[i];
top(aux);
for(int i=0; i<m; i++)
  tocheck[i] = aux[i];
}
```

2.1.12 The "powerof" function

The "powerof" function computes the recurrent representative of all the powers of "tocheck" using the "recurrent" function.

```
void csandpile::powerof(int * tocheck){
int *aux1, *aux2;
aux1 = new int [m];
aux2 = new int [m];
for(int i=0; i<m; i++)
  aux1[i] = tocheck[i];
recurrent(aux1);
for(int i=0; i<m; i++)
  aux2[i] = aux1[i];
fout<<" Configuration to check: ";
print(aux1);
fout<<endl;
fout<<" Powers"<<endl;
int j=1;
fout<<j++<<" - ";
print(aux2);
for(int i=0; i<m; i++)
  aux2[i] += aux1[i];
top(aux2);
while(!areequals(aux1,aux2)){
  fout<<j++<<" - ";
  print(aux2);
  for(int i=0; i<m; i++)
    aux2[i] += aux1[i];
  top(aux2);
  }
}
```

2.1.13 The "power" function

The "power" function uses the before "powerof" function in the "configuration" vector.

```
void csandpile::power(){
powerof(configuration);
}
```

2.1.14 The "identity" function

The "identity" function computes the recurrent representative of th identity using the following result:

Proposition A.3. *Let* $\sigma_{MAX} = (\deg(1) - 1, ..., \deg(n) - 1)$ *then*

$$s(2\sigma_{MAX} - s(2\sigma_{MAX}))$$

is recurrent and is a representative of the identity of the sandpile group.

Proof. Since $2\sigma_{MAX} - s(2\sigma_{MAX}) \in \langle \Delta_1, ..., \Delta_n, x_n \rangle$, then $2\sigma_{MAX} - s(2\sigma_{MAX})$ and $(0, ..., 0)$ are in the same equivalence class, that is, $s(2\sigma_{MAX} - s(2\sigma_{MAX}))$ and $(0, ..., 0)$ are in the same equivalence class.

On the other hand,

$$\begin{aligned} s(2\sigma_{MAX} - s(2\sigma_{MAX})) &= s(\sigma_{MAX} + \sigma_{MAX} - s(2\sigma_{MAX})) \\ &= s(\sigma_{MAX} + pos), \end{aligned}$$

where pos is a non-negative configuration. Therefore, by theorem 2.13 $s(2\sigma_{MAX} - s(2\sigma_{MAX})) = s(\sigma_{MAX} + pos)$ is recurrent. □

```
void csandpile::identity(){
int *aux = new int [m];
for(int i=0; i<m; i++)
  aux[i] = 2*(degree[i]-1);
top(aux);
for(int i=0; i<m; i++)
  aux[i] = 2*(degree[i]-1) - aux[i];
top(aux);
print(aux);
}
```

2.1.15 The "reprec" function

The "reprec" function computes and prints the recurrent configuration of the configuration vector given in the file "<my project>.gph".

```
void csandpile::reprec(){
int *aux = new int[m];
for(int i=0; i<m; i++)
  aux[i] = configuration[i];
recurrent(aux);
print(aux);
}
```

2.1.16 The "reprecinv" function

The "reprecinv" function computes the recurrent representative of the inverse of the configuration vector given in the file "<my project>.gph" using the following result:

Proposition A.4. *Let* c *be a non-negative configuration,* $\sigma_{MAX} = (\deg(1) - 1, ..., \deg(n) - 1)$, *and* $m = \max_{1 \leq i \leq n-1} \left\{ \left\lceil \frac{c[i]}{\sigma_{MAX}[i]} \right\rceil \right\} + 2$, *then*

$$s(m\sigma_{MAX} - s(m\sigma_{MAX}) - c)$$

is a recurrent representative of $-c$.

Proof. Since $m\sigma_{MAX} - s(m\sigma_{MAX}) \in \langle \Delta_1, ..., \Delta_n, x_n \rangle$, then $m\sigma_{MAX} - s(m\sigma_{MAX}) - c$ and $-c$ are in the same equivalence class, that is, $s(m\sigma_{MAX} - s(m\sigma_{MAX}) - c)$ and $-c$ are in the equivalence class.

On the other hand, since $m = \max_{1 \leq i \leq n-1} \left\{ \left\lceil \frac{c[i]}{\sigma_{MAX}[i]} \right\rceil \right\} + 2$, then $(m-2)\sigma_{MAX} \geq c$ and

$$\begin{aligned} s(m\sigma_{MAX} - s(m\sigma_{MAX}) - c) &= s(\sigma_{MAX} + \sigma_{MAX} - s(2\sigma_{MAX}) + (m-2)\sigma_{MAX} - c) \\ &= s(\sigma_{MAX} + pos), \end{aligned}$$

where pos is a non-negative configuration. Finally, by theorem 2.13 $s(2\sigma_{MAX} - s(2\sigma_{MAX}) + c) = s(\sigma_{MAX} + pos)$ is recurrent. □

```
void csandpile::reprecinv(){
int *aux = new int [m];
int max=1;
for(int i=0; i<m; i++)
  if((degree[i]-1)!=0)
    if(max<((int) configuration[i]/(degree[i]-1)+1))
      max = (int) configuration[i]/(degree[i]-1)+1;
max = max + 2;
for(int i=0; i<m; i++)
  aux[i] = max*(degree[i]-1);
top(aux);
for(int i=0; i<m; i++)
  aux[i] = max*(degree[i]-1) - aux[i] - configuration[i];
top(aux);
print(aux);
}
```

2.1.17 The "cofactor" function

The "cofactor" function computes the cofactor of a $n \times n$ matrix.

```
int csandpile::cofactor(int **matrix, int n){
if(n == 1)
  return matrix[0][0];
else if(n == 2)
  return (matrix[0][0]*matrix[1][1] - matrix[1][0]*matrix[0][1]);
int sum=0;
int **temp = new int *[n-1];
for(int i=0; i<(n-1); i++)
  temp[i] = new int [n-1];
for(int i=0; i<n; i++){
  for(int j=1; j<n; j++)
    for(int k=0; k<n; k++)
      if( k<i )
        temp[j-1][k] = matrix[j][k];
      else if( k>i )
        temp[j-1][k-1] = matrix[j][k];
  sum += matrix[0][i]* (int) pow(-1,i)*cofactor(temp,n-1);
}
return sum;
}
```

2.1.18 The "determinant" function

The "determinant" function computes the determinant of the reduced Laplacian matrix.

```
void csandpile::determinant(){
fout<<" Determinant: "<< cofactor(rest,m)<<endl;
}
```

2.1.19 The "group" function

The "group" function obtains the powers of the canonical base.

```
void csandpile::group(){
int * generator = new int [m];
for(int i=0; i<m; i++)
  generator[i]=0;
for(int i=0; i<m; i++){
  generator[i]=1;
  int * gaux = new int [m];
  for(int j=0; j<m; j++)
    gaux[j] = generator[j];
  fout<<" Generator "<<(i+1)<<": ";
  print(generator);
  powerof(gaux);
  fout<<endl;
  generator[i]=0;
}
}
```

2.1.20 The "main" function

Finally, we have the code of the "main" function. This function receives 3 parameters, the first one is the number of options and the second one is an array with each of the chosen options.

```
int main(int argc, char * argv[]){
if(argc == 1){
  cout<<"No input file";
}
else if(argc == 2){
  csandpile sand(argv[1]);
  char *output_file_name = new char [(strlen(argv[1])+3)];
  strcpy(output_file_name,argv[1]);
  strcat(output_file_name,".csp");
  fout.open(output_file_name, ios::trunc);
  fout<<endl<<" Write, after of the executable file, one of the following
options:";
  fout<<endl<<endl<<" -s to obtain the stable configuration";
```

```
   fout<<endl<<" -p to obtain the powers of the recurrent configuration";
   fout<<endl<<" -i to obtain the identity";
   fout<<endl<<" -r to obtain the recurrent configuration";
   fout<<endl<<" -ri to obtain the inverse recurrent configuration";
   fout<<endl<<" -group to obtain the powers of the standard base";
   fout<<endl<<" -complete n to create the Laplacian matrix of the
complete graph of n vertices";
   fout<<endl<<" -path n to create the Laplacian matrix of the path of n
vertices";
   fout<<endl<<" -cycle n to create the Laplacian matrix of the cycle of n
vertices";
   fout<<endl<<endl;
fout.close();
}
else if(argc == 3){
   csandpile sand(argv[1]);
   char *output_file_name = new char [(strlen(argv[1])+3)];
   strcpy(output_file_name,argv[1]);
   strcat(output_file_name,".csp");
   fout.open(output_file_name, ios::trunc);
   if( strcmp( argv[2] , "-s") == 0 )
     sand.stable();
   else if( strcmp(argv[2] , "-p") == 0 )
     sand.powerof();
   else if( strcmp(argv[2] , "-i") == 0 ){
     fout<<" Identity: ";
     sand.identity();
  }
  else if( strcmp(argv[2] , "-r") == 0 ){
     fout<<" The recurrent configuration of ";
     sand.print();
     fout<<" is";
     sand.reprec();
   }
   else if( strcmp(argv[2] , "-ri") == 0 ){
     fout<<" Inverse recurrent configuration: ";
     sand.reprecinv();
   }
   else if( strcmp(argv[2] , "-det") == 0 )
     sand.determinant();
   else if( strcmp(argv[2] , "-group") == 0 )
     sand.group();
   else
     fout<<"Error: Not recognized command"<<endl;
   fout.close();
}
else if(argc == 4)
   ofstream foutg;
   char *input_file_name = new char [(strlen(argv[1])+3)];
   strcpy(input_file_name,argv[1]);
   strcat(input_file_name,".gph");
```

```
 foutg.open(input_file_name, ios::trunc);
 int dim = atoi(argv[3]);
 if( strcmp(argv[2] , "-complete") == 0){
   foutg<<dim<<endl;
   if(dim>0){
     for(int i=0; i<dim; i++){
       for(int j=0; j<dim; j++)
         if(i==j)
           foutg<<(dim-1)<<" ";
         else
           foutg<<"-1 ";
       foutg<<endl;
     }
     foutg<<dim;
   }
 }
 else if( strcmp(argv[2] , "-path") == 0 ){
   foutg<<dim<<endl;
   if(dim>0){
     for(int i=0; i<dim; i++){
       for(int j=0; j<dim; j++)
         if(i==j)
           if(i==0 || i==dim-1)
             foutg<<"1 ";
           else
             foutg<<"2 ";
         else
           if(i==(j+1) || i==(j-1))
             foutg<<"-1 ";
           else
             foutg<<"0 ";
       foutg<<endl;
     }
     foutg<<dim;
   }
 }
 else if( strcmp(argv[2] , "-cycle") == 0 ){
   foutg<<dim<<endl;
   if(dim>0){
     for(int i=0; i<dim; i++){
       for(int j=0; j<dim; j++)
         if(i==j)
           foutg<<"2 ";
         else
           if(i==(j+1) || i==(j-1))
             foutg<<"-1 ";
           else
             foutg<<"0 ";
       foutg<<endl;
     }
     foutg<<dim;
```

```
    }
  }
  else
    fout<<"Error: Not recognized command"<<endl;
   foutg.close();
}
return 0;
}
```

Bibliography

[1] Alfaro, C., Valencia, C., A combinatorial description of the sandpile group of the cone of Q_d, in preparation.
[2] Bai H., On the critical group of the n-cube, Linear Algebra and its Applications, 369 (2003), 251–261.
[3] Bak P., Tang C., and Wiesenfeld K., Self-organized criticality: an explanation of the 1/f noise. Physics Review Letters, 59 (4) : 381–384, 1987.
[4] Bak P., Tang C. and Wiesenfeld K., Self-Organized Criticality, Phys. Rev. A 38 (1988), 364–374.
[5] Bertsimas D. and Tsitsiklis J. N., Introduction to Linear Optimization, Athena Scientific, Belmont, Massachusetts, 1997.
[6] Biggs N. L., Algebraic Graph Theory, second ed., Cambridge Mathematical Library, Cambridge University Press, Cambridge, 1993.
[7] Biggs N. L., Algebraic potential theory on graphs. Bull. London Math. Soc., 29 : 641–682, 1997.
[8] Biggs N. L., Chip-firing and the critical group of a graph, J. Algebraic Combin. 9 (1999) 25–45.
[9] Björner A., Lovász L., Shor P., Chip-firing games on graphs, European J. Combin, 12 (1991), no. 4, 283–291.
[10] Björner A. and Lovász L., Chip-firing games on directed graphs, Journal of Algebraic Combinatorics, 1: 304–328, 1992.
[11] Bollobás B., Graph Theory, GTM 63, Springer-Verlag, New York, 1979.
[12] Bondy J. A., Morty U. S. R., Graph Theory, GTM 244, Springer-Verlag, New York, 2008.
[13] Borgne Y. L. and Rossin D., On the identity of the sandpile group, Discrete Mathematics, 256 3 (2002), 775–790.
[14] Chen S. and Ye S. K., Critical groups for homeomorphism classes of graphs, Discrete Mathematics, 309 1, (2009), 255–258.
[15] Chen P., Hou Y., On the critical group of $P_4 \times C_n$, European Journal of Combinatorics 29 (2008) 532–534.
[16] Chen P., Hou Y., Woo C., On the critical group of the Möbius ladder graph, Australas. J. Combin. 36 (2006) 133–142.
[17] Chen W. and Schedler T., Concrete and abstract structure of the sandpile group for thick trees with loops, preprint, ArXiv:math/0701381v1.
[18] Christianson H., Reiner V., The critical group of a threshold graph, Linear Algebra Appl. 349 (2002) 233–244.
[19] Cori R., Rossin D., On the sandpile group of a graph, preprint, http://www.labri.u-bordeaux.fr/Perso/ cori/Articles/sable.ps
[20] Cori R., Rossin D., On the sandpile group of dual graphs, Eur. J. Combin. 21 (2000) 447–459.
[21] Cori R., Rossin D. and Salvy B., Polynomial ideals for sandpiles and their Gröbner bases, Theoretical Computer Science, 276 1-2 (2002) 1–15.
[22] Dartois A., Fiorenzi F., Francini P., Sandpile group on the graph D_n of the dihedral group, European J. Combin., 24 (2003) 815–824.
[23] Dhar D., Ruelle P., Sen S., and Verma D., Algebraic aspects of sandpile models. Journal of Physics, A 28 : 805–831, 1995.
[24] Dhar D., Self-organized critical state of sandpile automaton models. Phys. Rev. Lett. 64 (1990), no. 14, 1613–1616.
[25] Diestel R., Graph Theory, GTM 173, Springer-Verlag, New York, 2005.
[26] Godsil C., Royle G., Algebraic Graph Theory, GTM 207, Springer-Verlag, New York, 2001.
[27] Hell, P. and Nesetril, J., Graphs and homomorphims, Oxford lecture series in mathematics and its applications 28, Oxford University Press 2004, New York
[28] Hou Y., Woo C., Chen P., On the sandpile group of the square of a cycle C_n^2, Linear Algebra Appl. 418 (2006) 457–467.
[29] Hou Y., Lei T., Woo C., On the sandpile group of the graph $\mathcal{K}_3 \times C_n$, Linear Algebra and its Applications 428 (2008) 1886–1898.
[30] Jacobson B., Niedermaier A., Reiner V., Critical groups for complete multipartite graphs and Cartesian products of complete graphs, J. Graph Theory 44 (2003) 231–250.
[31] Jacobson N., Basic Algebra I, Second Edition, W. H. Freeman and Company, New York, 1985.
[32] Levine L., The sandpile group of a tree, European Journal of Combinatorics, 30 4 (2009), 1026–1035.
[33] Liang H., Pan Y. and Wang J., The critical group of $\mathcal{K}_m \times P_n$, Linear Algebra and its Applications, 428 11-12 (2008) 2723–2729.
[34] Lorenzini D.J., Arithmetical graphs, Math. Ann. 285 (1989) 481–501.
[35] Lorenzini D.J., A finite group attached to the Laplacian of a graph, Discrete Math. 91 (1991) 277–282.
[36] Merino C., The chip-firing game, Discrete Mathematics, 302 1-3 (2005),188–210.
[37] Merris R., Unimodular equivalence of graphs, Linear Algebra Appl. 173 (1992) 181–189.
[38] Shen J. and Hou Y., On the sandpile group of $3 \times n$ twisted bracelets, Linear Algebra and its Applications, 429 8-9, 16 (2008), 1894–1904.

[39] Speer E. R., Asymmetric Abelian sandpiles models, Journal of statistical physics, 71 1-2 (1993) 61-74.
[40] Toumpakari E., On the sandpile group of regular trees, Eur. J. Combin. 28 (2007) 822–842.
[41] Wagner D. G., The critical group of a directed graph, preprint, ArXiv:math/0010241v1

Index