Matroids, the Tutte Polynomial and the Chip Firing Game

Criel Merino
Somerville College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Michaelmas Term 1999

Contents

List of Figures

Chapter 1

Forests, Colourings and Acyclic Orientations of the Square Lattice

There is no known polynomial time algorithm which generates a random forest or counts forests or acyclic orientations in general graphs. On the other hand, there is no technical reason why such algorithms should not exist. These are key questions in the theory of approximately evaluating the Tutte polynomial which in turn contains several other specialisations of interest to statistical physics, such as the Ising, Potts, and random cluster models.

In this chapter we shall be considering three closely related counting problems on the square lattice.

These are the enumeration of k-colourings, acyclic orientations and forests. Apart from their intrinsic interest, our original motivation was the following. They are all key points of the Tutte plane and exact evaluation has been proved to be #P-hard even for planar bipartite graphs, see Vertigan and Welsh [**?**]. It is also the case that the square lattice is the fundamental separation point between the classes of graphs of bounded tree width and unbounded tree width, in the sense of Robertson and Seymour [**?**].

Thus, in a very technical sense, it can be regarded as the separation point between hard and easy problems. All evaluations of the Tutte polynomial are known to be in polynomial time for graphs of bounded tree width (see Andrzejak [**?**] and Noble [**?**]).

Not only are the quantities considered here hard to determine exactly but most of them have resisted all attempts to find good approximation schemes. There are a few exceptions. From Annan [**?**], see also [**?**], we know that the number of forests in a graph G has a fully polynomial randomised approximation scheme (fpras) whenever G is sufficiently dense, however the square lattice is not dense.

Also, Jerrum [**?**] and Salas and Sokal [**?**], using different Markov chain approaches, have obtained an fpras for the number of k-colourings of a graph G of maximum degree Δ whenever k is at least 2Δ.

We obtain some asymptotic counting results about the number of forests and the number of acyclic orientations on the $n \times n$ section of the square lattice together with some properties of the structure of the random forest.

Most of this material is a joint work with D.J.A. Welsh and it has been already published in [**?**]. Most recently, we have managed to improve our previous numerical results by using a completely different method. Here we include both methods, and their numerical results, as there is evidence [**?**] that there exists a computer program that can improve the numerical results of the first method but not necessarily the results of the second one. Also, better use of computer memory could improve our computer programs, and therefore our numerical results for the second method, but without given numerical results for the first method.

This is a long chapter, it includes two different methods for the same problem, but we decided not to divide it because, although different, the methods are closely related.

Section 1.1 gives the formalisation of these enumeration problems. Section 1.2 is devoted to known asymptotic enumeration results for the number of colourings of the square lattice, that gives some insight into the current research on the subject. The problem of the asymptotic behaviour of the number of acyclic orientations is treated in Section 1.3. Then, Section 1.4 does the same for the number of forests. Section 1.5 is about random forests on the square lattice. In Section 1.6 we use the transfer matrix method to help us with the computation of specific values of the Tutte polynomial of the $n \times n$ section of the square lattice. The two final sections include some improvements to the previous results of the chapter by using this approach of the transfer-matrix method.

1.1 Computational Complexity and the Square Lattice

The vertices of the *square lattice* L_n can be identified with the set of ordered pairs $\{(i,j) \in \mathbb{N}^2 |\ 0 \leq i,j \leq n-1\}$. There is an edge between the vertices (i,j) and (i',j') if $|i-i'|+|j-j'|=1$. Thus, the graph L_n has n^2 vertices and $2n(n-1)$ edges. See figure 1.1 for an example.

Figure 1.1: The square lattice L_{11}.

In this chapter we analyse the behaviour of the Tutte polynomial of the square lattice. It is easy to show that for any fixed (x, y), $T(L_n; x, y)$ is $O(\theta^{n^2})$ for a suitable $\theta = \theta(x, y)$. We are particularly interested in the limit of the sequence

$$\{(T(L_n, i, j))^{1/n^2}\}$$

for certain values of the integers i and j.

We first note that we already know from the results of Grimmett [**?**, **?**] and Biggs [**?**], that except in certain special cases, these limits exist. The proof is a straightforward argument using submultiplicativity. In [**?**, **?**], the author deals with the (Whitney) rank generating function of L_n, denoted by $W(L_n; x, y)$ but this is just a reparametrisation of the Tutte polynomial by the identity

$$W(G; x - 1, y - 1) = T(G; x, y),$$

which holds for any graph G.

We highlight in particular the special hyperbola $(x - 1)(y - 1) = 2$ of the Tutte plane where T is a reparametrisation of the partition function of the Ising model with zero external magnetic field. Thus on the positive branch of this hyperbola, which corresponds to the ferromagnetic version of the Ising model, convergence is to the classical limit of the Onsager solution, see for example [**?**].

When discussing complexity questions about the square lattice care must be taken over the exact formulation of the question.

In many or perhaps almost all cases, a statement such as "Problem Π is NP-hard for the square lattice" should be read as "Problem Π is NP-hard for the class of grid graphs". A graph is a *grid graph* if it can be embedded in the square lattice or more precisely, if it is isomorphic to a subgraph of the square lattice.

In what follows, we consider that the grid graphs are given already embedded in the square lattice. Here we have to notice that the problem of given a graph G, decide if G is a grid graph is NP-complete. This is a consequence of the following theorem in [**?**].

Theorem 1.1.1. *Given a tree T with vertices of degree at most 4, determining whether or not T has a VLSI layout with unit-length edges is NP-complete.*

The relation between a VLSI layout of G and an embedding of G in the square lattice is also explained in [**?**], from where we take it. A VLSI layout for a graph G is formally specified by mapping vertices of G to vertices of the square lattice L_n, for some n, together with an (incidence-preserving) assignment of edges of G to paths in the grid. The paths are restricted to follow edges in L_n and may not overlap one another, although they may cross at a point. Finally, a path may not cross a vertex to which the corresponding edge is not incident. The VLSI layout has unit-length if every edge of G goes to a path of length 1 in L_n.

We know of no problem which is NP-hard for L_n. Indeed it is not easy to see how one can be.

The square lattice is a planar bipartite graph, so several of the well known computational decision problems are "easy", either for being planar or bipartite. The following are some of the polynomial time decidable problems for grid graphs which are NP-hard in general. All are mentioned in [**?**]. They include STABLE SET, CLIQUE, CLIQUE PARTITION, CHROMATIC NUMBER, CHROMATIC INDEX and MAX CUT. Also, GRAPH ISOMORPHISM, which is a long-standing open problem for general graphs, is polynomial time computable for the grid graphs.

We know of just one problem that is polynomial time decidable for the square lattice, but is NP-complete for planar and bipartite graphs in general. This is the problem PARTITION INTO 4 CYCLES, see [**?**, **?**].

However, we should point out that there are a few decision problems which are NP-complete for grid graphs, three of them are HAMILTON CIRCUIT [**?**], DOMINATION SET [**?**] and STEINER TREE [**?**]. These three problems are certainly easy for L_n so our arguments in support of L_n being a threshold are perhaps not totally convincing.

Nevertheless, we do believe that because of its importance in statistical physics a better understanding of why so many problems on L_n are so hard is of fundamental interest.

1.2 The Number of Colourings

Let $\chi(L_n;k)$ be the number of k-colourings of the square lattice L_n. Clearly the number of 2-colourings of L_n is 2. Hence

$$\lim_{n\to\infty}(\chi(L_n;2))^{1/n^2}=1.$$

For $k>2$ the problem becomes much harder and exact results are not known. By starting at the bottom corner and colouring the vertices in order from left to right and bottom to top we see that the number of k-colourings of L_n must lie between $k(k-1)^{2n-2}(k-2)^{(n-1)^2}$ and $k(k-1)^{n^2-1}$. This argument gives

$$k-2\leq(\chi(L_n;k))^{1/n^2}\leq k-1.$$

For the rest of this section we assume $n>2$ to avoid trivialities. Let L_n^T be the graph obtained from the square lattice L_{n+1} by identifying the boundary vertices $(i,0)$ and (i,n), for $0\leq i\leq n$, and the vertices $(0,j)$ and (n,j), for $0\leq j\leq n$, and deleting any parallel edges. So L_n^T can be embedded in the torus and every vertex has degree four. This is often referred to as the toroidal square lattice. Let $\chi(L_n^T;k)$ be the number of k-colourings of L_n^T. It is known [**?**] that for a fixed integer $k\geq 3$ the limits of the sequences $\{(\chi(L_n^T;k))^{1/n^2}\}$ and $\{(\chi(L_n;k))^{1/n^2}\}$ are equal and we call this limit $\hat{\chi}(k)$.

In a tour de force, Lieb [**?**] showed that the number of ice configurations of L_n^T to the power $1/n^2$ is asymptotically $(4/3)^{3/2}$.

An *ice configuration* is an orientation of the edges so that at each vertex exactly two edges are directed in and two out. It is easy to see that this counts exactly the number of nowhere zero 3-flows on L_n^T. If we now assume L_n^T is self dual, which is not strictly true because it is nonplanar, it is generally accepted [**?**, pp. 56] that the result of Lieb implies that

$$\hat{\chi}(3)=(4/3)^{3/2}\approx 1.539600718.$$

Biggs and Meredith in [**?**] obtained the estimate

$$\hat{\chi}(k)\sim\frac{1}{2}(k-3+\sqrt{k^2-2k+5}),$$

where $g(k) \sim h(k)$ has its usual meaning that the limit as $k \to \infty$ of $g(k)/h(k)$ is 1.

Lower and upper bounds for $\hat{\chi}(k)$ were given by Biggs in [**?**]. He used the transfer-matrix technique to obtain

$$\frac{k^2-3k+3}{k-1} \leq \hat{\chi}(k) \leq \frac{1}{2}(k-2+\sqrt{k^2-4k+8}).$$

In [**?**], Nagel used an induced subgraph expansion for the chromatic polynomial to obtain the first terms of a power series that converges to $\hat{\chi}(k)$.

Kim and Enting [**?**] gave a more accurate approximation of the same power series by combining an expansion of $\hat{\chi}(k)$ due to de Neef and the transfer-matrix technique. Their approximation is

$$\frac{(k-1)^2}{k}[1+x^3+x^7+3x^8+4x^9+3x^{10}+3x^{11}+11x^{12}+24x^{13}+ \\ 8x^{14}-91x^{15}-261x^{16}-290x^{17}+254x^{18}+\cdots]$$

where $x = (k-1)^{-1}$. Numerical values obtained by using this approximation are given in the following table.

Value of k	$\hat{\chi}(k)$
3	1.53978474935
4	2.33605332564
5	3.25040489910
6	4.20009565692
7	5.16669659737
8	6.14286845544
9	7.12500489812
10	8.11111346089.

More recently Bakaev and Kabanovich [**?**] have extended the series of Kim and Enting from 18 terms to 36.

1.3 Acyclic Orientations

There are intimate connections between colourings and orientations of the edge set of a graph. The most well known result is that of Stanley [**?**] who proved that if $\alpha(G)$ is the number of acyclic orientations of G then

$$\alpha(G) = T(G;2,0) = (-1)^{|V(G)|}\chi(G;-1).$$

Less well known is the connection between colourings and the number of acyclic orientations of G in which there is exactly one given source. We denote this by $\alpha_0(G)$, and

$$\alpha_0(G) = T(G; 1, 0).$$

The proof can be found in [**?**].

Note that an immediate consequence of this is that $\alpha_0(G)$ is independent of the vertex chosen to be the unique source.

Let G be connected. Then, we know that

$$\chi(G; \lambda) = |\lambda T(G; 1 - \lambda, 0)|$$

so that, if

$$\chi(G; \lambda) = a_n\lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1\lambda$$

then

$$\begin{aligned} T(G; 1, 0) &= \lim_{\lambda \to 0} \lambda^{-1}|\chi(G; \lambda)| \\ &= |a_1|. \end{aligned}$$

It follows that

- The number of acyclic orientations with exactly one source is equal to $|a_1|$.
- Computing a_1 is #P-hard, even for bipartite planar graphs.

Let $\alpha(n)$ be the number of acyclic orientations of L_n and $\alpha_0(n)$ be the number of acyclic orientations with (0,0) as the only source.

Consider the following partial orientation of L_n. The edges $\{(i, j),\ (i, j + 1)\}$ are oriented away from (i, j), for $0 \leq i \leq n - 1$ and $0 \leq j \leq n - 2$. Also, orient the edges $\{(i, 0),\ (i + 1, 0)\}$ away from $(i, 0)$, for $0 \leq i \leq n - 2$. For any orientation of the remaining edges we get an acyclic orientation of L_n with (0,0) as the only source. So,

$$2^{(n-1)^2} \leq \alpha_0(n)$$

and thus

$$2 \leq \lim_{n \to \infty} (\alpha_0(n))^{1/n^2}.$$

A better lower bound is the following

Theorem 1.3.1.

$$\lim_{n\to\infty}(\alpha_0(n))^{1/n^2} \geq \frac{7}{3}$$
$$\lim_{n\to\infty}(\alpha(n))^{1/n^2} \geq \frac{22}{7}.$$

Proof. We enumerate the vertices of L_n by assigning to vertex (i,j) the number $ni+j+1$, for $0 \leq i \leq n-1$ and $0 \leq j \leq n-1$.

Let $L_{n,k}$ be $L_n \setminus \{1,\ldots,k\}$ and $L_{n,0} = L_n$. We prove that

$$\alpha_0(L_{n,k-1}) \geq \frac{7}{3}\,\alpha_0(L_{n,k}),$$

for $k < n(n-1)$ and $k \neq n, 2n, \ldots, n(n-2)$.

Once this result is proved, we apply it recursively in the order given to the vertices. Notice that for $k = n, 2n, \ldots, n(n-2)$, $\alpha_0(L_{n,k-1}) = \alpha_0(L_{n,k})$.

Consider an acyclic orientation of $L_{n,k-1}$ with the vertex labelled k as the only source. If we restrict this orientation to $L_{n,k}$, then $k+n$ and $k+1$ are the only possibilities for being a source in this restriction. Also, we know that this orientation of $L_{n,k}$ must have at least one source, as it is an acyclic orientation. So, we can partition the set of acyclic orientations of $L_{n,k-1}$ with k as the only source into three sets. The ones that restricted to $L_{n,k}$ have $k+n$ as the only source, the ones with $k+1$ as the only source and the ones with $k+n$ and $k+1$ as the only two sources. Call these sets R, S and P, respectively.

As we noted before, the number of acyclic orientations with exactly one source is independent of the vertex chosen as a source, therefore $|R| = |S|$.

Now, the set R can be partitioned further as follows. The orientations where the vertex $k+1$ is a sink, the ones that have the directed path $k+n+1 \to k+1 \to k+2$ and the ones that have the directed path $k+2 \to k+1 \to k+n+1$. Call these sets U, V and W respectively.

Clearly, any element of U can be transformed into a unique element of P, by reversing the orientation of the edges $(k+n+1, k+1)$ and $(k+1, k+2)$. Thus $|P| \geq |U|$. By the same argument $|P| \geq |V|$ and $|P| \geq |W|$. We conclude that $|P| \geq \frac{1}{3}|R|$ and the claim follows.

Now, consider the graph $\hat{L}_{n,k}$, which is obtained from $L_{n,k}$ by adding an apex a, that is, $V(\hat{L}_{n,k}) = V(L_{n,k}) \cup \{a\}$ and $E(\hat{L}_{n,k}) = E(L_{n,k}) \cup \{(a,v) | v \in V(L_{n,k})\}$. There is a clear bijection between acyclic orientations of $L_n = L_{n,0}$ and acyclic orientations of $\hat{L}_{n,0}$ with a as the only source.

We prove that

$$\alpha_0(\hat{L}_{n,k-1}) \geq \frac{22}{7}\ \alpha_0(\hat{L}_{n,k}),$$

for $k < n(n-1)$ and $k \neq n, 2n, \ldots, n(n-2)$.

As before, we apply the result recursively to the vertices $V(\hat{L}_{n,k}) \setminus \{a\}$ in the induced order given by $V(L_{n,k})$. It is important to note that here we are using again that the number of acyclic orientations with exactly one source is independent of the vertex chosen as a source.

Again, we partition the set of acyclic orientations of $\hat{L}_{n,k-1}$ with k as the only source. This time we have four sets. The ones that restricted to $\hat{L}_{n,k}$ have either $k+n$, $k+1$ or a as the only source, and the ones that have $k+n$ and $k+1$ as the only two sources. Call these sets R_1, R_2, R_3 and Q.

We have that $|R_1| = |R_2| = |R_3|$ because the number of acyclic orientations with exactly one source is independent of the vertex chosen as a source. Similarly as the previous argument, we partitione the set R_1 into 7 sets obtained by considering the possible orientations given to the edges $(k{+}1, a)$, $(k{+}1, k{+}n{+}1)$ and $(k{+}1, k{+}2)$. If we now call these sets $R_{1,i}$, $1 \leq i \leq 7$, we get that for a fixed i, any element of $R_{1,i}$ can be transformed into a unique element of Q, by reversing the orientation of the appropriate edges. Thus $|Q| \geq |R_{1,i}|$, $1 \leq i \leq 7$. We conclude that $|Q| \geq \frac{1}{7}|R_1|$. Summing these quantities gives the required factor of $\frac{22}{7}|R_1|$ and completes the proof. □

As far as upper bounds are concerned, there is the trivial upper bound of $\alpha(n) \leq 4^{n^2}$ obtained from all possible orientations of the edges. One rigorous improvement which we have on this is derived from the following theorem.

Theorem 1.3.2. *For any connected graph G with at least one cycle, $\alpha_0(G)$ is strictly less than the number of spanning trees and $\alpha(G)$ is strictly less than the number of forests.*

Proof. A basic property of T is that $T(G; x, y)$ can be written as $\sum t_{ij} x^i y^j$ with all coefficients t_{ij} nonnegative integers. The coefficient t_{ij} counts the number of spanning trees with *internal activity* i and *external activity* j, see [**?**] for definition and a proof. Provided G is connected with at least one cycle we can find spanning trees with positive external activity and this implies a strictly positive t_{ij}, for some $0 < j \leq E(G) - V(G) + 1$. Since $\alpha_0 = T(G; 1, 0) < T(G; 1, 1)$ and $\alpha(G) = T(G; 2, 0) < T(G; 2, 1)$ which equals the number of forests, the result follows. □

A better upper bound for $\alpha(n)$ is the following.

Theorem 1.3.3. *The number $\alpha(n)$ satisfies*

$$\lim_{n\to\infty}(\alpha(n))^{1/n^2} \leq 3.62330970816373\ldots.$$

We will prove this together with a similar result for the number of forests in our proof of Theorem 1.4.1 in the next section.

We now turn to the generation and enumeration of forests in L_n.

1.4 Forests

As far as complexity issues are concerned, counting spanning trees is no problem. It is one of the "easy points" of Theorem 2 in [**?**] and is easy for any graph by using the Kirchhoff determinantal formula.

However, as far as this chapter is concerned an alternative approach gives us a better idea of the asymptotics.

We use $t(n)$ to denote the number of spanning trees of L_n. Let a_n be the number of 1-factors or perfect matchings of L_{2n}. It is shown in [**?**] that

$$\begin{aligned}\lim_{n\to\infty}\frac{\ln a_n}{n^2} &= \frac{4}{\pi^2}\int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}}\ln(4\cos^2 x+4\cos^2 y)dxdy\\ &= c\approx 1.166243696.\end{aligned}$$

We obtained the approximated value of c by using `Mathematica` version 2.2.

Now, let b_n be the number of 1-factors in the graph L'_n, which is obtained from the $(2n-1)\times(2n-1)$ square lattice by taking out one corner vertex, that is, $L'_n = L_{2n-1}\setminus(0,0)$. In [**?**], a bijection has been established between the 1-factors of L'_n and the spanning trees of L_n. Since $b_n \leq a_n \leq b_{n+1}$ we get that $\lim_{n\to\infty} a_n^{1/n^2} = \lim_{n\to\infty} b_n^{1/n^2}$ and

$$\lim_{n\to\infty}(t(n))^{1/n^2} = e^c \approx 3.209912556. \tag{1.4.1}$$

The number of spanning forests of the square lattice L_n, which we denote by $f(n)$, seems an elusive quantity to approximate accurately. We found it difficult to do better than the trivial upper bound until Colin McDiarmid [**?**] showed by a probabilistic argument that there exists $\eta > 0$ such that

$$\lim_{n\to\infty}(f(n))^{1/n^2} \leq 4-\eta$$

where η was some unknown constant. An extension of his argument allows us to obtain the following explicit bound for $f(n)$ and similar techniques give us the proof of Theorem 1.3.3.

Theorem 1.4.1.

$$\lim_{n\to\infty}(f(n))^{1/n^2} \leq 3.78649853538319\ldots.$$

Proof. Let k be a fixed positive integer. From the lattice of side $2^p k$ we select the 4^p small lattices of sizes $k \times k$ whose bottom left hand corners are the points (ik, jk) for $0 \leq i, j \leq 2^p - 1$. Call $\mathcal{C}$ this set of subgraphs of $L_{2^p k}$.

We construct a set of subgraphs of $L_{2^p k}$ that properly contains all its forests. Choose for every subgraph of $\mathcal{C}$ a spanning forest of this subgraph and then choose any subset of the remaining $2(2^p)(2^p-1)k$ edges in $L_{2^p k}$, that is any subset of $E(L_{2^p k})\setminus \cup_{C\in\mathcal{C}}E(C)$. Any forest of $L_{2^p k}$ can be obtained in this way, we are, however, including some subgraphs that are not acyclic, so this is an overcount, but nevertheless

$$f(2^p k) \leq 2^{2(2^p)(2^p-1)k}(f(k))^{4^p}.$$

Hence

$$(f(2^p k))^{1/4^p k^2} \leq 2^{2/k}\left(\frac{1}{2}\right)^{2/2^p k}(f(k))^{1/k^2},$$

and taking the limit as $p \to \infty$ we get

$$\lim_{n\to\infty}(f(n))^{1/n^2} = \lim_{p\to\infty}(f(2^p k))^{1/4^p k^2} \leq 2^{2/k}(f(k))^{1/k^2}.$$

In Subsection 1.6.1, we obtain the value for $f(7) = 1.31592738937415 \times 10^{24}$, by computing $2^{2/7}(f(7))^{1/49}$ and using this we get the result. □

Proof of Theorem 1.3.3. We use the same construction as in the above proof to get the class $\mathcal{C}$. Then we choose for every subgraph of $\mathcal{C}$ an acyclic orientation of this subgraph and we orient the remaining edges in $L_{2^p k}$ arbitrarily. Using the same argument as before, we obtain

$$\lim_{n\to\infty}(\alpha(n))^{1/n^2} \leq 2^{2/k}(\alpha(k))^{1/k^2}.$$

For $k = 7$, by substituting the value $\alpha(7) = 1.519663682749935 \times 10^{23}$, that we will obtain in Subsection 1.6.1, in the left hand expression, we get the result. □

1.5 The random forest

In contrast with our very limited success in counting forests, we have made some progress on the related problem of describing the random forest in L_n. Jerrum [**?**] reports on the attempts (with V. Gore) to use the exact generation methods of Propp and Wilson [**?**] to generate a random forest in L_{30}. This appeared to be

the computational limit. First we present one somewhat surprising property of the random forest in general graphs.

Given a subset $A \subseteq E(G)$ we denote by $\sigma(A)$ the *closure* of A in the cycle matroid of G. In other words, $\sigma(A)$ consists of the unique maximal superset of A which contains A and has the same rank as A. Alternatively, if we consider the subgraph $G : A = (V(G), A)$ of G, and let $C_1, C_2, \ldots, C_k$ be its connected components, including isolated vertices, then edge $e \in \sigma(A)$ if and only if $e \in A$ or $e \notin A$ but both its endpoints are in the same connected component of $G : A$.

Theorem 1.5.1. *For any graph G, if F denotes the edge set of a forest chosen uniformly at random from all forests of G, then, if $\langle\ \rangle$ denotes expected value,*

$$\langle |F| \rangle + \langle |\sigma(F)| \rangle = |E(G)|.$$

Proof. Theorem 5.1 of [**?**] gives the following identity about the Tutte polynomial of any loop free matroid $M = (E, r)$,

$$|E|\ T(M; x, y) = \sum_{A \subseteq E} (x|\sigma(A)| + y(1-x)|\sigma(A) \setminus A|)\, t(A)$$

where $t(A) = (x-1)^{r(E)-r(A)}(y-1)^{|A|-r(A)}$. Now take M to be the cycle matroid of the graph G and put $x = 2$, $y = 1$. This gives

$$|E(G)|f(G) = \sum_{A \in \mathcal{F}(G)} (2|\sigma(A)| - |\sigma(A)| + |A|)$$

where $\mathcal{F}(G)$ is $\{A \in E(G) | A \text{ is a forest}\}$, and $f(G) = |\mathcal{F}(G)|$. Dividing the last equation by $f(G)$, we get the result. □

An alternative way of looking at this identity is the following. Consider a Markov chain X_t with set of states the forests of G and suppose the transition probabilities are governed by the following mechanism. At each epoch of time an edge e is picked uniformly at random from $E(G)$. If X_t contains e then $X_{t+1} = X_t \setminus e$, if $e \notin X_t$ and $X_t \cup \{e\}$ is a forest then $X_{t+1} = X_t \cup \{e\}$. It is easy to see that X_t converges to a uniform distribution on the set of forests of G. In equilibrium,

$$\Pr\{|X_{t+1}| = |X_t| + 1\} = \frac{|E| - |\sigma(X_t)|}{|E|}$$
$$\Pr\{|X_{t+1}| = |X_t| - 1\} = \frac{|X_t|}{|E|}.$$

Thus, intuitively, at equilibrium these probabilities should be equal. A formal proof using this idea was given by Steve Noble in his D.Phil. thesis [**?**].

Corollary 1.5.2. *The expected number of connected components of a random forest in L_n is at least n.*

Proof. Consider the case where $G = L_n$. Let F_n be the random forest and suppose it has k_n components so that $|F_n| = n^2 - k_n$. Also $|\sigma(F_n)| \geq |F_n|$. So applying the identity, we get

$$2n^2 - 2\langle k_n \rangle \leq |E(L_n)| = 2n^2 - 2n.$$

Thus $\langle k_n \rangle \geq n$ as required. □

However, asymptotically this is a very weak result because of the following.

Theorem 1.5.3. *The expected number $I(n)$ of isolated vertices of a random forest in L_n satisfies*

$$I(n) \geq (n^2 + 4n + 4)/16.$$

Proof. For a graph G let $G \setminus v$ denote the subgraph obtained by deleting v and all edges incident with v. Consider the probability $p_0(v)$ that in a random forest of G, v has degree zero.

But, given any forest of $G \setminus v$ it may be augmented to a forest of G in at most $2^{d(v)}$ ways. Hence,

$$p_0(v) = \frac{f(G \setminus v)}{f(G)} \geq \frac{1}{2^{d(v)}},$$

where $d(v)$ denotes the degree of v in G. Since $I(n) = \sum_{v \in L_n} p_0(v)$ the result follows. □

It is almost immediate that the probability $p_1(v)$ that in the random forest of G, vertex v has degree 1 is given by $p_1(v) = d(v)p_0(v)$. Applying this to L_n gives.

Corollary 1.5.4. *The expected number of degree 1 vertices of a random forest in L_n is at least $(n^2 + 2n)/4$.*

1.6 A transfer-matrix approach

We now introduce the $n \times m$ *lattice* $L_{n,m}$. The vertices of $L_{n,m}$ can be identified with the set of ordered pairs $\{(i, j) \in \mathbb{N}^2 |\ 0 \leq i \leq n-1, 0 \leq j \leq m-1\}$. And there is an edge between the vertices (i, j) and (i', j') if $|i - i'| + |j - j'| = 1$. Thus, the graph $L_{n,m}$ has nm vertices and $m(n-1)+n(m-1)$ edges. Note that $L_{n,n} = L_n$. Figure 1.2 shows $L_{15,11}$.

We also define the *m-comb graph*, P'_m to be the subgraph of $L_{2,m}$ with the same set of vertices and edges

$$\{(1,j) \sim (2,j) |\, 0 \leq j \leq m-1\} \cup \{(2,j) \sim (2,j+1) |\, 0 \leq j \leq m-2\}.$$

Figure 1.2 also shows the graph P'_{11}.

Any subset of edges in $L_{n,m}$ can be seen as constructed from a subset of edges in $L_{n-1,m}$ and a subset of edges from the m-comb graph, we formalise this later, but the intuitive idea shall be clear from Figure 1.2. This characteristic of $L_{n,m}$ allows us to compute the Tutte polynomial of $L_{n,m}$ by means of the *transfer-matrix method*.

Figure 1.2: The lattice $L_{15,11}$ and the 11-comb graph P'_{11}.

The theoretical background of the transfer-matrix method, taken from [**?**], is described below.

A *directed graph* or *digraph* $\vec{G}$ is a triple (V, E, ϕ), where $V = \{v_1, \ldots, v_p\}$ is a set of vertices, E is a finite set of *directed edges* or *arcs*, and ϕ is a map from E to $V \times V$. If $\phi(e) = (u, v)$, then e is called an edge from u to v, with *initial* vertex u and *final* vertex v. A *directed walk* Γ in $\vec{G}$ of *length* n from u to v is a sequence $e_1, \ldots, e_n$ of n edges such that the final vertex of e_i is the initial vertex of e_{i+1}, for $1 \leq i \leq n-1$.

Now let $w : E \to R$ be a *weight function* on E with values in some commutative ring R. If $\Gamma = e_1, \ldots, e_n$ is a walk, then the weight of Γ is defined by $w(\Gamma) = w(e_1) \cdots w(e_n)$. For $1 \leq i, j \leq p$ and $n \in \mathbb{N}$, we define

$$A_{i,j}(n) = \sum_{\Gamma} w(\Gamma),$$

where the sum is over all walks Γ in $\vec{G}$ of length n from v_i to v_j. In particular, $A_{i,j}(0) = \delta_{ij}$. The fundamental problem treated by the transfer-matrix method is the

evaluation of $A_{i,j}(n)$. The idea is to interpret $A_{i,j}(n)$ as an entry in a certain matrix. Define a $p \times p$ matrix $T = (T_{i,j})$ by

$$T_{i,j} = \sum_e w(e),$$

where the sum is over all edges e satisfying that its initial vertex is v_i and its final vertex is v_j. In other words, $T_{i,j} = A_{i,j}(1)$. The matrix T is called the *adjacency matrix* of $\vec{G}$, with respect to the weight function w.

Theorem 1.6.1. *Let $n \in \mathbb{N}$. Then the (i,j)-entry of T^n is equal to $A_{i,j}(n)$. (Here we define $A^0 = I_p$ even if A is not invertible, where I_p is the identity matrix).*

Proof. See [**?**]. □

Now, we formalise the intuitive idea at the beginning of the section. Here we denote by $E_{n,m}$ the edge-set of $L_{n,m}$. We also use E_m^k, $k \geq 2$, for the set of edges

$$\{(k-2,i) \sim (k-1,i) |\, 0 \leq i \leq m-1\} \cup \{(k-1,i) \sim (k-1,i-1) |\, 1 \leq i \leq m-1\}$$

of $L_{n,m}$, $n \geq k$. In the case $k = 1$, we define $E_m^1 = E_{1,m}$, that is the edges of a path of length m. By our definition of $L_{n,m}$, we have that $E_{n-1,m} \subseteq E_{n,m}$, and also $E_m^k \subseteq E_{n,m}$, for all $k \leq n$.

The *right-row decomposition* of a set of edges A in $L_{n,m}$ is the set $\{A_i\}_{i=0}^n$ of subsets of edges of $L_{n,m}$, where $A_i = A \cap E_{i,m}$, $1 \leq i \leq n$, and $A_0 = \emptyset$. In particular, $A_n = A$. The *block decomposition* of A is the partition of the set A given by $\{B_i\}_{i=1}^n$, where $B_i = A \cap E_m^i = A_i \setminus A_{i-1}$, for $1 \leq i \leq n$.

A way to compute the Tutte polynomial of $L_{n,m}$ is to compute for every subset of edges $A \subseteq E_{n,m}$, the term

$$(u-1)^{r(E(L_{n,m}))-r(A)}(v-1)^{|A|-r(A)},$$

and then sum over all possible subsets A. First we make the change of variable $u - 1 = x$ and $v - 1 = y$, and we also factor the term $x^{r(E(L_{n,m}))}$. So, we want to compute, for every set $A \subseteq E_{n,m}$,

$$x^{r(E(L_{n,m}))}(x^{-r(A)}y^{|A|-r(A)}). \tag{1.6.1}$$

Here is where our decompositions become useful.

Lemma 1.6.2. *Let A be a subset of edges of $L_{n,m}$, then*

$$x^{-r(A)}y^{|A|-r(A)} = x^{-r(B_1)}y^{|B_1|-r(B_1)}\prod_{i=1}^{n-1} x^{-r(A_{i+1})+r(A_i)}y^{|B_{i+1}|-r(A_{i+1})+r(A_i)}, \qquad (1.6.2)$$

where $\{A_i\}_{i=0}^n$ is the right-row decomposition of A and $\{B_i\}_{i=1}^n$ is the block decomposition of A.

Proof. Equation 1.6.2 is a consequence of the following:

$$\sum_{i=1}^{n-1}(-r(A_{i+1}) + r(A_i)) = -r(A_n) + r(A_1) = -r(A_n) + r(B_1),$$

as $B_1 = A_1$. Thus, the exponent of x in the right-hand side of equation 1.6.2 is $-r(B_1) - r(A_n) + r(B_1) = -r(A)$, as $A_n = A$.

Also,

$$\begin{aligned}\sum_{i=1}^{n-1}(|B_{i+1}| - r(A_{i+1}) + r(A_i)) &= (\sum_{i=1}^{n-1}|B_{i+1}|) - r(A_n) + r(B_1)\\ &= (\sum_{i=1}^{n}|B_i|) - |B_1| - r(A_n) + r(B_1)\\ &= |A| - |B_1| - r(A_n) + r(B_1).\end{aligned}$$

The reason for the equality $\sum_{i=1}^n(|B_i|) = |A|$ is that $\{B_i\}_{i=1}^n$ is a partition of A. Thus, the exponent of y in the right-hand side of equation 1.6.2 is

$$|B_1| - r(B_1) + |A| - |B_1| - r(A_n) + r(B_1) = |A| - r(A).$$

□

An advantage of equation 1.6.2 is that each factor of the right-hand side expression can be computed using very little information from A, and this will save us having to keep track of all subsets of edges of $L_{n,m}$.

To formalise this, we denote the set of vertices $\{(r,j)|0 \le j \le m-1\}$ of $L_{n,m}$ by $\mathcal{V}_r$, with $1 \le r \le n$. Consider an arbitrary subgraph H of $L_{n,m}$. We enumerate the connected components of H, seen as a spanning graphs of $L_{n,m}$, and we label every vertex with the number of the component it belongs to. For a fixed r, the *r-state* of H is the ordered m-tuple of labels of $\mathcal{V}_r$, $(l_m, \ldots, l_1)$, where l_i is the label of vertex $(r, i-1)$, with $1 \le i \le m$. We denote the r-state of H by $\sigma_r(H)$. We also define the *r-phase* of H, $\psi_r(H)$, to be the set of connected components to which the vertices $\mathcal{V}_r \cup \mathcal{V}_{r-1}$ belong to. For an r-state β, we denote by $|\beta|$, the number of different labels in $\beta = \sigma_r(H)$, that is, the cardinality of the set of components of H to which the vertices $\mathcal{V}_r$ belong.

Lemma 1.6.3. *Let $n, m \geq 2$ be two integers. Let $A \subseteq E_{n,m}$, with right-row decomposition $\{A_i\}_{i=0}^{n}$, then, for $1 \leq i \leq n$,*

$$-r(A_{i+1}) + r(A_i) = |\psi_{i+1}(A_{i+1})| - |\sigma_i(A_i)| - m. \tag{1.6.3}$$

Proof. Let $\{B_i\}_{i=1}^{n}$ be the block decomposition of A. By definition of rank in a graph, for $1 \leq i \leq n$,

$$r(A_i) = (nm - 1) - \omega(A_i),$$

where, for a subset of edges H of $L_{n,m}$, $\omega(H)$ is the number of connected components of H, see Chapter **??**.

But the set of connected components of A_{i+1} can be partitioned into the set of components which the vertices $\mathcal{V}_i \cup \mathcal{V}_{i+1}$ belong to, that we denote by $\mathcal{A}$, and the set of components left, denoted by $\mathcal{B}$. On the other hand, the set of connected components of A_i can be partitioned into the set of components which the vertices $\mathcal{V}_{i-1} \cup \mathcal{V}_i$ belong to, that we denote by $\mathcal{A}'$, and the set of components left, denoted by $\mathcal{B}'$. As $A_{i+1} = A_i \cup B_{i+1}$, the connected components of A_i that do not contain any of the vertices in $\mathcal{V}_i \cup \mathcal{V}_{i+1}$ are not going to change when we put the edges of B_{i+1}, so $|\mathcal{B}'| = |\mathcal{B}|$. Thus,

$$\begin{aligned} -r(A_{i+1}) + r(A_i) &= \omega(A_{i+1}) - \omega(A_i) \\ &= |\mathcal{A}| + |\mathcal{B}| - |\mathcal{A}'| - |\mathcal{B}'| \\ &= |\mathcal{A}| - |\mathcal{A}'|. \end{aligned}$$

But every vertex in $\mathcal{V}_{i+1}$ is a connected component of A_i, so $|\mathcal{A}'| = |\sigma_i(A_i)| + m$, and by definition, $|\mathcal{A}| = |\psi_{i+1}(A_{i+1})|$. So the result follows. □

Observe that, for $1 \leq i \leq n$, the right-hand side of equation 1.6.3 depends only on m, the number of different labels of the i-state of A_i and the $i+1$-phase of A_{i+1}. In fact, the $i+1$-phase of A_{i+1} depends just on the distribution of the labels in the i-state of A_i and the set of edges B_{i+1}.

We are going to make this more precise by considering a *standard form* of an r-state of a subset of edges of $L_{n,m}$. Given an r-state $\beta = (l_m, \ldots, l_1)$ of H, we consider the different labels $\{l_{a_1}, \ldots, l_{a_p}\}$, where we assume a_i is the first ocurrence in β of the label l_{a_i}, that is, for all $n \leq i \leq a_i$, $l_i \neq l_{a_i}$, the standard form of β is the ordered set obtained by relabelling $l_{a_i} \to i$. We denote the standard form of β by $\overline{\beta}$. Note that, as we are just relabelling, $|\beta| = |\overline{\beta}|$.

For a given m, the set

$$\mathcal{S}_m = \{\overline{\sigma_r(H)} |\ H \subseteq E_{n,m}, n \geq 2, r \leq n\}$$

is finite, as it is contained in the set of all ordered m-tuples of the elements $1, \ldots, m$, thus $|\mathcal{S}_m| \leq m^m$. It is known that $|\mathcal{S}_m|$ is the $(m+1)$-Catalan number, c_{m+1}, [**?**]. The Catalan numbers have several equivalent definitions [**?**], here we just note that

$$c_{m+1} = \frac{1}{m+1}\binom{2m}{m}.$$

Now, for an element $\beta = (\beta_1, \ldots, \beta_m) \in \mathcal{S}_m$ and a subset of edges B of P'_m, we construct a graph with vertices

$$\{(1,i)|\, 0 \leq i \leq m-1\} \cup \{(2,i)|\, 0 \leq i \leq m-1\}$$

and edges $B \cup B'$, where B' is the set of edges

$$\{(1,i) \sim (1,j)|\, \beta_i = \beta_j, 1 \leq i,j \leq m-1\}.$$

We denote this graph by $G_{\beta,B}$.

Lemma 1.6.4. *Let $m \geq 2$ be a fixed integer. If $\beta \in \mathcal{S}_m$ and B is a subset of edges of P'_m, then there exists a subset $H \in E_{n,m}$, for some $n \geq 2$ such that :*

1. *if $H_1 = H \cap E_{n-1,m}$, then $\overline{\sigma_{n-1}(H_1)} = \beta$.*
2. *$|\psi_n(H)| = \omega(G_{\beta,B})$.*

Proof. As $\beta \in \mathcal{S}_m$, we know that there exists $H' \in E_{n,m}$, for some n, such that $\overline{\sigma_r(H')} = \beta$, for some $r \leq n$. We take n and r such that $r = n-1$. Now, there is a natural isomorphism in graphs between B and a subset of edges in E^n_m, by our definition of E^n_m, call this set C. Thus, the graph $H = H' \cup C$ has the property that $H' = H \cap E_{n-1,m}$, with $\overline{\sigma_{n-1}(H')} = \beta$.

The vertices of $G_{\beta,B}$ can be identified with $\mathcal{V}_{n-1} \cap \mathcal{V}_n$, just by changing $(1,j) \to (n-1,j)$ and $(2,j) \to (n,j)$, for $0 \leq j \leq m-1$. Even more, there is an edge $(1,i) \sim (1,j)$ in $G_{\beta,B}$ if and only if there is a path from $(n-1,i)$ to $(n-1,j)$ in H', by definition of $G_{\beta,B}$ and $\sigma_{n-1}(H')$. And by construction of C, there is an edge $(1,j) \sim (2,j)$ or $(2,j) \sim (2,j')$ in B if and only if there is an edge $(n-1,j) \sim (n,j)$ or $(n,j) \sim (n,j')$ in C

Therefore, if the vertices a, b of $G_{\beta,B}$ are in the same connected component, then the corresponding vertices in $\mathcal{V}_{n-1} \cup \mathcal{V}_n$ will be in the same connected component of H. Thus, $|\psi_n(H)| = \omega(G_{\beta,B})$. □

A converse of the previous result is also true.

Lemma 1.6.5. *If $H \subseteq E_{n,m}$, where $n, m \geq 2$ are integers, then for the standard state $\beta = \overline{\sigma_{n-1}(H_1)}$, where $H_1 = H \cap E_{n-1,m}$, and the subset of edges $B = H \cap E_m^n$, we have that*

$$|\psi_n(H)| = \omega(G_{\beta,B}).$$

Proof. This follows directly from the definition of a standard form and the graph $G_{\beta,B}$. □

Now, it is more clear what we meant at the end of Lemma 1.6.3; the evaluation of the exponent of the variables in the right-hand side of equation 1.6.3 will be done by using just the information of the standard forms of the A_i's and the B_i's, then the evaluation of the product in Lemma 1.6.2 will be done by the transfer-matrix method.

For a fixed $m \geq 2$, we define a *transformation of an element $\beta \in \mathcal{S}_m$ by a subset of edges $B \in P'_m$*. The new standard state will be $\gamma = \overline{\sigma_{n-1}(H)}$, where n and H are defined by Lemma 1.6.4. This element depends only on β and B and not the choice of n or H, as H was constructed using just β and B, also n was determined by β. We denote this by $\beta \xrightarrow{B} \gamma$.

It is also convenient to have the following notion. For a standard state $\gamma \in \mathcal{S}_m$, we say that it is *realizable* if there exists $B \subseteq E_{1,m}$ such that $\overline{\sigma_1(B)} = \gamma$, and we call B the *realization* of γ. Observe that not every element of $\mathcal{S}_m$ has to be realizable, for example, with $m = 3$, $(1, 2, 1)$ is not realizable. Also, it is worth noticing that the realization of a standard state γ is unique.

Finally, we are ready to use the transfer-matrix method. Let $m \geq 2$, consider the directed graph $\vec{G}_m$, whose vertices are the standard states in $\mathcal{S}_m$, and if $\beta, \gamma \in \mathcal{S}_m$ is a pair of standard states such that exists B, a subset of edges of P'_m, with $\beta \xrightarrow{B} \gamma$, then there is an edge $\beta \to \gamma$. We assign to this edge the weight

$$x^{\omega(G_{\beta,B})-|\beta|-m} y^{|B|+\omega(G_{\beta,B})-|\beta|-m}.$$

The weights are elements of the ring of polynomials $\mathbb{Z}[x, x^{-1}, y]$, where we assume $xx^{-1} = 1$. Observe that there may be many directed edges from β to γ, all with different weight.

Consider the adjacency matrix T_m of $\vec{G}_m$ with respect to this weight assignment, then we have the following

Theorem 1.6.6. *Let β and γ be elements of $\mathcal{S}_m$, where β has realization B_1, then*

$$x^{nm-1} x^{|B_1|} (T_m^{n-1})_{\beta,\gamma} = \sum_{\substack{A \subseteq E_{n,m} \\ \sigma_1(A)=\beta \\ \sigma_n(A)=\gamma}} x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)}$$

Proof. Let A be a subset of $E_{n,m}$, with $\sigma_1(A) = \beta$ and $\sigma_n(A) = \gamma$. Consider its right-row decomposition $\{A_i\}_{i=0}^n$ and its block decomposition $\{B_i\}_{i=1}^n$, we show that the path Γ in $\vec{G}_m$ defined by

$$\beta = \gamma_1 \xrightarrow{B_2} \cdots \xrightarrow{B_n} \gamma_n = \gamma$$

where $\gamma_i = \sigma_i(A_i)$, is such that

$$x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)} = x^{mn-1} x^{|B_1|} w(\Gamma). \tag{1.6.4}$$

As the term $|B_1| - r(B_1) = 0$, because B_1 is an independent set of edges, see page **??**, we can introduce to the right-hand side expression of equation 1.6.4 the factor $y^{|B_1|-r(B_1)}$, we also expand $w(\Gamma)$, so the right-hand side of equation 1.6.4 is equal to

$$x^{mn-1} x^{|B_1|} y^{|B_1|-r(B_1)} \prod_{i=1}^{n-1} x^{\omega(G_{\gamma_i,B_{i+1}})-|\gamma_i|-m} y^{|B_{i+1}|+\omega(G_{\gamma_i,B_{i+1}})-|\gamma_i|-m}.$$

By Lemma 1.6.4, this expression is equal to

$$x^{mn-1} x^{|B_1|} y^{|B_1|-r(B_1)} \prod_{i=1}^{n-1} x^{|\psi_{i+1}(A_{i+1})|-|\sigma_i(A_i)|-m} y^{|B_{i+1}|+|\psi_{i+1}(A_{i+1})|-|\sigma_i(A_i)|-m},$$

and by Lemma 1.6.3, this is equal to

$$x^{mn-1} x^{|B_1|} y^{|B_1|-r(B_1)} \prod_{i=1}^{n-1} x^{-r(A_{i+1})+r(A_i)} y^{|B_{i+1}|-r(A_{i+1})+r(A_i)}.$$

Finally, by Lemma 1.6.2, this last expression is equal to

$$x^{nm-1} x^{-r(A)} y^{|A|-r(A)} = x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)}.$$

But every path Γ from β to γ gives an $A \subseteq E_{n,m}$ such that $\sigma_1(A) = \beta$ and $\sigma_n(A) = \gamma$. If Γ is the path

$$\beta \xrightarrow{B_2} \cdots \xrightarrow{B_n} \gamma,$$

take $A = \bigcup_{i=1}^n B_i$. Therefore, to every path Γ there exists by construction a unique A such that equation 1.6.4 holds. By summing over all paths we get the result. □

We define the vector $\overrightarrow{X_m}$ of c_{m+1} entries over $\mathbb{Z}[x]$ by

$$(\overrightarrow{X_m})_\gamma = \begin{cases} x^{|B|} & \text{if } \gamma \text{ has realization } B, \\ 0 & \text{otherwise.} \end{cases} \tag{1.6.5}$$

Now, we have the main theorem of this section.

Theorem 1.6.7. *For integers $n, m \geq 2$ we have*

$$T(L_{n,m}; x+1, y+1) = x^{nm-1}\overrightarrow{X_m}^t T_m^{n-1}\vec{1}, \tag{1.6.6}$$

where $\vec{1}$ is the c_{m+1} vector with all entries equal to 1 and $\overrightarrow{X_m}$ is the vector defined by equation 1.6.5.

Proof. The definition of the Tutte polynomial implies that

$$T(L_{n,m}; x+1, y+1) = \sum_{A \subseteq E_{n,m}} x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)}.$$

On the other hand,

$$x^{nm-1}\overrightarrow{X_m}^t T_m^{n-1}\vec{1} = \sum_{\beta}\sum_{\gamma}\sum_{\substack{A \subseteq E_{n,m} \\ \sigma_1(A)=\beta \\ \sigma_n(A)=\gamma}} x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)},$$

where the first sum runs over all realizable elements β of $\mathcal{S}_m$, and the second sum runs over the all elements of $\mathcal{S}_m$. Because the last term of the previous sequence of equations includes the term $x^{r(E(L_{n,m}))-r(A)} y^{|A|-r(A)}$ exactly once for every $A \subseteq E_{n,m}$, as we show at the end of the proof of Theorem 1.6.6, we obtain the result. □

1.6.1 Numerical values

In principle, the above method can be used to compute the Tutte polynomial of the lattice $L_{n,m}$, $n, m \geq 2$, but computationally it is not feasible, as the required space to store the transfer-matrix grows exponentially, as an example, for $m = 10$ the transfer-matrix is a 16792-by-16792 matrix. Even for small values of n and m, the above computation involves storing large polynomials for each entry of the transfer-matrix, so that although possible, it is very cumbersome. Other possibility is to evaluate the polynomial at sufficient many points and then interpolate, this is more practical but we have not implemented this option in our programs.

However, for some small values of n and m we can evaluate the Tutte polynomial at particular points easily. By Theorem 1.6.7, to evaluate $T(L_{n,m}; x_0+1, y_0+1)$, we just have to evaluate

$$x_0^{nm-1}\overrightarrow{X_m}^t|_{x=x_0} \left(T_m|_{\substack{x=x_0 \\ y=y_0}}\right)^n \vec{1}$$

where $\overrightarrow{X_m}^t|_{x=x_0}$ is the vector defined in 1.6.5 with the substitution $x = x_0$, and $T_m|_{\substack{x=x_0 \\ y=y_0}}$ is the transfer matrix defined in page 32 with the substitution, $x = x_0$ and $y = y_0$.

For small m, the matrix $T_m|_{\substack{x=x_0\\y=y_0}}$ can be created by using the programs `indices.c` in Section B.1 and `matrix.c` in Section D.1; also the vector $\overrightarrow{X_m}^t|_{x=x_0}$ can be created by using the program `vector.c` in Section F.1. Using this procedure with the values $(x_0, y_0) = (1, 0)$ and $(x_0, y_0) = (1, -1)$, for the square lattices of side n, $2 \leq n \leq 7$, we compute $f(n)$ and $\alpha(n)$ for $2 \leq n \leq 7$. The values are shown in Table 1.1 and Table 1.2. Unfortunately, this method of evaluating the Tutte polynomial at a particular point does not allow us to compute values with a substitution involving $x_0 = 0$, as the entries of the matrix T_n may have polynomials with terms in x^{-1}, thus we cannot compute evaluations of the Tutte polynomial at the points $(1, y_0)$, $y_0 \in \mathbb{R}$.

Forests

Side n	Number of forests $f(n)$	Upper bound $2^{2/n} f(n)^{1/n^2}$
2	15	3.93597934253086
3	3102	3.87834878392052
4	8790016	3.84161954194043
5	3.410086174080000e+11	3.81700397677008
6	1.810755082420676e+17	3.79951640436763
7	1.315927389374152e+24	3.78649853538319

Table 1.1: This table displays the values of $f(n)$ and $2^{2/n} f(n)^{1/n^2}$ for $1 \leq n \leq 7$.

The following tables display the values of $f(n)$ and $\alpha(n)$ for $1 \leq n \leq 7$. Table 1.1 also show the values $2^{2/n} f(n)^{1/n^2}$, which give upper bounds for $\lim_{k\to\infty}(f(k))^{1/k^2}$ by Theorem 1.4.1. And Table 1.2 shows the values $2^{2/n}\alpha(n)^{1/n^2}$ which are upper bounds for $\lim_{k\to\infty}(\alpha(k))^{1/k^2}$ by the proof of Theorem 1.3.3.

Acyclic orientations

Side n	Number of acyclic orientations $\alpha(n)$	Upper bound $2^{2/n}\alpha(n)^{1/n^2}$
2	14	3.86867284053534
3	2398	3.76899413308793
4	5015972	3.70925927917375
5	1.280914342660000e+11	3.67039497637590
6	3.993185613821266e+16	3.64326780246972
7	1.519663682749935e+23	3.62330970816373

Table 1.2: This table displays the values of $\alpha(n)$ and $2^{2/n}\alpha(n)^{1/n^2}$ for $1 \leq n \leq 7$.

1.7 Forests and acyclic orientations again

The procedure described in subsection 1.6.1 allows us to actually compute the number of forests of $L_{n,m}$, that from now on we denote by $f(n,m)$, for a fixed m and an arbitrary n. In fact, it allows us more, namely, to have a very good bound on $\lim_{n\to\infty} f(n,m)^{1/n}$ by using linear algebra.

During this section we denote $T_m|_{\substack{x=1\\y=0}}$ by A_m. To compute $f(n,m)$ we have to evaluate $\overrightarrow{X_m}^t|_{x=1} A_m^{n-1}\vec{1}$, but the vector $\overrightarrow{X_m}^t|_{x=1}$ has just 0-1 entries, as $\overrightarrow{X_m}^t$ is given by 1.6.5. Then, we are interested in evaluating $aA_m^{n-1}\vec{1}$, where a is a 0-1 vector.

The first observation is that $aA\vec{1} \leq \|A\|_1$, where a is a 0-1 vector, A is a $k\times k$ real matrix and $\|\cdot\|_1$ is the l_1 *matrix norm*, that is $\|A\|_1 = \sum_i \sum_j |A_{ij}|$. This is clear as

$$aA\vec{1} = \sum_{i:a_i=1}\sum_{j=1}^{k} A_{ij} \leq \sum_{i=1}^{k}\sum_{j=1}^{k} |A_{ij}| = \|A\|_1. \tag{1.7.1}$$

Also, there is a well known result in linear algebra, see [**?**], that relates any matrix norm of a matrix with its *spectral radius*. If A is a $k\times k$ matrix, the spectral radius of A is defined by

$$\rho(A) = \max\{|\lambda| \mid \lambda \text{ is an eigenvalue of } A\}$$

Theorem 1.7.1. *Let $\|\cdot\|$ be a matrix norm on $\mathcal{M}_k$, the $k\times k$ real matrices. Then, for $A\in\mathcal{M}_k$,*

$$\rho(A) = \lim_{k\to\infty} \|A^k\|^{1/k}.$$

Combining these two results we obtain the following theorem

Theorem 1.7.2. *For any fixed natural number m,*

$$\lim_{n\to\infty} f(n,m)^{1/n} \leq \rho(A_m).$$

Proof. From the above discussion we have

$$f(n,m) = aA_m^{n-1}\vec{1} \leq \|A_m^{n-1}\|_1,$$

for some 0-1 vector a. Now, using this and that $\frac{1}{n} = \frac{1}{n-1} + \frac{-1}{n(n-1)}$ we get

$$\begin{aligned}
\lim_{n\to\infty} f(n,m)^{\frac{1}{n}} &\leq \lim_{n\to\infty} \|A_m^{n-1}\|_1^{\frac{1}{n}}. \\
&= \lim_{n\to\infty} \|A_m^{n-1}\|_1^{\frac{1}{(n-1)}} \|A_m^{n-1}\|_1^{\frac{-1}{n(n-1)}} \\
&= \frac{\lim_{n\to\infty}\|A_m^{n-1}\|_1^{\frac{1}{(n-1)}}}{\lim_{n\to\infty}\left(\|A_m^{n-1}\|_1^{\frac{1}{(n-1)}}\right)^{\frac{1}{n}}}.
\end{aligned}$$

By Theorem 1.7.1, the sequence $\|A_m^{n-1}\|_1^{1/(n-1)}$ converges to $\rho(A_m) > 0$. Therefore, the limit in the denominator of the last expression converges to 1 and the above limit converges to $\rho(A_m)$. □

The upper bound for $\lim_{n\to\infty} f(n,m)^{1/n}$ has a direct implication on $\lim_{n\to\infty} f(n)^{1/n^2}$, as we prove in the following theorem.

Theorem 1.7.3. *For $k \geq 1$,*

$$\lim_{n\to\infty} f(n)^{1/n^2} \leq 2^{1/k}(\rho(A_k))^{1/k}.$$

Proof. Let k be a fixed integer. From a square lattice of side kp we select p lattices $L_{p,pk}$, whose bottom left-hand corners are the points $(ki, 0)$, $0 \leq i \leq p-1$. Call $\mathcal{C}$ this set of subgraphs.

Choose for every subgraph in $\mathcal{C}$ a spanning forest and then choose any subset of the remained $(p-1)kp$ edges in L_{kp}. Any forest of L_{kp} can be obtained in this way, but this is clearly an over counting, so, as in Theorem 1.4.1, we conclude that

$$f(kp) \leq 2^{(p-1)kp}(f(k,kp))^p \leq 2^{kp^2}(f(k,kp))^p.$$

Hence

$$f(kp)^{1/(kp)^2} \leq 2^{1/k} f(k,kp)^{1/k^2 p}$$

By taking the limit as $p \to \infty$ we get the result using Theorem 1.7.2. □

Using the programs `indices.c` and `matrix.c` together with `matlab`, we compute the values of $\rho(A_n)$ for $2 \leq n \leq 7$. In table 1.3 we show the values of $\rho(A_n)$ and $2^{1/n}(\rho(A_n))^{1/n}$, $1 \leq n \leq 7$. These numerical values and Theorem 1.7.3 imply the following.

Corollary 1.7.4.

$$\lim_{n\to\infty} f(n)^{1/n^2} \leq 3.7469814013994.$$

We now turn to acyclic orientations. If we denote by A'_m the matrix $T_m\big|_{\substack{x=1\\y=-1}}$, we can apply the above observation 1.7.1 and Theorem 1.7.1 to obtain a similar result to Theorem 1.7.2 but for the number of acyclic orientations of $L_{n,m}$, that we denote by $\alpha(n,m)$.

Theorem 1.7.5. *For any fixed natural number m,*

$$\lim_{n\to\infty} \alpha(n,m)^{1/n} \leq \rho(A'_m).$$

Forests

Side n	Spectral radius of A_n	Upper bound $2^{1/n}(\rho(A_n))^{1/n}$
2	7.46410161513775	3.8637033051563
3	27.66113326848412	3.8103656631516
4	1.023825083033856e+02	3.7828048765881
5	3.788239653174334e+02	3.7661142283605
6	1.401519651761211e+03	3.7549561053822
7	5.184904556369020e+03	3.7469814013994

Table 1.3: This table displays the spectral radius of A_n and the bound $2^{1/n}(\rho(A_n))^{1/n}$ for $1 \leq n \leq 7$.

Proof. The proof is the same as in Theorem 1.7.2, but with the matrix A'_m. □

Also, the same technique as used to prove Theorem 1.7.3 can be used for the proof of the following.

Theorem 1.7.6. *For $k \geq 1$,*

$$\lim_{n\to\infty} \alpha(n)^{1/n^2} \leq 2^{1/k}(\rho(A'_k))^{1/k}.$$

Proof. Let k be a fixed integer. We again partition the square lattice of side kp into p smaller lattices $L_{p,k}$, whose bottom left-hand corners are the points $(ki, 0)$, $0 \leq i \leq p-1$. Call $\mathcal{C}'$ this set of subgraphs.

We choose for every subgraph in $\mathcal{C}'$ an acyclic orientation and then choose any arbitrary orientation for the remained $(p-1)kp$ edges in L_{kp}. Any acyclic orientation of L_{kp} can be obtain in this way, but again we have over counted, so we conclude that

$$\alpha(kp) \leq 2^{(p-1)kp}(\alpha(k, kp))^p \leq 2^{kp^2}(\alpha(k, kp))^p.$$

Hence

$$\alpha(kp)^{1/(kp)^2} \leq 2^{1/k}\alpha(k, kp)^{1/k^2p}$$

By taking the limit as $p \to \infty$ we get the result using Theorem 1.7.5. □

In table 1.4 we show the values of $\rho(A'_n)$ and $2^{1/n}(\rho(A'_n))^{1/n}$, $1 \leq n \leq 7$. Using these values and Theorem 1.7.6 we get the following.

Corollary 1.7.7.

$$\lim_{n\to\infty} \alpha(n)^{1/n^2} \leq 3.5632215047716.$$

Acyclic orientations

Side n	Spectral radius of A'_n	Upper bound $2^{1/n}(\rho(A'_n))^{1/n}$
2	7	3.7416573867740
3	24.46585609973064	3.6576050153895
4	85.48940334443925	3.6160595209437
5	2.987015449475036e+02	3.5913149457935
6	1.043650023238710e+03	3.5749020517894
7	3.646445690249791e+03	3.5632215047716

Table 1.4: This table displays the spectral radius of A'_n and the bound $2^{1/n}(\rho(A'_n))^{1/n}$ for $1 \leq n \leq 7$.

1.8 Some improvement on the lower bounds

In the last section we used the transfer-matrix method to improve the upper bounds given in [**?**]. In this section we improve the lower bounds of the same reference.

The *n-fan* graph $\mathcal{F}_n$ is the graph whose vertices are $\{\hat{0}\} \cup \{(j,1) | 0 \leq j \leq n-1\}$ and whose edge set consists of all the edges $\hat{0} \sim (j,1)$, $0 \leq j \leq n-1$, and $(j,1) \sim (j+1,1)$, $0 \leq j \leq n-2$, see Figure 1.3. By computing the generating function of the number of forests of $\mathcal{F}_n$, denoted $f(\mathcal{F}_n)$, Marc Noy [**?**] proved the following

Theorem 1.8.1.

$$\lim_{n\to\infty} f(n)^{1/n^2} \geq 2 + \sqrt{2}.$$

Marc Noy and I later extend this to get better lower bounds, here we include this work.

Figure 1.3: The $15, 5$-fan graph $\mathcal{F}_{15}^5$ and the 9-fan graph $\mathcal{F}_9$.

We define the n, k-*fan* graph $\mathcal{F}_n^k$, $k \geq 1$, as the graph with vertex set $\{\hat{0}\} \cup \{(i,j) | 0 \leq i \leq n-1,\ 1 \leq j \leq k-1\}$. There is an edge between vertices (i,j) and

(i', j') if $|i - i'| + |j - j'| = 1$, for all $0 \leq i \leq n-1,\ 1 \leq j \leq k$; also we have all the edges $\hat{0} \sim (i, 1)$, for $0 \leq i \leq n-1$. See Figure 1.3.

Thus, the $n, 1$-fan graph $\mathcal{F}_n^1$ is the n-fan graph. The number of forests of $\mathcal{F}_n^k$ will be denoted by $f(\mathcal{F}_n^k)$ and the number of acyclic orientations by $\alpha(\mathcal{F}_n^k)$.

For the proofs of the following two theorems one more definition is required. We define the n, k-*comb* graph P_n^k to be the graph with vertex set $\{(i, j) | 0 \leq i \leq n-1,\ 0 \leq j \leq k\}$. And there is an edge between vertices (i, j), (i', j') if $|i - i'| + |j - j'| = 1$, $0 \leq i, i' \leq n-1,\ 0 \leq j, j' \leq k$; also we have all the edges $(i, 0) \sim (i, 1)$, $0 \leq i \leq n-1$, see Figure 1.4. Observe that the $n, 1$-comb graph is the n-comb graph P'_n defined in page 25. Also note that there is a natural bijection, $\phi_{n,k}$, from the set of edges of P_n^k to the set of edges of $\mathcal{F}_n^k$ given by the identity function on all the edges $(i, j) \sim (i', j')$ with $0 \leq i \leq n-1$, $1 \leq j \leq k$, and the edge $(i, 0) \sim (i, 1)$ goes under $\phi_{n,k}$ to the edge $\hat{0} \sim (i, 1)$, for $0 \leq i \leq n-1$.

Figure 1.4: The $15, 5$-comb graph P_{15}^5.

Theorem 1.8.2. *For an arbitrary but fixed integer k,*

$$\left(\lim_{m\to\infty} (f(\mathcal{F}_m^k))^{1/m}\right)^{1/k} \leq \lim_{n\to\infty} f(n)^{1/n^2}.$$

Proof. From a square lattice of side $kp+1$ we select p different $kp+1, k$-comb graphs, G_i, $0 \leq i \leq p-1$, whose bottom left-hand corners are the points $(0, ki)$, $0 \leq i \leq p-1$. Call $\mathcal{C}$ this set of subgraphs. Observe that there are kp edges left at the bottom of L_{kp+1} that do not belong to any of the G_i's.

Choose for every subgraph G_i in $\mathcal{C}$ a spanning forest B'_i in $\mathcal{F}_{kp+1}^k$ and take the edges in G_i that correspond (under the bijection $\phi_{kp+1,k}$) to this forest, say B_i, $0 \leq i \leq p-1$.

The set of edges $B = \bigcup_{i=0}^{p-1} B_i$ forms a forest. The reason is the following: suppose there is a cycle C in B, then it would intersect some of the subgraphs B_i. The cycle C cannot be inside some B_i, as this would contradict our choice of B'_i. Let j_0 be the maximum j such that B_j intersects C. Thus, B_{j_0} contains a path from (l, j_0k) to (h, j_0k) for some $0 \leq l, h \leq kp$. This path in B_{j_0} maps under $\phi_{kp+1,k}$ onto a cycle in $\mathcal{F}^k_{kp+1}$ and this contradicts our choice of B'_{j_0}. Therefore there is not such a cycle C.

Any such choice of the B'_i, $0 \leq i \leq p-1$ will give a different forest of L_{pk+1}, so

$$\big(f(\mathcal{F}^k_{kp+1})\big)^p \leq f(kp+1),$$

then

$$\Big(\big(f(\mathcal{F}^k_{kp+1})\big)^{\frac{1}{pk+1}}\Big)^{\frac{p}{kp+1}} \leq (f(kp+1))^{\frac{1}{(pk+1)^2}},$$

and by taking the limit when $p \to \infty$ we get the result. □

With some changes, the last proof also gives a proof of the following

Theorem 1.8.3. *For an arbitrary but fixed integer k,*

$$\Big(\lim_{m\to\infty} (\alpha(\mathcal{F}^k_m))^{1/m}\Big)^{1/k} \leq \lim_{n\to\infty} \alpha(n)^{1/n^2}.$$

Proof. Choose for every subgraph G_i in $\mathcal{C}$, as defined in the proof of Theorem 1.8.2, an acyclic orientation A'_i in $\mathcal{F}^k_{kp+1}$ and consider the (acyclic) orientation in G_i that corresponds (under the bijection $\phi_{kp+1,k}$) to this orientation, say A_i, $0 \leq i \leq p-1$. Orient the remaining kp edges in any arbitrary but fixed way. This orientation, say A, is an acyclic orientation of L_{kp+1}. To see this, we suppose there is an oriented cycle C in A. The oriented cycle C cannot be part of the orientation A_i, as this would contradict the choice of A'_i, for all $0 \leq i \leq p-1$. Let j_0 be the maximum j such that A_j contains part of C. Then, A_{j_0} contains a directed path from (l, j_0k) to (h, j_0k) for some $0 \leq l, h \leq kp$. But this means that there is an oriented cycle in A'_{j_0} which is not possible. Thus, there is not such an oriented cycle in A.

Again, any such choice of the A'_i, $0 \leq i \leq p-1$, will give an acyclic orientation of L_{pk+1}, so

$$\big(\alpha(\mathcal{F}^k_{kp+1})\big)^p \leq \alpha(kp+1),$$

and the proof follows as in Theorem 1.8.2. □

It is easy to find and solve the recurrence relation for the sequence $f(\mathcal{F}_m)$. The recurrence relation is given by

$$f(\mathcal{F}_m) = 4f(\mathcal{F}_{m-1}) - 2f(\mathcal{F}_{m-2}),$$

with initial conditions $f(\mathcal{F}_1) = 2$ and $f(\mathcal{F}_2) = 7$. The solution is given by

$$f(\mathcal{F}_m^1) = f(\mathcal{F}_m) = \frac{1+\sqrt{2}}{4}(2+\sqrt{2})^m + \frac{1-\sqrt{2}}{4}(2-\sqrt{2})^m.$$

Theorem 1.8.1 follows by Theorem 1.8.2 and the last expression.

It is also relatively easy to find a recurrence relation for $\alpha(\mathcal{F}_m^2)$, namely

$$\alpha(\mathcal{F}_m^2) = 13\alpha(\mathcal{F}_{m-1}^2) - 27\alpha(\mathcal{F}_{m-2}^2),$$

with the initial conditions $\alpha(\mathcal{F}_1^2) = 4$ and $\alpha(\mathcal{F}_2^2) = 42$. By solving this recurrence we get

$$\alpha(\mathcal{F}_m^2) = c_1\left(\frac{13+\sqrt{61}}{2}\right)^m + c_2\left(\frac{13-\sqrt{61}}{2}\right)^m,$$

where $c_1 \approx 0.3890957718$ and $c_2 \approx 0.01872540139$. Then, by Theorem 1.8.3 we obtain

Corollary 1.8.4.

$$\lim_{n\to\infty} \alpha(n)^{1/n^2} \geq \sqrt{\frac{13+\sqrt{61}}{2}}.$$

Using contraction and deletion, we could find, in principle, recurrence relations for the sequences $f(\mathcal{F}_m^k)$ and $\alpha(\mathcal{F}_m^k)$ for a fixed $k > 1$ and $m = 1, \ldots$, then we solve the recurrence to obtain an explicit expression of these sequences. This is, however, very cumbersome. Already for $f(\mathcal{F}_m^2)$ we have to express $f(\mathcal{F}_m^2)$ as a solution of several linear recurrence relations with many variables, [**?**].

Here, we prefer to use the transfer-matrix method already developed and compute $\lim_{n\to\infty}(f(\mathcal{F}_n^k))^{1/n}$ for some small values of k.

Let $H = \{e_1, \ldots, e_{n-1}\}$ be the set of edges $e_1 = (0,0) \sim (1,0)$, $\ldots$, $e_{n-1} = (n-2,0) \sim (n-1,0)$. Clearly, the graphic matroid $M(\mathcal{F}_n^{k-1})$ is isomorphic to the graphic matroid $M(L_{n,k}/H)$, where $L_{n,k}/H$ is the graph $L_{n,k}$ with the edges in H contracted. Even more, we can consider their ground sets the same as the identified vertices in $M(L_{n,k}/H)$ can be considered to be $\hat{0}$, the same vertex as in $\mathcal{F}_n^{k-1}$, so, we consider here $E(\mathcal{F}_n^{k-1}) = E(L_{n,k}/F)$. If r denotes the rank function of $M(L_{n,k})$ and r' denotes the rank function of $M(\mathcal{F}_n^{k-1})$, for a set $B \subseteq E(\mathcal{F}_n^{k-1})$, these functions are related by

$$\begin{aligned} r'(B) &= r(B \cup \{e_1, \ldots, e_{n-1}\}) - r(\{e_1, \ldots, e_{n-1}\}) \\ &= r(B \cup \{e_1, \ldots, e_{n-1}\}) - (n-1), \end{aligned} \tag{1.8.1}$$

where the first equality follows from Proposition **??**. In particular

$$r'(E(\mathcal{F}_n^{k-1})) = r(E(L_{n,k})) - (n-1). \tag{1.8.2}$$

The definition of the Tutte polynomial gives for $M(\mathcal{F}_m^{k-1})$ that

$$T(\mathcal{F}_n^{k-1}; x+1, y+1) = \sum_{B \subseteq E(\mathcal{F}_n^{k-1})} x^{r'(E(\mathcal{F}_n^{k-1}))-r'(B)} y^{|B|-r'(B)}. \tag{1.8.3}$$

If for each $B \subseteq E(\mathcal{F}_n^{k-1})$ we take $A = B \cup \{e_1, \ldots, e_{n-1}\}$, then, for a particular B, the exponent of x in 1.8.3 equals, by equations 1.8.1 and 1.8.2,

$$r(E(L_{n,m})) - (n-1) - r(A) + (n-1) = r(E(L_{n,m})) - r(A)$$

and the exponent of y equals

$$|A| - (n-1) - r(A) + (n-1) = |A| - r(A).$$

So, we obtain

$$T(\mathcal{F}_n^{k-1}; x+1, y+1) = \sum_{\substack{A = B \cup \{e_1, \ldots, e_{n-1}\} \\ B \subseteq E(\mathcal{F}_n^{k-1})}} x^{r(E(L_{n,k}))-r(A)} y^{|A|-r(A)}. \tag{1.8.4}$$

We use again the transfer-matrix method to give an expression for this Tutte polynomial. For $k \geq 2$, we define a subdigraph $\vec{G'}_k$ of the digraph $\vec{G}_k$ considered in page 32. Its vertices are the standard forms in $\mathcal{S}_k$, and there is a directed edge $\beta \to \gamma$, for $\beta, \gamma \in \mathcal{S}_k$ if $\beta \xrightarrow{B} \gamma$ as defined in page 32, where B is a subset of edges of P'_k containing the edge $(1,0) \sim (2,0)$. The weights of the edges are the same as in page 32.

Consider the adjacency matrix T'_k of $\vec{G'}_k$ with respect to these weights. We have the following analogue of Theorem 1.6.6

Theorem 1.8.5. *Let β and γ be elements of $\mathcal{S}_k$, where β has realization B_1, then*

$$x^{kn-1} x^{|B_1|} (T'_k)^{n-1}_{\beta,\gamma} = \sum_{\substack{A \subseteq E_{n,k} \\ \{e_1, \ldots, e_{n-1}\} \subset A \\ \sigma_1(A) = \beta \\ \sigma_n(A) = \gamma}} x^{r(E(L_{n,k}))-r(A)} y^{|A|-r(A)}$$

Proof. It follows from the proof of Theorem 1.6.6 by considering just subsets A of $E(L_{n,k}) = E_{n,k}$ such that A contains all the edges $\{e_1, \ldots, e_{n-1}\}$. □

Take the vector $\overrightarrow{X_k}$ as defined in 1.6.5, then the analogue to Theorem 1.6.7 is the following

Theorem 1.8.6. *For $k, n \geq 2$ two integers we have*

$$T(\mathcal{F}_n^{k-1}; x+1, y+1) = x^{kn-1} \overrightarrow{X_k}^t (T_k')^{n-1} \vec{1}, \tag{1.8.5}$$

where $\vec{1}$ is the vector with all entries equal to 1.

Proof. By 1.8.4, we have

$$T(\mathcal{F}_n^{k-1}; x+1, y+1) = \sum_{\substack{A = B \cup \{e_1, \ldots, e_{n-1}\} \\ B \subseteq E(\mathcal{F}_n^{k-1})}} x^{r(E(L_{n,k})) - r(A)} y^{|A| - r(A)}.$$

Now, the proof continues as the proof of Theorem 1.6.7 where now we just consider sets A containing the edges $\{e_1, \ldots, e_{n-1}\}$. □

As explained in Subsection 1.6.1, we can now evaluate for a particular point $(x_0 + 1, y_0 + 1)$ the polynomial $T(\mathcal{F}_n^{k-1}; x_0 + 1, y_0 + 1)$ by computing

$$x_0^{kn-1} \overrightarrow{X_k}^t|_{x=x_0} \, (T_k'|_{\substack{x=x_0 \\ y=y_0}})^{n-1} \, \vec{1}.$$

But due to Theorem 1.8.2 and Theorem 1.8.3, we are more interested in evaluating the $\lim_{n\to\infty}(f(\mathcal{F}_n^{k-1}))^{1/n}$ and $\lim_{n\to\infty}(\alpha(\mathcal{F}_n^{k-1}))^{1/n}$. So, we consider here just the evaluations $x_0 = 1$ and $y_0 = 0$ or $y_0 = -1$. We denote for the rest of this section the square real matrix $T_k'|_{\substack{x=1 \\ y=0}}$ by D_k; the matrix $T_k'|_{\substack{x=1 \\ y=-1}}$ by D_k'; and the column vector $\overrightarrow{X_k}^t|_{x=1}$ by a_k.

Before going into the main theorems of the second part of this section, we take a detour on the Perron-Frobenius Theorem. Most of this is from [**?**].

Definition 1.8.7. *Let $A = (a_{ij})$ be an $n \times n$ real matrix. We say that $A \geq 0$, that is A is nonnegative, if all its entries a_{ij} are nonnegative. We say that $A > 0$, A is positive, if all its entries a_{ij} are positive.*

Later in the section we show that the matrices D_k and D_k' are nonnegative. First an important definition for the Perron-Frobenius Theorem.

Definition 1.8.8. *An $n \times n$ real matrix A is said to be reducible if either*

(a) *$n = 1$ and $A = 0$; or*

(b) *$n \geq 2$, there is an $n \times n$ permutation matrix P (that is a square matrix, all of whose entries are 0 or 1 and in each column and each row of P there is precisely one 1), and there is some integer l with $1 \leq l \leq n-1$ such that*

$$P^t A P = \begin{pmatrix} B_1 & B_3 \\ 0 & B_2 \end{pmatrix}$$

where B_1 is an $l \times l$ matrix, B_2 is an $(n-l) \times (n-l)$ matrix, B_3 is an $l \times (n-l)$ matrix and $\vec{0}$ is the $l \times (n-l)$ zero matrix.

Definition 1.8.9. *An $n \times n$ real matrix A is irreducible if it is not reducible.*

We are going to prove later that the matrices D_k and D'_k are irreducible, but for that we need a criteria based on a graph associated to any matrix.

Definition 1.8.10. *The directed graph of an $n \times n$ real matrix A, denoted by $\Gamma(A)$, is the directed graph on n nodes $c_1, \ldots, c_n$ such that there is a directed arc in $\Gamma(A)$ from c_i to c_j if and only if $(A)_{ij} \neq 0$.*

Then there is the following theorem, see [**?**].

Theorem 1.8.11. *Let A be a $n \times n$ nonnegative real matrix. Then A is irreducible if and only if $\Gamma(A)$ is strongly connected.*

In order to state the Perron-Frobenius Theorem, we also need the notion of *primitivity.*

Definition 1.8.12. *An $n \times n$ nonnegative real matrix A is said to be primitive if it is irreducible and has only one eigenvector of maximum modulus.*

Theorem 1.8.13 (Perron-Frobenius Theorem)**.** *Let A be an $n \times n$ real matrix and suppose that A is primitive and nonnegative. Then*

(a) *$\rho(A) > 0$;*

(b) *$\rho(A)$ is an eigenvalue of A;*

(c) *There is a positive vector x such that $Ax = \rho(A)x$; and*

(d) $\lim_{m\to\infty}(\rho(A)^{-1}A)^m = L > 0$,

where $L = xy^t$, $Ax = \rho(A)x$, $A^t y = \rho(A)y$, $x > 0$, $y > 0$ and $x^t y = 1$.

Proof. See [**?**]. □

To check that a large matrix is primitive will be very difficult by using just the definition, fortunately from Lemma 8.52 and Lemma 8.55 in [**?**] one can obtain a more manageable test for primitivity.

Lemma 1.8.14. *If A is an $n \times n$ nonnegative and irreducible real matrix, and if all the main diagonal entries of A are positive, then A is primitive.*

Now, we apply this powerful theory to the matrices D_k and D'_k, for $k \geq 2$.

First recall that we have the notation $\beta \xrightarrow{B} \gamma$, $\beta, \gamma \in \mathcal{S}_k$, and B is a subset of edges of P'_k, see page 31. As mentioned before, we are interested in subsets of edges of P'_k that contain the edge $(1,0) \sim (2,0)$, that is subsets of $E(P'_k/e_0)$, where $e_0 = (1,0) \sim (2,0)$. We introduce the set

$$U_{\beta\gamma} = \{B \subseteq E(P'_k/e_0) | \beta \xrightarrow{B} \gamma\}.$$

Now, the entry (β, γ) in T'_k is a non-zero polynomial if and only if $U_{\beta\gamma}$ is non-empty. To prove that D_k is a nonnegative matrix we need the following technical lemma.

Lemma 1.8.15. *Let $k \geq 2$ and $\beta, \gamma \in \mathcal{S}_k$. If $U_{\beta\gamma}$ is non-empty, the evaluation of $(T'_k)_{\beta\gamma}$ is positive.*

Proof. The entry $(T'_k)_{\beta\gamma}$ is the sum of the terms

$$x^{\omega(G_{\beta,B})-|\beta|-k} y^{|B|+\omega(G_{\beta,B})-|\beta|-k}$$

over all B in $U_{\beta\gamma}$. Every term, when evaluated at $x = 1$, $y = 0$ is either 1 or 0. It is enough to prove that there exists B in $U_{\beta\gamma}$ such that the corresponding term is 1. For that, it is enough to prove that there exists B in $U_{\beta\gamma}$ such that

$$|B| + \omega(G_{\beta,B}) - |\beta| - k = 0. \tag{1.8.6}$$

Suppose that there is no edge of B belonging to a cycle of $G_{\beta,B}$. Thus, the removal of any edge in B increases the number of connected components by exactly 1. By this argument it follows that $\omega(G_{\beta,B} \setminus B) = \omega(G_{\beta,B}) + |B|$. But by definition of $G_{\beta,B}$, see page 30, $G_{\beta,B} \setminus B$ has $|\beta| + k$ connected components. Thus, if B has no edge belonging to a cycle of $G_{\beta,B}$, B satisfies 1.8.6.

We now show that we can find such a B in $U_{\beta\gamma}$. By hypothesis, $U_{\beta\gamma}$ is non-empty, so there exists $C_0 \in U_{\beta\gamma}$. If C_0 has no edge belonging to a cycle of G_{β,C_0}, we have the result. Suppose that C_0 has the edge a_0 in a cycle of G_{β,C_0}, note that such an edge can be taken to be different from e_0, and consider $C_1 = C_0 \setminus a_0$. We claim that C_1 also belongs to $U_{\beta\gamma}$. By definition of $\beta \xrightarrow{C_0} \gamma$, there exists H and m such that $\overline{\sigma_{m-1}(H_1)} = \beta$ and $\overline{\sigma_m(H)} = \gamma$, where $H_1 = H \cap E_{m-1,k}$; also, this H has the property that if it has block decomposition $\{B_i\}_{i=1}^m$, then B_m is isomorphic to C_0. Thus, the edge a'_0 of B_m corresponding to the edge a_0 in C_0 is in a cycle of H and its removal does not change the connected components of H. So, $\overline{\sigma_{m-1}(H_1 \setminus a'_0)} = \beta$,

$\overline{\sigma_m(H \setminus a_0')} = \gamma$ and B_m' is isomorphic to C_1, where $H \setminus a_0'$ has block decomposition $\{B_i'\}_{i=1}^m$. We can now repeat the argument with C_1 instead of C_0. As this process is finite, we end up with a set B with no edges in a cycle of $G_{\beta,B}$ and the proof is complete. □

Lemma 1.8.16. *Let $k \geq 2$. The real matrix D_k is nonnegative. Even more, the main diagonal entries of D_k are positive.*

Proof. The matrix D_k is the evaluation of T_k' at $x = 1$, $y = 0$. For every entry (β, γ) of T_k', $\beta, \gamma \in \mathcal{S}_k$, there are two possibilities. Either $U_{\beta\gamma}$ is empty, in which case $(T_k')_{\beta\gamma}$ is the zero polynomial and $(D_k)_{\beta\gamma}$ is zero; or $U_{\beta\gamma}$ is non-empty, in which case, by Lemma 1.8.15, $(T_k')_{\beta\gamma}$ when evaluated at $x = 1$, $y = 0$ is positive and $(D_k)_{\beta\gamma}$ is positive. We conclude that D_k is nonnegative.

To see that the main diagonal entries of D_k are positive, we just have to check, by Lemma 1.8.15, that $U_{\beta\gamma}$ is non-empty. But this is clear as the set B_I

$$B_I = \{(1, j) \sim (2, j) | 0 \leq j \leq k - 1\}$$

is always in $U_{\beta,\beta}$. □

A similar result is true for the matrix D_k' but the proof involves so much notation that we decided to omit it, and instead we analyse the matrix D_k' for the particular values of $2 \leq k \leq 8$ that we need for the main result of the section.

Theorem 1.8.17. *Let $2 \leq k \leq 8$. If $\beta, \gamma \in \mathcal{S}_k$, then $(D_k')_{\beta\gamma}$ is positive if and only if $(D_k)_{\beta\gamma}$ is positive and $(D_k')_{\beta\gamma}$ is zero if and only if $(D_k)_{\beta\gamma}$ is zero.*

Proof. This was done by computing for $2 \leq k \leq 8$ the matrices D_k' and D_k and comparing entry by entry. □

The goal is to prove that D_k is a primitive matrix, for $k \geq 2$. In view of Theorem 1.8.11 we need the following

Lemma 1.8.18. *The digraph $\Gamma(D_k)$ is strongly connected, for $k \geq 2$.*

Proof. By definition 1.8.7, the digraph $\Gamma(D_k)$ has $c_{k+1} = \frac{1}{k+1}\binom{2k}{k}$ vertices that we identify with the set $\mathcal{S}_k$, and there is an arc from β to γ if $(D_k)_{\beta,\gamma}$ is non-zero. Then, by Lemma 1.8.15, there is an arc $\beta \longrightarrow \gamma$ if $U_{\beta,\gamma}$ is not empty. So, to prove that $\Gamma(D_k)$ is strongly connected, we just have to give, for every pair (β, γ) with $U_{\beta,\gamma}$ non-empty, a sequence $\beta = \beta_0, \ldots, \beta_p = \gamma$, such that $U_{\beta_i,\beta_{i+1}}$ is non-empty for $0 \leq i \leq p - 1$.

Observe that we always have $U_{\beta,\overline{1}} \neq \emptyset$, where $\overline{1}$ is the standard form in $\mathcal{S}_k$ with all the entries equal to 1. This is because $\overline{1}$ is a realizable standard form, see page 32 for the definition, with realization

$$A_{\overline{1}} = \{(2,j) \sim (2,j+1) | 0 \leq j \leq k-2\}.$$

Then, $A_{\overline{1}} \cup \{e_0\}$ belongs to $U_{\beta,\overline{1}}$.

Thus, it is enough to prove that for every γ in $\mathcal{S}_k$, there is a sequence $\overline{1} = \beta_0, \ldots, \beta_p = \gamma$, such that $U_{\beta_i,\beta_{i+1}}$ is non-empty for $0 \leq i \leq p-1$. For this it is enough to give a subset $H \subseteq E_{p,k}$ with $\overline{\sigma_1(H \cap E_{1,k})} = \overline{1}$ and $\overline{\sigma_p(H)} = \gamma$, where H contains all the edges $(0,0) \sim (1,0)$, ..., $(p-2,0) \sim (p-1,0)$.

In order to avoid complicated notation with subindices, we make the following consideration. Take a standard form $\gamma \in \mathcal{S}_m$, then there exists $H \subseteq E_{n,m}$ such that $\overline{\sigma_n(H)} = \gamma$. Now, construct $H^\perp$ as the reflection along the horizontal symmetry axis of $L_{n,m}$, that is, vertex (i,j) goes to vertex $(i,m-j)$, and we obtain the standard form $\gamma^\perp = \overline{\sigma_n(H^\perp)}$. Observe that if $\gamma = (\gamma_1, \ldots, \gamma_m)$ with labels $1 = l_0 < \cdots < l_q$, then $\gamma^\perp = (R(\gamma_m), \ldots, R(\gamma_1))$, where R is the relabelling such that if $\gamma_i = l_{j_0}$ then $R(\gamma_i) = l_{p-j_0}$.

So, the previous claim is equivalent to the following. For all $\gamma \in \mathcal{S}_k$, there exists $H \subseteq E_{p,k}$, for some $p \geq 1$, with $\overline{\sigma_0(H \cap E_{1,k})} = \overline{1}$ and $\overline{\sigma_p(H)} = \gamma$, where H contains all the edges $(0,k-1) \sim (1,k-1)$, ..., $(p-2,k-1) \sim (p-1,k-1)$.

The case $\gamma = \overline{1}$ is trivial. Suppose $\gamma = (\gamma_1, \ldots, \gamma_k) \neq \overline{1}$ has p different labels, $1 < l_2 < \cdots < l_p$, $p > 1$. Construct for $1 \leq s \leq p-1$ the set of edges B_s consisting of all the edges

$$\{(s,(k-1)-j) \sim (s+1,(k-1)-j) | 0 \leq j \leq k-1,\ \gamma_{j+1} \in \{1,\ldots,s\}\},$$

and all the edges

$$\{(s+1,(k-1)-j) \sim (s+1,(k-1)-(j-1)) | 1 \leq j \leq k-1,\ \gamma_j, \gamma_{j-1} \notin \{1,\ldots,s\}\},$$

(recall the definition of r-state in page 28). We also take the set B_0 as the already defined set $A_{\overline{1}} \cup \{e_0\}$. The difficulty of understanding the subindices can probably be overcome by looking at Figure 1.5, where we show an example of the construction with the standard form $(1,2,3,1,4,5,6,4,1)$.

We claim that $H = \bigcup_{s=0}^{p-1} B_s$ is such that $\overline{\sigma_1(H \cap E_{1,k})} = \overline{1}$ and $\overline{\sigma_p(H)} = \gamma$. The first part was already discussed.

For the second part, we first observe that any standard form $\beta = (\beta_1, \ldots, \beta_m)$ satisfies the following property: If $\beta_i = \beta_j$, with $1 \leq i < j \leq m$, then there is no

Figure 1.5: Example of the construction in the proof of Lemma 1.8.18.

$1 \leq i' < j' \leq m$ with $\beta_{i'} = \beta j'$ such that $i < i' < j < j'$ and $\beta_i \neq \beta_{i'}$. We call this the *parenthesis* property. This follows as there exists $H_0 \subseteq E_{n,m}$, for some $n \geq 2$, such that $\overline{\sigma_n(H_0)} = \beta$. Then if $\beta_i = \beta_j$, the vertices $(m-1, i)$ and $(m-1, j)$ are in the same connected component of H_0; also, if $\beta_{i'} = \beta_{j'}$, the vertices $(m-1, i')$ and $(m-1, j')$ are in the same connected component of H. Thus, $i < i' < j < j'$ implies that the four vertices are in the same connected component; the path from $(m-1, i)$ to $(m-1, j)$ has to intersect the path from $(m-1, i')$ to $(m-1, j')$. Also, any standard form $\beta = (\beta_1, \ldots, \beta_m)$ satisfies that if $\beta_j = l > 1$, then for all $1 \leq i \leq l$, there exists $j_i < j$ such that $\beta_{j_i} = i$.

Now, returning to the proof, it is just left to prove that $\overline{\sigma_p(H)} = \gamma$.

First, take $1 \leq i < j \leq k$ such that $\gamma_i = \gamma_j = l_s$. We claim that there is a path in B_{s-1} from (s, i) to (s, j). If there was no such a path, there would be a vertex (s, h) with $i < h < j$ and $l_{s'} = \gamma_h < l_s$; but as we pointed out, there would be an h' with $\gamma_{h'} = l_{s'}$ and $h' < i$, that contradicts the parenthesis property. Hence, no such a vertex exists and there is the mentioned path in B_{s-1}. Thus, it is clear that there is a path from (p, i) to (p, j) (that goes through vertices (s, i) and (s, j)).

Now, take $1 \leq i, j \leq k$ such that $l_s = \gamma_i < \gamma_j = l_{s'}$. By construction, a path from the vertex (p, i) to any other vertex (p, j'), $1 \leq j' \leq k$, has to pass through (s, i). By the same reason, a path from vertex (p, j) to any other vertex (p, i'), $1 \leq i' \leq k$, has to go through vertex (s', j). But by construction, there is no path from (s, i) to

(s', j) as $s < s'$. We conclude that there is no path from vertex (p, i) to vertex (p, j), if $\gamma_i \neq \gamma_j$.

Thus, vertices (p, i) and (p, j) are in the same connected component in H if and only if $\gamma_i = \gamma_j$, $1 \leq i, j \leq k$. Therefore $\overline{\sigma_p(H)} = \gamma$ and the proof is complete. □

Theorem 1.8.19. *For $k \geq 2$, the matrices D_k is a nonnegative primitive matrix. Also, for $2 \leq k \leq 8$, D'_k is a nonnegative primitive matrix.*

Proof. From the previous lemma and Theorem 1.8.11, we know that D_k is irreducible. Also Lemma 1.8.16 says that D_k is nonnegative and have its main diagonal entries are positive. Thus, by Lemma 1.8.14, D_k is primitive.

Let $2 \leq k \leq 8$, by Claim 1.8.17 and definition of $\Gamma(D'_k)$, $\Gamma(D'_k) = \Gamma(D_k)$, thus $\Gamma(D'_k)$ is strongly connected and by Theorem 1.8.11, we know that D'_k is irreducible. Also, by the same claim and Lemma 1.8.16, D'_k is nonnegative and has its main diagonal entries positive. Thus, by Lemma 1.8.14, D'_k is primitive. □

Theorem 1.8.20. *Let $k \geq 2$, then*

$$\lim_{m\to\infty} (f(\mathcal{F}_m^{k-1}))^{1/m} = \rho(D_k).$$

Proof. We know by our discussion in page 47 that

$$f(\mathcal{F}_{m+1}^{k-1}) = a_k D_k^m \vec{1}.$$

Now, we apply the Perron-Frobenius Theorem to obtain the following result.

$$\begin{aligned} \lim_{m\to\infty} \frac{f(\mathcal{F}_{m+1}^{k-1})}{\rho(D_k)^m} &= \lim_{m\to\infty} \frac{a_k D_k^m \vec{1}}{\rho(D_k)^m} \\ &= a_k \left(\lim_{m\to\infty} \frac{D_k^m}{\rho(D_k)^m} \right) \vec{1} \\ &= a_k L \vec{1}. \end{aligned}$$

where $L = zy^t$, $D_k z = \rho(D_k) z$, $D_k^t y = \rho(D_k) y$, $z > 0$, $y > 0$ and $z^t y = 1$. The last equation follows from Theorem 1.8.13 part (d) and Theorem 1.8.19

Notice that $a_k > 0$ and $L > 0$ thus the real number $\theta = a_k L \vec{1}$ is positive and we obtain

$$\lim_{m\to\infty} \frac{f(\mathcal{F}_{m+1}^{k-1})}{\rho(D_k)^m} = \theta > 0.$$

Then

$$\lim_{m\to\infty} f(\mathcal{F}_{m+1}^{k-1})^{1/(m+1)} = \lim_{m\to\infty} \theta^{1/(m+1)} \rho(D_k)^{m/(m+1)} = \rho(D_k).$$

□

Forests

Value k	Spectral radius of D_k	Lower bound $\rho(D_k))^{1/(k-1)}$
2	3.41421356237309	3.41421356237309
3	12.44247067856202	3.52738864864109
4	45.81188435340786	3.57815694155781
5	1.691398651379422e+02	3.60629703655053
6	6.251093502414258e+02	3.62402511724480
7	2.311325901206500e+03	3.63616457160850
8	8.547958845274989e+03	3.64497565338648

Table 1.5: This table displays the spectral radius of D_k and the value $\rho(D_k))^{1/(k-1)}$ for $2 \leq k \leq 8$.

A similar proof gives the following

Theorem 1.8.21. *Let $8 \geq k \geq 2$, then*

$$\lim_{m\to\infty} (\alpha(\mathcal{F}_m^{k-1}))^{1/m} = \rho(D'_k).$$

Finally, using Theorem 1.8.2 and Theorem 1.8.3 together with the last two theorems we obtain the following

Corollary 1.8.22. *For any fixed $k \geq 2$, we have that*

$$\rho(D_k)^{1/(k-1)} \leq \lim_{n\to\infty} f(n)^{1/n^2},$$

and for $2 \leq k \leq 8$

$$\rho(D'_k)^{1/(k-1)} \leq \lim_{n\to\infty} \alpha(n)^{1/n^2}.$$

The programs `indices.c` in Section B.1 and `matrix2.c` in Section D.2 can generate the matrices D_k and D'_k for small values of k. We did this and then we used `matlab` to obtain the values shown in Table 1.5 and Table 1.6. Observe that the lower bound for $k = 2$ in Table 1.5 is $3.41421356237309 \approx 2 + \sqrt{2}$, which corresponds to Theorem 1.8.1; and the lower bound for $k = 3$ in Table 1.6 is $3.2256975738518 \approx \sqrt{(13 + \sqrt{61})/2}$, which corresponds to Corollary 1.8.4. Now, we have the main Theorem of the section.

Theorem 1.8.23.

$$\begin{aligned} 3.64497565338648\ldots &\leq \lim_{n\to\infty} f(n)^{1/n^2} \\ 3.41358097503492\ldots &\leq \lim_{n\to\infty} \alpha(n)^{1/n^2}. \end{aligned}$$

Acyclic orientations

Value k	Spectral radius of D'_k	Lower bound $\rho(D'_k))^{1/(k-1)}$
2	3	3.00000000000000
3	10.405124837953	3.22569757385180
4	36.28240963973893	3.31053899212290
5	1.266819950037992e+02	3.35489338669197
6	4.425006985929521e+02	3.38207155765814
7	1.545892644925946e+03	3.40039906808105
8	5.400966279211179e+03	3.41358097503492

Table 1.6: This table displays the spectral radius of D'_k and the value $\rho(D'_k))^{1/(k-1)}$ for $2 \leq k \leq 8$.

1.9 Conclusion and Open Problems

To sum up, the above arguments show that

$$\frac{7}{3} \leq \lim_{n\to\infty} (\alpha_0(n))^{1/n^2} \leq \lim_{n\to\infty} (t(n))^{1/n^2} \approx 3.209912556,$$
$$3.41358097503492\ldots \leq \lim_{n\to\infty} (\alpha(n))^{1/n^2} \leq 3.5632215047716\ldots,$$

and

$$3.64497565338648\ldots \leq \lim_{n\to\infty} (f(n))^{1/n^2} \leq 3.7469814013994\ldots.$$

We close by highlighting a tantalising problem on trees and acyclic orientations, namely which of $\alpha(n)$ and $t(n)$ is bigger. We know by Theorem 1.8.23 and (1.4.1) that

$$\lim_{n\to\infty} (t(n))^{1/n^2} < \lim_{n\to\infty} (\alpha(n))^{1/n^2}.$$

So, for some big N and all $n \geq N$, $t(n) < \alpha(n)$. Also, we have checked using table 1.2 that $t(n) < \alpha(n)$ for $2 \leq n \leq 7$. Therefore, it is very likely that $\alpha(n)$ is strictly bigger than $t(n)$ for all n, but we do not see a way to prove this.

A more general form of this problem is to decide for which graphs the number of acyclic orientations is bigger than the number of spanning trees. The extreme cases are the complete graph K_n, where the number of spanning trees, n^{n-2}, is more than the number of acyclic orientations, $n!$, and the cycle C_n, where the number of spanning trees is n and the number of acyclic orientations is $2^n - 2$.

Consider the evaluation of the Tutte polynomial at $(2, 0)$ and $(0, 2)$, which are dual evaluations in the sense of **??**, for the complete graph K_n and the cycle C_n. These

quantities are $T(K_n;2,0) = n!$, $T(K_n;0,2) = (n-1)!2^{\frac{n(n-1)}{2}-n}$, $T(C_n;2,0) = 2^n - 2$ and $T(C_n;0,2) = 1$, so $T(K_n;0,2) \geq T(K_n;1,1)$ and $T(C_n;2,0) \geq T(C_n;1,1)$. This suggest that either the number of the acyclic orientations or the number of totally cyclic orientations should be greater that the number of spanning trees. In other words we believe that the following is true

Conjecture 1.9.1. *Let G be a connected graph with no loops and no edges belonging to every spanning tree, then*

$$\max\{T(G;2,0), T(G;0,2)\} \geq T(G;1,1).$$

An even stronger form of this conjecture is that for this same class of graphs the Tutte polynomial is a convex function on the line segment joining $(2,0)$ to $(0,2)$.

Chapter 2

The chip firing game and the Tutte polynomial

We have considered approximations for some values of the Tutte polynomial on the square lattice. Now, we turn to a new interpretation of the Tutte polynomial for general graphs. This interpretation consists of a process on the given graph, that is called the chip firing game which since its appearance in 1983 has been proved to be fruitful for different fields such as graph theory, algebraic combinatorics, self-organized criticality and statistical physics.

First we describe the game in one of its versions. Then we restate some known properties of the critical configurations of the game and prove them in a slightly different way by making use of the theory of greedoids. In section 2.5 we prove a relation of the chip firing process with the Tutte polynomial. More precisely, that the generating function of the critical configurations is an evaluation of the Tutte polynomial along the line $\{(1, y)|y \in \mathbb{R}\}$. This result has been published in [**?**].

We close with two applications of the main result. One is another approach to the interpretation of abelian sandpiles as the limit when $Q \to 0$ of the Potts model. The other is a proof for graphic matroids of one long standing conjecture in the context of matroid complexes due to Richard Stanley.

2.1 Introduction

In 1983, Joel Spencer [**?**] was studying the following problem: Given $k \in \mathbb{N}$ and n vectors $\{V_1, \ldots, V_n\}$ in $\mathbb{R}^m$, $\|V_i\|_\infty \leq 1$, $0 \leq i \leq n$, do there exist $e_1, \ldots, e_n$, $e_i \in \{-1, 1\}$ for $1 \leq i \leq n$, such that $\| \sum_{i=1}^{n} e_i V_i\|_\infty \leq kn^{1/2}$?

His answer uses a "balancing game". We have a pile of N chips in the centre of a long path, at each move we take $\lceil N/2 \rceil$ chips to the right and $\lfloor N/2 \rfloor$ to the left.

Now, the game continues with these two new piles and so on. Later, Anderson et. al. in [**?**] extended the game by allowing the player to move one chip at a time either to the right or to the left, and starting in an arbitrary distribution of chips.

In 1991, Björner, Lovász and Shor [**?**] studied the natural generalisation of this game to simple graphs. We put some chips on each vertex of G; we say that a vertex is *ready* if it has at least as many chips as its degree, in which case we can fire it and the result is that it distributes one chip to each of its neighbours, this can cause another vertex to be ready and so on. This game was called the *chip firing game*. Here, they were mainly interested in the final distribution of chips after a sequence of moves that follow the rules, we called this a legal game, and the duration of the process. Even though they were working in the context of simple graphs, exactly the same analysis can be done for graphs with multiple edges, and we state their main results in this more general context.

Theorem 2.1.1 (Björner, Lovász and Shor)**.** *Given a connected multi-graph and an initial distribution of chips, either every legal game can be continued indefinitely, or every legal game terminates after the same number of moves with the same final position. The number of times a given vertex is fired is the same in every legal game.*

Theorem 2.1.2 (Björner, Lovász and Shor)**.** *Given a connected multi-graph G and an initial distribution of N chips:*

1. *If $N > 2|E(G)| - |V(G)|$, then the game is infinite.*
2. *If $|E(G)| \leq N \leq 2|E(G)| - |V(G)|$, then there exists an initial configuration which terminates after a finite number of firings and also one which continues indefinitely.*
3. *If $N < |E(G)|$, the game is finite.*

It was Norman Biggs [**?**] who came up with a process that was related to this game and to what is known in self-critical systems as Abelian sandpiles. In this game we also have a graph G, but this time we are given a special vertex q. The rules of this new game are as above for every vertex except for q, but q has a debit of chips equal to the number of chips on the graph and is ready only when every other vertex is not, then q is fired until some vertex is ready. The last rule ensures an infinite game. In [**?**], the game is called the *dollar game*, and dollars are used instead of chips. The vertex q plays the role of the government and the whole game is a simulation of the economy. Here, however, we stick to the term chip firing game for this new game.

2.2 The chip firing game

We consider throughout this section labelled connected graphs which may have loops and multiple edges, so it is useful for a graph G and a vertex $v \in V(G)$ to denote by $indeg(v)$ twice the number of loops at v, and by $exdeg(v)$ the number of edges that are incident to v but are not loops, so $deg(v) = indeg(v) + exdeg(v)$. Also, for $v, w \in V(G)$, we define $\nu(v, w)$ to be the number of edges joining v and w.

We restate the definition of the chip firing game with some mathematical notation. Let G be a graph and $q \in V(G)$. A *configuration* is a function

$$\theta : V(G) \to \mathbb{Z}$$

where $\theta(v) \geq 0$ for all $v \neq q$ and $\theta(q) = -\sum_{v \neq q} \theta(v)$. A vertex $v \neq q$ is *ready* if $\theta(v) \geq deg(v)$, q is *ready* if every other vertex is not ready. For a configuration θ and a ready vertex w, firing w results in the new configuration θ' defined by:

$$\theta'(v) = \begin{cases} \theta(v) + \nu(v, w), & \text{if } v \neq w, \\ \theta(w) - exdeg(w), & \text{if } v = w. \end{cases}$$

Notice that if we have a loop at vertex w, a firing of the vertex w sends two chips through the loop that return again to the vertex, so loops do not affect the redistribution of chips, but to define the process for graphs with loops is important as we are working with the Tutte polynomial.

An example of the firing of two vertices is shown in Figure 2.1. The two vertices with a circle around them are fired one after the other and the number next to each vertex corresponds to its number of chips.

Figure 2.1: An example of the firing of two vertices in a chip firing game.

We can express the firing rule by means of the Laplace matrix of the graph. For a graph G, its *Laplace matrix* Q is the $|V(G)| \times |V(G)|$ matrix whose rows and columns are indexed by the vertices of G and

$$Q_{v,w} = \begin{cases} -\nu(v,w), & \text{if } v \neq w, \\ exdeg(w), & \text{if } v = w. \end{cases}$$

Then, if we consider a configuration θ as a vector in $\mathbb{R}^{|V|}$ indexed by the vertices, firing vertex w corresponds to taking the new vector $\theta - Qe_w$, where the vector e_w is zero in every entry except in entry w that is 1. And in general, a sequence s of firings can be represented as $\theta - Q[s]$, where the value of $[s]$ at entry w is the number of times we have fired vertex w. Of course, all this can be done only if the firings are legal.

A *legal sequence* for a configuration $\theta = \theta_1$ is a sequence σ of vertices (v_1 ,... , v_k), such that v_1 is ready in θ_1 and at each moment i, v_i is ready in θ_i, and θ_i is obtained from θ_{i-1} by firing v_{i-1}, $2 \leq i \leq k$, so a legal sequence is a sequence of vertices that can be fired in a game with starting configuration θ. If we can go from an initial configuration θ to a configuration θ' by a legal sequence we write $\theta \to \theta'$. A configuration θ is *stable* if for every vertex $v \neq q$, $\theta(v) < deg(v)$, so, in a stable configuration, the only vertex that is ready is q. Suppose we start with a configuration θ and after a legal sequence of length k, where $k \geq 1$ if the graph does not consists just of loops, we arrive at the same configuration, we say that θ is a *recurrent* configuration. A stable configuration that is also recurrent is called a *critical* configuration. The following statement is known but we supply a proof for the sake of completeness.

Theorem 2.2.1. *For a connected graph G and initial configuration θ, there exists a legal firing which produces a critical configuration.*

Proof. This follows by the pigeon-hole principle. We start firing vertices as long as we obey the rules of the game. First note that the vertices $V(G) \setminus q$ cannot be fired infinitely without firing q. Such a sequence of firings would imply that a vertex is fired infinitely often and, because there is only a finite number of chips, so are all its neighbours. As G is connected, this argument implies that a neighbour of q is fired infinitely often but that would mean an infinite loss of chips and this contradicts the finiteness of the number of chips. Therefore, at some moment we have to fire q and we have arrived at a stable configuration. Now we can fire q as many times as necessary and start again until another stable configuration is reached and so on.

But the number of stable configurations is finite, therefore at some point we have to repeat one, say θ'. Then, the legal sequence from θ to the first time we obtain θ' satisfies the statement. □

2.3 Greedoids with repetition

Before continuing with the chip firing game we summarise some results of the theory of greedoids with repetition. Greedoids were invented around 1980 by B. Korte and L. Lovász as a generalization of the matroid concept within the context of combinatorial optimization. A greedoid is defined as a collection of ordered sets satisfying a version of the matroid axioms where the order of the sets is important, see [**?**]. Greedoids with repetition were introduced by allowing that these ordered sets have repetition of elements. For more about greedoids see [**?**] and for greedoids with repetition see [**?**]. Here we follow [**?**] as we just need results that deal with the chip firing game.

Let E be a finite set of cardinality n. A *word* over E is any finite sequence of elements of E and we denote the set of all possible words by E^*. A *language* $\mathcal{L}$ over E is any non-empty finite subset of E. A *subword* v of a word w is any subsequence of w. For a word w, we denote by $|w|$ the length of w, and by $[w]$ its *score*, that is, the vector in $\mathbb{Z}^n$ which is defined by

$$[w]_i = k, \quad \text{if } i \in E \text{ occurs } k \text{ times in } w.$$

A language $\mathcal{L}$ is *left-hereditary* if whenever a word w belongs to $\mathcal{L}$, then any prefix of w also belongs to $\mathcal{L}$. The language is *locally free* if the following holds:

- Let $w \in \mathcal{L}$ and $i \neq j$ be two elements of E such that wi and wj belong to $\mathcal{L}$, then wij also belongs to E.

We said that $\mathcal{L}$ is *permutable* if whenever w and w' are two words in $\mathcal{L}$ such that $[w] = [w']$ and $wi \in \mathcal{L}$, for some $i \in E$, then we also have $w'i \in \mathcal{L}$.

The importance of these properties is that they imply a very strong exchange property on the language $\mathcal{L}$. The property is the following:

ST If w and w' belong to $\mathcal{L}$, then w contains a subword v such that $w'v \in \mathcal{L}$ and $[w'v] = [w] \vee [w']$, where for two vectors a and b in $\mathbb{Z}^n$, $[a] \vee [b]$ denotes the coordinate-wise maximum.

The property (ST) implies, for example, that:

GE If w and w' belong to $\mathcal{L}$ and $|w| > |w'|$, then there exists an element i in w such that $w'i \in \mathcal{L}$.

If $\mathcal{L}$ is just left-hereditary and satisfies this last property, then it is a greedoid with repetition [**?**]. We call the property (GE), the *greedoid property*.

The following lemma states the equivalence between the first three properties and (ST); the proof is in [**?**].

Lemma 2.3.1. *Every locally free, permutable, left-hereditary language has the strong exchange property. Conversely, every language with the strong exchange property is locally free and permutable.*

As with matroids, some greedoids and greedoids with repetition have special elements that work as bases. We say that a word $w \in \mathcal{L}$ is *basic* if it is not the prefix of any other word in $\mathcal{L}$. Suppose that $\mathcal{L}$ has a basic word w, then any other basic word has to have the same length as w and there is no word of greater length by (GE). This common length is called the *rank* of the language. If there is not a basic word in $\mathcal{L}$, then its rank is infinite as every word can be extended indefinitely, but here we just work with languages with finite rank.

If we further suppose that $\mathcal{L}$ is a left-hereditary language that satisfies (ST), then we can conclude that all basic words have the same score. This is all that we need for our analysis of the chip firing game, however, the last observation has been generalised in [**?**] as follows. Two words w and w' in a left-hereditary language are said to be *equivalent*, $w \sim w'$, if for every string v such that $wv \in \mathcal{L}$, we also have $w'v \in \mathcal{L}$. The equivalence classes of this relation are called *flats*. We can assign an order relation to the set of flats of $\mathcal{L}$ by saying that a flat A is a *subflat* of a flat B, if every word in A can be extended to a word in B.

Lemma 2.3.2. *Let $\mathcal{L}$ be a locally free, permutable, left-hereditary language of finite rank and w and w' be two words in $\mathcal{L}$. Then, the flat defined by w is a subflat of the flat defined by w' if and only if $[w] \leq [w']$. In particular, $w \sim w'$ if and only if $[w] = [w']$.*

Again, the proof can be found in [**?**].

2.4 Critical configurations

We now return to the chip firing game to analyse very closely the critical configurations of a given graph. The results we are presenting here have been proved in

different ways. One approach is to use direct counting arguments as in [**?**]. Another is to apply group theory arguments to the Picard group as in [**?**]. We use an approach hinted at in [**?**] based on the theory of greedoids with repetition. First we notice that Theorem 2.2.1 allows us to define the following language:

$\mathcal{L}(\theta) = \{\sigma | \sigma$ is a legal firing such that starting from θ it can be extended to a legal firing that produces a critical configuration and any configuration produced during this firing is not critical$\}$.

We now apply the theory of greedoids with repetition to $\mathcal{L}(\theta)$.

Lemma 2.4.1. *For a graph G and initial configuration θ, the language $\mathcal{L}(\theta)$ is locally-free, permutable and left-hereditary.*

Proof. Clearly, if $\sigma \in \mathcal{L}(\theta)$ and σ' is a prefix of σ, then σ' can be extended to σ and then to a critical configuration, so $\mathcal{L}(\theta)$ is left-hereditary.

Note that if we have two legal firings σ and σ' with the same score, that is $[\sigma] = [\sigma']$, then $\theta - Q[\sigma] = \theta - Q[\sigma']$, where Q is the Laplace matrix, and both sequences lead to the same configuration. Therefore, if there exists a vertex w such that σw is a member of $\mathcal{L}(\theta)$, $\sigma' w$ can be extended to produce a critical configuration as σw and $\sigma' w$ lead to the same configuration. Also after the firing of σ we cannot arrive at a critical configuration, as σw is in the language, so σ' cannot arrive at a critical configuration. We conclude that the language is permutable.

Let $\sigma \in \mathcal{L}(\theta)$. Suppose there exist two different vertices w and w' such that σw and $\sigma w'$ can be extended to produce a critical configuration; this implies $w, w' \neq q$ by the rules, so after the legal firing σ we cannot arrive at a critical configuration, that is stable by definition. Also, vertices w and w' have at least as many chips as their degrees, so we can fire both and the firing $\sigma w w'$ produces a configuration that can now be extended to a critical configuration by Theorem 2.2.1. This proves $\mathcal{L}(\theta)$ is locally-free and completes the proof of the lemma. □

Theorem 2.4.2. *For a graph G and a configuration θ, there exists a unique critical configuration c such that $\theta \to c$. In particular, if θ is critical, then $\theta = c$.*

Proof. We know by the proof of Theorem 2.2.1 that $\mathcal{L}(\theta)$ is not empty. This language has as basic words all the legal firings that starting from θ finish in a critical configuration and they do not pass through any other critical configuration; lets take one of these configurations, say c.

As $\mathcal{L}(\theta)$, by Lemma 2.3.1 has the greedoid exchange property, all basic words have the same length; also any two basic words are equivalent, thus, by Lemma 2.3.2, they have the same score and lead to the same critical configuration. We conclude that this c is unique. □

For a critical configuration θ, we called the basic words of $\mathcal{L}(\theta)$ *critical sequences*. Critical sequences have been studied before [**?**, **?**] for loopless graphs with multiple edges. We include a minor extension of Lemma 3.6 from [**?**, pp. 5] that we require later.

Lemma 2.4.3. *Let G be a graph and c be a critical configuration, then any critical sequence consists of firing all the vertices of G exactly once.*

Proof. For a graph G with loops and a critical starting configuration c, we have a legal sequence that arrives at the same starting configuration, but as G is connected this means that every vertex must be fired at least once. Not firing a vertex means not firing any of its neighbours and so on. During a firing, once two chips travel through a loop, they stay in the vertex of that loop and because c is a recurrent configuration, these chips should have been there since the beginning, therefore, for all $v \neq q$, $c(v) \geq indeg(v)$. We construct the graph $\hat{G}$ from G by removing all the loops and the configuration c' from c by defining $c'(v) = c(v) - indeg(v)$ for all $v \neq q$, the above argument proves that c' is in fact a configuration. It is clear that any legal sequence for c in G is one for c' in $\hat{G}$ and so c is a critical configuration for G if and only if c' is one for $\hat{G}$. The result of the theorem follows now from the loopless case in [**?**]. □

For a configuration θ, we define its *weight*, $w(\theta)$, to be

$$w(\theta) = \sum_{v \neq q} \theta(v).$$

If θ is critical, we also define its *level* as

$$\text{level}(\theta) = w(\theta) - |E(G)| + deg(q).$$

This definition seems less natural but shortly we explain why it is the right quantity to be associated to a critical configuration.

We want to view a critical configuration as a configuration in the general chip firing game of Björner, Lovász and Shor in order to apply the powerful Theorem 2.1.1. For a graph G and a stable configuration θ of the chip firing game, we get a configuration for the general chip firing game by redefining the value of $\theta(q)$ to be $deg(q)$; this configuration θ_q of the general chip firing game is called the *extension* of θ.

Lemma 2.4.4. *Let G be a graph and θ be a stable configuration for the chip firing game. If θ is critical, then the corresponding extension θ_q produces an infinite chip firing game.*

Proof. We know that θ has a firing consisting of firing each vertex once, and the vertex q is the first in this sequence. This will produce the same configuration θ. If we mimic this firing for θ_q, that has enough chips to start this sequence, we arrive at the same configuration θ_q. We keep on repeating this sequence and we get the infinite firing. □

Theorem 2.4.5. *Let G be a graph and θ a critical configuration, then*

$$0 \leq \text{level}(\theta) \leq |E(G)| - |V(G)| + 1.$$

Proof. By Lemma 2.4.4, we can associate to θ a configuration θ_q for the general chip firing game that produce an infinite game, so, by Theorem 2.1.2, the total number of chips on θ_q is at least $|E(G)|$. But θ_q and θ have the same number of chips at every vertex except for q, so $w(\theta) + deg(q) \geq |E(G)|$ and then $\text{level}(\theta) \geq 0$.

For the right inequality just note that as every critical configuration is stable,

$$w(\theta) \leq \sum_{v \neq q} (deg(v) - 1) = 2|E(G)| - deg(q) - |V(G)| + 1.$$

□

As an example we consider the graph G in Figure 2.2. If c is one critical configuration, then we consider it as a point of $\mathbb{R}^2$, we simply take the point $(c(a), c(b))$ as c. In Figure 2.3, we have drawn all the critical configurations of G using this identification together with their corresponding level.

Figure 2.2: A graph used to show the levels of its critical configurations

Figure 2.3: The critical configurations of the graph in Figure 2.2.

2.5 Critical configurations and the Tutte polynomial

Already in [**?**, **?**] it is proved that for a graph G, the number of critical configurations equals the number of spanning trees. Also for a loopless graph, using the proof of Theorem 2.1.2 in [**?**], we can infer that there is an injection from the set of acyclic orientations with exactly one predefined source to the set of critical configurations of level 0. These quantities are, respectively, evaluations of the Tutte polynomial at the points $(1,1)$ and $(1,0)$. So, it was natural to try to count the number of critical configurations of a given level and expect a relation with the Tutte polynomial. For a graph G, we define, for every $i \geq 0$, c_i to be the number of critical configurations with level i. Of course, by Theorem 2.4.5, $c_i = 0$ for all $i > |E(G)| - |V(G)| + 1$. We now take the generating function of the critical configurations, that is, the polynomial

$$P_q(G;y) = \sum_{i=0}^{|E(G)|-|V(G)|+1} c_i y^i.$$

All these definitions were proposed by Biggs [**?**]. We now prove his conjecture that the above polynomial is an evaluation of the Tutte polynomial.

Theorem 2.5.1. *For a graph G and a vertex q, we have that the generating function of the critical configurations is the Tutte polynomial of G along the line $x = 1$, that is,*

$$P_q(G;y) = T(G;1,y).$$

Thus, the polynomial on the left side is independent of the choice if the special vertex q.

Proof. Let G be a graph and q a vertex in $V(G)$. Take an edge e that has q as an endpoint, and let u be the other end, where u could be equal to q. First, we prove some relations among critical configurations and then we use induction to prove the statement.

Let us suppose G has e as its only edge. Then there are just two possible cases. If e is an isthmus, the only critical configuration θ has value zero at u and $\text{level}(\theta) = -|E(G)| + deg(q) = 0$, so $P_q(G;y) = 1 = T(G;1,y)$. If e is a loop, the only critical configuration θ, and in fact the only configuration, assigns a value of 0 to q, and $\text{level}(\theta) = -|E(G)| + deg(q) = -1 + 2 = 1$. So, $P_q(G;y) = y = T(G;1,y)$.

Now suppose G has at least 2 edges. We now analyse the possibilities for e. First, the case that the edge e is a coloop. From a critical configuration θ' of G/e, we get a

configuration θ of G by defining $\theta(w) = \theta'(w)$ for every $w \neq u, q$ and $\theta(u) = deg(u)-1$; the value at q is determined by using the definition of configuration. The configuration θ is critical as after firing q, we can fire u and repeat a critical sequence for θ', so this map sends a critical configuration of G/e into a critical configuration of G. We claim that this map is a bijection. Let θ be a critical configuration of G. Define the configuration θ' of G/e by $\theta'(w) = \theta(w)$ for every vertex $w \neq q$ of $V(G/e)$. Now, if we replace the first two vertices in a critical sequence for θ, that must be q and u, by q, we get a critical sequence for θ' and so θ' is a critical configuration. As every critical configuration of G must assign a value of $deg(u) - 1$ to the vertex u, the map is a bijection and

$$\begin{aligned}
\text{level}(\theta') &= \sum_{w \neq q,u} \theta'(w) - |E(G/e)| + deg_{G/e}(q) \\
&= \sum_{w \neq q,u} \theta(w) - (|E(G)| - 1) + (deg_G(q) + deg_G(u) - 2) \\
&= \sum_{w \neq q,u} \theta(w) + deg_G(u) - 1 - |E(G)| + deg_G(q) \\
&= \sum_{w \neq q} \theta(w) - |E(G)| + deg_G(q) \\
&= \text{level}(\theta),
\end{aligned}$$

so we conclude that the polynomials are equal,

$$P_q(G; y) = P_q(G/e; y). \tag{2.5.1}$$

Second the case that the edge e is a loop. A similar argument as above shows that if we define, from a critical configuration θ' of $G \setminus e$, a configuration θ of G by $\theta(w) = \theta'(w)$ for every $w \neq q$, we obtain a critical configuration. This map is a bijection and

$$\begin{aligned}
\text{level}(\theta') &= \sum_{w \neq q} \theta'(w) - |E(G \setminus e)| + deg_{G-e}(q) \\
&= \sum_{w \neq q} \theta(w) - (|E(G)| - 1) + (deg_G(q) - 2) \\
&= \text{level}(\theta) - 1,
\end{aligned}$$

so the two polynomials are related by

$$P_q(G;y) = yP_q(G \setminus e;y). \tag{2.5.2}$$

We finally suppose e is neither an isthmus nor a loop, and G has more than 1 edge. We partition the set of critical configurations of G into two sets, $\mathcal{A}$ the set of critical configurations θ of G with $\theta(u) = deg(u) - 1$, and $\mathcal{A}'$ the rest of the critical configurations.

Now, for θ in $\mathcal{A}$, we associate a critical configuration θ' in G/e just by defining $\theta'(w) = \theta(w)$ for all vertices w different from u or q. To see that θ' is in fact a critical configuration, take a critical sequence for θ and as $\theta(u) = deg(u) - 1$ we can assume that the first two firings are q and then u by Lemmas 2.4.1, that ensure any legal firing is locally free, but after the firing of u the situation in the other vertices is the same in both graphs, so we can reproduce the rest of the sequence in G/e with the same result. Therefore, we arrive at the initial configuration θ' and the construction gives a critical configuration for G/e. This construction can be reversed by defining from a critical configuration θ' in G/e, the configuration in G, $\theta(w) = \theta'(w)$ for all $w \neq u$ and $\theta(u) = deg(u) - 1$. This configuration is critical in G as we can fire q and then u. We are now in the same situation as firing q in G/e, so we can follow a critical sequence for θ'; the final configuration is going to agree with θ in every vertex different from u. Furthermore, after firing q in G, u has $deg(u) - 1 + \nu(q,u)$ chips, after the firing of u, u loses $degex(u)$ chips and after the firing of all its neighbours, u gains $degex(u) - \nu(q,u)$ chips, so the final total is $deg(u) - 1$ chips. We conclude that θ constructed in this way is critical. Thus, we have a bijection and for corresponding θ and θ'

$$\begin{aligned}
\text{level}(\theta) &= \sum_{w \neq q} \theta(w) - |E(G)| + deg_G(q) \\
&= \sum_{w \neq q,u} \theta'(w) + deg_G(u) - 1 - (|E(G/e)| + 1) + deg_G(q) \\
&= \sum_{w \neq q,u} \theta'(w) - |E(G/e)| + deg_{G/e}(q) \\
&= \text{level}(\theta').
\end{aligned}$$

It is clear that using the same argument as above, the elements in $\mathcal{A}'$ can be placed in a bijective correspondence with the critical configurations of $G \setminus e$. First, for a critical configuration θ in G, define a configuration θ' in $G \setminus e$ by $\theta'(w) = \theta(w)$, i.e. $\theta' = \theta$, for every $w \neq q$ and now note that any legal firing in $G \setminus e$ is a legal firing

in G. After q fires, $\theta'(u)$ is one fewer than $\theta(u)$; but in compensation $deg_{G\setminus e}(u)$ is now smaller also. Moreover, for θ in $\mathcal{A}'$ and the corresponding θ', we have

$$\begin{aligned} \text{level}(\theta) &= \sum_{w\neq q} \theta(w) - |E(G)| + deg_G(q) \\ &= \sum_{w\neq q} \theta'(w) - (|E(G\setminus e)| + 1) + (deg_{G\setminus e}(q) + 1) \\ &= \text{level}(\theta'). \end{aligned}$$

We conclude that

$$P_q(G; y) = P_q(G \setminus e; y) + P_q(G/e; y). \tag{2.5.3}$$

It is clear that by this process we can arrive at the first two trivial cases. Now, use induction on the number of edges of the graph and the recursions (2.5.1), (2.5.2), and (2.5.3) to apply the Recipe Theorem **??**, thus, we have that $P_q(G; y) = T(G; 1, y)$. □

Using the same identification as the one in Figure 2.3, we explain by means of Figure 2.5, how the inductive step of the proof works in a particular example. The graph to be considered here has been drawn in Figure 2.4.

Figure 2.4: The graph used to exemplify the proof of Theorem 2.5.1.

2.6 Abelian sandpiles and the Potts model

Self-organised criticality is a concept widely considered in various domains since Bak, Tang and Wiesenfeld [**?**] introduced it ten years ago. One of the paradigms in this framework is the abelian sandpile model, introduced by Dhar [**?**].

Figure 2.5: Exemplification of the inductive step of Theorem 2.5.1.

We start by recalling the definition of the general abelian sandpile model on a set of N sites labelled 1, 2, ..., N, that we referred to as the system. At each site the height of the sandpile is given by an integer h_i. The set $\vec{h} = \{h_i\}$ is called the *configuration* of the system. For every site i, a threshold H_i is defined; configurations with $h_i < H_i$ are called *stable*. For every stable configuration, the height h_i increases in time at a constant rate (depending on i), this is called the *loading* of the system. This loading continues until at some site i, its height h_i exceeds the threshold H_i, then the site i topples and all the values h_j, $1 \leq j \leq N$, are updated according to the rule:

$$h_j = h_j - \Delta_{ij}, \quad \text{for all } j, \tag{2.6.1}$$

where Δ_{ij} is an integer matrix satisfying

$$\Delta_{ii} > 0, \quad \Delta_{ij} \leq 0 \quad \text{and} \quad s_i = \sum_j \Delta_{ij} \geq 0.$$

If after this redistribution some height exceeds its threshold we apply the toppling rule (2.6.1) and so on, until we arrive at a stable configuration and the loading resumes. When there are two or more possible toppling, we choose one arbitraryly and we continue with the remaining topplings. The sequence of topplings is called an *avalanche*. We assume that an avalanche is "instantaneous", that is, no loading occurs during an avalanche.

The value s_i is called the *dissipation* at site i. It may happen that an avalanche continues without end. We can avoid this possibility by requiring that from every

non-dissipative site i, i.e. $s_i = 0$, there exists a path to a *dissipative* site j, i.e. $s_j > 0$. In other words, there is a sequence $i_0, \ldots, i_n$, with $i_0 = i$, $i_n = j$ and $\Delta_{i_{k-1},i_k} < 0$, for $k = 1, \ldots, n$. In this case we said that the system is *weakly-dissipative* [**?**]. From now on, we assume that the system is always weakly-dissipative.

When the matrix Δ_{ij} is symmetric and the loading of the system at site i equals the dissipation at i, the abelian sandpile model coincides with the chip-firing game on a graph [**?**]. We now explain this.

Every site of the abelian sandpile model corresponds to a vertex in a graph G containing $N+1$ vertices, that is the number of vertices is one more than the number of sites in the system. We label the vertices 0,1, ..., N. The graph has multiple edges, and the number of edges between sites i and j, i and j both nonzero, equals $|\Delta_{ij}|$. For all $i \neq 0$, we connect site i to site 0 using $|\sum_{j=1}^{N} \Delta_{ij}|$ edges.

Every vertex i, $1 \leq i \leq N$, has a number of chips θ_i that represents its height h_i (when seen as a site of the system) at every moment of time and vertex 0 has a negative number of chips given by $\sum_{i=1}^{N}(-h_i)$. A toppling at site i corresponds to *firing* vertex i. The loading of the system is represented by the firing of the vertex 0, in this case the height of site i (its number of chips in G) is increased by the number of edges from 0 to i. The vertex 0 may (must) fire only when no ordinary vertex can fire.

Then, the model has an important abelian property, namely the stable configuration of the system after an avalanche, and the number of breaks at any site during an avalanche, do not depend on the order of breaks during the avalanche.

We recall that the partition function of the Potts model is the evaluation of the Tutte polynomial given by **??** in Chapter **??**.

More generally, for the random cluster model, see [**?**], we have:

$$Z_{\mathrm{RC}}(G; p, Q) = p^{r(E)} q^{r^*(E)} Q^{k(G)} T(G; 1 + \frac{Qq}{p}, \frac{1}{q}), \tag{2.6.2}$$

where $q = 1 - p$.

Several authors, see Wu [**?**], have considered the formal limiting behaviour of the Q-state Potts model as $Q \to 0$. This makes more sense in the context of the random cluster model which we recall is defined for all $Q > 0$. Let us now consider this convergence in more detail.

Suppose in the random cluster model, p and Q both tend to zero with p/Q kept constant at 1. Then easy calculations show that in this case

$$\lim \frac{Z_{\mathrm{RC}}(G; p, Q)}{p^{r(E)}} = T(G; 2, 1).$$

In other words the limit is the number of forests of G.

There are various other cases to consider.

a) For the part of the degenerate hyperbola $H_0 = \{(x,y)|(x-1)(y-1) = 0\}$, consisting of $y = 1$, $x \geq 1$, it is clear that if we let $Q \to 0$ in such a way that Q/p is fixed at $\alpha > 0$, then in the random cluster model

$$x = 1 + \frac{Qq}{p} \to 1 + \alpha$$
$$y = \frac{1}{1-p} \to 1$$

and so

$$p^{-r}(1-p)^{-|E|+r(E)} Z_{\mathrm{RC}}(G;Q,p) \to T(G;1+\alpha,1).$$

We have already mentioned the case $\alpha = 1$ where the limit is the number of forests. More generally we have the interesting specialisation

$$\lambda^r T(G;1+1/\lambda,1) = i(P(G),\lambda)$$

where i is the Ehrhart polynomial of a particular family of zonotopes $P(G)$ determined by a graph G. See [**?**].

b) If $Q \to 0$ with p fixed then

$$\lim_{Q\to 0} Z_{\mathrm{RC}}(G;p,Q) = cT(G;1,\frac{1}{1-p})$$

where c is a constant. In other words we are getting

1) the reliability probability,

2) the chip-firing game, that is the abelian sandpile model,

as two different realisations of this limiting behaviour.

This interpretation of the sandpile model has been considered before by Majumdar and Dhar [**?**] but their approach was quite different.

2.7 Chip firing and shellable complexes

In this section we use the chip firing game and its relation with the Tutte polynomial given by Theorem 2.5.1 to prove a long standing conjecture of Richard Stanley [**?**] in the particular case of cographic matroids.

2.7.1 Simplicial complexes

A *simplicial complex* Δ is a collection of subsets of a set of vertices V such that

- if $F \in \Delta$ and $G \subseteq F$, then $G \in \Delta$,
- for all v in V, $\{v\} \in \Delta$.

The elements of Δ are called *faces*. Maximal faces are called *facets* and if all the facets have the same cardinality, then Δ is called a *pure* simplicial complex. The *dimension* of a face is its size minus one and the dimension of a pure simplicial complex is the dimension of any of its facets.

Associated to any simplicial complex Δ of dimension $d-1$ we have its *face vector* or *f-vector* $(f_0,\ f_1,\ \ldots,\ f_d)$, where f_i is the number of faces of size i of Δ. The generating function of the f–vector, or *face enumerator* is defined by

$$f_\Delta(x) = \sum_{i=0}^{d} f_i x^{d-i},$$

and the *Euler characteristic* of Δ is defined by

$$\chi(\Delta) = -f_0 + f_1 - \ldots (-1)^{d-1} f_d.$$

Note that

$$(-1)^{d-1}\chi(\Delta) = f_\Delta(-1).$$

For a pure simplicial complex Δ, a *shelling* is a linear order of the facets F_1, F_2, $\ldots$, F_t such that, for $1 \leq l \leq t$, F_l meets the complex generated by its predecessors, called Δ_{l-1}, in a non-void union of maximal proper faces. A complex is said to be *shellable* if it is pure and admits a shelling.

Define, for $1 \leq l \leq t$,

$$\mathcal{R}(F_l) = \{x \in F_l |\ F_l \setminus x \in \Delta_{l-1}\},$$

where here $\Delta_0 = \emptyset$. The number of facets such that $|F_l - \mathcal{R}(F_l)| = i$ is denoted by h_i and it does not depend on the particular shelling, see [**?**]. The vector $(h_0,\ h_1,\ \ldots,\ h_d)$ is called the *h–vector* of Δ. The generating function of the h–vector, or *shelling polynomial* is given by

$$h_\Delta(x) = \sum_{i=0}^{d} h_i x^{d-i}.$$

Then, it is well known, see for example [**?**], that the face enumerator and the shelling polynomial satisfy the relation

$$h_\Delta(x+1) = f_\Delta(x),$$

and so, the coefficients satisfy

$$f_k = \sum_{i=0}^{d} h_i \binom{d-i}{k-i},\ k = 0, \ldots, d$$

and

$$h_k = \sum_{i=0}^{d} (-1)^{i+k} f_i \binom{d-i}{k-i},\ k = 0, \ldots, d.$$

2.7.2 Matroid complexes

If $M = (E, \mathcal{B})$ is a matroid of rank r, the family of all independent sets forms a simplicial complex of dimension $r-1$ that we denote by $\Delta(M)$. The facets of $\Delta(M)$ are the bases of the matroid M and therefore, $\Delta(M)$ is pure. Complexes of this kind are called *matroid complexes*. Matroid complexes are known to be shellable, see [**?**], and it is also known [**?**] that the shelling polynomial of $\Delta(M)$ is an evaluation of the Tutte polynomial, that is,

$$T(M; x, 1) = h_{\Delta(M)}(x)$$

and, by duality, we also have

$$T(M; 1, y) = h_{\Delta(M^*)}(y).$$

2.7.3 A conjecture of Richard Stanley

An *order ideal* (or *down-set*) of a poset P is a subset I of P such that if $x \in I$ and $y \leq x$, then $y \in I$. If we take as a poset P the set of all monomials over indeterminates $z_1, \ldots, z_n$ and the order given by divisibility, then an order ideal of P is called a *multicomplex* over $z_1, \ldots, z_n$.

If we form the poset $(\mathbb{N}^n, \leq)$, where $a \leq b$ if $a(i) \leq b(i)$ for $1 \leq i \leq n$, then a multicomplex $\mathcal{M}$ can also be seen as an order ideal of $(\mathbb{N}^n, \leq)$. More explicitly, if $\mathcal{M}$ is a multicomplex over $z_1, \ldots, z_n$, then the image of the function $\mu : \mathcal{M} \to \mathbb{N}^n$ defined by

$$\mu(z_1^{i_1}, \ldots, z_n^{i_n}) = (i_1, \ldots, i_n)$$

is an order ideal of $(\mathbb{N}^n, \leq)$. So, we can use interchangeably both definitions of multicomplex.

A multicomplex whose maximal elements are all of the same rank, where the rank of an element is the sum of the value of its entries, is called *pure*. The vector $(h_0, \ldots, h_d)$, where h_i is the number of monomials of rank i, is the degree sequence of the multicomplex. A vector $(h_0, \ldots, h_d)$ is called a *(pure) O-sequence* if it is the degree sequence of some (pure) multicomplex. R. Stanley proved that the h-vector of a shellable simplicial complex is an O-sequence and made the following conjecture [**?**].

Conjecture 2.7.1. *The h–vector of a matroid complex is a pure O-sequence.*

As a direct application of our Theorem 2.5.1 we have the following

Theorem 2.7.2. *If M is a cographic matroid, then the h-vector of the matroid complex $\Delta(M)$ is a pure O-sequence. In other words the conjecture of Stanley is true for cographic matroids.*

Proof. Let M be a cographic matroid. Then its dual M^* is a graphic matroid and there exists a graph G such that $M^* = M(G)$. Let $|V(G)| - 1 = n$ and $q \in V(G)$. We are going to prove that the set of critical configurations of G, $\mathcal{C}$, can be transformed to be a pure multicomplex whose O-sequence is the h-vector of $\Delta(M)$.

First consider any critical configuration c and suppose that c' is a stable configuration such that $c' \geq c$, here we consider configurations as elements of $(\mathbb{N}^n, \leq)$. We will prove that c' is also critical. Let $d(v) = c'(v) - c(v)$, for $v \neq q$, and σ any critical sequence for c. At each vertex v, we mark $d(v)$ chips as "permanent" and the rest as "movable". Then, starting from c', we can follow the legal firing σ without problem, if we just use the "movable" chips. At the end, as c is critical, we have again $c(v)$ "movable" chips in each $v \neq q$, and also $d(v)$ "permanent" chips that did not move. So, c' is a critical configuration.

From the above argument, we get that $\bigvee_{c \in \mathcal{C}} c$, the coordinate-wise maximum of all elements of $\mathcal{C}$ in $(\mathbb{N}^n, \leq)$ is a critical configuration, namely the configuration defined by

$$c_\perp(v) = deg(v) - 1, \quad v \neq q.$$

Now, let I be the down-set in $(\mathbb{N}^n, \leq)$ generated by $c_\perp$, that is, $a \in I$ if $a \leq c_\perp$. Clearly $\mathcal{C}$ is contained in I. Consider the function $f : I \to \mathbb{N}^n$ given by

$$f(c)(v) = c_\perp(v) - c(v).$$

Note that $f(c) \in I$ and $f(f(c)) = c$. Also it is clear that if c and c' are critical configurations such that $c \leq c'$, then $f(c) \geq f(c')$.

Let $a \in f(\mathcal{C})$ and $b \in \mathbb{N}^n$ such that $b \leq a$, then $\vec{0} \leq b \leq a$ and the images of these elements satisfy $f(a) \geq f(b) \geq f(\vec{0}) = c_\perp$. By the second inequality we conclude that $f(b)$ is a stable configuration. We know that there is a critical configuration c such that $f(a) = c$, so by the preceding argument and the first inequality we get that $f(b)$ is a critical configuration and $b = f(f(b)) \in f(\mathcal{C})$. Therefore $f(\mathcal{C})$ is a multicomplex.

The rank of an element of $(\mathbb{N}^n, \leq)$ is just $\sum_{i=1}^n a(i)$. For a critical configuration c of level i, $f(c)$ has rank

$$\begin{aligned}
\sum_{v \neq q} f(c)(v) &= \sum_{v \neq q} (c_\perp(v) - c(v)) \\
&= \sum_{v \neq q} (deg(v) - 1) - \sum_{v \neq q} c(v) \\
&= 2|E(G)| - deg(q) - |V(G)| + 1 - w(c) \\
&= 2|E(G)| - deg(q) - |V(G)| + 1 - \text{level}(c) - |E(G)| + deg(q) \\
&= |E(G)| - |V(G)| + 1 - i.
\end{aligned}$$

So, critical configurations of the same level are mapped by f to elements of the same rank.

Also, observe that if c is a critical configuration then there exists a critical configuration c^* of minimal level such that $c^* \leq c$. First consider the case when G has no loops. Take a legal sequence for c that makes it recur. We know by Lemma 2.4.3 that this is a permutation of the vertices. We repeat this firing but this time we will label some chips. All the chips sent by q are labelled q. When we fire vertex v_i, we mark any unmarked chip that is moved with the label v_i. Any already marked chip will return to the vertex of its label. This is possible because we have a critical configuration. Clearly, at the end, the number of marked chips that are not labelled q is $|E| - deg(q)$. So these marked chips induce a critical configuration c^* of minimal level (level 0) such that $c^* \leq c$. In the case that G has loops, as shown in Lemma 2.4.3, we can fix two chips for every loop and continue as in the loopless case.

Thus the maximal elements of $f(\mathcal{C})$ are the image under f of the minimal critical configurations, as all these have the same rank, we obtain that $f(\mathcal{C})$ is a pure multicomplex whose pure O-sequence is $(c_{|E(G)|-|V(G)|+1}, \ldots, c_0)$.

To finish the proof, we use Theorem 2.5.1 to get

$$\begin{aligned}\sum_{i=0}^{r} c_i y^i &= T(G;1,y) \\ &= \sum_{i=0}^{r} h_i y^{r-i},\end{aligned}$$

where $r = |E(G)| - |V(G)| + 1$ is the rank of M. Then $c_{r-i} = h_i$, for $0 \leq i \leq r$, as required. □

Chapter 3

An extension of the chip firing game to unimodular matroids

Most of the research on chip firing games on undirected and directed graphs started as a way of analysing dynamic behaviour of processes in other areas, like Markov chains and self-critical systems [**?**, **?**]. Later, Norman Biggs put a particular instance of chip firing games in the context of algebraic graph theory and showed its intrinsic relation with algebraic and combinatorial invariants of graphs, such as the Picard group or the number of spanning trees of the graph, [**?**]. The last chapter showed even more, that this chip firing game on graphs is actually related with the algebraic and topological structure of graphic matroids; specifically, with the Tutte polynomial of the graph and the h-vector of the matroid complex of the graph. So, the natural question was to extend this process to a broader class of matroids. Here we present a first attempt to do this. We have managed to define a chip firing game for regular matroids that when restricted to graphic matroids conserves all the algebraic properties of chip firing on graphs. However, this extension cannot be considered to be total as it is not clear that it also contains the information of the Tutte polynomial or the h-vector of the matroid.

In section 3.1, we give some introduction to the theory of lattices, discrete vector groups, that is used to define a process without the notion of vertex. Section 3.2 gives an extension of the notion of cut and cycle space of graphs to regular matroids. Similarly, section 3.3 extends the notion of integral cut and integral flow to regular matroids to obtain analogue results to the ones in [**?**]. Section 3.4 is devoted to a very general definition of the chip firing process that is due to Björner and Lóvasz. Then we choose a special type of processes to generalise the chip firing game, in section 3.5. Finally, in section 3.6, we apply all this theory to define a chip firing process for

regular matroids and also state the algebraic results that relate this generalisation to the game on graphs.

3.1 Lattice theory

A *lattice* Λ is a subset of $\mathbb{R}^n$ such that:

- $\vec{0} \in \Lambda$;
- if x, $y \in \Lambda$ then $x - y \in \Lambda$;
- Λ has no accumulation points.

That is, Λ is a discrete vector group in a linear manifold $\mathbb{R}^r$ through the origin of $\mathbb{R}^n$, see Figure 3.1 for an example.

It can be proved, see [**?**], that every lattice Λ has a $\mathbb{Z}$–basis, say $\{B_1, \ldots, B_r\}$, so

$$\Lambda = \Lambda(B_1, \ldots, B_r) = \{\sum_{i=1}^{r} \lambda_i B_i |\ \lambda_i \in \mathbb{Z}, i = 1, \ldots, r\}.$$

We represent such a basis by an $r \times n$ matrix B, where the vectors of the basis are the rows of the matrix B.

If B is a basis of a lattice $\Lambda(B)$ and $A \in \mathrm{GL}(r, \mathbb{Z})$, that is, A is an $r \times r$ matrix with integer coefficients and A has determinant ± 1, then it is a standard result, see [**?**], that AB is also a basis. It also can be proved, [**?**], that if B and B' are two bases for a lattice Λ, then there exists $A \in \mathrm{GL}(r, \mathbb{Z})$ such that $B' = AB$. Two lattices $\Lambda(B)$ and $\Lambda(B')$ whose bases are related by $B' = ABO$, where $A \in \mathrm{GL}(r, \mathbb{Z})$ and O is a real orthogonal matrix ($OO^t = I_n$), are called *congruent* lattices, [**?**]. For two congruent lattices $\Lambda(B)$ and $\Lambda(B')$ we have

$$|B'(B')^t| = |ABOO^tB^tA^t| = |BB^t|,$$

where $|\cdot|$ denotes the determinant of a square matrix. Therefore, the quantity $\sqrt{|BB^t|}$ is independent of the choice of the basis of the lattice Λ, and does not change for any lattice congruent to $\Lambda(B)$; this quantity is generally denoted $\det(\Lambda)$. The matrix $Q = BB^t$ is called the *Gram matrix* of $\Lambda(B)$.

For a lattice $\Lambda = \Lambda(B)$, the parallelotope consisting of the points

$$\theta_1 B_1 + \cdots + \theta_r B_r \quad , 0 \leq \theta_i < 1,$$

Figure 3.1: An example of a lattice in $\mathbb{R}^2$. Also a basis of the lattice and its associated fundamental parallelotope.

is called the *fundamental parallelotope* of $\Lambda(B)$. Geometrically speaking, $\det(\Lambda)$ is the (common) volume of each fundamental parallelotope, see Figure 3.1.

Also important is the *Voronoi polyhedron* of $\Lambda(B)$ defined by

$$\mathrm{Vor}(\Lambda) = \{v \in \mathbb{R}^n |\ (v,v) \leq (v-x, v-x) \text{ for all } x \in \Lambda \setminus \{\vec{0}\}\}.$$

The Voronoi polyhedron of a lattice Λ is well understood, and there is a good characterisation of its facets. A vector $v \in \Lambda$ is called *relevant* if it is orthogonal to a facet of $\mathrm{Vor}(\Lambda)$, that is, if $v \neq \vec{0}$ and the hyperplane given by

$$\mathcal{H} = \{x \in \mathbb{R}^n |\ (x,x) = (x-v, x-v)\}$$

intersects $\mathrm{Vor}(\Lambda)$ in a facet, see Figure 3.2. A vector $c \in \Lambda$ is *reduced* if $(c,c) \leq (v,v)$ for all $v \in c + 2\Lambda$. The following is a theorem of Voronoi, see [**?**], but the exposition we use is from [**?**].

Theorem 3.1.1. *A vector $v \in \Lambda \setminus \{\vec{0}\}$ is relevant if and only if v is reduced and v and $-v$ are the only reduced vectors of Λ that are contain in $v + 2\Lambda$.*

Any lattice Λ has a *dual lattice* Λ^* in the space $\mathcal{H}$ that Λ spans, given by

$$\Lambda^* = \{x \in \mathcal{H} |\ (x,v) \in \mathbb{Z}, \quad \text{for all} \quad v \in \Lambda\}.$$

Figure 3.2: The Voronoi polyhedron of the lattice in Figure 3.1. Also one of its relevant vectors.

The set Λ^* is also a lattice. Furthermore, if $\{B_1, \ldots, B_r\}$ is a basis of Λ then the vectors $\{D_1, \ldots, D_r\}$ defined by

$$D_i B_j = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

form a basis of Λ^*, see [**?**]. It follows that $(\Lambda^*)^* = \Lambda$ and $\det(\Lambda)\det(\Lambda^*) = 1$. A lattice is called *unimodular* if it is equal to its dual lattice.

A *sublattice* Γ of a lattice Λ is just a subset of Λ that is a lattice. As lattices are Abelian groups, we can consider the *quotient group* Λ/Γ. It can be proved, [**?**], that the order of the group Λ/Γ, called the *index* of Γ in Λ, and denoted by $[\Lambda : \Gamma]$, is equal to $\det(\Gamma)/\det(\Lambda)$.

A lattice Λ is *integral* if the inner product of any two lattice vectors is an integer, that is, if and only if $\Lambda \subseteq \Lambda^*$. An important group associated with an integral lattice is its *dual quotient group* Λ^*/Λ which has order $(\det(\Lambda))^2$. The following theorem of [**?**] that also appears in [**?**] relates all these concepts.

Theorem 3.1.2. *Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two orthogonal subspaces of $\mathbb{R}^n$ not equal to $\vec{0}$, such that $\mathbb{R}^n = \mathcal{H}_1 \oplus \mathcal{H}_2$. Let $\pi_{\mathcal{H}_1}$ (respectively $\pi_{\mathcal{H}_2}$) be the orthogonal projection of $\mathbb{R}^n$ onto $\mathcal{H}_1$ (respectively $\mathcal{H}_2$). Suppose that $\mathbb{Q}^n = (\mathcal{H}_1 \cap \mathbb{Q}^n) \oplus (\mathcal{H}_2 \cap \mathbb{Q}^n)$. Let Λ be a unimodular integral lattice. Set $\Gamma_1 = \Lambda \cap \mathcal{H}_1$ and $\Gamma_2 = \Lambda \cap \mathcal{H}_2$. Then*

- Γ_1 *is a lattice in* $\mathcal{H}_1$ *and* Γ_2 *is a lattice in* $\mathcal{H}_2$.
- $\Gamma_1^* = \pi_{\mathcal{H}_1}(\Lambda)$ *and* $\Gamma_2^* = \pi_{\mathcal{H}_2}(\Lambda)$.
- *The dual quotient groups* Γ_1^*/Γ_1 *and* Γ_2^*/Γ_2 *are isomorphic.*

3.2 The circuit space of a regular matroid

Let G be a connected graph with n vertices and m edges. If we consider, for an arbitrary orientation of G, the incidence matrix B of G as a linear transformation from $\mathbb{R}^m$ to $\mathbb{R}^n$, the kernel of B is a vector space of dimension $m - n + 1$, that is the rank of the matroid $M^*(G)$. This is called the *cycle space* of G. The orthogonal complement of the cycle space is called the *cut space* of G. The cycle space and cut space of a graph are well known and have been very extensively studied before, see for example [**?**, **?**]. It is well known in the matroid theory field that there is a natural extension to regular matroids of these concepts; in this section, we make more precise this extension.

Let M be a binary matroid. In [**?**] the *circuit matrix* $D(M)$ of M is defined to be the incidence matrix of circuits against elements. The *cocircuit matrix* $D^*(M)$ is the incidence matrix of cocircuits against elements. Clearly, $D^*(M) = D(M^*)$. A binary matroid is *orientable* if it is possible to assign negative signs to some of the non-zero entries in $D = D(M)$, $D^* = D^*(M)$ to get matrices $\hat{D}$ and $\hat{D}^*$ in such a way that the scalar product of any row of $\hat{D}$ with any row of $\hat{D}^*$ is zero over the ring of integers. We called this signing of the circuit and cocircuit matrices an *orientation* of M. We also call the row space of $\hat{D}$ the *oriented circuit* space of M and the row space of $\hat{D}^*$ the *oriented cocircuit* space of M. The following result is due to Minty, [**?**] (in the reference, orientable is called signable).

Theorem 3.2.1. *For a binary matroid* M*,* M *is regular if and only if it is orientable.*

In fact, it is known, see [**?**], that if M is a regular matroid with a totally unimodular matrix representation $B = [I_r|A]$, then for each cocircuit C of M, the row space of B contains a signing of the incidence vector of C. Also, for every circuit C' of M, the row space of the unimodular matrix $B^* = [-A^t|I_{n-r}]$ contains a signing of the incidence vector of C'. The signing of $D(M)$ and $D^*(M)$ by this procedure is clearly an orientation of M because every row of B is orthogonal to every row of B^*.

It also can be proved, [**?**], that if we have an orientation of M, $\hat{D}^*$ and $\hat{D}$, $\hat{D}^*$ is a real representation of M. If M has rank r and n elements , we can find an $r \times n$

submatrix B of $\hat{D}^*$ of the form $B = [I_r|A]$, for example, by choosing the rows of $\hat{D}^*$ corresponding to the fundamental cocircuits of some fixed basis of M and possibly permuting some columns. As the rows of B are linearly independent, they generate the row space of $\hat{D}^*$ and then B is also a real representation of M ([**?**] 6.3). But if $B' = [I_r|A']$ is a real representation of a regular matroid, then B' is unimodular ([**?**]). So we conclude that B is unimodular.

Even more, $B^* = [-A^t|I_{n-r}]$ is a unimodular representation of M^*, [**?**], whose rows are contained in the rows of $\hat{D}$ and generate the same space as $\hat{D}$, this is because the rows of B^* are orthogonal to the rows of B and its incidence vectors are circuits of M.

Therefore, if we have an orientation of M, $\hat{D}(M)$, and $\hat{D}^*(M)$, the row space of $\hat{D}^*(M)$ is equal to the row space of B, for some unimodular representation of M; and similarly, the row space of $\hat{D}(M)$ is equal to the row space of B^*, for some unimodular representation of M^*.

Since, given any two $r \times n$ unimodular matrix representations of a regular matroid M, A and A', one can transform one into the other by multiplying by an $r \times r$ unimodular matrix U, then interchanging columns or multiplying some columns by -1, [**?**], the row spaces of A and A' are isomorphic vector spaces. So the oriented circuit and cocircuit spaces are, up to isomorphism, independent of the particular orientation of the matroid.

Thus, we just simply call any particular oriented cocircuit space of a regular matroid M , the *cocircuit space* of M and we denote it by $\mathcal{C}(M)$. Similarly, any particular oriented circuit space of M will be called the *circuit space* of M and we denote it by $\mathcal{C}^*(M)$, and just when we want to refer to a particular orientation, we will use $\mathcal{C}(M(B))$ and $\mathcal{C}^*(M(B))$. Clearly, the circuit space of M^* is the cocircuit space of M and the cocircuit space of M^* is the circuit space of M. Also, from now on, we will assume that if M is rank-r, $r > 0$, and regular, B is a unimodular matrix representation with exactly r rows. In the case that M has rank 0, that is, M is just a collection of loops, B will be a row of zeros.

As we said at the beginning of this section, the circuit and cocircuit space provide a generalisation of the cycle and cocycle space of a graph. The incidence matrix of G is a unimodular matrix representation of $M(G)$ for any orientation of G, [**?**], and the last row is minus the sum of the previous rows, so when considering the incidence matrix minus the last row it is clear that the cycle space of G is the circuit space of $M(G)$ and the cut space of G is the cocircuit space of $M(G)$.

3.3 The integral lattice of cocircuits of a regular matroid

Let G be a graph with m edges, $\mathcal{C}$ be its cut space and $\mathcal{C}^*$ be its cycle space. In [**?**], the lattice $N(G) = \mathcal{C} \cap \mathbb{Z}^m$ is called the *lattice of integral cuts* and $L(G) = \mathcal{C}^* \cap \mathbb{Z}^m$, the *lattice of integral flows*. These are very well known concepts in graph theory, [**?**]. Here we give a generalisation of them to regular matroids, that as far as we know has not been published before, and prove some extension to regular matroids of results in [**?**].

Let M be a rank-r regular matroid and B be a unimodular matrix representation of M with r rows. The lattice $\Lambda(B)$ is contained in the cocircuit space of M, $\mathcal{C}(M) = \mathcal{C}(M(B))$. So we call, for any particular unimodular matrix representation B of M with r rows, the lattice $\Lambda(B)$, the *lattice of cocircuits* of M. As we said in the last section, any other unimodular representation, say B', with r rows is obtained from B by multiplying by a matrix $U \in \mathrm{GL}(r, \mathbb{Z})$, and multiplying by -1 and permuting some columns of B, then the lattices $\Lambda(B)$ and $\Lambda(B')$ are congruent, see 3.1. Thus the lattice of cocircuits is, up to congruence on lattices, independent of the representation chosen, and for a regular matroid M we denote this lattice by $\Lambda(M)$. When we want to remark which basis we are using, we write $\Lambda(B)$.

Also, by the previous section, we know that there exists a unimodular matrix representation, B^*, of M^* such that every row of B^* is orthogonal to every row of B. The lattice of cocircuits of M^*, $\Lambda(B^*)$, is called *the lattice of circuits* of M. The lattice of circuits will be then $\Lambda(M^*)$.

Lemma 3.3.1. *If M is an n element regular matroid of rank r with unimodular matrix representation B, having r rows, then*

$$\Lambda(B) = \mathbb{Z}^n \cap \mathcal{C}(M(B)).$$

Therefore, the lattice of cocircuits consists of the integer-valued vectors in the cocircuit space of M.

Proof. If M has rank 0, $\mathcal{C}(M(B)) = \{\vec{0}\}$ and also $\Lambda(B) = \{\vec{0}\}$ so the result follows.

By the discussion in the previous section, we can assume that $B = [I_r|A]$, where r is the rank of M. It is clear that the elements of $\Lambda(B)$ are integer-valued vectors, so $\Lambda(B)$ is a sublattice of $\mathbb{Z}^n \cap \mathcal{C}(M(B))$. Let a belong to $\mathbb{Z}^n \cap \mathcal{C}(M(B))$, then we can choose a representative vector $\hat{a}$ of the coset of a in the quotient group $(\mathbb{Z}^n \cap \mathcal{C}(M))/\Lambda(B)$, that is in the fundamental paralletope of $\Lambda(B)$. The vector $\hat{a}$ has

integer entries, as $\hat{a} - a \in \Lambda(B)$ and a is an integer-valued vector. But $\hat{a} = \sum_{i=1}^{r} \lambda_i B_i$, $0 \leq \lambda_i < 1$, $1 \leq i \leq r$, where B_i is the i row of B. Then the choice of B implies that $\lambda_i = 0$ for all $1 \leq i \leq r$, and $a \in \Lambda(B)$. □

Recall that for a graph G, $N(G)$ is its lattice of integral cuts and $L(G)$ is its lattice of integral flows. Lemma 3.3.1 says that $L(G)$ is the same as the lattice of circuits of $M(G)$ and $N(G)$ is the lattice of cocircuits of $M(G)$.

We now extend some of the results in [**?**] for the lattice of integral flows and integral cuts to the lattice of integral cocircuits of a regular matroid.

Theorem 3.3.2. *Let M be a regular matroid, then the determinant of the lattice $\Lambda(M)$ equals the square root of the number of bases of M. Therefore, the order of the dual quotient group $\Lambda^*(M)/\Lambda(M)$ equals the number of bases of M.*

Proof. Let M be a rank-r regular matroid. If $r = 0$, that is M is a collection of loops, then M has one basis, the empty set. But $\Lambda(M) = \{\vec{0}\}$, so $\Lambda^*(M) = \{\vec{0}\}$ and $\Lambda^*(M)/\Lambda(M)$ is the trivial group of 1 element. Thus the statement is true.

If $r > 0$, take B a unimodular representation with r rows. The determinant of the lattice $\Lambda(M)$ is the determinant of the Gram-matrix BB^t. By the Binet-Cauchy Theorem, this determinant equals

$$|BB^t| = \sum |R|^2,$$

where the sum runs over all $r \times r$ submatrices R of B, and $|\cdot|$ denotes the determinant of a square matrix. As B is unimodular, $|R|^2 = 0, 1$. If $|R|^2 = 1$, the set of columns of R correspond to a basis of M, and conversely, thus the determinant $|BB^t|$ equals the number of bases of M. □

For a connected graph G the dual quotient group of the lattice of integral flows is called *the Jacobian of the connected graph G*, see [**?**]. So, the Jacobian of G is isomorphic to $\Lambda^*(M^*)/\Lambda(M^*)$. The following result is an extension of Theorem 8 of [**?**].

Theorem 3.3.3. *Let M be a regular matroid. Then, the groups $\Lambda^*(M)/\Lambda(M)$ and $\Lambda^*(M^*)/\Lambda(M^*)$ are isomorphic.*

Proof. Let M be a rank-r regular matroid with n elements. If $r = 0$, as in the proof of Theorem 3.3.2, $\Lambda^*(M)/\Lambda(M)$ is the trivial group. Similarly, $\Lambda^*(M^*)/\Lambda(M^*)$ is also the trivial group as $\Lambda(M^*)$ is $\mathbb{Z}^n$, and therefore $\Lambda^*(M^*)$ is $\mathbb{Z}^n$. Thus, the statement is true in this case. The result follows from duality in the case $r = n$.

If $0 < r < n$, take B a unimodular representation of M with r rows and B^* a unimodular representation of M^* with $n - r$ rows and such that every row of B is orthogonal to every row of B^*. Let $\mathcal{H}_1 =< B >$ be the space spanned by the rows of B and $\mathcal{H}_2 =< B^* >$ be defined similarly. As $0 < r < n$, $\mathcal{H}_1$ and $\mathcal{H}_2$ are not the trivial space $\{\vec{0}\}$. Also a vector $v \in \mathbb{Q}^n$ has a decomposition $v = v_1 + v_2$, where v_1 is the projection of v over $\mathcal{H}_1$ and v_2 is the projection of v over $\mathcal{H}_2$.

Now, the result follows directly from Theorem 3.1.2 and Lemma 3.3.1 by taking the unimodular integral lattice $\mathbb{Z}^n$, provided v_1 and v_2 are in $\mathbb{Q}^n$, so we prove this last claim. Observe that $(v_1, B_i) = (v, B_i) \in \mathbb{Q}$, as B_i has integer-valued entries for every row B_i of B. Consider $n \in \mathbb{Z}$ such that $(nv_1, B_i) = n(v_1, B_i) \in \mathbb{Z}$, for all $1 \leq i \leq r$. Then, $v_1' = nv_1 \in \Lambda^*(M(B))$, and so $v_1' = \sum_{i=1}^{r} \lambda_i B_i'$, where λ_i are integers for $1 \leq i \leq r$ and $B_1', \ldots, B_r'$ is a basis for $\Lambda^*(M(B))$. The claim will be proved if $B_i' \in \mathbb{Q}^n$, for all $1 \leq i \leq r$, but this is true because the vectors $B_1, \ldots, B_r$ belong to $\mathbb{Q}^n$, are linearly independent and also belong to $\Lambda^*(M(B))$. So, by Theorem 18 in [**?**] (also known, in a slightly different form, as the Hermitian normal form Theorem) $\Lambda^*(M(B))$ has a basis whose elements are rational combinations of the vectors $B_1, \ldots, B_r$ (the Hermitian normal form Theorem states more, it describes the form of this combinations). □

More interesting is the following extension of Proposition 6 of [**?**] .

Theorem 3.3.4. *Let M be a regular matroid with orientation induced by a unimodular matrix representation B. There is a natural bijection between the oriented circuits of M and the facets of the Voronoi polyhedron of $\Lambda(M)$. More precisely, a vector $\lambda \in \Lambda(B)$ is relevant if and only if it is the vector of an oriented circuit of $M(B)$.*

For the proof we need some results. The following theorem is in [**?**] but it is implicit in the work of Tutte on chain groups. The *support* of a vector v, $\mathrm{supp}(v)$, is the set of components that are not zero.

Theorem 3.3.5. *Let A be an $r \times n$ matrix over a field F and $M = M(A)$. Then the set of cocircuits of M coincides with the set of minimal nonempty supports of vectors from the row space of A.*

The following lemma has a trivial proof in the theory of oriented matroids. Here we just state some of the facts of oriented matroids that are needed and all are from [**?**].

The *sign function* is defined as a map, $\text{sign} : \mathbb{R} \to \{+, -, 0\}$,

$$z \to \begin{cases} + & \text{if } z > 0, \\ 0 & \text{if } z = 0, \\ - & \text{if } z < 0. \end{cases}$$

We extend this function to vectors componentwise, so if $v \in \mathbb{R}^n$, $\text{sign}(v) \in \{+, -, 0\}^n$.

For an $r \times n$ real matrix B, its space of *linear dependences* is

$$\text{Dep}(B) = \{v \in \mathbb{R}^n | Bv = \vec{0}\} \subseteq \mathbb{R}^n.$$

The *signed vectors* of B are given by

$$\mathcal{V}(B) = \{\text{sign}(v) \in \{+, -, 0\}^n | v \in \text{Dep}(B)\},$$

and the signed circuits of B are the signed vectors of minimal nonempty support. Then, by Theorem 3.3.5, the oriented circuits of M correspond, under the sign function, to the signed circuits of (the oriented matroid of) B.

There is a natural *partial order on the set of signs* $\{+, -, 0\}$. We set $0 < +$ and $0 < -$, while $+$ and $-$ are incomparable. On the set of sign vectors, we use componentwise partial ordering. So, if $U, V \in \{+, -, 0\}^n$ are sign vectors, $U \leq V$ if $U_i \leq V_i$ holds for all positions $1 \leq i \leq n$, and we write $U < V$ if $U \leq V$ and $U \neq V$.

For a set of sign vectors $\mathcal{S}$, we define $\text{Min}(\mathcal{S})$ to be the set of all minimal nonzero vectors in $\mathcal{S}$, that is,

$$\text{Min}(\mathcal{S}) = \{U \in \mathcal{S} \setminus \{\vec{0}\} | \text{ there is no } U' < U \text{ with } U' \in \mathcal{S}\}.$$

It is known, [?], that $\text{Min}(\mathcal{V}(B))$ is equal to the set of signed circuits of B. It follows that:

Lemma 3.3.6. *Let M be a regular matroid with unimodular matrix representation B. If v is any vector in the circuit space of $M(B)$, then there exists an oriented circuit c of $M(B)$ such that sign$(c) \leq$ sign(v).*

Proof of Theorem 3.3.4. If c is an oriented circuit of $M(B)$, the entries of c have values in $\{1, -1, 0\}$, and if $v \in c + 2\Lambda(B)$, v has even values in the entries that do not belong to $\text{supp}(c)$ and odd values in the entries of $\text{supp}(c)$. If $v \neq c, -c$, there exists at least one entry in which v is bigger than c in absolute value, so $(v, v) > (c, c)$, and c is reduced, and by Theorem 3.1.1, c is relevant.

Let v be a relevant vector, then by Theorem 3.1.1, v is reduced. Take an oriented circuit c such that $\text{sign}(c) \leq \text{sign}(v)$. This can be done by Lemma 3.3.6. The choice of c guarantees that $(v, c) \geq (c, c)$. On the other hand,

$$(v - 2c, v - 2c) = (v, v) + 4(c, c) - 4(v, c) \geq (v, v) \tag{3.3.1}$$

as v is reduced. The right-hand side of equation 3.3.1 implies that $4(c, c) - 4(v, c) \geq 0$ and thus $(c, c) \geq (v, c)$. Therefore $(v, c) = (c, c)$, and all the entries of v are in $\{1, -1, 0\}$. This, together with the left-hand side of equation 3.3.1, also implies $(v - 2c, v - 2c) = (v, v)$.

If $v - c$ is not zero, again by Lemma 3.3.6, there exists an oriented circuit c' with $\text{sign}(c') \leq \text{sign}(v - c)$. Also we have that $\text{sign}(v - c) < \text{sign}(v)$ because v and c have the same sign in every common nonzero entry and c is not the zero vector. As $(v, c) = (c, c)$, the later implies that $c \neq c'$. By the above argument, using c' instead of c, we have $(v - 2c', v - 2c') = (v, v)$. Thus

$$(v, v) = (v - 2c, v - 2c) = (v - 2c', v - 2c').$$

But v, $v - 2c$ and $v - 2c'$ are all different vectors and this contradicts that v and $-v$ are the only reduced vectors in $v + 2\Lambda(B)$. Therefore, $v = c$ and the proof is complete. □

The *minimal norm* of a lattice Λ is the number $\min\{(v, v) |\ v \in \Lambda, v \neq \vec{0}\}$. From the previous theorem we obtain the following result that is an extension of Theorem 1 (ii) in [**?**].

Corollary 3.3.7. *Let M be a regular matroid. Then, the minimal norm of $\Lambda(M)$ is equal to the size of a circuit of M with minimum length .*

Proof. A vector of minimum norm in a lattice is a relevant vector, and by the previous theorem, it corresponds to an oriented circuit of M, that has norm equal to the size of its support, that is the size of the circuit. To any vector v in $\Lambda(M)$, there is a relevant vector λ with norm less or equal to the norm of v. As all the relevant vectors of $\Lambda(M)$ correspond to oriented circuits of M, a circuit with minimum length has minimal norm in $\Lambda(M)$. □

3.4 A general chip firing process

Chip firing games on undirected graphs, see [**?**, **?**, **?**], and chip firing games on directed graphs, see [**?**], are special cases of so called "vector addition systems".

In [**?**], a *vector addition system* is defined as a triple (C, V, b), where C is a polyhedral cone in $\mathbb{R}^n$ (with apex in $\vec{O}$), V is a finite set of vectors in $\mathbb{R}^n$ and b is any fixed vector in C. Also the language of all sequences $v_1, \ldots, v_k$ of vectors in V for which $b + v_1 + \cdots + v_j \in C$, for every $1 \leq j \leq k$, is refered to as a *vector addition language*. Vector addition systems were introduced by Karp and Miller [**?**] with the further requirement that b and the vectors in V have integer components and that C is the first orthant in $\mathbb{R}^n$ (they are also known as *general Petri nets*, see [**?**]).

We now try to extend some of the ideas of the last chapter. First some definitions. Given an $r \times n$ matrix A, a polyhedral cone (with apex in $\vec{0}$) $C = C(A)$ in $\mathbb{R}^n$ is the intersection of a finite family of closed halfspaces of $\mathbb{R}^n$, given by $\mathcal{H}_i^+ = \{v \in \mathbb{R}^n | \ (a_i, v) \geq 0\}$, where a_i is the i-th row of A. The family $\{\mathcal{H}_i^+ | \ 1 \leq i \leq r\}$ is called *irredundant* provided

$$\bigcap_{\substack{1 \leq i \leq r \\ i \neq j}} \mathcal{H}_i^+ \neq C,$$

for each $i = 1, \ldots, r$. We always assume that the given family of halfspaces is irredundant. Now, given such a cone $C(A)$, there will exist many different A giving rise to the same $C(A)$.

Lemma 3.4.1. *For any given $r \times n$ matrix A and cone $C(A) \subseteq \mathbb{R}^n$ there exists matrix A^* such that $C(A) = C(A^*)$ and moreover for each facet F of C, there exists only one row a of A^* such that $F = \mathcal{H} \cap C$, where $\mathcal{H} = \{v \in \mathbb{R}^n | \ (a, v) = 0\}$.*

Proof. We can suppose that $(a_i, v) = 0$, for $i < r_0$, and $(a_i, v) \geq 0$, for $r_0 \leq i$, for all $v \in C$, and also that for each a_i, $r_0 \leq i$, there exists at leat one $v \in C$ such that $(a_i, v) > 0$. So,

$$\bigcap_{1 \leq i < r_0} \mathcal{H}_i^+ \supseteq < C >,$$

where $< C >$ is the subspace spanned by the vectors in C, and

$$C = < C > \cap (\bigcap_{r_0 \leq i \leq r} \mathcal{H}_i^+).$$

We, then, can assume that $\bigcap_{1 \leq i < r_0} \mathcal{H}_i^+ = < C >$, as $< C >$ can be expressed as a finite intersection of halfspaces (it is the kernel of the projection on its orthogonal space).

If we consider the space $\mathcal{H} =< C >$, then

$$C = \bigcap_{r_0 \leq i \leq r} (\mathcal{H}_i^+ \cap \mathcal{H}).$$

So C is a full-dimensional cone in $\mathcal{H}$, and the family of halfspaces of $\mathcal{H}$ is irredundant. Then it is known ([**?**]) that $F_i = C \cap (\mathcal{H}_i^+ \cap \mathcal{H})$, $r_0 \leq i \leq r$, is a facet of C, that is, $C \cap \mathcal{H}_i^+$ is a facet of C. Therefore, we can find an $r' \times n$ matrix A^* such that for the first r'_0 rows, $(a, v) = 0$, for all $v \in C$, where $0 \leq r'_0 \leq r'$. Also, each of rows left defines a different facet of C. □

We call such an A^* a *canonical matrix* for the cone $C(A)$, although it is not unique, and henceforth we will assume that $A = A^*$. In other words, when speaking of $C(A)$ we will assume that A is a canonical matrix.

Here we are more interested in properties of the vector addition language when we vary the vector b, so we make the following definitions.

Definition 3.4.2. *A game system $\mathcal{G}$ over $\mathbb{R}^n$ is an ordered triple $(C(A), V, I)$ consisting of:*

1. *A polyhedral cone $C(A)$ in $\mathbb{R}^n$.*
2. *A finite set V of vectors in $\mathbb{R}^n$. We call the members of V, "the rules of the game".*
3. *A set $I \subseteq C(A)$. We call the members of I, "the initial configurations".*

To any game system $\mathcal{G}$ we associate a family of languages $\{\mathcal{L}(b)\}_{b \in I}$, where

$$\mathcal{L}(b) = \{v_1 \cdots v_k | b + v_1 + \cdots + v_j \in C(A), 1 \leq j \leq k,\ k \in \mathbb{N}\}.$$

And we refer to any word σ in $\mathcal{L}(b)$ as a *legal firing* with initial configuration b. Clearly, for every $b \in I$ the language $\mathcal{L}(b)$ is a vector addition language. We also introduce the concept of *sub-game*.

Definition 3.4.3. *Given two matrices A and A', we define a game system $\mathcal{G} = (C(A), V, I)$ over $\mathbb{R}^n$ to be a sub-game system of $\mathcal{G}' = (C(A'), V', I')$ over $\mathbb{R}^m$ if there exists a linear transformation $f : \mathbb{R}^n \to \mathbb{R}^m$ such that*

1. *$f(C(A)) \subseteq C(A')$.*
2. *$f(V) \subseteq V'$ and if $x, y \in V$ with $x \neq y$, then $f(x) \neq f(y)$.*

3. $f(I) \subseteq I'$.

If $\mathcal{G}$ and $\mathcal{G}'$ are game systems and $\mathcal{G}$ is a sub-game system of $\mathcal{G}'$ determined by the linear function f, their associated languages are related by the function $\hat{f}$ from the set of all words over V to the set of all words over V' defined by $\hat{f}(x_1 \cdots x_k) = f(x_1) \cdots f(x_k)$. This function satisfies the condition that $\hat{f}(\mathcal{L}(b)) \subseteq \mathcal{L}(f(b))$ for all $b \in I$, furthermore, this function is just a relabelling of the set of symbols of V, so we can think of the language $\mathcal{L}(b)$ to be contained in the language $\mathcal{L}(f(b))$, for every $b \in I$.

We say that two game systems $\mathcal{G}$ and $\mathcal{G}'$ are *equivalent* if $\mathcal{G}$ is a sub-game of $\mathcal{G}'$ and $\mathcal{G}'$ is a sub-game of $\mathcal{G}$. This clearly gives an equivalence relation among all game systems. Also, in this case, we can think the associated languages to be the same, in the sense discussed above. We use this concept as we want a game process on regular matroids that does not depend on particular unimodular representations of the matroid.

As pointed out in [**?**], if $\mathcal{G} = (C(A), V, I)$ is a game system, then for every $b \in I$, the language $\mathcal{L}(b)$ is left-hereditary and permutable but in general not locally free. To ensure $\mathcal{L}(b)$ is a locally free language, we assume the following property.

***** for every row a of A, there exists at most one vector $v \in V$ with $(a, v) < 0$.

Note, this is a very strong condition. In [**?**] it is proved that a vector addition system with this property has an associated vector addition language that is locally free. We state this in our context and provide a proof for the sake of consistency.

Lemma 3.4.4. *If $\mathcal{G} = (C(A), V, I)$ is a game system that satisfies (*), then for every $b \in I$, the language $\mathcal{L}(b)$ is locally free.*

Proof. Let $b \in I$ and let $\sigma = v_1 \cdots v_k \in \mathcal{L}(b)$ be such that σx and σy are in $\mathcal{L}(b)$ for two different elements $x, y \in V$, we want to prove $\sigma xy \in \mathcal{L}(b)$. Let $w = b + v_1 + \cdots v_k$, it is enough to prove that $w + x + y \in C(A)$. For any row vector a of A, (a, x) and (a, y) cannot be both negative. Suppose $(a, x) \geq 0$, then $(a, w+x+y) = (a, w+y)+(a, x) \geq 0$ as by hypothesis $(a, w + y) \geq 0$. Therefore for every row a of A, $(a, w + x + y) \geq 0$, so $w + x + y \in C(A)$. □

Example 1. Let G be a connected graph with n vertices and Q be its Laplace matrix with columns indexed by the vertices. Consider the game system $(C(A), V, I)$, where A is the $n \times n$ identity matrix I_n, that is, the cone is the positive orthant; V consists of the columns vectors of $-Q$ and I is the set of integer points inside $C(I_n)$.

This game system clearly satisfies property (*) and corresponds to the general chip firing game of [**?**].

We introduce the following definition that is our generalisation of the chip firing game in Chapter 2.

Definition 3.4.5. *If $\mathcal{G} = (C(A), V \cup \{q\}, I)$ is a game system, we call the pair $(\mathcal{G}, q)$ a chip firing process if the following are true.*

- *$\mathcal{G}$ satisfies (*).*
- *V is a set of linearly independent vectors.*
- *$q \in \Lambda(V)$, where $\Lambda(V)$ is the lattice generated by the vectors in V.*
- *$I = C(A) \cap \Lambda'$, where Λ' is a lattice that contains $\Lambda(V)$ as a sublattice.*

In a chip firing process $(\mathcal{G}, q)$, we say that $\sigma \in \mathcal{L}(b)$ is a *q-legal firing* with initial configuration b if $\sigma = v_1 \cdots v_k$ is such that whenever $q = v_j$, for some $1 \leq j \leq k$, then $\sigma' = v_1 \cdots v_{j-1}x$ does not belong to $\mathcal{L}(b)$ for any $x \in V$, where if $j = 1$, $\sigma' = x$. Associated to any chip firing process we have the family of languages $\{\mathcal{L}_q(b)\}_{b \in I}$, where $\mathcal{L}_q(b)$ is the set of q-legal firings with initial configuration b. Now, we prove a similar result to Lemma 2.4.1.

Lemma 3.4.6. *If $(\mathcal{G}, q)$ is a chip firing process, then $\mathcal{L}_q(b)$ is a left-hereditary permutable locally free language for all $b \in I$.*

Proof. As $\mathcal{L}(b)$ is left-hereditary, permutable and locally free, the properties can be now checked for $\mathcal{L}_q(b)$ easily. □

Following the analogy with the previous chapter, for a chip firing process $(\mathcal{G}, q)$, we say that an element $b' \in I$ is a *final configuration* after a q-legal firing σ with initial configuration b if $\sigma = v_1 \cdots v_k$ and $b' = b + v_1 + \cdots + v_k$, and we denote this by $b' \to b$. The definition of chip firing process guarantees that $b' \in I$. An initial configuration b is *stable* if $\mathcal{L}_q(b)$ has q as a prefix of every word. Suppose we start with an initial configuration b and after a q-legal firing $\sigma = v_1 \cdots v_k$, with $k \geq 1$, we arrive at the same configuration b, then we said that b is a *recurrent configuration*. And finally, an initial configuration is *critical* if it is stable and recurrent. In general, the sets of stable, recurrent or critical configurations could be empty.

Let $(\mathcal{G}, q)$ be a chip firing process. As in the previous chapter, we consider for every $b \in I$ the language

$\overline{\mathcal{L}}_q(b) = \{\sigma \in \mathcal{L}_q(b) | \sigma$ is a q-legal firing such that starting from b it can be extended to a legal firing that produces a critical configuration and for any proper prefix $v_1 \cdots v_k$ of σ , $b + v_1 + \cdots + v_k$ is not critical $\}$.

We have a complete analogue of Lemma 2.4.1.

Lemma 3.4.7. *If $(\mathcal{G}, q)$ is a chip firing process, then $\overline{\mathcal{L}}_q(b)$ is a left-hereditary permutable locally free language for all $b \in I$.*

Proof. The proof is similar to the proof of Lemma 2.4.1. □

We say that two chip firing processes $(\mathcal{G}, q)$ and $(\mathcal{G}', q')$ are *equivalent* if the game systems $\mathcal{G}$ and $\mathcal{G}'$ are equivalent using the linear functions f and g, and $f(q) = q'$, $g(q') = q$. Again, the two pairs of languages $\mathcal{L}_q(b)$, $\mathcal{L}_{f(q)}(f(b))$ and $\overline{\mathcal{L}}_q(b)$, $\overline{\mathcal{L}}_{f(q)}(f(b))$ are taken to be the same due to the discussion in page 104.

Example 2. Let G be a graph with n vertices and Laplace matrix Q with columns indexed by the vertices and let $v \in V(G)$. Consider the game system $\mathcal{G} = (C(A), V \cup \{q\}, I)$ on $\mathbb{R}^n$, where $C(A)$ is generated by the vectors $e_i - e_v$, and the e_i's are the standard basis of $\mathbb{R}^n$ with coordinates labelled as the vertices of G. $V \cup \{q\}$ is the set of column vectors of $-Q$, q is the column corresponding to vertex v and I is $\mathbb{Z}^n \cap C(A)$. Observe that a matrix for $C(A)$ is I'_n, where the first two rows of I'_n are the vectors $\vec{1}$ and $-\vec{1}$, $\vec{1}$ is the vector of all entries equal 1, and the next $n-1$ rows correspond to e_i, $1 \leq i \leq n$, $e_i \neq e_v$. Now it is clear that $(\mathcal{G}, q)$ is a chip firing process that corresponds to the chip firing game of Chapter 2.

Notice that using the projection over the v coordinate we obtain an equivalent chip firing process but over $\mathbb{R}^{n-1}$. Figures 2.3 and 2.5 were created having this in mind.

Example 3. Let $\vec{G}$ be an oriented graph with n vertices and a special vertex v. Consider the matrix Q' defined by

$$Q'_{ij} = \begin{cases} d_{ij} & \text{if } j \neq i, \\ -d^+(i) & \text{if } j = i, \end{cases}$$

where d_{ij} is the number of edges that go from vertex i to vertex j, and $d^+(i)$ is the outdegree of vertex i. Thus $d^+(i) = \sum_j d_{i,j}$.

The game system $\mathcal{G} = (C(A), V \cup \{q\}, I)$ on $\mathbb{R}^n$, where $C(A)$ is as in the previous example, $V \cup \{q\}$ is the set of row vectors of Q', q is the row corresponding to vertex v and I is $\mathbb{Z}^n \cap C(A)$, is a chip firing process. With the matrix I'_n of the previous example it is easy to check condition (*), and the other conditions of Definition 3.4.5

are trivial. This is closely related to the chip firing game on directed graphs of [**?**], but is, as far as I know, a new chip firing game on directed graphs that deserves more research.

3.5 The conical chip firing process

An easy way to construct a chip firing process is the following: if we have two lattices Λ and Λ' in $\mathbb{R}^n$, with Λ a sublattice of Λ', we can construct $(\mathcal{G}, q)$, where $\mathcal{G} = (C(A), V \cup \{q\}, I)$, just by taking $V = \{v_1, \ldots, v_k\}$, a base of Λ, as the set of rules. The cone $C(A)$ will be the cone generated by $\{-v_1, \ldots, -v_k\}$, that is $C(A) = \{\sum_{i=1}^{k} \lambda_i(-v_i) |\ \lambda_i \geq 0, 1 \leq i \leq k\}$; this cone has k facets, as every $k-1$ vectors generate a facet, and any facet is obtained as a positive linear combination of these vectors. The vector q will be $\sum_{i=1}^{k} v_i$ and $I = \Lambda' \cap C(A)$. The only thing that has to be proved is property (*), but this is easy as for each $v_i \in V$, v_i belongs to $k-1$ facets of $C(A)$, so $(a, v_i) < 0$ for exactly one row of A, given that $-v_i \in C(A)$.

As an example consider the conical chip firing process where Λ is generated by the vectors $(-1, -2)$ and $(-2, -1)$ in $\mathbb{R}^2$, and Λ' is its dual lattice. Figure 3.3 shows a region of $\mathbb{R}^2$, where the black squares represent elements of the lattice Λ and the dots represents elements of Λ^*. The shaded region is part of the cone $C(A)$ and the x's inside this region are the initial configurations of the process. The two vectors in solid lines correspond to the vectors $(1, 2)$ and $(2, 1)$, and the vector in dotted line corresponds to the vector q.

If a chip firing process can be obtained by this procedure, we call it a *conical chip firing process*. Observe that if $(\mathcal{G}, q)$ and $(\mathcal{G}', q')$ are conical chip firing processes obtained from the same lattices Λ and Λ', then they are equivalent chip firing processes. As we said, in a general chip firing process, the set of stable configurations could be empty, or even infinite, and so critical configuration can very well not exist in the general case, but for conical chip firing games this is not the case and in fact they are like chip firing on graphs. The shaded parallepiped of Figure 3.3 contains the stable configuration of the conical chip firing process of the previous example.

Theorem 3.5.1. *If $(\mathcal{G}, q)$ is a conical chip firing game, then the following statements are true.*

- *The set of stable configurations is finite.*
- *For any initial configuration b, the language $\overline{\mathcal{L}}_q(b)$ is not empty and has finite rank.*

Figure 3.3: A illustration of the conical chip firing process.

Proof. Let $\mathcal{G} = (C(A), V \cup \{q\}, I)$ be the conical chip firing game, where $V = \{v_1, \ldots, v_k\}$. We can assume that row i of A, a_i, is the row vector that defines the facet of $C(A)$ generated by $V \setminus \{v_i\}$. We prove that the set of stable configurations is the set $S = I \cap \{b \in C(A) | \, 0 \leq (b, a_i) < (-v_i, a_i), 1 \leq i \leq k\}$.

The initial configuration b is stable if and only if $b \in I$ and for all $1 \leq i \leq k$, $b + v_i \notin C(A)$. This happens if and only if $b \in I$ and for all $1 \leq i \leq k$, there exists a row a of A such that $0 \leq (a, b) < (a, -v_i)$. This a has to be the i-th row of A and we have that $b \in S$.

If b is any initial configuration, then the rule q ensures that $\overline{\mathcal{L}}_q(b)$ has words of infinite length. Even more, if $b' \in C(A)$, then for row j of A, say a_j,

$$\begin{aligned} (a_j, b' + v_i) &< (a_j, b') \text{ if } i = j, \\ (a_j, b' + v_i) &= (a_j, b') \text{ if } i \neq j. \end{aligned}$$

Therefore, as we cannot have an infinite sequence of consecutive elements of V, an infinite length word of $\overline{\mathcal{L}}_q(b)$ must have the symbol q an infinite number of times. Every time we use q, we have arrived at a stable configuration, and as there are only a finite number of these, one stable configuration has to be repeated, this is, then, a critical configuration. We conclude that $\overline{\mathcal{L}}_q(b)$ is not empty. Also, as it contains

a word of finite length, it follows from the discussion in page 65, that it has finite rank. □

The language $\overline{\mathcal{L}}_q(b)$ is a greedoid with repetition, see [**?**], and it would be interesting to see the connection, if any, with the greedoids obtained from Coxeter groups in [**?**].

Corollary 3.5.2. *Let $(\mathcal{G}, q)$ be a conical chip firing process, then to any initial configuration b, there exists a unique critical configuration c such that $b \to c$. In particular, if b is critical, $b = c$.*

Proof. We know by Theorem 3.5.1 that $\overline{\mathcal{L}}_q(b)$ is not empty. This language has as basic words all the q-legal firings that starting from b finish in a critical configuration and they do not pass through any other critical configuration; lets take one of these configurations, say c.

As $\overline{\mathcal{L}}_q(b)$, by Lemma 3.4.7 and Lemma 2.3.1, has the strong exchange property, all basic words have the same length; also any two basic words are equivalent, thus, by Lemma 2.3.2, they have the same score, see page 64, and lead to the same critical configuration. We conclude that this c is unique. □

For a conical chip firing process, $(\mathcal{G}, q)$, we denote by $K(\mathcal{G})$ the set of all critical configurations. Also, if we have an initial configuration b, $\gamma(b)$ will be the unique critical configuration that is guaranteed by Corollary 3.5.2. In what follow, we extend the ideas of [**?**] on the algebraic structure of the critical configurations of the chip firing game to our conical chip firing process.

Lemma 3.5.3. *Let $\Lambda \subseteq \Lambda'$ be two k-dimensional lattices in $\mathbb{R}^n$, and let $(\mathcal{G}, q)$ be the conical chip firing process obtained from Λ and Λ' by the above method. Then, the function $\phi : K(\mathcal{G}) \to \Lambda'/\Lambda$ defined by $\phi(c) = [c]_\Lambda$, where $[c]_\Lambda$ is the coset of c in Λ'/Λ, is a bijection.*

Proof. Let $\{v_1, \ldots, v_k\}$ be a base for Λ. Suppose we have $c_1, c_2 \in K(\mathcal{G})$ such that $[c_1]_\Lambda = [c_2]_\Lambda$, then $c_1 = \sum_{i=1}^k \lambda_i v_i$ and $c_2 = \sum_{i=1}^k \lambda'_i v_i$, where $\lambda'_i, \lambda_i \in \mathbb{Z}$, for $1 \leq i \leq k$. But as c_1 and c_2 are in the cone generated by $\{-v_1, \ldots, -v_k\}$, we can assume that $\lambda'_i, \lambda_i \leq 0$, $1 \leq i \leq k$. The vector $b = \sum_{i=1}^k (\lambda_i + \lambda'_i) v_i$ is, then, an initial configuration and $b \to c_1$ and $b \to c_2$, so by the previous theorem, $c_1 = c_2$ and ϕ is injective.

Now, take $[b]_\Lambda \in \Lambda'/\Lambda$. We can choose a representative b in the set of initial configurations as the cone of $\mathcal{G}$ contains an unbounded open set of $< \Lambda >$, the space span by Λ, that has to intersect the coset $[b]_\Lambda$ because $< \Lambda > /(b + \Lambda)$ is compact

by definition of discrete vector group, see page 88. Then, $\gamma(b) = b + \sum_{i=1}^{k} \lambda_i v_i$, with $\lambda_i \in \mathbb{Z}$ for $1 \leq i \leq k$, and $\phi(\gamma(b)) = [b]_\Lambda$. Therefore, ϕ is onto. □

As in [**?**], it follows that the set of critical configurations has a natural group operation $\bullet$, namely, if c and c' are critical configurations, $c \bullet c'$ is the unique critical configuration c'' such that $[c]_\Lambda + [c']_\Lambda = [c'']_\Lambda$. But by the proof of Lemma 3.5.3 we know that $\gamma(c + c')$ is such a critical configuration, therefore $c \bullet c' = \gamma(c + c')$ and we have the following

Theorem 3.5.4. *Let $\Lambda \subseteq \Lambda'$ be two k-dimensional lattices in $\mathbb{R}^n$, and let $(\mathcal{G}, q)$ be the conical chip firing process obtained by the above method from Λ and Λ'. Then, the set of critical configurations can be given a structure of Abelian group and the order of the group equals the order of the group Λ'/Λ.*

3.6 A chip firing game for regular matroids

Let M be a regular matroid, then we can consider the lattice of cocircuits $\Lambda(M)$ and its dual lattice $\Lambda^*(M)$ to construct a conical chip firing process, where we assume an underlying unimodular matrix representation of M. We define the *chip firing game of a regular matroid M* to be this conical chip firing process. As we note, a particular unimodular representation B of M gives a particular lattice $\Lambda(B)$ in $\mathcal{C}(M(B))$, but any two of these lattices are congruent and the spaces they generate are related by an isomorphism, so we obtain equivalent conical chip firing processes.

We continue the analogy with the chip firing game and obtain the following result that extends Theorem 7.3 of [**?**].

Theorem 3.6.1. *Let M be a regular matroid. The number of critical configurations of the chip firing game of M equals the number of bases of M.*

Proof. By Theorem 3.5.4, the number of critical configurations of M equals the order of the group $\Lambda^*(M)/\Lambda(M)$ that in turns equals the number of bases of M by Theorem 3.3.2. □

We also define for a regular matroid M the *critical group of M* to be the group of critical configurations of its chip firing game. This group is isomorphic to the group $\Lambda^*(M)/\Lambda(M)$ by Theorem 3.5.4. As we note before, in the case of a graphic matroid $M(G)$, this group is the Jacobian group of G.

The *Picard group* of a graph G is defined in [**?**] as $\ker(\tau)/\mathrm{Im}Q$, where the Laplace matrix Q of G is consider as an homomorphism, $Q : \mathbb{Z}^n \to \mathbb{Z}$, and τ is the homomorphism $\tau : \mathbb{Z}^n \to \mathbb{Z}$ given by $\tau((a_1, \ldots, a_n)) = \sum_{i=1}^{n} a_i$. The Picard group appears in algebraic geometry, but here we will not go on the details of its definition, see [**?**]. We know from [**?**] that the Picard group of a graph G is isomorphic to the Jacobian group of G, and thus the Picard group of G is isomorphic to the critical group of $M(G)$.

The Picard group of G is, up to an isomorphism of groups, the same as the *critical group of* G defined in [**?**], also refered to as the *sandpile group* of G in [**?**]. This is the reason for our notation. For a planar graph there is the following result from [**?**].

Theorem 3.6.2. *For a planar graph G and any of its duals G^*, the sandpile group of G and the sandpile group of G^* are isomorphic.*

We also obtain the following analogue to this theorem.

Theorem 3.6.3. *If M is a regular matroid, then the critical group of M is isomorphic to the critical group of M^*.*

Proof. Follows from 3.3.3. □

It would be nice to be able to provide a matroid interpretation of the main theorem of Chapter 2, namely, interpreting the generating function of critical configurations, given by some level function on the configurations, as an evaluation of the Tutte polynomial.

Unfortunately, I see no way to define the level of a configuration in this extension of the chip firing game which gives the desired result for regular matroids.

Another natural question is why I have just considered only regular matroids and not general binary matroids. The reason is that I see no way of extending the definition of chip firing process in a natural way. One obstruction being that computing the number of bases of a binary matroid given any binary representation is believed to be NP-hard. Thus, there is unlikely to be a determinant formula for it and the tools from lattice theory that I am using cannot be applied.

Chapter 4

The Matroid Bases Problem

The problem of counting the number of bases of a matroid has a long history and is very important in several fields of combinatorics and computer science in particular for its relevance in the problem of determining the reliability of a network. One promising algorithmic approach rests on the long standing conjecture of Mihail and Vazirani [**?**] that the bases-exchange graph of any matroid has cut set expansion at least 1. The verifying of this conjecture will give an "efficient" algorithm to approximate the number of bases of several classes of matroids, including binary matroids and , in general, representable matroids.

A partial answer was given by Feder and Mihail [**?**] when they defined balanced matroids and proved that this class of matroids satisfies the conjecture. More recently, Gambin [**?**] has defined exchange preserving matroids and proved that every matroid in this class has cut set expansion at least 1/2.

Here we take two different approaches to the conjecture. One is to extend the notion of balanced matroids to obtain the more general notion of α-balanced matroids. This class of matroids, like balanced matroids, are closed under minors, direct sum and 2-sum, but we have not found an expression for their cut set expansion yet. The other approach is to strengthen the conjecture and get a more algebraic structure that hopefully helps for the problem, with this aim we define k-expander matroids. A matroid is k-expander if its bases-exchange graph has strong cut set expansion at least k. We prove that balanced matroids are contained properly in the class of 2-expander matroids and provide several sufficient conditions for a matroid to be a 2-expander.

However, the truth of the conjecture even for binary matroids seems a very elusive and difficult problem and we were unable to settle it.

4.1 Bases–exchange graph

Associated with a matroid M we have its *bases polytope*, $P(M)$, introduced by Edmonds in [**?**], which is the convex hull of the incidence vectors of the bases of M. Using the definition of $\mathcal{B}$ and of convexity it can be proved that there is an edge between vertices B_1 and B_2 in $P(M)$ if and only if their symmetric difference is 2, that is, there exists $x \in B_1 - B_2$ and $y \in B_2 - B_1$ such that $(B_1 - x) \cup y = B_2$, this is called a *bases exchange*. Also note that if $|E| = n$, and the rank of M is r, then the bases polytope is contained in the r-slice of the n-cube, that is, the intersection of the n-cube with the hyperplane $\{y \in \mathbb{R}^n | \sum_i y_i = r\}$. It can be proved that a polytope contained in the r-slice of the n-cube is the bases polytope of some matroid if and only if every edge of the polytope has length 2 (using hamming distance) [**?**].

The 1-skeleton of the bases polytope of a matroid M is a graph that has as vertices the bases of M and there is an edge between two bases if they are related by a bases exchange, see page **??**. We call this graph the *bases-exchange graph of* M, and we denote it by $G(M)$. See figure 4.1 for an example.

Figure 4.1: The Euclidean representation of the matroid $U_{2,4}$ and also its bases polytope.

There are several results about the structure of the bases-exchange graph. For example, Bondy [**?**] showed that it is pancyclic. Holzmann and Harary [**?**] proved that if $G(M)$ has at least two cycles then, for every edge in the graph there is a Hamiltonian cycle containing it and one avoiding it. Also, Holzmann [**?**] showed that two matroids M and N have isomorphic bases-exchange graphs if and only if $M = M_1 \oplus \ldots \oplus M_s$ and $N = N_1 \oplus \ldots \oplus N_s$ and there is a permutation σ such that $M_i \cong N_{\sigma(i)}$ or $M_i \cong N^*_{\sigma(i)}$. Note that the matroids M and M^* have isomorphic bases-exchange graphs. An important result on this topic is due to Maurer who provided

a characterisation of graphs which are isomorphic to bases-exchange graphs of some matroid, we refer the reader to [**?**, **?**].

Here we are interested in the expansion properties of the bases-exchange graphs, more specifically, we are interested in the following notions.

Definition 4.1.1. *For a graph $G = (V, E)$, its cut set expansion is*

$$\min_{\substack{\emptyset \subset A \subset V \\ 0<|A| \leq |V|/2}} \frac{|C(A)|}{|A|}, \tag{4.1.1}$$

where $C(A) = \{(x, y) \in E | x \in A, y \in V - A\}$ is the cut set determined by A.

The strong cut set expansion of G is

$$\min_{\emptyset \subset A \subset V} \frac{|C(A)||V|}{|A||V - A|}.$$

Cut set expansion is closely related to the concept of *conductance* which is a key concept in approximate counting. Let X_t, $t = 0, 1 \ldots$, be a Markov chain over state-space Ω and transition probabilities $\{p_{ij}\}$, i, $j \in \Omega$. We say that X_t is *irreducible* if any state i can be reached from any state j in a finite number of steps, and it is *aperiodic* if $\gcd\{t | \ p_{i,j}^t > 0\} = 1$ for all i, $j \in \Omega$, where $p_{i,j}^t$ is the probability of going from state i to state j in t steps. Thus, if X_t is finite, irreducible and aperiodic, it converges to a unique stationary distribution $\pi = \{\pi_i\}$, $i \in \Omega$, and we say that X_t is *ergodic*. We also say that an ergodic Markov chain is *time-reversible* if it satisfies the "detailed balance" property, that is, for all i and j in Ω, $p_{ij}\pi_i = p_{ji}\pi_j$.

The *conductance* of an ergodic Markov chain X_t is the quantity

$$\mathbf{\Phi} = \min_{A \subseteq \Omega : \sum_{i \in A} \pi_i \leq 1/2} \frac{\sum_{i \in A} \sum_{j \in \bar{A}} \pi_i p_{ij}}{\sum_{i \in A} \pi_i}.$$

It is explained in [**?**] that the number of steps required for an ergodic time-reversible Markov chain to lose its memory and approach stationary distribution is $O(\mathbf{\Phi}^{-2} \ln(1/\pi_{\min}))$, where $\pi_{\min}$ is the minimum stationary probability over all states in Ω. When this number of steps is a polynomial in $\ln(|\Omega|)$, we call the chain *rapidly mixing*.

We now apply these concepts to the following Markov chain. Given a matroid M, the set of states of the Markov chain is $\mathcal{B}(M)$. If X_t is the state (basis) at time t, then with probability at least $1/2$, $X_{t+1} = X_t$, otherwise choose e from X_t and f from E uniformly at random and, if $X' = X_t - e \cup f \in \mathcal{B}$, then $X_{t+1} = X'$, else $X_{t+1} = X_t$.

By using the bases exchange property, this Markov chain is easily seen to be irreducible, and by definition, aperiodic. Furthermore, by definition, it is symmetric, therefore it converges to the uniform distribution over $\mathcal{B}$ and is time-reversible.

In the case of the previous Markov chain we have that the transition probabilities are bounded below by $|E(M)|^{-2}$, and clearly, $|E(M)| \sim \log(|\mathcal{B}|)$. Therefore, if we have an oracle to test whether a given set is a basis in M, this random walk can be used as an efficient almost uniform sampling scheme for the set of bases of the matroid M.

If, as in [**?**], an almost uniform sampling scheme can be used to approximate the ratio $r = |\mathcal{B}_e|/|\mathcal{B}|$ up to arbitrary accuracy, where $\mathcal{B}_e$ are the set of bases containing the element $e \in E(M)$, then the question of approximating $|\mathcal{B}|$ reduces to that of approximating the number of bases of M/e. Thus the size of the problem reduces efficiently from n to $n-1$. We conclude that if the bases-exchange graphs of a matroid M and all its minors possess inverse polynomial cut set expansion, then there is an efficient scheme to approximate $|\mathcal{B}|$. Here we have to mention that Azar, Broder and Frieze [**?**] have proved that it is impossible to get a good approximation in deterministic polynomial time of the number of bases of a matroid if the matroid is given to us by a basis oracle. So the only hope to approximate this #P-hard problem is by using randomised algorithms.

For the expansion of the bases-exchange graph there is a well-known conjecture due to Mihail and Vazirani [**?**].

Conjecture 4.1.2 (Matroid-Expansion Conjecture)**.** *For any matroid M, the bases-exchange graph $G(M)$ has cut set expansion at least 1.*

This conjecture can be also stated as follows: "for any bipartition of the vertices of a bases-exchange graph, the number of edges incident to both partition classes is at least as large as the size of the smaller partition class".

The Matroid-Expansion conjecture is an instance of a more general conjecture also from [**?**], namely

Conjecture 4.1.3. *Let P be any 0-1 polytope, that is, a polytope whose vertices are vectors with entries in $\{0,1\}^n$ for some natural number n, then the graph of the 1-skeleton of P has cut set expansion at least 1.*

4.2 Balanced Matroids

Among all possible bases-exchange graphs, one that is common in graph theory as well as easy to work with is the *n-cube*. This graph has as vertex set all the n-tuples with entries in the set $\{0, 1\}$, and two n-tuples are jointed by an edge if they differ just in one coordinate. This graph is, apart form several other interpretations, the bases-exchange graph of the graphic matroid $M = P_1 \oplus \cdots \oplus P_n$, where each P_i consists of a cycle of size 2.

There are several methods to prove good expansion in the n-cube. For example, there is a proof using isoperimetric inequalities [**?**], one using an inductive/balance argument [**?**] and one using canonical paths [**?**]. The extension of the last two ideas to bases-exchange graphs led to the key concept of *balance*. In this and the next section we describe, with some minor extensions, the work of Feder and Mihail on balanced matroids.

As we work with bases of the matroid that contain or avoid a specific element, we need some notation. If M has set of bases $\mathcal{B}$, then the set of bases that contain (avoid) $e \in E(M)$ is denoted by $\mathcal{B}_e$ ($\mathcal{B}_{\bar{e}}$). Thus, as there is a natural bijection between $\mathcal{B}_e$ ($\mathcal{B}_{\bar{e}}$) and the set of bases of M/e ($M \backslash e$), the notation can be used to denote the bases of any minor of M, say $M \setminus X/Y$, by $\mathcal{B}_{\bar{x_1},\ldots,\bar{x_r},y_1,\ldots y_s}$, where $X = \{x_1, \ldots, x_r\}$ and $Y = \{y_1, \ldots, y_s\}$. Even more, we use this notation for any subset $X \subseteq \mathcal{B}$, so X_e will denote $X \cap \mathcal{B}_e$ and $X_{\bar{e}}$ will denote $X \cap \mathcal{B}_{\bar{e}}$.

The n-cube has n natural bipartitions of its vertex set by considering a fixed coordinate j, $1 \leq j \leq n$, and taking the set of vertices with coordinate j equal to 0 and the set of vertices with coordinate j equal to 1. There are as well some natural bipartite subgraphs of the bases-exchange graph that we call the *restricted bases-exchange graphs*.

Definition 4.2.1. *Let M be a matroid, and $G(M)$ be its bases-exchange graph. The graph $G_e(M)$ is the bipartite graph with vertex set $\mathcal{B}_e \dot{\cup} \mathcal{B}_{\bar{e}}$ and there is an edge (B_1, B_2) with $B_1 \in \mathcal{B}_e$ and $B_2 \in \mathcal{B}_{\bar{e}}$, if there exists $f \in E(M) - e$, such that $(B_1 - e) \cup f = B_2$, that is, there is a bases exchange between bases B_1 and B_2 involving e.*

But when we have the bipartition of the n-cube into 2 $(n-1)$-cubes, the edges in this cut have the strong property that every set in one bipartition class has a neighbour set of proportional size, in fact they have the same cardinality. So, for the restricted bases-exchange graph of a matroid we ask for this property.

Definition 4.2.2. *A matroid M is said to enforce ratios on an element $e \in E(M)$ if the following conditions hold.*

$$\begin{array}{rl} \textit{For all } A \subset \mathcal{B}_e \textit{ we have} & \frac{|\Gamma(A)|}{|\mathcal{B}_{\bar{e}}|} \geq \frac{|A|}{|\mathcal{B}_e|}, \quad \textit{and} \\ \textit{for all } A \subset \mathcal{B}_{\bar{e}} \textit{ we have} & \frac{|\Gamma(A)|}{|\mathcal{B}_e|} \geq \frac{|A|}{|\mathcal{B}_{\bar{e}}|}, \end{array}$$

where $\Gamma(A)$ is the set of the neighbours of A in $G_e(M)$.

If M enforces ratios for some $e \in E(M)$ we said that M enforces ratios partially. If M enforces ratios for all elements e then we said that M enforces ratios.

Mihail and Sudan [**?**] prove that if all matroids enforce ratios then the Matroid-Expansion conjecture holds. Examination of their proof shows that it relies only on the fact that M enforces ratios for some element e and that M/e and $M \setminus e$ have cut set expansion at least 1. We state this mild extension but first we need a technical definition. Here, for two disjoint sets A_1 and A_2, the notation $A_1 \dot{\cup} A_2$ used in this chapter makes emphasis on the fact that A_1 and A_2 are disjoint sets.

Definition 4.2.3. *Let M be a matroid and e an element of M. Then, M is said to have the* crossed-cut *property on the element e if its bases-exchange graph satisfies the conditions that for any subset X of the set of bases of M, with $X = X_e \dot{\cup} X_{\bar{e}}$ and $|X_e| \geq |\mathcal{B}_e|/2$ or $|X_{\bar{e}}| \geq |\mathcal{B}_{\bar{e}}|/2$ but not both,*

$$|C(X_e, \mathcal{B}_{\bar{e}} \setminus X_{\bar{e}})| + |C(X_{\bar{e}}, \mathcal{B}_e \setminus X_e)| \geq ||X_e| - \frac{|\mathcal{B}_e|}{2}| + |\frac{|\mathcal{B}_{\bar{e}}|}{2} - |X_{\bar{e}}||$$

where $C(X_e, \mathcal{B}_{\bar{e}} \setminus X_{\bar{e}})$ denotes the set of edges that go from X_e to $\mathcal{B}_{\bar{e}} \setminus X_{\bar{e}}$ in the restricted bases-exchange graph of M, $C(X_{\bar{e}}, \mathcal{B}_e \setminus X_e)$ is defined similarly.

Lemma 4.2.4. *If a matroid M enforces ratios on an element e, then M has the crossed-cut property on the same element e.*

Proof. Let $X = X_e \dot{\cup} X_{\bar{e}}$ with $X_e \subset \mathcal{B}_e$, $|X_e| \geq |\mathcal{B}_e|/2$ and $X_{\bar{e}} \subset \mathcal{B}_{\bar{e}}$, $|X_{\bar{e}}| < |\mathcal{B}_{\bar{e}}|/2$.

By the hypothesis

$$\frac{|\Gamma(X_e)|}{|\mathcal{B}_{\bar{e}}|} \geq \frac{|X_e|}{|\mathcal{B}_e|} \geq \frac{1}{2},$$

so we can select a subset $Y \subseteq \Gamma(X_e)$ such that $Y \cap X_{\bar{e}} = \emptyset$ and $|Y| = |\mathcal{B}_{\bar{e}}|/2 - |X_{\bar{e}}|$. Let $C(X_e, Y)$ be the set of edges that go from X_e to Y, then clearly

$$|C(X_e, Y)| \geq |Y| \geq \frac{|\mathcal{B}_{\bar{e}}|}{2} - |X_{\bar{e}}|.$$

Take now the subset $Y' = \mathcal{B}_{\bar{e}} \setminus (Y \cup X_{\bar{e}})$ that, by choice of Y, has size $\mathcal{B}_{\bar{e}}/2$. Again, by enforcement of ratios we get that

$$\frac{|\Gamma(Y')|}{|\mathcal{B}_e|} \geq \frac{|Y'|}{|\mathcal{B}_{\bar{e}}|} \geq \frac{1}{2},$$

so $|\Gamma(Y') \cap X_e| \geq |X_e| - |\mathcal{B}_e|/2$. Let $C(X_e, Y')$ be the set of edges that go from X_e to Y', then

$$|C(X_e, Y')| \geq |\Gamma(Y') \cap X_e| \geq |X_e| - |\mathcal{B}_e|/2.$$

As

$$|C(X_e, \mathcal{B}_{\bar{e}} \setminus X_{\bar{e}})| + |C(X_{\bar{e}}, \mathcal{B}_e \setminus X_e)| \geq |C(X_e, Y)| + |C(X_e, Y')|$$

we conclude that

$$|C(X_e, \mathcal{B}_{\bar{e}} \setminus X_{\bar{e}}| + |C(X_{\bar{e}}, \mathcal{B}_e \setminus X_e))| \geq |\mathcal{B}_e|/2 \geq ||X_e| - \frac{|\mathcal{B}_e|}{2}| + |\frac{|\mathcal{B}_{\bar{e}}|}{2} - |X_{\bar{e}}||$$

The other case is similar. □

The cross-cut property is what is needed to prove cut set expansion inductively.

Theorem 4.2.5. *If M has the crossed-cut property on an element e and M/e and $M \setminus e$ have cut set expansion at least 1, then M has cut set expansion at least 1.*

Proof. Let X be a subset of bases with $|X| \leq |\mathcal{B}|/2$, and again $X = X_e \dot{\cup} X_{\bar{e}}$, where $X_e \subseteq \mathcal{B}_e$ and $X_{\bar{e}} \subseteq \mathcal{B}_{\bar{e}}$. If $|X_e| \leq |\mathcal{B}_e|/2$ and $|X_{\bar{e}}| \leq |\mathcal{B}_{\bar{e}}|/2$, then $|C(X)|$ is at least, $|C(X_e)|$ (with X_e in $G(M/e)$) plus $|C(X_{\bar{e}})|$ (with $X_{\bar{e}}$ in $G(M \setminus e)$). By hypothesis this number is at least $|X_e| + |X_{\bar{e}}|$.

If $|X_e| \geq |\mathcal{B}_e|/2$ and $|X_{\bar{e}}| \leq |\mathcal{B}_{\bar{e}}|/2$ then we have that $|C(X)|$ is

$$|C(X_e)| + |C(X_{\bar{e}})| + |C(X_e, \mathcal{B}_e \setminus X_e)| + |C(X_{\bar{e}}, \mathcal{B}_{\bar{e}} \setminus X_{\bar{e}})|$$

The first two terms sum to at least $|\mathcal{B}_e| - |X_e| + |X_{\bar{e}}|$, as M/e and $M \setminus e$ have cut set expansion at least 1. The last two terms sum to at least $|X_e| - \frac{|\mathcal{B}_e|}{2} + \frac{|\mathcal{B}_{\bar{e}}|}{2} - |X_{\bar{e}}|$ by the crossed-cut property. These two partial results give a total sum of at least $|\mathcal{B}|/2$ that is bigger than $|X|$. The other case is similar. □

Corollary 4.2.6. *If all matroids in a minor-closed class $\mathcal{M}$ enforce ratios then all matroids in $\mathcal{M}$ have cut set expansion at least 1.*

Proof. If M has 1 element, then either M is a coloop or a loop, in the first case $G(M)$ has cut set expansion 1, and we can consider that if M is a loop, then $G(M)$ has cut set expansion 1, as $\mathcal{B}(M) = \{\emptyset\}$.

Suppose that the result is true for all matroids of size n. Then, let M be a matroid of size $n+1$, we have that M satisfies the hypothesis of Theorem 4.2.5, that is, M enforces ratios for some element e, and $M \setminus e$ and M/e have cut set expansion 1, so, we can conclude that M has cut set expansion 1. □

It is, therefore, important to know which matroids enforce ratios. Observe that if e and f are two distinct elements of M, the subset of bases $A = \mathcal{B}_{ef}$ is such that $\Gamma(A) = \mathcal{B}_{\bar{e}f}$, where as before $\Gamma(A)$ is the set of neighbours of A in $G_e(M)$. Because M enforces ratios we have that $|\mathcal{B}_e||\mathcal{B}_{\bar{e}f}| \geq |\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}}|$. But this condition is equivalent to a notion in matroid theory that was studied much early, see [**?**], for other reasons and now turns to be very important in the Matroid-Expansion Conjecture. We say that:

Definition 4.2.7. *A matroid M is negatively correlated if for all distinct $e, f \in E(M)$,*

$$|\mathcal{B}_e||\mathcal{B}_f| \geq |\mathcal{B}_{ef}||\mathcal{B}|. \tag{4.2.1}$$

We also say that if two particular elements e and f of the matroid M satisfy 4.2.1 then they are negatively correlated. This property of negative correlation can be put in several different forms as we see in the following theorem. Here we denote by $\Pr[f]$ the probability that a basis chosen uniformly at random contains the element f and, by $\Pr[f|e]$, the probability that a basis chosen uniformly at random from the set $\mathcal{B}_e$ contains the element f.

Theorem 4.2.8. *The following statements about a matroid M are equivalent.*

1. *M is negatively correlated.*
2. *For all e and f, $|\mathcal{B}_e||\mathcal{B}_{\bar{e}f}| \geq |\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}}|$.*
3. *For all e and f, $|\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}f}| \geq |\mathcal{B}_{\bar{e}\bar{f}}||\mathcal{B}_{ef}|$.*
4. *For all e and f, $|\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}}| \geq |\mathcal{B}_{\bar{e}\bar{f}}||\mathcal{B}_e|$.*
5. *M^* is negatively correlated.*
6. *For all e and f, $\Pr[f] \geq \Pr[f|e]$, where e is not a loop of M.*

7. *For all e and f,* $\Pr[f|\bar{e}] \geq \Pr[f|e]$,*where e is not a loop of M.*

Proof. Straightforward algebraic manipulation. □

Every uniform matroid and all projective geometries $PG(r, q)$ are negatively correlated matroids [**?**]. Even more, Feder and Mihail [**?**] prove that all regular matroids are negatively correlated. However, this is not true for all matroids as the following examples show.

Example 1. The matroid defined by the Euclidean representation of Figure 4.2 is called S_8, see [**?**]. Calculations give $|\mathcal{B}_g| = 28$, $|\mathcal{B}_f| = 20, |\mathcal{B}_{ef}| = 12$ and $|\mathcal{B}| = 48$, so S_8 is not negatively correlated [**?**].

Figure 4.2: The Euclidean representation of the matroid S_8.

Example 2. For the matroid defined as the 6-truncation of the graph in Figure 4.3, that is, its bases are the spanning forests of size 6, we have $|\mathcal{B}_g| = 112$, $|\mathcal{B}_f| = 272, |\mathcal{B}_{ef}| = 80$ and $|\mathcal{B}| = 384$, so this matroid is not negatively correlated.

Figure 4.3 also shows this matroid as a transversal matroid, see page **??** . Here, the family of sets is $A_1 = \{1, 2, h\}$, $A_2 = \{3, 4, h\}$, $A_3 = \{5, 6, h\}$, $A_4 = \{2, 4, 6, 8, 10, g, h\}$, $A_5 = \{7, 8, h\}$ and $A_6 = \{9, 10, h\}$. So, the vertices in $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, h, g\}$ of a given matching in the bipartite graph of the figure correspond to a partial transversal of the family $(A_1, \ldots, A_6)$, that is, an independent set of the induced transversal matroid. If a matroid is the dual of a transversal matroid it is called a *strict gammoid*, see [**?**].

Corollary 4.2.9. *There exist binary matroids, transversal matroids and strict gammoids which are not negatively correlated.*

Figure 4.3: The truncation of a graph which is not negatively correlated, also its representation as a transversal matroid

As the matroid S_8 is a minor of $PG(3,2)$ we conclude that negative correlation is not closed under minors. In order to have this important property balanced matroids were defined.

Definition 4.2.10. *A matroid M is said to be balanced if it and every minor of M is negatively correlated.*

From the previous discussion we have that uniform and regular matroids are balanced but binary, transversal, and therefore real representable matroids, need not be. It is known that every binary matroid that does not contain S_8 as a minor is balanced [**?**] but we have not found any published proof. By asking some people we have reconstructed a proof that we include in Appendix **??**, this because the proof involves several matroid concepts that are not relevant for this chapter. However, we give some more results about matroid operations on balanced matroids that are important for Appendix **??** and this chapter.

Theorem 4.2.11. *Let M and N be balanced matroids then the following statements are true.*

1. *M^* is balanced.*
2. *The direct sum of M and N, $M \oplus N$, is balanced.*
3. *The 2-sum of M and N, $M \oplus_2 N$, is balanced.*

The first statement follows from Theorem 4.2.8. We prove the other two in a more general context in Section 4.4.

Theorem 4.2.12. *Every binary matroid that does not contain S_8 as a minor is balanced.*

Proof. See Appendix **??**. □

4.3 Near-balanced matroids

In this section we continue showing the inductive/balance argument that was used by [**?**] to prove that balanced matroids have cut set expansion 1. We have made a mild generalisation to the proof in [**?**] to include an important matroid for the class of balanced matroids, that is the matroid S_8. We also outline the canonical path argument that proves a better cut set expansion for balanced matroids and give us the possibility of strengthening the Matroid-Expansion conjecture. First, we extend the concept of balanced matroid to *near-balanced* matroids.

Definition 4.3.1. *A matroid M is near-balanced if there exists an element $e \in E(M)$ such that for all f, e and f are negatively correlated and every proper minor of M is balanced.*

Theorem 4.3.2. *For every balanced matroid the bases-exchange graph enforces ratios*

Proof. Let M be a balanced matroid with n elements and rank r, we prove the statement by induction on the rank r of M.

First we delete all the loops of M, if any, as this does not change the bases-exchange graph of M. Let $X_e \subseteq \mathcal{B}_e$ and $\hat{X}_{\bar{e}} = \Gamma(X_e)$ in the restricted bases-exchange graph. We have to show that $|\hat{X}_{\bar{e}}|/|\mathcal{B}_{\bar{e}}| \geq |X_e|/|\mathcal{B}_e|$.

If M has rank 1, then either $|X_e| = 1$, in which case $|\hat{X}_{\bar{e}}| = n-1$ and $|\hat{X}_{\bar{e}}|/|\mathcal{B}_{\bar{e}}| = 1 = |X_e|/|\mathcal{B}_e|$, or $|X_e| = 0$, in which case $|\hat{X}_{\bar{e}}|/|\mathcal{B}_{\bar{e}}| = 0 = |X_e|/|\mathcal{B}_e|$. If M has just one element, $n = 1$, we assume the result as we can consider $1 = 0/0$ and for this particular case we have $|\hat{X}_{\bar{e}}|/|\mathcal{B}_{\bar{e}}| = 0/0$.

Suppose now that M has rank r at least 2 and that the statement is true for all matroids of rank $r-1$. Let $f \in E(M)\backslash e$ and consider $X_{ef} = X_e \cap \mathcal{B}_{ef}$, $X_{e\bar{f}} = X_e \cap \mathcal{B}_{e\bar{f}}$, $\hat{X}_{\bar{e}f} = \hat{X}_{\bar{e}} \cap \mathcal{B}_{\bar{e}f}$ and $\hat{X}_{\bar{e}\bar{f}} = \hat{X}_{\bar{e}} \cap \mathcal{B}_{\bar{e}\bar{f}}$. We have the following decomposition.

$$\frac{|X_e|}{|\mathcal{B}_e|} = \frac{|X_{ef}|}{|\mathcal{B}_{ef}|}\frac{|\mathcal{B}_{ef}|}{|\mathcal{B}_e|} + \frac{|X_{e\bar{f}}|}{|\mathcal{B}_{e\bar{f}}|}\frac{|\mathcal{B}_{e\bar{f}}|}{|\mathcal{B}_e|} \tag{4.3.1}$$

$$\frac{|\hat{X}_{\bar{e}}|}{|\mathcal{B}_{\bar{e}}|} = \frac{|\hat{X}_{\bar{e}f}|}{|\mathcal{B}_{\bar{e}f}|}\frac{|\mathcal{B}_{\bar{e}f}|}{|\mathcal{B}_{\bar{e}}|} + \frac{|\hat{X}_{\bar{e}\bar{f}}|}{|\mathcal{B}_{\bar{e}\bar{f}}|}\frac{|\mathcal{B}_{\bar{e}\bar{f}}|}{|\mathcal{B}_{\bar{e}}|}. \tag{4.3.2}$$

Now, we are going to use the averaging principle which says that if α_1 and α_2 are two positive real numbers less or equal to 1 with $\alpha_1 \geq \alpha_2$, then for any two real numbers $x \geq y$ we have

$$\alpha_1 x + \beta_1 y \geq \alpha_2 x + \beta_2 y,$$

where $\beta_1 = 1 - \alpha_1$ and $\beta_2 = 1 - \alpha_2$.

As e and any $f \in E(M) \setminus e$ are negatively correlated we have by Theorem 4.2.8 $|\mathcal{B}_{ef}|/|\mathcal{B}_e| \leq |\mathcal{B}_{\bar{e}f}|/|\mathcal{B}_{\bar{e}}|$. Suppose there exists f such that $|X_{ef}|/|\mathcal{B}_{ef}| \geq |X_{e\bar{f}}|/|\mathcal{B}_{e\bar{f}}|$, then by the averaging principle,

$$\frac{|X_e|}{|\mathcal{B}_e|} \leq \frac{|X_{ef}|}{|\mathcal{B}_{ef}|}\frac{|\mathcal{B}_{\bar{e}f}|}{|\mathcal{B}_{\bar{e}}|} + \frac{|X_{e\bar{f}}|}{|\mathcal{B}_{e\bar{f}}|}\frac{|\mathcal{B}_{\bar{e}\bar{f}}|}{|\mathcal{B}_{\bar{e}}|}. \tag{4.3.3}$$

Where we use the substitution $\alpha_1 = |\mathcal{B}_{\bar{e}f}|/|\mathcal{B}_{\bar{e}}|$, $\alpha_2 = |\mathcal{B}_{ef}|/|\mathcal{B}_e|$, $x = |X_{ef}|/|\mathcal{B}_{ef}|$ and $y = |X_{e\bar{f}}|/|\mathcal{B}_{e\bar{f}}|$.

Note that the neighbours of X_{ef} in $G_e(M/f)$ are contained in $\hat{X}_{\bar{e}f}$ and the neighbours of $X_{e\bar{f}}$ in $G_e(M\setminus f)$ are contained in $\hat{X}_{\bar{e}\bar{f}}$. By the induction hypothesis M/f and $M \setminus f$ enforce ratios therefore $|X_{ef}|/|\mathcal{B}_{ef}| \leq |\hat{X}_{\bar{e}f}|/|\mathcal{B}_{\bar{e}f}|$ and $|X_{e\bar{f}}|/|\mathcal{B}_{e\bar{f}}| \leq |\hat{X}_{\bar{e}\bar{f}}|/|\mathcal{B}_{\bar{e}\bar{f}}|$ and from 4.3.3 and 4.3.2 we get the result.

It just remains to prove that there exists f such that $|X_{ef}|/|\mathcal{B}_{ef}| \geq |X_{e\bar{f}}|/|\mathcal{B}_{e\bar{f}}|$ that by equation 4.3.1 is equivalent to $|X_{ef}|/|\mathcal{B}_{ef}| \geq |X_e|/|\mathcal{B}_e|$. This follows from the equalities

$$\sum_{f \neq e} \frac{|X_{ef}|}{|X_e|} = r(M) - 1 = \sum_{f \neq e} \frac{|\mathcal{B}_{ef}|}{|\mathcal{B}_e|}.$$

To prove, for example, the first equality, construct a bipartite graph with vertex set $X_e \dot{\cup} (E(M) \setminus e)$ and where there is an edge between a basis $B \in X_e$ and element $g \in E(M) \setminus e$ if $g \in B$. Then, the number of edges is equal to $\sum_{f \neq e} |X_{ef}|$ and also is $(r(M) - 1)|X_e|$.

To finish the proof of the Theorem, we just note that being balanced is a self dual property and the bases-exchange graphs of M and M^* are isomorphic, so the same argument can be used to prove that for all $X \subseteq \mathcal{B}_{\bar{e}}$, $|\Gamma(X)|/|\mathcal{B}_e| \geq |X|/|\mathcal{B}_{\bar{e}}|$. □

Theorem 4.3.3. *If M is near-balanced then it enforces ratios partially.*

Proof. Let e be an element of M such that for all f, e and f are negatively correlated. As any minor of M is balanced, we can proceed as in the inductive step of the previous theorem. □

Here, we obtain a minor extension to the work in [**?**].

Corollary 4.3.4. *If a matroid is near-balanced then it has cut set expansion 1.*

Proof. By Theorem 4.3.2, Lemma 4.2.4 and Theorem 4.2.5 balanced matroids have cut set expansion 1. By the previous corollary, Lemma 4.2.4, Theorem 4.2.5 and the previous comment all near-balanced matroids have cut set expansion 1. □

Feder and Mihail also introduced the concept of a fractional matching. This is a property of bipartite graphs.

Definition 4.3.5. *A bipartite graph $G(X, A)$, where $X = X_1 \dot\cup X_2$, has a fractional matching if there exists an assignment of nonnegative weights to the edges in A such that for each vertex $u \in X_1$, the sum of the weights of edges incident to u is $|X_2|$, and for each vertex $v \in X_2$, the sum of the weights of the edges incident to v is $|X_1|$.*

We include a result of [**?**] with a slightly different statement but the same proof.

Theorem 4.3.6. *Suppose that M enforces ratios on an element $e \in E(M)$, then $G_e(M)$ admits a fractional matching.*

Proof. We construct the following bipartite graph G'_e. The vertices are

$$V_1 \dot\cup V_2 = \{(B, i) | B \in \mathcal{B}_e, 1 \leq i \leq |\mathcal{B}_{\bar e}|\} \dot\cup \{(D, j) | D \in \mathcal{B}_{\bar e}, 1 \leq j \leq |\mathcal{B}_e|\},$$

that is, G'_e has $|\mathcal{B}_{\bar e}|$ copies of each vertex in G_e corresponding to a basis in $\mathcal{B}_e$ and $|\mathcal{B}_e|$ copies of each vertex in G_e corresponding to a basis in $\mathcal{B}_{\bar e}$. For vertices $(B, i) \in V_1$ and $(D, j) \in V_2$, there is an edge if B and D have an edge in G_e.

The hypothesis that M enforces ratios on e implies that if A is a subset of vertices in V_1, then for

$$\hat A = \{B | (B, j) \in A, \text{ for some } 1 \leq j \leq |\mathcal{B}_{\bar e}|\} \subseteq \mathcal{B}_e$$

we have that

$$|A| \leq |\hat A||\mathcal{B}_{\bar e}| \leq |\Gamma_{G_e}(\hat A)||\mathcal{B}_{\bar e}|.$$

But, by construction, $|\Gamma_{G'_e}(A)| = |\Gamma_{G_e}(\hat A)||\mathcal{B}_{\bar e}|$, where here $\Gamma_H(W)$ is the set of neighbours of W in the graph H. Thus, G'_e satisfies Hall's condition and G'_e admits a perfect matching P. So, if we identify the copies of each basis in G'_e to obtain G_e, and we assign to each edge a weight equal to the number of edges of P that correspond to it, we obtain a fractional matching for G_e. □

Here, again, we note that negative correlation is a necessary property for fractional matching.

Theorem 4.3.7. *If for every $e \in E(M)$, $G_e(M)$ admits a fractional matching, then M is negatively correlated.*

Proof. Let e and f belong to $E(M)$, then the sum of the weights of the vertices $\mathcal{B}_{\bar{e},\bar{f}}$ is $|\mathcal{B}_e||\mathcal{B}_{\bar{e},\bar{f}}|$. As all these edges have also an end at vertices in $\mathcal{B}_{e,\bar{f}}$, and the sum of the weights on this set of vertices is $|\mathcal{B}_{\bar{e}}||\mathcal{B}_{e,\bar{f}}|$ then we have $|\mathcal{B}_e||\mathcal{B}_{\bar{e},\bar{f}}| \leq |\mathcal{B}_{\bar{e}}||\mathcal{B}_{e,\bar{f}}|$. By Theorem 4.2.8 that is equivalent to M being negatively correlated. □

Using fractional matching, Feder and Mihail managed to define paths between every pair of vertices of the bases-exchange graph of a balanced matroid that do not saturate any edge, that is, they ensure that the number of (directed) paths that go through any edge is at most half the number of vertices. Their construction can be used to define the same paths in a near-balanced matroid with a slight change. We now give a description of this construction.

First, choose an order in which we pick the elements of M, say $e_1, \ldots, e_m$. Now, we follow exactly the original construction, which we take from [**?**]. We notice that this change will make invalid other results in [**?**] that rely on a random choice of the elements in $E(M)$.

Let $B' = B_{\dot{e}_1,\ldots,\dot{e}_m}$, we construct all the paths to B', where each $\dot{e}_i$ is either e_i or $\bar{e}_i$ depending if e_i belongs or not to B'. At the beginning of the construction all the paths to B' are distributed uniformly among the bases $\mathcal{B}$, in the sense that each basis contains the first vertex of one path to B'. The construction ensures that after i steps, the paths destined to B' are uniformly arranged over $\mathcal{B}_i = \mathcal{B}_{\dot{e}_1,\ldots,\dot{e}_i}$, the set of bases that agree with B' in the first i elements, in the sense that the expected number of paths at each such basis after i steps is the same for all of them. We notice that this condition holds initially when $i = 0$ and $\mathcal{B}_i = \mathcal{B}$. Assume that after i steps, the paths destined to B' are uniformly arranged over $\mathcal{B}_i = \mathcal{B}_{i+1} \cup \mathcal{B}'_{i+1}$, where $\mathcal{B}_{i+1} = \mathcal{B}_{\dot{e}_1,\ldots,\dot{e}_i,\dot{e}_{i+1}}$ and $\mathcal{B}'_{i+1} = \mathcal{B}_{\dot{e}_1,\ldots,\dot{e}_i,\bar{\dot{e}}_{i+1}}$. The paths at $\mathcal{B}_i$ that are already at $\mathcal{B}_{i+1}$ are left there for the $(i+1)$-step. The paths at $\mathcal{B}_i$ that are currently at $\mathcal{B}'_{i+1}$ are sent to B_{i+1} by the following procedure.

Consider the minor $N = M_{\dot{e}_1,\ldots,\dot{e}_i}$, because M is balanced or near-balanced, this minor enforces ratios on e_{i+1} and then the restricted bases-exchange graph $G_{e_{i+1}}(N)$ has a fractional matching. In the case that $i = 0$ and M is near-balanced, we know by definition that there exists an element $e \in E(M)$ that enforces ratios, so without loss of generality, we can suppose that $e = e_1$. Now, to continue the construction of the paths, we assign probabilities to each possible exchange, chosen proportionally to

the associated weights in the fractional matching, to ensure that after $(i+1)$-steps, the paths destined to B' are uniformly arranged over $\mathcal{B}_{i+1}$.

For example, if p is the expected number of paths at each vertex in $V_1 = \mathcal{B}'_{i+1}$ and $V_2 = \mathcal{B}_{i+1}$, then we can assign to each edge a the quantity $\omega(a)/|V_2|$, the probability to be used to extend a path, where $\omega(a)$ is the weight of the edge in the fractional matching. Thus, for a particular edge $a = (B', B)$, the expected number of paths using that edge would be $p\omega(a)/|V_2|$, and the expected number of paths that will use a particular vertex B in $\mathcal{B}_{i+1}$ is

$$\sum_{\substack{a=(B',B)\\ B'\in V_1}} \frac{p\omega(a)}{|V_2|} = \frac{p|V_1|}{|V_2|}$$

So, the $p|V_1|$ paths in $\mathcal{B}'_{i+1}$ are uniformly distributed in $\mathcal{B}_{i+1}$.

After m steps, all the paths are at $\mathcal{B}_m = \{B'\}$, as desired. Observe that with this construction, at step i the number of paths that have to cross from $\mathcal{B}'_i$ to $\mathcal{B}_i$, when we consider all possible destinations, is

$$\frac{|\mathcal{B}|}{|\mathcal{B}_{i-1}|}|\mathcal{B}_i||\mathcal{B}'_i|.$$

This is because for each vertex in $\mathcal{B}_i$, there are $(|\mathcal{B}|/|\mathcal{B}_{i-1}|)|\mathcal{B}'_i|$ paths in $|\mathcal{B}'_i|$ to it, that is, $|\mathcal{B}|/|\mathcal{B}_{i-1}|$ paths for each vertex in $|\mathcal{B}'_i|$. From this it is easy to get the following results of [**?**].

Theorem 4.3.8. *At step i, the expected number of paths leaving a basis B' in $\mathcal{B}_{\dot{e}_1,\ldots,\bar{\dot{e}}_i}$ at step i is $|\mathcal{B}|\Pr[\dot{e}_i|\dot{e}_1\ldots\dot{e}_{i-1}]$ and the expected number of paths entering B in $\mathcal{B}_{\dot{e}_1,\ldots,\dot{e}_i}$ is $|\mathcal{B}|\Pr[\bar{\dot{e}}_i|\dot{e}_1\ldots\dot{e}_{i-1}]$.*

Note that a particular bases exchange involving two elements e_i and e_j, with $i<j$ can only be used during step i, so it also follows from the previous discussion that the expected number of paths using such an exchange at step i over all destinations is at most $|\mathcal{B}|\min(\Pr[\bar{\dot{e}}_i|\dot{e}_1\ldots\dot{e}_{i-1}], \Pr[\dot{e}_i|\dot{e}_1\ldots\dot{e}_{i-1}])$. In particular we have

Theorem 4.3.9. *The expected number of paths through any edge (B, B') is at most $|\mathcal{B}|/2$.*

Now, we have a result that was not mentioned in [**?**].

Theorem 4.3.10. *If M is a near-balanced matroid then it has strong cut set expansion at least 2.*

Proof. For $A \subseteq \mathcal{B}$, there are at least $|A||\mathcal{B} \setminus A|$ paths from A to $\mathcal{B} \setminus A$, as through any edge there pass at most $\frac{1}{2}|\mathcal{B}|$ of these paths. Thus, we have that $\frac{1}{2}|C(A)||\mathcal{B}| \geq |A||\mathcal{B} \setminus A|$. □

We will prove in Section 4.5 that the matroid S_8 is near-balanced. However, the class of near-balanced matroids is not big enough to include even binary matroids. For example, the matroid $S_8 \oplus_2 S_8$, where the base point is one of the elements h or g of Example 1, is neither near-balanced nor enforces ratios on any element.

4.4 α-Balanced Matroids

As we saw in the previous section, not all matroids are balanced but balance is a desirable property in a matroid. We extend the notion of balance with the hope that this new property will also help to prove good cut set expansion.

Definition 4.4.1. *Let α be a real number in the interval [0,1]. A matroid $M = (E, B)$ is α-correlated if for all e and f in E, $e \neq f$, we have*

$$\alpha|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}\bar{f}}| \leq |\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}f}|.$$

We can deduce some elementary properties. If M is α-correlated then, for all $\beta \in$[0,1] such that $\alpha \geq \beta$, we have that M is β-correlated. Also every matroid is 0-correlated and all balanced matroids are 1-correlated.

First, we prove that α-correlation implies similar properties to negative correlation.

Theorem 4.4.2. *For any $\alpha \in [0, 1]$, if M is α-correlated then the following statements are true.*

1. *M^* is α-correlated.*
2. *For all e and f, $|\mathcal{B}_e||\mathcal{B}_{\bar{e}f}| \geq \alpha|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}}|$.*
3. *For all e and f, $|\mathcal{B}_e||\mathcal{B}_f| \geq \alpha|\mathcal{B}||\mathcal{B}_{ef}|$.*
4. *For all e and f, $|\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}}| \geq \alpha|\mathcal{B}_{\bar{e}\bar{f}}||\mathcal{B}_e|$.*
5. *For all e and f, $\Pr[f|\bar{e}] \geq \alpha \Pr[f|e]$, where e is not a loop of M.*
6. *For all e and f, $\Pr[f] \geq \alpha \Pr[f|e]$, where e is not a loop of M.*

Proof. The first statement follows as the definition of α-correlation is self dual. For the second statement we have by hypothesis that

$$|\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}f}| \geq \alpha|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}\bar{f}}|.$$

As $\alpha \leq 1$, we also have the inequality

$$|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}f}| \geq \alpha|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}f}|,$$

so, adding the corresponding terms in both inequalities we have

$$|\mathcal{B}_{e\bar{f}}||\mathcal{B}_{\bar{e}f}| + |\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}f}| \geq \alpha\left(|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}\bar{f}}| + |\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}f}|\right),$$

that is equivalent to

$$|\mathcal{B}_{e}||\mathcal{B}_{\bar{e}f}| \geq \alpha|\mathcal{B}_{ef}||\mathcal{B}_{\bar{e}}|.$$

The proofs of the third and the fourth statements are similar to this one.

The fifth statement can be proved from the second statement by the following inequalities:

$$\Pr[f|\bar{e}] = \frac{|\mathcal{B}_{\bar{e}f}|}{|\mathcal{B}_{\bar{e}}|} \geq \alpha\frac{|\mathcal{B}_{ef}|}{|\mathcal{B}_{e}|} = \alpha\Pr[f|e].$$

The proof of the last statement is similar to this one. □

Of course, every matroid is α-correlated for some $\alpha \in [0,1]$ but the aim is to find a class of matroids with an α-correlation that makes it closer to the notion of being balanced, so we have the following definitions.

Definition 4.4.3. *For $\alpha \in [0,1]$, a matroid M is α-balanced, if every minor of M is α-correlated.*

The balance number *of a matroid M is the real number*

$$a(M) = \sup\{\alpha \in [0,1] | M \text{ is } \alpha\text{-balanced}\}.$$

Example 1 revisited. As we saw, the matroid S_8 is not negatively correlated, but from Table 4.4.1 we get that the balance number of S_8 is $\frac{8}{9}$.

x and y	$\mathcal{B}_{x\bar{y}}$	$\mathcal{B}_{\bar{x}y}$	$\mathcal{B}_{xy}$	$\mathcal{B}_{\bar{x}\bar{y}}$	$\mathcal{B}_{x\bar{y}}\mathcal{B}_{\bar{x}y}$ and $\mathcal{B}_{xy}\mathcal{B}_{\bar{x}\bar{y}}$
g, h	16	8	12	12	$128 < 144$
g, a_0	16	12	12	8	$192 > 96$
h, a_0	12	16	8	12	$192 > 96$
a_1, a_2	16	16	8	8	$256 > 64$
a_3, a_4	13	13	11	11	$139 > 121$

(4.4.1)

Here a_0 is any element in $\{a, b, c, d, e, f\}$. The pair $\{a_1, a_2\}$ is any of the pairs $\{a, b\}$, $\{c, d\}$ or $\{e, f\}$. Elements a_3 and a_4 are any pair in $\{a, b, c, d, e, f\}$ that is not any of the previous cases. □

As for balanced matroids, given a fixed α, α-balanced matroids are a structured class of matroids in the sense that it is a class of matroids closed under minors (by definition), direct sum and 2-sum. For the proof of this statement we need first the following technical lemma.

Lemma 4.4.4. *Let M and N be α-balanced matroids then the following statements are true.*

1. *The direct sum $M \oplus N$, is α-correlated.*

2. *The 2-sum of $M \oplus_2 N$, is α-correlated.*

Proof. For the proof we introduce some notation. For a matroid M, the quantity $|\mathcal{B}_{\dot{a_1},\ldots,\dot{a_k}}|$ is denoted by $m_{\dot{a_1},\ldots,\dot{a_k}}$, where $\dot{a_i}$ is either a_i or $\bar{a}_i$. So, for example, $m_{e,\bar{f}}$ is the number of bases containing element e and not containing f and m denotes the total number of bases.

For the first statement note the important fact that if $e \in M$, then $(M \oplus N)/e = (M/e) \oplus N$ and $(M \oplus N) \setminus e = (M \setminus e) \oplus N$, see [**?**]. Let $D = M \oplus N$, we want to prove that for all e and f, $\alpha d_{e,f} d_{\bar{e},\bar{f}} \leq d_{\bar{e},f} d_{e,\bar{f}}$. Let e and $f \in E(D)$, then we have three cases. If e and f are in $E(M)$ then $d_{e,f} = m_{e,f} n$, $d_{\bar{e},\bar{f}} = m_{\bar{e},\bar{f}} n$, $d_{\bar{e},f} = m_{\bar{e},f} n$ and $d_{e,\bar{f}} = m_{e,\bar{f}} n$. By hypothesis

$$\alpha m_{e,f} m_{\bar{e}} \leq m_{\bar{e},f} m_{e,\bar{f}},$$

so, we obtain that

$$\alpha d_{e,f} d_{\bar{e},\bar{f}} = \alpha m_{e,f} m_{\bar{e},\bar{f}} n^2 \leq m_{\bar{e},f} m_{e,\bar{f}} n^2 = d_{\bar{e},f} d_{e,\bar{f}}.$$

The case e and f in $E(N)$ is similar. If e is in $E(M)$ and f is in $E(N)$ then $d_{e,f} = m_e n_f$, $d_{\bar{e},\bar{f}} = m_{\bar{e}} n_{\bar{f}}$, $d_{\bar{e},f} = m_{\bar{e}} n_f$ and $d_{e,\bar{f}} = m_e n_{\bar{f}}$. So, as $\alpha \leq 1$,

$$\alpha d_{e,f} d_{\bar{e},\bar{f}} \leq d_{e,f} d_{\bar{e},\bar{f}} = m_e n_f m_{\bar{e}} n_{\bar{f}} = d_{\bar{e},f} d_{e,\bar{f}}.$$

For the second statement note that for $e \in M \setminus g$, where g is the base point for the 2-sum of M and N, $(M \oplus_2 N)/e = (M/e) \oplus_2 N$ and $(M \oplus_2 N) \setminus e = (M \setminus e) \oplus_2 N$, see [**?**]. Also, if B is a basis of $M \oplus_2 N$, then $B = B_1 \dot{\cup} B_2$, where B_1 is a basis of M and B_2 is a basis of N and either $g \in B_1$ or $g \in B_2$ but nor both, see page **??**.

Let $S = M \oplus_2 N$, we want to prove that for all e and f,

$$\alpha s_{e,f} s_{\bar{e},\bar{f}} \leq s_{\bar{e},f} s_{e,\bar{f}}. \tag{4.4.2}$$

Let e and $f \in E(S)$ and g be the base point in M and N. Again we have three cases. If e and f are in $E(M) \setminus g$, then

$$\begin{aligned}
s_{e,f} &= m_{g,e,f} n_{\bar{g}} + m_{\bar{g},e,f} n_g, \\
s_{\bar{e},\bar{f}} &= m_{g,\bar{e},\bar{f}} n_{\bar{g}} + m_{\bar{g},\bar{e},\bar{f}} n_g, \\
s_{\bar{e},f} &= m_{g,e,\bar{f}} n_{\bar{g}} + m_{\bar{g},e,\bar{f}} n_g, \\
s_{e,\bar{f}} &= m_{g,\bar{e},f} n_{\bar{g}} + m_{\bar{g},\bar{e},f} n_g.
\end{aligned}$$

First, we expand left-hand side of 4.4.2 and add the expression

$$\begin{aligned}
0 \equiv\ & \alpha n_g n_{\bar{g}} (m_{g,e,f} m_{g,\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}) \\
& - \alpha n_g n_{\bar{g}} (m_{g,e,f} m_{g,\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}).
\end{aligned}$$

$$\begin{aligned}
\alpha s_{e,f} s_{\bar{e},\bar{f}} &= \alpha n_{\bar{g}}^2 (m_{g,e,f} m_{g,\bar{e},\bar{f}}) + \alpha n_g^2 (m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}) \\
&\quad + \alpha n_g n_{\bar{g}} (m_{g,e,f} m_{\bar{g},\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{g,\bar{e},\bar{f}}) \\
&= \alpha n_{\bar{g}} (n_{\bar{g}} - n_g)(m_{g,e,f} m_{g,\bar{e},\bar{f}}) + \alpha n_g (n_g - n_{\bar{g}})(m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}) \\
&\quad + \alpha n_g n_{\bar{g}} (m_{g,e,f} m_{\bar{g},\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{g,\bar{e},\bar{f}} + m_{g,e,f} m_{g,\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}),
\end{aligned}$$

now, we apply the following simplification

$$\begin{aligned}
& m_{g,e,f} m_{\bar{g},\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{g,\bar{e},\bar{f}} + m_{g,e,f} m_{g,\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}} \\
&= m_{g,e,f} m_{\bar{e},\bar{f}} + m_{\bar{g},e,f} m_{\bar{e},\bar{f}} \\
&= m_{\bar{e},\bar{f}} m_{e,f}.
\end{aligned}$$

So, the last term in the previous inequalities can be simplified to

$$\alpha n_{\bar{g}} (n_{\bar{g}} - n_g)(m_{g,e,f} m_{g,\bar{e},\bar{f}}) + \alpha n_g (n_g - n_{\bar{g}})(m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}}) + \alpha n_g n_{\bar{g}} (m_{\bar{e},\bar{f}} m_{e,f}).$$

Now, we use that M, M/g and $M \setminus g$ are α-balanced to get the following inequalities

$$\begin{aligned}
\alpha m_{\bar{e},\bar{f}} m_{e,f} &\leq m_{\bar{e},f} m_{\bar{e},f}, \\
\alpha m_{g,e,f} m_{g,\bar{e},\bar{f}} &\leq m_{g,e,\bar{f}} m_{g,\bar{e},f}, \\
\alpha m_{\bar{g},e,f} m_{\bar{g},\bar{e},\bar{f}} &\leq m_{\bar{g},\bar{e},f} m_{\bar{g},e,\bar{f}}.
\end{aligned}$$

Therefore the last term obtained in our inequalities is less than or equal to

$$n_{\bar{g}}(n_{\bar{g}} - n_g)(m_{g,e,\bar{f}}m_{g,\bar{e},f}) + n_g(n_g - n_{\bar{g}})(m_{\bar{g},\bar{e},f}m_{\bar{g},e,\bar{f}}) + n_g n_{\bar{g}}(m_{\bar{e},f}m_{e,\bar{f}}),$$

expanding the term $m_{\bar{e},f}m_{e,\bar{f}}$ we obtain

$$\begin{aligned}
&n_{\bar{g}}(n_{\bar{g}} - n_g)(m_{g,e,\bar{f}}m_{g,\bar{e},f}) + n_g(n_g - n_{\bar{g}})(m_{\bar{g},\bar{e},f}m_{\bar{g},e,\bar{f}}) \\
&\quad + n_g n_{\bar{g}}(m_{g,\bar{e},f}m_{g,e,\bar{f}} + m_{g,\bar{e},f}m_{\bar{g},e,\bar{f}} + m_{\bar{g},\bar{e},f}m_{g,e,\bar{f}} + m_{\bar{g},\bar{e},f}m_{\bar{g},e,\bar{f}}),
\end{aligned}$$

and simplifiying the expression we finally get

$$\begin{aligned}
&n_{\bar{g}}^2(m_{g,\bar{e},f}m_{g,e,\bar{f}}) + n_g^2(m_{\bar{g},\bar{e},f}m_{\bar{g},e,\bar{f}}) \\
&\quad + n_g n_{\bar{g}}(m_{g,\bar{e},f}m_{\bar{g},e,\bar{f}} + m_{\bar{g},\bar{e},f}m_{g,e,\bar{f}}) \\
&= (m_{g,\bar{e},f}n_{\bar{g}} + m_{\bar{g},\bar{e},f}n_g)(m_{g,e,\bar{f}}n_{\bar{g}} + m_{\bar{g},e,\bar{f}}n_g) \\
&= s_{\bar{e},f}s_{e,\bar{f}}.
\end{aligned}$$

The case where e and f are in N is similar. If e is in $M \setminus g$ and f is in $N \setminus g$ we have:

$$\begin{aligned}
s_{e,f} &= m_{g,e}n_{\bar{g},f} + m_{\bar{g},e}n_{g,f}, \\
s_{\bar{e},\bar{f}} &= m_{g,\bar{e}}n_{\bar{g},\bar{f}} + m_{\bar{g},\bar{e}}n_{g,\bar{f}}, \\
s_{\bar{e},f} &= m_{g,\bar{e}}n_{\bar{g},f} + m_{\bar{g},\bar{e}}n_{g,f}, \\
s_{e,\bar{f}} &= m_{g,e}n_{\bar{g},\bar{f}} + m_{\bar{g},e}n_{g,\bar{f}}.
\end{aligned}$$

As M and N are α-balanced we have,

$$0 \leq (m_{g,\bar{e}}m_{\bar{g},e} - \alpha m_{g,e}m_{\bar{g},\bar{e}})(n_{\bar{g},f}n_{g,\bar{f}} - \alpha n_{g,f}n_{\bar{g},\bar{f}}),$$

expanding this expression we have

$$\alpha m_{g,e}m_{\bar{g},\bar{e}}n_{\bar{g},f}n_{g,\bar{f}} + \alpha m_{\bar{g},e}m_{g,\bar{e}}n_{g,f}n_{\bar{g},\bar{f}} \leq m_{g,\bar{e}}m_{\bar{g},e}n_{\bar{g},f}n_{g,\bar{f}} + \alpha^2 m_{g,e}m_{\bar{g},\bar{e}}n_{g,f}n_{\bar{g},\bar{f}}.$$

Using first this inequality and then the fact that $\alpha \in [0,1]$ we obtain,

$$\begin{aligned}
\alpha s_{e,f}s_{\bar{e},\bar{f}} &= \alpha m_{g,e}m_{g,\bar{e}}n_{\bar{g},f}n_{\bar{g},\bar{f}} + (\alpha m_{g,e}m_{\bar{g},\bar{e}}n_{\bar{g},f}n_{g,\bar{f}} \\
&\quad + \alpha m_{\bar{g},e}m_{g,\bar{e}}n_{g,f}n_{\bar{g},\bar{f}}) + \alpha m_{\bar{g},e}m_{\bar{g},\bar{e}}n_{g,f}n_{g,\bar{f}} \\
&\leq \alpha m_{g,e}m_{g,\bar{e}}n_{\bar{g},f}n_{\bar{g},\bar{f}} + (m_{g,\bar{e}}m_{\bar{g},e}n_{\bar{g},f}n_{g,\bar{f}} \\
&\quad + \alpha^2 m_{g,e}m_{\bar{g},\bar{e}}n_{g,f}n_{\bar{g},\bar{f}}) + \alpha m_{\bar{g},e}m_{\bar{g},\bar{e}}n_{g,f}n_{g,\bar{f}} \\
&\leq m_{g,e}m_{g,\bar{e}}n_{\bar{g},f}n_{\bar{g},\bar{f}} + m_{g,\bar{e}}m_{\bar{g},e}n_{\bar{g},f}n_{g,\bar{f}} \\
&\quad + m_{g,e}m_{\bar{g},\bar{e}}n_{g,f}n_{\bar{g},\bar{f}} + m_{\bar{g},e}m_{\bar{g},\bar{e}}n_{g,f}n_{g,\bar{f}} \\
&= s_{\bar{e},f}s_{e,\bar{f}}.
\end{aligned}$$

□

Theorem 4.4.5. *Let M and N be α-balanced matroids, with $\alpha \in [0,1]$, then the following statements are true.*

1. *M^* is α-balanced.*
2. *$M \oplus N$ is α-balanced.*
3. *$M \oplus_2 N$ is α-balanced.*

Proof. The first statement follows from Theorem 4.4.2. The second statement follows from Lemma 4.4.4 and the fact that any minor of $M \oplus N$ is the direct sum of a minor of M and a minor or N, see [**?**]. The third also follows from Lemma 4.4.4 and the fact that any minor of $M \oplus_2 N$ is the direct sum of a minor of M and a minor or N, see [**?**]. □

Proof of Theorem 4.2.11. Note that taking $\alpha = 1$ in Corollary 4.4.5, we have a proof of the second and third statement of Theorem 4.2.11. □

4.5 A New Conjecture

While working on the conjecture of Mihail–Vazarani, we realised how difficult the expression 4.1.1 is to relate to any operation on matroids, such as direct sum or two sum, so we came up with the idea of taking a slightly stronger property that was more malleable. We, therefore, propose the following property.

Definition 4.5.1. *For $k \in \mathbb{N}$, a matroid M is said to be a k–*expander if the bases–exchange graph $G(M)$ has strong cut set expansion at least k, that is, for all subsets A of bases we have

$$|C(A)| \geq \frac{k|A||\mathcal{B} - A|}{|\mathcal{B}|}, \tag{4.5.1}$$

where as in definition 4.1.1, $C(A)$ is the cut determined by A.

To check the property given by the equation 4.5.1 we need the matroid to have at least 2 bases. In what follow we assume that matroids with just 1 or 0 bases are k expanders to avoid specifying in every statement if M has or has not at least 2 bases.

We have already pointed out that all balanced matroids have strong cut set expansion at least 2, so all balanced matroids are 2-expander matroids. Even more, the class of 2-expander matroids contains properly the class of balanced matroids as the following proposition shows.

Theorem 4.5.2. *The matroid S_8 is near–balanced and a 2-expander matroid.*

Proof. Consider the matroid S_8 as in example 1, take the element a, then it is displayed in Table 4.4.1 that $\mathcal{B}_{x\bar{a}}\mathcal{B}_{\bar{x}a} > \mathcal{B}_{xa}\mathcal{B}_{\bar{x}\bar{a}}$ for all $x \neq a$. Also, every minor of S_8 is regular, so, by definition 4.3.1, S_8 is near–balanced. Using Theorem 4.3.10 we conclude that S_8 is a 2–expander matroid. □

Suppose now that there exists a matroid M and a subset of its bases $A \subset \mathcal{B}(M)$, with $|A| \leq |\mathcal{B}(M)|(1 - \frac{1}{\sqrt{2}})$ and $C(A) = |A|$, then the matroid $M \oplus M$ would not satisfy the Matroid–Expansion Conjecture as the bases $X = \{B \in \mathcal{B}(M \oplus M) | B' \subset B \text{ for some } B' \in A\}$ would be a set of bases of $M \oplus M$ with size less than $\frac{|\mathcal{B}(M\oplus M)|}{2}$ and $|C(X)| < |X|$. So, either the bound in the Matroid–Expansion Conjecture is not tight and we can strengthen it or the Conjecture is not true. We hope for the first possibility and as we already have a stronger property than cut set expansion 1, we propose the following stronger conjecture.

Conjecture 4.5.3. *All matroids are 2–expander matroids.*

A nice property of general k–expander matroids, for a fixed k, is that they are closed under the operation of taking direct sums. To prove this result we need a technical lemma.

For an $m \times n$ matrix R with entries in $\{0, 1\}$ we denote by x_i the number of non–zero entries in the row i and by y_j the corresponding number in column j. Also, we define the function $f(x, p) = \frac{x(p-x)}{p}$, where $0 \leq x \leq p$. Obviously, we have $\sum_{i=1}^{m} x_i = \sum_{j=1}^{n} y_j$ but also we have

Lemma 4.5.4.

$$\sum_{i=1}^{m} f(x_i, n) + \sum_{j=1}^{n} f(y_j, m) \geq f(\sum_{i=1}^{m} x_i, nm). \tag{4.5.2}$$

In order to prove Lemma 4.5.4 we use the following result.

Lemma 4.5.5. *Inequality 4.5.2 is equivalent to the following inequality about the row and column sums of the matrix R.*

$$\sum_{i=1}^{m} x_i + \frac{1}{nm}\left(\sum_{i=1}^{m} x_i\right)^2 \geq \frac{1}{n}\sum_{i=1}^{m} x_i^2 + \frac{1}{m}\sum_{j=1}^{n} y_j^2. \tag{4.5.3}$$

Proof. We have the following series of equivalent inequalities:

$$\begin{aligned}
\sum_{i=1}^{m} f(x_i, n) + \sum_{j=1}^{n} f(y_j, m) &\geq f(\sum_{i=1}^{m} x_i, nm) \\
\sum_{i=1}^{m} \frac{x_i(n - x_i)}{n} + \sum_{j=1}^{n} \frac{y_j(m - y_j)}{m} &\geq \frac{(\sum_{i=1}^{m} x_i)(nm - \sum_{i=1}^{m} x_i)}{nm} \\
\sum_{i=1}^{m} x_i - \sum_{i=1}^{m} \frac{x_i^2}{n} + \sum_{j=1}^{n} y_j - \sum_{j=1}^{n} \frac{y_j^2}{m} &\geq \sum_{i=1}^{m} x_i - \frac{(\sum_{i=1}^{m} x_i)^2}{nm} \\
\sum_{i=1}^{m} x_i + \frac{(\sum_{i=1}^{m} x_i)^2}{nm} &\geq \frac{1}{n}\sum_{i=1}^{m} x_i^2 + \frac{1}{m}\sum_{j=1}^{n} y_j^2.
\end{aligned}$$

□

To prove inequality 4.5.3 we construct a triple of directed graphs in which the number of edges correspond to the above numbers and then we use a counting argument.

Suppose R is an $m \times n$ matrix with entries in $\{0, 1\}$. Define a graph G as follows: the vertices are the mn entries of the matrix, that is $\{R_{i,j}\}$, and if $R_{i,j} = 1$ then there is an arrow to $R_{k,l}$ if $R_{k,l} = 0$ and there are two arrows if $R_{k,l} = 1$. The total number of arrows is $\sum_{i=1}^{m} x_i(nm + \sum_{i=1}^{m} x_i)$.

Now, we construct graphs G_1 and G_2. The vertex set of G_1 is also the set of entries of R and if $R_{i,j} = 1$ then there is an arrow $R_{i,j} \to R_{k,l}$, for all $k \in \{1, \ldots, m\}$ if $R_{i,l} = 1$. In a particular row, say i, every entry $R_{i,j} = 1$ is going to have mx_i edges going out, so the total number of arrows going out from the ith row is mx_i^2. Therefore, the total number of arrows in G_1 is $m\sum_{i=1}^{m} x_i^2$. We repeat this construction, now by columns, to get G_2 with a total number of arrows of $n(\sum_{i=1}^{m} y_i^2)$.

With these two graphs we construct $G' = G_1 + G_2$. We are going to prove that G has more arrows than G'.

First we compare the number of arrows between vertices of the same row, say i, and consider the entry $R_{i,j}$. If $R_{i,j} = 0$, it does not have arrows going out in either graph. If $R_{i,j} = 1$, it has two loops in G and two loops in G'. If there is an entry $R_{i,l} = 1$, with $l \neq j$, then there are two arrows from $R_{i,j}$ to $R_{i,l}$ in G and just one in G'. If there is an entry $R_{i,l} = 0$, with $l \neq j$, then there is an arrow from $R_{i,j}$ to $R_{i,l}$ in G and none in G'. The same analysis works by columns.

For the arrows between vertices that are not in the same row or column we proceed by pairs of vertices that form the corners of a rectangle, that is for the quadruple $R_{i,j}$, $R_{i,l}$, $R_{k,j}$ and $R_{k,l}$, for $i < k$ and $j < l$. In the Figure 4.4 we denote by a full dot if the

corresponding entry is one, and by a non-full dot if it is zero. In each case the number of arrows in a quadruple in G is at least the number of arrows in the corresponding quadruple in G'.

Figure 4.4: The different quadruples of vertices in G and G' together with their edges for the proof of Lemma 4.5.4.

We conclude that the number of arrows in G is at least the number of arrows in G', therefore

$$\sum_{i=1}^{m} x_i(nm + \sum_{i=1}^{m} x_i) \geq m \sum_{i=1}^{m} x_i^2 + n \sum_{j=1}^{n} y_j^2. \tag{4.5.4}$$

And this finishes the proof of Lemma 4.5.4 □

Theorem 4.5.6. *Let k be a positive integer, then if M and N are k–expander matroids, $M \oplus N$ is also a k–expander matroid.*

Proof. The bases–exchange graph of the matroid $M \oplus N$ is the product graph of the two graphs $G(M)$ and $G(N)$. So, each basis S in $M \oplus N$ is a pair of bases (B, C), where B is a basis of M and C is a basis of N. The edges of $G(M \oplus N)$ go between pairs with the same first entry and whose second entries are neighbours in N or with the same second entry and whose first entries are neighbours in M. Suppose that $|\mathcal{B}(M)| = m$ and $|\mathcal{B}(N)| = n$, then, every basis of $M \oplus N$ labels an entry of a $m \times n$ matrix. Now, let A be a subset of bases of $M \oplus N$, we want to bound the number of edges in the corresponding cut. Construct the matrix R of size $m \times n$ with an entry equal to 1 if that entry corresponds to a basis in A and 0 otherwise. The edges in the cut go between bases in the same row or the same column, and by hypothesis this number is, in each row i, $kf(x_i, n)$ and in each column j is $kf(y_j, m)$. Now, we can apply the previous lemma and obtain that the number of edges in the cut $C(A)$ is at least $kf(\sum_{i=1}^{m} x_i, nm)$. □

We are unable to prove Conjecture 4.5.3 and while trying to prove it by an induction argument, the following notion seems both natural and interesting.

Definition 4.5.7. *For $k \in \mathbb{N}$, a matroid M is said to k–expand weakly if for all $e \in E(M)$ and all $A \subset \mathcal{B}$ we have that*

$$|C(A) \cap E(G_e(M))| \geq \frac{kN_1N_2}{N_1 + N_2} \left(\frac{n_1}{N_1} - \frac{n_2}{N_2} \right)^2 \tag{4.5.5}$$

where $n_1 = |A \cap \mathcal{B}_e|$, $n_2 = |A \cap \mathcal{B}_{\bar{e}}|$, $N_1 = |\mathcal{B}_e|$ and $N_2 = |\mathcal{B}_{\bar{e}}|$.

Clearly, for $k > k'$, if a matroid M k–expands weakly, then it k'-expands weakly. The relation between being a k–expander matroid and to k-expand weakly is partially answered in Theorem 4.5.9 and Corollary 4.5.10. First, a technical lemma.

Lemma 4.5.8. *Let $0 \leq n_1 \leq N_1$ and $0 \leq n_2 \leq N_2$ then for the function $f(x, p) = \frac{x(p-x)}{p}$ we have*

$$f(n_1, N_1) + f(n_2, N_2) + \frac{kN_1N_2}{N_1 + N_2} \left(\frac{n_1}{N_1} - \frac{n_2}{N_2} \right)^2 = f(n_1 + n_2, N_1 + N_2).$$

Proof. Consider the quantity

$$f(n_1 + n_2, N_1 + N_2) - f(n_1, N_1) - f(n_2, N_2),$$

that is

$$\frac{k(n_1+n_2)(N_1+N_2-n_1-n_2)}{N_1+N_2}-\frac{kn_1(N_1-n_1)}{N_1}-\frac{kn_2(N_2-n_2)}{N_2}.$$

Expanding this expression we obtain the following long expression

$$\begin{aligned}&\frac{k(n_1N_1^2N_2+n_1N_1N_2^2-n_1^2N_1N_2-n_1n_2N_1N_2)}{(N_1+N_2)N_1N_2}\\&+\frac{k(n_2N_1^2N_2+n_2N_1N_2^2-n_2^2N_1N_2-n_1n_2N_1N_2)}{(N_1+N_2)N_1N_2}\\&-\frac{k(n_1N_1^2N_2+n_1N_1N_2^2-n_1^2N_1N_2-n_1^2N_2^2)}{(N_1+N_2)N_1N_2}\\&-\frac{k(n_2N_1^2N_2+n_2N_1N_2^2-n_2^2N_1^2-n_2^2N_1N_2)}{(N_1+N_2)N_1N_2}.\end{aligned}$$

After some simplification we get

$$\begin{aligned}&=\frac{k(n_1^2N_2^2-2n_1n_2N_1N_2+n_2^2N_1^2)}{(N_1+N_2)N_1N_2}=\frac{k(n_1N_2-n_2N_1)^2}{(N_1+N_2)N_1N_2}\\&=\frac{kN_1N_2}{N_1+N_2}\left(\frac{n_1}{N_1}-\frac{n_2}{N_2}\right)^2,\end{aligned}$$

and the result follows. □

Theorem 4.5.9. *Let $k\in\mathbb{N}$ and M be a matroid. Suppose there exists an element e such that $M\setminus e$ and M/e are k–expander matroids and also that M and e satisfy equation 4.5.5 for k and all subsets A of bases, then M is a k–expander matroid.*

Proof. Let A be a subset of bases of M. Then $C(A)$ can be partitioned into the edges that are in the subgraph of $G(M)$ that is isomorphic to $G(M\setminus e)$, the edges in the subgraph isomorphic to $G(M/e)$ and the edges in $C(A)\cap E(G_e(M))$. Let $|A\cap\mathcal{B}_e|=n_1$, $|A\cap\mathcal{B}_{\bar{e}}|=n_2$, $|\mathcal{B}_e|=N_1$ and $|\mathcal{B}_{\bar{e}}|=N_2$, then by hypothesis the first two sets of edges have total cardinality

$$\frac{kn_1(N_1-n_1)}{N_1}+\frac{kn_2(N_2-n_2)}{N_2}.$$

Also by hypothesis we have that

$$|C(A)\cap E(G_e(M))|\geq\frac{kN_1N_2}{N_1+N_2}\left(\frac{n_1}{N_1}-\frac{n_2}{N_2}\right)^2.$$

Adding up, we get that the cardinality of $C(A)$ is such that

$$\begin{aligned}|C(A)| &\geq k\left(\frac{n_1(N_1-n_1)}{N_1}+\frac{n_2(N_2-n_2)}{N_2}+\frac{N_1N_2}{N_1+N_2}\left(\frac{n_1}{N_1}-\frac{n_2}{N_2}\right)^2\right)\\ &= k\left(\frac{(n_1+n_2)(N_1+N_2-n_1-n_2)}{N_1+N_2}\right)\\ &= k\frac{|A||\mathcal{B}(M)\setminus A|}{|\mathcal{B}(M)|}.\end{aligned}$$

As this was for an arbitrary $A \subset \mathcal{B}(M)$, we conclude that M is a k-expander matroid. □

Corollary 4.5.10. *If every matroid 2–expands weakly then every matroid is a 2–expander matroid.*

Proof. By using Theorem 4.5.9 and induction on the size of the matroid. □

It is easy to check that for $0 \leq n_1 \leq N_1$ and $0 \leq n_2 \leq N_2$ then

$$\frac{N_1N_2}{N_1+N_2}\left(\frac{n_1}{N_1}-\frac{n_2}{N_2}\right)^2 \leq \left|n_1-\frac{N_1}{2}\right|+\left|n_2-\frac{N_2}{2}\right|. \tag{4.5.6}$$

Therefore, if M is balanced, by Theorem 4.2.4, it has the cross-cut property and by equation 4.5.6 it 1-expands weakly. Also, by doing several cases, it can be checked that S_8 1-expands weakly so it is reasonable to believe that all matroids 1–expand weakly. However, we think that the following stronger conjecture is true.

Conjecture 4.5.11. *All matroids 2–expand weakly.*

4.6 Conclusion and open problems

We have introduced several classes and properties on matroids For a matroid M, the notion of balanced fractional matching and enforcing of ratios are related by the following diagram.

M is balanced. $\Leftrightarrow$ For all minors N of M, N enforces ratios. $\Leftrightarrow$ For all minors N of M, $G_e(N)$ admits a fractional matching, for all $e \in E(N)$.

$\Downarrow$

For all minors N of M, N has the crossed-cut property on every element of $E(N)$.

$\Downarrow$

M has cut set expansion ≥ 1.

All these implications were discussed or proved in Section 4.2 and Section 4.3.

The classes of α-balanced matroids, for $0 \leq \alpha \leq 1$, are clearly related by

$$\text{Balanced matroids} \subset \alpha\text{-balanced matroids},$$
$$\alpha_1\text{-balanced matroids} \subset \alpha_2\text{-balanced matroids},$$

where $0 \leq \alpha \leq 1$ and $0 \leq \alpha_1 \leq \alpha_2 \leq 1$.

Finally, balanced, near-balanced and k-expander matroids are related by

$$\begin{array}{ccccccc} \text{Balanced} \\ \text{matroids} & \subset & \text{near-balanced} \\ & & \text{matroids} & \subseteq & \text{2-expander} \\ & & & & \text{matroids} & \supseteq & \text{3-expander} \\ & & & & & & \text{matroids} & \supseteq \cdots, \end{array}$$

And for a matroid M we also have

$$\begin{array}{l} \text{For all minors } N \text{ of } M, \\ N \text{ } k\text{-expands weakly.} \end{array} \Rightarrow \begin{array}{l} M \text{ is a } k\text{-expander} \\ \text{matroid.} \end{array}$$

The notion of α–balanced is a very natural generalisation of balance and is of a certain intrinsic interest but probably the most important question about it is if there is a relation, as we believe, between the conjecture of Mihail and Vazirani, and values of α close to 1. Ideally, one would like for a fixed α_0, that there exists a polynomial, $p(x)$, such that if M is an α_0-balanced matroid, then M has cut set expansion $1/p(|E(M)|)$. For example, in the case of balanced matroids, $\alpha_0 = 1$ and $p(x)$ is the constant polynomial 1.

Conjecture 4.5.3 seems to be more appropriate to work with such a structured notion as matroids, and Theorem 4.5.6 gives evidence for this thought. The next step would be to prove that this theorem remain true if we replace direct sum by 2–sum. The more general problem is to determine which class of matroids is the one that satisfies this conjecture, that we already know that contains balanced matroids and so all regular matroids.

Appendix A

Indices.c

Appendix B

Indices.c

There follows a printout of the `C` program which I wrote to generate the indices of the matrix T_n defined in page 32. The indices correspond to the standard forms $\mathcal{S}_n$, defined in page 29, for a fixed integer n which is the input of the program.

The program is based on the result mentioned during the proof of Theorem 1.8.18, that every standard form has the parenthesis property. In fact, and we omit the proof of this, all standard forms in $\mathcal{S}_n$ can be constructed by concatenate every standard form of $\mathcal{S}_{n-1}$ with a label in $\{1, \ldots, n\}$, and consider all of those which satisfy the parenthesis property. Now, this construction can be done easily by using a stack and the program consists on a straightforward implementation of the stack plus a function that generates all the labels that can be concatenated to an already constructed standard form checking that the new sequence satisfies the parenthesis property.

B.1 `indices.c`

```
#include<stdio.h>
#include<stdlib.h>

#define MAXLEVEL 20  /* Maximum possible size of the standard forms. */
#define NO 0
```

Stack structure that represents the indices of the transfer-matrix.

```
struct node{
        int val;
        struct node *next; /* List of possible sons. */
}  stack[MAXLEVEL+1];         /* Stack of components. */
```

```
int top;                        /* The top of the stack. */
```

Stack functions.

```
void init_stack(void);
```

List functions.

```
void delete(int place);
void insert(int val, int place);
```

Function that creates the possible labels.

```
void create_sons(int top, int level);
```

File functions.

```
int on_screen( void);
```

I-O functions.

```
int read_level(void);

main()
{
   int i,j;                     /* Counters. */
   long int row;                /* Number of indices. */
   FILE *index;                 /* File to keep the standard forms. */
   int screen;                  /* Option to display. */
   int level;                   /* The size of the standard forms. */
   char file[20];               /* The name of the file to store. */
                                /* the standard forms. */

   /* Read option to write the results in a file or the screen. */
   screen = on_screen();

   if ( screen == NO ){
     printf("Give the file to store the indices\n");
     scanf("%s", file);
     printf("The data will be stored in file %s",file);
```

```
  if ( (index=fopen(file, "wb"))== NULL ){
    printf("I cannot open the file to store data, sorry\n");
    exit(1);
  }

  printf("So the data will be stored in file %s\n", file);
} /* End of if */

level= read_level();

init_stack();              /* Initialisation of stack. */

top=1;  row=1;             /* Initialisation. */

if ( screen ) printf("The indices are:\n");

else{

/* The first integer is the size of the standard forms. */
  fwrite(&level, sizeof(int), 1, index);

/* The second is the total of the indices. */
  fwrite(&row, sizeof(long int), 1, index);

}

stack[top].val=1;  /* The first value is always 1. */

create_sons(top, level);

while (top > 0 && top < level){

  if ( top == level-1 ) {
    /* We found some standard form of desired size. */

    while( stack[top].next!= NULL){

      /* We print all the standard forms that start with */
      /* stack[1].val,..., stack[level-1].val */
      /* and the last value is an element of the queue */
      /* stack[level-1]. */
```

```
      stack[top+1].val=stack[top].next->val;

      if ( screen ) printf("%d\t", row);
      for(j=1; j< level+1; j++){

        if ( screen )
          printf(" %d ", stack[j].val);
        else
          fwrite(&(stack[j].val), sizeof(char), 1, index);

      }

      if ( screen ) printf("\n");

      row++;

      delete(top);

    } /* End of nested while. */

    top--; /* Pop queue. */

} /* End of if. */

else{

    /* We take another element from the queue at the top. */
    if ( stack[top].next==NULL )
      /* We finish the queue at the top */
      /* and go down to a previous queue. */
        top--; /* Pop */

    else{

        /* We take the next element in the queue and */
        /* construct all possible extensions. */
        stack[top+1].val=stack[top].next->val;
        delete(top);
        top++; /* Push queue. */
        create_sons(top, level);
```

```
      } /* End of nested else. */

    } /* End of main else. */

  } /* End of main while. */

  /* Adjustment as we increase row during */
  /* the operation when we left the while. */

  row--;

  /* Present the results of the program. */

  if ( screen )
    printf("The total number of indices is %d\n", row);

  else{

    if ( fseek(index, sizeof(int),SEEK_SET) ==0){

      fwrite(&row, sizeof(long int), 1, index);
      printf("There has been stored %ld", row);
      printf(" rows of %d entries\n", level);
      printf("that is a total of  %ld bites\n", row*level+8);
      printf("in the file %s\n", file);

    }

    else{
      printf("There is an error when trying to store");
      printf(" the number of indices\n");
      exit(2);
    } /* End of nested else. */

    fclose(index);

  } /* End of main else. */

  exit(0);

} /* End of function main*/
```

Implementation of stack functions.

```
void init_stack(void)
{
   int i;

   for (i=0; i<  MAXLEVEL; i++){
     stack[i].val=0;
     stack[i].next=NULL;
   }

} /* End of function init_stack. */
```

Implementation of list procedures.

```
void delete(int place)
{

  if (stack[place].next == NULL)
     printf("There is an error in delete fuction\n");

  else
    stack[place].next = stack[place].next->next;

} /* End of function delete . */

void insert(int val, int place)
{

  struct node *new;

  new = (struct node *) malloc( sizeof(struct node));
  if (new == NULL)
    printf("There is a fatal error\n");

  else{

    new->val = val;
    new->next= stack[place].next;
    stack[place].next= new;
```

```
  }
}/* End of function insert. */
```

This function creates all the possible extension to a given standard form, that is, find all possible labels $\{1, \ldots,$ `level`$\}$ which when concatenate to the standard form of size `level`-1, already constructed in the data structure `stack`, will produce a standard form of size `level`. The algorithm works as follows: a label is forbidden if we encounter it previously surrounded by a pair of equal labels. So we just go through all the labels, finding all the forbidden ones and the rest of them are possible extensions.

```
void create_sons(int top, int level)
{
  int forbiden[MAXLEVEL+1];      /* List of forbidden sons. */
  int new_top,                   /* To simulate a new stack. */
       min,                      /* Minimum label seen. */
       max_total,                /* Maximum label in the stack. */
       last_seen,                /* Last label seen. */
       i;                        /* Counter. */

  new_top=top;

  max_total=0;

  min=stack[new_top].val;

  for (i=0; i < level+1; i++)
    forbiden[i]=0;

  while( new_top > 0) {

    last_seen = stack[new_top].val;
    new_top--;

    if (last_seen > max_total)
        max_total= last_seen;
    if (last_seen < min )
            min = last_seen;
    else if (last_seen > min)
         forbiden[last_seen]=1;
```

```
  } /* End of while. */

  if ( (max_total+1) < level )
    for( i=max_total+2; i< level+1; i++)
       forbiden[i]=1;

  for(i=1; i< level+1; i++)
    if (forbiden[i]!= 1)
       insert(i, top);

} /* End of function create_sons. */
```

Read option to store the results in a file or display on the screen.

```
int on_screen(void)
{

char c='\0';

  printf("Do you want the data to be stored in a file y/n\n");

  scanf("%c", &c);

  while ( (c!= 'y') && (c!= 'n') ) {
    printf("Type y or n");
    scanf("%c", &c);
  }

  if ( c == 'n' )
    return(1);    /* The result will be display in the screen. */
  else
    return(0);    /* The result will be stored in a file. */
} /* End of function in_screen. */
```

Read the side of the standard forms.

```
int read_level(void)
{

int lev;
```

```
  printf("Give the size of the standard forms,");
  printf(" the maximum is 20\n");

  while ( scanf("%d", &lev)!= 1)
    printf("Type an integer between 1 and 20\n");
  return(lev);

} /* End of function read_level. */
```

Appendix C

Matrix.c

Appendix D

Matrix.c

We now present a printout of the `C` program which I wrote to compute the matrix $T_n|_{\substack{x=x_0\\y=y_0}}$ of page 35, and which has as input the point (x_0, y_0).

Also in this appendix, we present a program to compute $T'_n|_{\substack{x=x_0\\y=y_0}}$ of page 47.

D.1 `matrix.c`

The program assumes that we have in a file the indices of the matrix T_n, that are constructed using the program of Appendix B. The program is a straightforward implementation of the theory of Section 1.6.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAXLEVEL 20     /* Maximum possible size of the indices. */

int dad[2*MAXLEVEL+1]; /* The tree for the union-find structure. */
```

Operation for the union-find data structure

```
void ini_dad(void);
int root(int x);
int find( int x, int y, int doit);
```

To generate a set of horizontal edges

```
void edges(int *a, unsigned x, int level);
```

To find a standard form in the table of standard forms

```
int look_place( int *conf, char *table, int level, long int total);

main()
{
  FILE *index;            /* File where the standard forms are stored. */
  FILE *result;           /* File to store the adjacency matrix. */
  char *file;             /* Name of the file with the indices. */
  int level;              /* Size of the indices. */
  int total;              /* Total number of standard forms. */
  int x_val, y_val;       /* Values to compute the Tutte polynomial. */

  int old_comp;           /* The number of components in the old and */
  int new_total_comp;     /* new standard form. */

  int top_hor;            /* Total number of possible horizontal and */
  int top_ver;            /* vertical edges in n-comb graph. */

  unsigned iter1;         /* Counters for the three levels of iteration */
  unsigned iter2;         /* for the construction of the matrix. */
  unsigned iter3;

  int size;               /* Size of the set of edges extending the */
                          /* square lattice. */
  int rank;               /* Increase on the rank that will */
                          /* produce the extension. */

  char buffer[MAXLEVEL]; /* Array to store a standard form. */
  char *table;            /* Set of standard forms. */
  int i,j;                /* General counters. */

  int  vertical[MAXLEVEL];    /* Set of vertical and horizontal edges */
  int  horizontal[MAXLEVEL]; /* of n-comb graph. */

  int  old[MAXLEVEL];     /* Labels of components in old and new */
  int  new[MAXLEVEL];     /* set of standard forms. */
  double *row;            /* Buffer for the values of a row in the matrix. */

  /********** MAIN PROGRAM **************/
```

```
/* Initialisation of buffer. */
for (i=0; i< MAXLEVEL; i++)
  buffer[i]=0;

/* Open the file where the standard forms are stored. */
printf("Give the file where the  standard forms are stored\n");
scanf("%s", file);

if ( (index=fopen(file, "rb"))== NULL ){
  printf("I cannot open the file %s to read the data, sorry\n", file);
  exit(1);
}

printf("So the data will be read from file %s\n", file);

if ( fread(&level, sizeof(int), 1, index) == 1  )
  printf("The size of the square lattice is %d\n", level);

else{
  printf("There is an error in the file\n");
  exit(1);
}

if ( fread(&total, sizeof(int), 1, index) == 1  )
  printf("And the number of standard forms is %d\n",total);

else{
  printf("There is an error in the file\n");
  exit(1);
}

/* Creating space to store the standard forms. */
if( (table = (char *) calloc( level*total, sizeof(char)))==NULL){
  printf("There is not enough memory to store the indices\n");
  exit(2);
}
```

```
/* Reading the standard forms. */
j = 0;
while ( fread(buffer, sizeof(char), level, index) == level  ){
  for( i=0; i<level; i++)
     table[level*j+i]= buffer[i];
  j++;
}

/* Close the file of standard forms. */
fclose(index);

/* Space for the row of the adjacency matrix. */
if( (row = (double *) calloc( total, sizeof(double)))==NULL){
  printf("There is not enough memory to compute ");
  printf(" the matrix\n sorry\n");
  exit(2);
}

/* Read point (x_val, y_val) to evaluate the Tutte Polynomial. */
printf("Give the value of x where you wish to compute T(x, y)\n");
scanf("%d", &x_val);
printf("Give the value of y where you wish to compute T(x, y)\n");
scanf("%d", &y_val);

/* Open the file to store the matrix. */
printf("Give the file to store the matrix\n");
printf("The matrix will occupy %d bites\n", total*total*8);
scanf("%s", file);

if ( (result=fopen(file, "wb"))== NULL ){
  printf("I cannot open a file to stored data, sorry\n");
  exit(1);
}

printf("So the matrix will be stored in file %s\n", file);

/* We store the dimension of the matrix in the file. */
fwrite(&total, sizeof(int), 1, result);
```

```
/* Computing the adjacency matrix at value x_val, y_val. */

/* Number of iterations for each row of the matrix. */
top_hor=(unsigned) ldexp(1, level);
top_ver= (unsigned) ldexp(1, level-1);

/* Iteration through all indices of the matrix. */
for( iter1=0; iter1< total; iter1++){

  /* Initialisation of the graph G_{β,B}, */
  /* see page 32. */
  for(i=0; i<level; i++)
    old[i]=table[iter1*level+i];
  for(i=0; i<total; i++)
    row[i]=0.0;

 /* We iterate over all possible extensions of horizontal edges. */
  for(iter2=0; iter2 < top_hor ; iter2++){
     for(i=0; i< level;i++) horizontal[i]=0;
     edges(horizontal,iter2,level); /* Create horizontal edges. */

     /* We iterate over all possible extensions of vertical edges. */
     for(iter3=0; iter3 < top_ver; iter3++){
        for(i=0; i< level-1; i++) vertical[i]=0;
        edges(vertical,iter3,level-1); /* Create vertical edges. */

        size = 0;
        ini_dad();

        /* At the beginning every vertex is in a different component. */
        for(i=0; i<level; i++)
          new[i]= i+1;

        /* First, we construct the connected components */
        /* for the old standard form. */
        for ( i=0; i+1 < level; i++ )
          for(j=i+1; j < level; j++)
            if ( old[i]==old[j])
              find(i+level+1, j+level+1, 1);
```

```
/* We count the number of components of the previous */
/* standard form. */
old_comp = 0;
for ( i=level+1; i < 2*level+1; i++)
  if ( dad[i] == 0 ) /* It is a root and a new component. */
    old_comp++;

/* We change the connected components due to */
/* the addition of the vertical edges. */
for ( i=0; i <  level-1; i++)
  if ( vertical[i]==1 ){ /* New vertical edge. */
    find(i+1, i+2, 1);
    size++;
  }

/* And we change the connected components due to */
/* the addition of the horizontal edges. */
for (i=0; i < level; i++ )
  if ( horizontal[i]==1 ){ /* New horizontal edge. */
    find(i+level+1, i+1, 1);
    size++;
  }

/* Change the label of the roots in the new standard form. */
for (i=1, j=1; i < level+1; i++ )
  if ( dad[i]==0 ){ /* Its a root. */
    new[i-1]=j;
    j++;
  }

/* Using the components we relabel the new standard form. */
for (i=1; i < level+1; i++ ){
  j= root(i);
  new[i-1]=new[j-1];
}
```

```
/* Compute the new total of components after adding */
/* horizontal and vertical edges. */
new_total_comp= 0;
for ( i=1; i < 2*level+1; i++)
  if ( dad[i] == 0 ) /* it is a root and a new component. */
    new_total_comp++;

/* We obtain the increase or decrease in the rank. */
rank= new_total_comp-old_comp-level;


/* Find the index in the adjacency matrix of the new standard form. */
i = look_place( new, table, level, total);

/* We compute the value x^{-rank} y^{size-rank} */
/* at the point x =x_val and y =y_val. */

if ( rank+size == 0 ){
  if (rank==0)
    row[i]=row[i]+1;
  else
    row[i]=row[i]+pow(x_val,rank);
} /* End of main if. */

else{
  if(rank==0)
    row[i]=row[i]+pow(y_val, rank+size);
  else
    row[i]=row[i]+pow(x_val,rank)*pow(y_val, rank+size);
} /* End of main else */

  } /*End of iteration over extensions of vertical edges. */
} /*End of iteration over extensions of horizontal edges. */

/* Store the result for this row of the adjacency matrix. */
 fwrite(row, sizeof(double), total, result);

}/* End of iteration through all indices of the matrix. */
```

```
  /* Close the file of the matrix. */
  fclose(result);
  printf("We succesfully store the matrix\n");
}/* End on main */
```

Function implementation

Implementation of the Union-Find data structure

```
void ini_dad(void)
```

Initialisation of the Union-Find structure.

```
{
int i;

 for (i=1; i < 2*MAXLEVEL+1; i++ )
     dad[i]=0;
}/*End of function ini_dad. */

int root(int x)
```

Find the connected component of vertex `x`.

```
{
 int i=x;

 while ( dad[i] > 0 ) i=dad[i];
 return(i);
}/*End of function root. */

int find( int x, int y, int doit)
```

The function returns 0 if the vertices `x` and `y` are in the same connected component; and 1 if they are in different connected components. Additionally, if the option `doit` is 1, it makes the both vertices to be in the same connected component.

```
{
 int i=x, j=y;

 while ( dad[i] > 0 ) i = dad[i];
 while ( dad[j] > 0 ) j = dad[j];
```

```
 if ( (doit != 0) && (i != j) ){
   if ( j > i ) dad[j] = i;
   else dad[i] = j;
 }

 return( i != j );
}/*End of function find. */
```

Figure D.1: Example for the function `edges`

```
void edges(int *a, unsigned x, int level)
```

To generate a set of horizontal (vertical) edges, we identify an edge with a bit, so the presence of edge means bit on. For example, the graph in figure D.1 has number $1 + 0 \times 2^1 + 1 \times 2^2 = 5$.

```
{
    int i;

    for( i=0; (i< level) && ( x!= 0); x>>=1){
      if( x & 01 )
        a[i]=1;
      else
        a[i]=0;
      i++;
    }
}/*End of function edges. */
```

```
int look_place( int *conf, char *table, int level, long int total)
```

Find the place of the standard form in the table of indices by using binary search.

```
{

int l;
int r;
int place;
int i;

  l=0; r=total-1;

  while ( r >= l ) {

    place= (l+r)/2;
    i=0;

    while( (i< level) && ( conf[i]==table[place*level+i] ) )
      i++;
    if ( i >= level ) return(place);
    if ( conf[i] <= table[place*level+i] )
      l = place+1; /* It is in the lower half. */
    else
      r = place-1; /* It is in the upper half. */

  } /* End of main while. */

  return(-1);
} /*End of function look_place. */
```

D.2 matrix2.c

The program `matrix2.c` is almost the same as `matrix.c`, the only changes are in the following code

⋮

```
/* Number of iterations for each row of the matrix. */
top_hor=(unsigned) ldexp(1, level);
top_ver= (unsigned) ldexp(1, level-1);

/* Iteration through all indices of the matrix. */
for( iter1=0; iter1< total; iter1++){

  /* Initialisation of the graph G_{β,B}, */
  /* see page 32. */
  for(i=0; i<level; i++)
    old[i]=table[iter1*level+i];
  for(i=0; i<total; i++)
    row[i]=0.0;

 /* We iterate over all possible extensions of horizontal edges. */
  for(iter2=0; iter2 < top_hor ; iter2++){
     for(i=0; i< level;i++) horizontal[i]=0;
     edges(horizontal,iter2,level); /* Create horizontal edges. */
```

$$\vdots$$

So, we change the previous code by the following.

$$\vdots$$

```
/* Number of iterations for each row of the matrix */
/* Here is a big chance with respect to matrix.c. */
/* Now, we always have the edge (1,0) ~ (2,0) presented.*/
top_hor=(unsigned) ldexp(1, level-1);

/* The next line is unchanged. */
top_ver= (unsigned) ldexp(1, level-1);

/* Iteration through all indices of the matrix. */
for( iter1=0; iter1< total; iter1++){

  /* Initialisation of the graph G_{β,B}, */
  /* see page 32. */
  for(i=0; i<level; i++)
    old[i]=table[iter1*level+i];
  for(i=0; i<total; i++)
    row[i]=0.0;
```

```
/* We iterate over all possible extensions of horizontal edges. */
 for(iter2=0; iter2 < top_hor ; iter2++){
    for(i=0; i< level;i++) horizontal[i]=0;
    /* Here, there is another change. */
    /* The use of the number iter2*2+1 in the function */
    /* edges, ensures that the edge (1,0) ~ (2,0) is */
    /* always present. */
    edges(horizontal,iter2*2+1,level); /* Create horizontal edges. */
```

$\vdots$

All the rest of the program remains the same.

Appendix E

Vector.c

Appendix F

Vector.c

We present a printout of the `C` program which I wrote to compute the vector $\overrightarrow{X_n}^t|_{x=x_0}$ of 1.6.5 and the vector $\vec{1}$ of all entries 1, both of which are important for the computation of 1.6.1. The program has as input x_0 and uses the file generated by `indices.c` to get the indices of $\overrightarrow{X_n}$, and it is a straightforward computation of 1.6.5.

F.1 vector.c

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAXLEVEL 20   /* Maximum possible size of the indices. */
#define MAXSIZE   25  /* Maximum length of file name. */

int dad[MAXLEVEL+1]; /* The tree for the union-find structure. */
```

Operation for the union-find data structure.

```
void ini_dad(void);
int root(int x);
int find( int x, int y, int doit);
```

To generate a set of edges.

```
void edges(int *a, unsigned x, int level);
```

To find a standard form in the table of standard forms.

```
int look_place( int *conf, char *table, int level, long int total);
```

```
main()
{
FILE *index;            /* File where the standard forms are stored. */
FILE *result;           /* File to store the adjacency matrix. */
char file[MAXSIZE];     /* Name of the file with the indices. */
char file2[MAXSIZE];    /* Name of the file with the results. */

int level;              /* Size of the indices. */
int total;              /* Total number of standard forms. */
int x_val;              /* Value to compute the Tutte polynomial. */

int top_ver;            /* Vertical edges in n-comb graph. */

unsigned count;         /* Counters for the iteration. */

int size;               /* Size of the set of edges of the */
                        /* realization, see page 32.*/

char buffer[MAXLEVEL]; /* Array to store a standard form. */
char *table;            /* Set of standard forms. */
int i,j;                /* General counters. */

int  vertical[MAXLEVEL]; /* Set of edges. */

int  new[MAXLEVEL];    /* Labels of components set of */
                       /* standard forms. */

 double *vector;       /* The vector defined in equation 1.6.5 */
                       /* evaluated at x_val. */
 double *ones;         /* Vector with all entries equal to 1. */
```

********** MAIN PROGRAM **********

```
  /* Initialisation of buffer. */
  for (i=0; i< MAXLEVEL; i++)
    buffer[i]=0;

 /* Open the file where the standard forms are stored. */
  printf("Give the file where the  indices are stored\n");
  scanf("%s", file);
```

```
if ( (index=fopen(file, "rb"))== NULL ){
  printf("I cannot open the file %s to read the data, sorry\n",file);
  exit(1);
}

printf("So the data will be read from file %s\n", file);

if ( fread(&level, sizeof(int), 1, index) == 1  )
  printf("The size of the square lattice is %d\n", level);

else{
  printf("There is an error in the file\n");
  exit(1);
}

if ( fread(&total, sizeof(int), 1, index) == 1  )
  printf("And the number of indices is %d\n",total);

else{
  printf("There is an error in the file\n");
  exit(1);
}

/* Read the x_val to evaluate the vector in equation 1.6.5. */
printf("At which value of x are you going to evaluate\n");
scanf("%d",&x_val);

/* Open the file to store the vector. */
printf("Give the file to store the vector\n");
printf("The file will occupy %d bites\n", (total*2+1)*8);
scanf("%s", file2);

if ( (result=fopen(file2, "wb"))== NULL ){
  printf("I cannot open the file to store data, sorry\n");
  exit(1);
}

printf("So the vectors will be stored in the file %s\n", file2);
```

```
/* Creating space to store the standard forms. */
if( (table = (char *) calloc( level*total, sizeof(char)))==NULL){
  printf("There is not enough memory to store the indices\n");
  exit(2);
}

/* Reading the standard forms. */
j = 0;
while ( fread(buffer, sizeof(char), level, index) == level  ){
  for( i=0; i<level; i++)
     table[level*j+i]= buffer[i];
  j++;
}

/* Close the file of standard forms. */
fclose(index);

/* Space for vector and ones arrays. */
if( (vector = (double *) calloc( total, sizeof(double)))==NULL){
  printf("There is not enough memory to store the indices\n");
  exit(2);
}

if( (ones = (double *) calloc( total, sizeof(double)))==NULL){
  printf("There is not enough memory to store the one's vector\n");
  exit(2);
}

/* Initialisation of the vector array. */
for(i=0; i< total; i++)
  vector[i]=0;

/* Number of all possible realizations, */
/* see page 32. */
top_ver= (unsigned) ldexp(1, level-1);

/* Iteration through all possible realizations. */
for(count=0; count < top_ver; count++){
```

```
ini_dad();
size=0;

/* Construction of a realization. */
for(i=0; i< level-1; i++) vertical[i]=0;
edges(vertical,count,level-1);

/* Construction of the standard form */
/* corresponding to this realization. */
for(i=0; i<level; i++)
  new[i]= i+1;
for ( i=0; i <  level-1; i++)
  if ( vertical[i]==1 ){
    /* New edge. */
    size++;
    find(i+1, i+2, 1);
  }

/* Change the label of the roots. */
for (i=1, j=1; i < level+1; i++ )
  if ( dad[i]==0 ){ /* Its a root. */
    new[i-1]=j;
    j++;
  }
/* Using the components we relabel to get */
/* a standard form */
for (i=1; i < level+1; i++ ){
  j= root(i);
  new[i-1]=new[j-1];
}
/* Find the index in the vector of this standard form. */
i = look_place( new, table, level, total);
```

```
    /* Put the evaluation in the entry of the vector. */
    if( size == 0)
      vector[i]=1.0;
    else{
      if(x_val==0)
        vector[i]=0;
      else
        vector[i]=pow(x_val,size);
    }/* End of main if. */

  }/*End of iteration over realizations. */

  /* Create the vector with entries equal to 1. */
  for(i=0; i<total; i++)
    ones[i]=1.0;

  /* We store the dimension of the vectors */
  /* and write the vectors in the corresponding file. */
  fwrite(&total, sizeof(int), 1, result);
  printf("I wrote the size of the vectors\n");

  fwrite(vector, sizeof(double), total, result);
  printf("I wrote the initial vector\n");

  fwrite(ones, sizeof(double), total, result);
  printf("I wrote the 1's vector\n");
  fclose(result);
} /* End on main. */
```

For the implementation of the functions and procedures, see Section D.1 form page 173 to 176.

Appendix G

Matrix2.c

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAXLEVEL 20     /* Maximum possible size of the square lattice. */

int dad[2*MAXLEVEL+1]; /* The tree for the union-find structure. */
```

Operation for the union-find data structure.

```
void ini_dad(void);
int root(int x);
int find( int x, int y, int doit);
```

To generate a set of horizontal edges.

```
void edges(int *a, unsigned x, int level);
```

To find a standard form in the table of standard forms.

```
int look_place( int *conf, char *table, int level, long int total);

main()
{
  FILE *index;            /* File where the standard forms are stored. */
  FILE *result;           /* File to store the adjacency matrix. */
  char *file;             /* Name of the file with the indices. */
  int level;              /* Size of the square matrix. */
  int total;              /* Total number of standard forms */
  int x_val, y_val;       /* Values to compute the Tutte polynomial. */
```

```
int old_comp;           /* The number of components in the old and */
int new_total_comp;     /* new standard form. */

int top_hor;            /* Total number of possible horizontal and */
int top_ver;            /* vertical edges in n-comb graph. */

unsigned iter1;         /* Counters for the three levels of iteration */
unsigned iter2;         /* for the construction of the matrix. */
unsigned iter3;

int size;               /* Size of the set of edges extending the */
                        /* square lattice. */
int rank;               /* Increase on the rank that will */
                        /* produce the extension. */

char buffer[MAXLEVEL]; /* Array to store a standard form. */
char *table;            /* Set of standard forms. */
int i,j;                /* General counters. */

int  vertical[MAXLEVEL];   /* Set of vertical and horizontal edges */
int  horizontal[MAXLEVEL]; /* of n-comb graph. */

int  old[MAXLEVEL];     /* Labels of components in old and new */
int  new[MAXLEVEL];     /* set of standard forms. */
double *row;            /* Buffer for the values of a row in the matrix. */

/********** MAIN PROGRAM **************/

/* Initialisation of buffer. */
for (i=0; i< MAXLEVEL; i++)
  buffer[i]=0;

/* Open the file where the standard forms are stored. */
printf("Give the file where the  standard forms are stored\n");
scanf("%s", file);

if ( (index=fopen(file, "rb"))== NULL ){
  printf("I cannot open the file %s to read the data, sorry\n", file);
  exit(1);
}
```

```
printf("So the data will be read from file %s\n", file);

if ( fread(&level, sizeof(int), 1, index) == 1  )
  printf("The size of the square lattice is %d\n", level);

else{
  printf("There is an error in the file\n");
  exit(1);
}

if ( fread(&total, sizeof(int), 1, index) == 1  )
  printf("And the number of standard forms is %d\n",total);

else{
  printf("There is an error in the file\n");
  exit(1);
}

/* Creating space to store the standard forms. */
if( (table = (char *) calloc( level*total, sizeof(char)))==NULL){
  printf("There is not enough memory to store the indices\n");
  exit(2);
}

/* Reading the standard forms. */
j = 0;
while ( fread(buffer, sizeof(char), level, index) == level  ){
  for( i=0; i<level; i++)
     table[level*j+i]= buffer[i];
  j++;
}

/* Close the file of standard forms. */
fclose(index);

/* Space for the row of the adjacency matrix. */
if( (row = (double *) calloc( total, sizeof(double)))==NULL){
  printf("There is not enough memory to compute ");
  printf(" the matrix\n sorry\n");
  exit(2);
}
```

```
/* Read point (x_val, y_val) to evaluate the Tutte Polynomial. */
printf("Give the value of x where you wish to compute T(x, y)\n");
scanf("%d", &x_val);
printf("Give the value of y where you wish to compute T(x, y)\n");
scanf("%d", &y_val);

/* Open the file to store the matrix. */
printf("Give the file to store the matrix\n");
printf("The matrix will occupy %d bites\n", total*total*8);
scanf("%s", file);

if ( (result=fopen(file, "wb"))== NULL ){
  printf("I cannot open a file to stored data, sorry\n");
  exit(1);
}

printf("So the matrix will be stored in file %s\n", file);

/* We store the dimension of the matrix in the file. */
fwrite(&total, sizeof(int), 1, result);

/* Computing the adjacency matrix at value x_val, y_val. */

/* Number of iterations for each row of the matrix */
/* Here is a big chance with respect to matrix.c. */
/* Now, we always have the edge (1,0) ~ (2,0) presented.*/
top_hor=(unsigned) ldexp(1, level-1);

/* The next line is unchanged. */
top_ver= (unsigned) ldexp(1, level-1);

/* Iteration through all indices of the matrix. */
for( iter1=0; iter1< total; iter1++){

  /* Initialisation of the graph G_{β,B}, */
  /* see page 32. */
  for(i=0; i<level; i++)
    old[i]=table[iter1*level+i];
  for(i=0; i<total; i++)
    row[i]=0.0;
```

```
/* We iterate over all possible extensions of horizontal edges. */
 for(iter2=0; iter2 < top_hor ; iter2++){
    for(i=0; i< level;i++) horizontal[i]=0;
    /* Here, there is another change */
    /* The use of the number iter22 + 1 in the function */
    /* edges, ensure that the edge (1, 0) ∼ (2, 0) is */
    /* always present. */
    edges(horizontal,iter2*2+1,level); /* Create horizontal edges. */

    /*
```