Journal of Combinatorial Theory, Series A 120 (2013) 164–182

Contents lists available at SciVerse ScienceDirect

Journal of Combinatorial Theory, Series A

www.elsevier.com/locate/jcta

Chip-firing games, potential theory on graphs, and spanning trees ☆

Matthew Baker, Farbod Shokrieh

Georgia Institute of Technology, Atlanta, GA 30332-0160, USA

ARTICLE INFO

Article history:
Received 27 July 2011
Available online 31 July 2012

Keywords:
Chip-firing
Potential theory
Energy pairing
Reduced divisors
Matrix-tree theorem
Random spanning trees

ABSTRACT

We study the interplay between chip-firing games and potential theory on graphs, characterizing reduced divisors (G-parking functions) on graphs as the solution to an energy (or potential) minimization problem and providing an algorithm to efficiently compute reduced divisors. Applications include an "efficient bijective" proof of Kirchhoff's matrix-tree theorem and a new algorithm for finding random spanning trees. The running times of our algorithms are analyzed using potential theory, and we show that the bounds thus obtained generalize and improve upon several previous results in the literature.

1. Introduction

Chip-firing games on graphs arise in several different fields of research: in theoretical physics they relate to the "Abelian sandpile" or "Abelian avalanche" models in the context of self-organized critical phenomena [4,25,26]; in arithmetic geometry, they appear implicitly in the study of component groups of Néron models of Jacobians of algebraic curves [42,36,6]; and in algebraic graph theory they relate to the study of flows and cuts in graphs [3,10,11]. We recommend the recent survey article [34] for a short but more detailed overview of the subject.

There is a close connection between chip-firing games and potential theory on graphs. In this paper, we explore some new aspects of this interplay. Conceptually, this connection should not come as a surprise; in both settings the Laplacian operator plays a crucial rule. However, in chip-firing games an extra "integrality condition" is imposed; in the language of optimization theory, chip-firing

☆ We would like to thank Omid Amini, Jan Draisma, Xander Faber, Greg Lawler, Ye Luo, Sam Payne, and the anonymous referees for their comments on an earlier version of this manuscript. The authors' work was supported in part by NSF grants DMS-0901487 and CCF-0902717.

E-mail addresses: mbaker@math.gatech.edu (M. Baker), shokrieh@math.gatech.edu (F. Shokrieh).

0097-3165/$ – see front matter
http://dx.doi.org/10.1016/j.jcta.2012.07.011

games lead to integer programing problems whose associated linear programming relaxations can be solved using potential theory on graphs. Our potential theory methods allow us to prove some new results about chip-firing games and to give new proofs and/or generalizations of some known results in the subject. We also show that certain "ad hoc" techniques in the literature are naturally explained or unified by our approach.

Our main potential-theoretic tool is the *energy pairing* (see Section 3.3), which is a canonical positive definite bilinear form defined on the set of divisors of degree zero[1] on G. This pairing can be computed using any *generalized inverse* of the Laplacian matrix of G. The energy pairing can be used to define two functions $\mathcal{E}_q$ and b_q (for a fixed vertex q) on $\mathrm{Div}(G)$ which interact in a useful way with *chip-firing moves*; after firing a set of vertices A (not containing the exceptional vertex q), the value of $\mathcal{E}_q$ goes down by at least the size of the associated cut (see Proposition 4.1), and the value of b_q goes down by *exactly* the size of A (see Proposition 4.5).

Chip-firing moves induce a natural equivalence relation on $\mathrm{Div}(G)$ called *linear equivalence* of divisors. If we once again fix a vertex q, then a particularly nice set of representatives for linear equivalence classes is given by the *q-reduced divisors* (see Section 4.3). We show that the q-reduced divisor equivalent to a given divisor D can be characterized as the unique element of $|D|_q$ (the set of divisors $D' = \sum_v a'_v(v)$ linearly equivalent to D for which $a'_v \geqslant 0$ whenever $v \neq q$) minimizing the functional $\mathcal{E}_q$; see Theorem 4.12. A similar result holds with $\mathcal{E}_q$ replaced by b_q; see Theorem 4.14. Using this result we are able to give a new proof of the important fact that there is a unique q-reduced divisor in each linear equivalence class.

In order to check whether or not a given divisor is q-reduced, according to the definition (given in Section 4.3), one needs to check a certain inequality for *all subsets* of $V(G)\backslash\{q\}$. But there is in fact a much more efficient procedure called *Dhar's burning algorithm* (after Dhar [25]); see Section 5.1. Using a modification of Dhar's burning algorithm, it is possible to obtain an "activity preserving" bijection between q-reduced divisors and spanning trees of G; this was originally discovered (using different terminology) by Cori and Le Borgne [23]. In Section 5.2 we formulate the Cori–Le Borgne algorithm in the language of reduced divisors.

We then turn to the problem of *computing* the q-reduced divisor equivalent to a given divisor. Dhar's algorithm shows that one can efficiently *check* whether a given divisor is q-reduced; we show in Section 5.3 (specifically Algorithm 4) that one can efficiently *find* the q-reduced divisor equivalent to a given divisor as well. Algorithm 4 can be viewed as the *search* version of Dhar's *decision* algorithm. The main challenge here is the running-time analysis; we use potential theory (specifically, the function b_q) to give a bound on the running time of Algorithm 4. As we have already mentioned, the key point is that after firing a set $A \subseteq V(G)\backslash\{q\}$, the value of b_q goes down by exactly the size of A; this makes the function b_q a powerful tool for running-time analysis in chip-firing processes. The seemingly different techniques of Tardos [46], Björner, Lovász and Shor [14], Chung and Ellis [19], van den Heuvel [28], and Holroyd, Levine, Mészáros, Peres, Propp and Wilson [29] all give bounds which are specializations of the running-time bound which we derive using b_q; see Remark 5.9.

We next turn to applications of Algorithm 4. The first application (see Section 6.1) is an "efficient bijective" proof of Kirchhoff's celebrated matrix-tree theorem (stated in a more canonical way than usual in Theorem 6.2). The efficient bijective matrix-tree theorem provides a new approach to the *random spanning tree* problem, which we state in Section 6.2. This problem has been extensively studied in the literature and there are essentially two known types of algorithms for it: determinant based algorithms (e.g. [27,20,33]) and random walk based algorithms (e.g. [16,1,48,31] and [37, Chapter 4]). See Remark 6.3 for possible advantages of our new approach.

The paper is structured as follows. In Section 2 we fix our notation. In Section 3 we recall some basic facts from potential theory on graphs and define the energy pairing. The functionals $\mathcal{E}_q$ and b_q are introduced in Section 4 and the interplay between chip-firing dynamics and potential theory on graphs is studied. The algorithmic applications of this interplay, most notably Algorithm 4, are

[1] A *divisor* on a finite graph G is an element $\sum_{v\in V(G)} a_v(v)$ of the free Abelian group $\mathrm{Div}(G)$ on the set $V(G)$ of vertices of G. The *degree* of a divisor is the sum of the a_v over all v.

discussed in Section 5. Some applications of Algorithm 4, including an "efficient bijective" proof of Kirchhoff's matrix-tree theorem and a new algorithm for the random spanning tree problem, are discussed in Section 6.

2. Notation and terminology

Throughout this paper, a *graph* will mean a finite, connected, unweighted multigraph with no loop edges. The set of vertices of a graph G is denoted by $V(G)$ and the set of edges by $E(G)$. We let $n = |V(G)|$ and $m = |E(G)|$. For $A \subseteq V(G)$ and $v \in A$, we denote by $\mathrm{outdeg}_A(v)$ the number of edges between v and $V(G)\backslash A$.

Let $\mathrm{Div}(G)$ be the free Abelian group generated by $V(G)$. An element $\sum_{v\in V(G)} a_v(v) \in \mathrm{Div}(G)$ is called a *divisor* on G. The coefficient a_v of (v) in D is denoted by $D(v)$. For $D \in \mathrm{Div}(G)$, let $\deg(D) = \sum_{v\in V(G)} D(v)$ and let $\mathrm{Div}^0(G)$ be the subgroup of $\mathrm{Div}(G)$ consisting of divisors of degree zero. We denote by $\mathcal{M}(G) = \mathrm{Hom}(V(G), \mathbb{Z})$ the group of integer-valued functions on the vertices. For $A \subseteq V(G)$, $\chi_A \in \mathcal{M}(G)$ denotes the $\{0,1\}$-valued characteristic function of A; note that $\{\chi_{\{v\}}\}_{v\in V(G)}$ generates $\mathcal{M}(G)$.

The *Laplacian operator* $\Delta : \mathcal{M}(G) \to \mathrm{Div}(G)$ is defined by $\Delta(f) = \sum_{v\in V(G)} \Delta_v(f)(v)$, where

$$\Delta_v(f) = \sum_{\{v,w\}\in E(G)} \big(f(v) - f(w)\big).$$

This definition naturally extends to all rational or real-valued functions on vertices.

Let $\mathrm{Prin}(G)$ (the group of *principal divisors*) be the image of the Laplacian operator $\Delta : \mathcal{M}(G) \to \mathrm{Div}(G)$. It is easy to see that $\mathrm{Prin}(G) \subseteq \mathrm{Div}^0(G)$ and that both $\mathrm{Prin}(G)$ and $\mathrm{Div}^0(G)$ are free Abelian groups of rank $n-1$. As a consequence, the quotient group

$$\mathrm{Jac}(G) = \mathrm{Div}^0(G)/\mathrm{Prin}(G)$$

is finite. Following [3], $\mathrm{Jac}(G)$ is called the *Jacobian* of G.

Let $\{v_1, \ldots, v_n\}$ be a labeling of $V(G)$. With respect to this labeling, the *Laplacian matrix* Q associated to G is the $n \times n$ matrix $Q = (q_{ij})$, where q_{ii} is the degree of vertex v_i and $-q_{ij}$ $(i \neq j)$ is the number of edges connecting v_i and v_j. It is well known (and easy to verify) that Q is symmetric, has rank $n-1$, and that the kernel of Q is spanned by $\mathbf{1}$, the all-1's vector (see, e.g., [9,15]).

The labeling $\{v_1, \ldots, v_n\}$ of $V(G)$ induces isomorphisms between the Abelian groups $\mathrm{Div}(G)$, $\mathcal{M}(G)$, and the group of $n \times 1$ column vectors with integer coordinates. We use $[D]$ to denote the column vector corresponding to $D \in \mathrm{Div}(G)$ and $[f]$ to denote the column vector corresponding to $f \in \mathcal{M}(G)$. Under these isomorphisms, the Laplacian operator $\Delta : \mathcal{M}(G) \to \mathrm{Div}(G)$ corresponds to the matrix Q thought of as a homomorphism $Q : \mathbb{Z}^n \to \mathbb{Z}^n$, i.e., for $f \in \mathcal{M}(G)$ we have $[\Delta(f)] = Q[f]$.

3. Potential theory

3.1. Generalized inverses

A matrix has an inverse only if it is square and has full rank. But one can define a "partial inverse" for any matrix.

Definition. Let A be a matrix. A matrix L satisfying $ALA = A$ is called a *generalized inverse* of A.

Every matrix A has at least one generalized inverse. In fact more is true: every matrix has a unique *Moore–Penrose pseudoinverse*.[2]

[2] The Moore–Penrose pseudoinverse of A is a generalized inverse of A having the following additional properties: (i) $LAL = L$ and (ii) AL and LA are both symmetric. See [7] for additional details.

Let Q be the Laplacian matrix of a (connected) graph G. Since Q has rank $n-1$, it does not have an inverse in the usual sense. But there are several natural ways to obtain generalized inverses for Q. Here are some examples.

Construction 3.1. Fix an integer $1 \leqslant i \leqslant n$ and let Q_i be the invertible $(n-1)\times(n-1)$ matrix obtained from Q by deleting i-th row and i-th column from Q (Q_i is sometimes called the *reduced Laplacian* of G with respect to i). Let $L_{(i)}$ be the $n\times n$ matrix obtained from Q_i^{-1} by inserting a row of all zeros after the $(i-1)$-st row and inserting a column of all zeros after the $(i-1)$-st column. Then $L_{(i)}$ is a generalized inverse of Q. Indeed, one checks that $Q\,L_{(i)} = I + R_{(i)}$, where I is the $n\times n$ identity matrix and $R_{(i)}$ has all -1 entries in the i-th row and is zero elsewhere; as $R_{(i)}\,Q = 0$, we obtain $Q\,L_{(i)}\,Q = Q$.

Construction 3.2. Let J be the $n\times n$ all 1's matrix. Then $Q + \frac{1}{n}J$ is nonsingular and $Q^+ = (Q + \frac{1}{n}J)^{-1} - \frac{1}{n}J$ is a generalized inverse of Q. In fact, Q^+ is the Moore–Penrose pseudoinverse of Q, since one easily verifies that $Q\,Q^+ = Q^+Q = I - \frac{1}{n}J$ and $Q^+Q\,Q^+ = Q^+$.

One can use the matrices $L_{(i)}$ from Construction 3.1 to obtain other generalized inverses for Q:

Construction 3.3. Let $\mu = (\mu_1, \mu_2, \ldots, \mu_n)^T \in \mathbb{R}^n$ satisfy $\sum_{i=1}^n \mu_i = 1$. Then $L_\mu = \sum_{i=1}^n \mu_i L_{(i)}$ is a generalized inverse for Q. The matrix L_μ has the additional property that $L_\mu\mu = c_\mu \mathbf{1}$ for some $c_\mu \in \mathbb{R}$; this follows from the calculation

$$Q\,L_\mu\mu = \left(I + \sum_{i=1}^n \mu_i R_{(i)}\right)\mu = \mu - \mu = 0.$$

If J is the all-1's matrix as in Construction 3.2, then $G_\mu = L_\mu - c_\mu J$ is also a generalized inverse and has the additional property that $G_\mu\mu = 0$. The special case where $\mu_i = 1/n$ for all i gives the Moore–Penrose pseudoinverse Q^+ from the previous construction.

3.2. The j-function

We can think of a graph G as an electrical network in which each edge is a resistor having unit resistance.

Definition. For $p, q, v \in V(G)$, let $j_q(p, v)$ denote the electric potential at v if one unit of current enters a network at p and exits at q, with q grounded (i.e., zero potential).

From a more mathematical point of view, $j_q(p,\cdot)$ is the unique (rational-valued) solution to the Laplace equation $\Delta f = (p) - (q)$ satisfying $f(q) = 0$; alternatively, one can define $j_q(p, v)$ to be the (p, v)-entry of the matrix $L_{(q)}$ in Construction 3.1 (see, e.g., [18,5]; note also that $(pv \parallel q)$ in [10] is the same as our $j_q(p, v)$ up to scaling).

The following properties of the j-function are proved, for example, in [5]:

- $j_q(p, q) = 0$.
- $j_q(p, v) = j_q(v, p)$.
- $0 \leqslant j_q(p, v) \leqslant j_q(p, p)$.
- $r(p, q) = j_q(p, p) = j_p(q, q)$, where $r(p, q)$ denotes the effective resistance between p and q.

3.3. The energy pairing

Let L be any generalized inverse of the Laplacian matrix Q. Then the bilinear form $\langle\cdot,\cdot\rangle : \mathrm{Div}^0(G) \times \mathrm{Div}^0(G) \to \mathbb{Q}$ defined by

$$\langle D_1, D_2\rangle = [D_1]^T L[D_2] \tag{3.4}$$

is independent of the choice of L. (Indeed, since there are functions $f_i \in \mathrm{Hom}(V(G), \mathbb{Q})$ such that $[D_i] = Q[f_i]$ for $i = 1, 2$, we have $\langle D_1, D_2\rangle = [f_1]^T Q[f_2]$ and it is easy to check that the right-hand side does not depend on the choice of f_1, f_2.) We call the canonical bilinear form $\langle\cdot,\cdot\rangle$ the *energy pairing* on $\mathrm{Div}^0(G)$.

Lemma 3.5. *The energy pairing is positive definite.*

Proof. Let B be the incidence matrix of the graph. Then $Q = BB^T$, so if $[D] = Q[f]$ we have $\langle D, D\rangle = [f]^T BB^T[f] = \|B^T[f]\|_2^2$. □

Definition. The *energy* of a divisor $D \in \mathrm{Div}^0(G)$ is

$$\mathcal{E}(D) = \langle D, D\rangle = [D]^T L[D].$$

Since the energy pairing is positive definite, $\mathcal{E}(D) \geqslant 0$ with equality iff $D = 0$.

Remark 3.6. The name "energy pairing" comes from the fact that if $D \in \mathrm{Div}^0(G)$ represents an external current in the network, where $D(v)$ units of current enter the network at v if $D(v) > 0$ and $-D(v)$ units of current exit the network at v if $D(v) < 0$, then $\mathcal{E}(D)$ is precisely the total energy dissipated (per unit time) in the network.

We emphasize that the energy pairing is independent of the choice of L only because the divisors are assumed to have degree zero. One can extend the energy pairing to arbitrary divisors by fixing a vertex q and defining the *q-energy pairing* by

$$\langle D, E\rangle_q = \big\langle D - \deg(D)(q), E - \deg(E)(q)\big\rangle$$

for $D, E \in \mathrm{Div}(G)$.

3.4. The maximum principle

Lemma 3.7. *Let $f \in \mathcal{M}(G)$. Let $A_{\max}$ (resp. $A_{\min}$) be the set of vertices where f achieve its maximum (resp. minimum) value. Then*:

(a) *For $v \in A_{\max}$, $\Delta_v(f) \geqslant \mathrm{outdeg}_{\mathcal{A}_{\max}}(v)$.*
(b) *For $v \in A_{\min}$, $\Delta_v(f) \leqslant -\mathrm{outdeg}_{\mathcal{A}_{\min}}(v)$.*

Proof. For part (a) let $v \in \mathcal{A}_{\max}$. For an edge $e = vw$, if $w \in A_{\max}$ then $f(v) = f(w)$, and if $w \notin A_{\max}$ then $f(v) - f(w) \geqslant 1$. Since $\Delta_v(f) = \sum_{\{v,w\}\in E(G)}(f(v) - f(w))$, the result follows. Part (b) follows from part (a) by replacing f with $-f$. □

One obtains the following well-known corollary:

Corollary 3.8 *(Maximum principle). Suppose $f \in \mathcal{M}(G)$ is nonconstant. Then f achieves its maximum (resp. minimum) value at a vertex v for which $\Delta_v(f) > 0$ (resp. $\Delta_v(f) < 0$).*

4. Chip-firing dynamics and potential theory

4.1. Chip-firing dynamics on graphs

Following [6], we define an equivalence relation $\sim$ (called *linear equivalence*) on the group $\mathrm{Div}(G)$ as follows:

Definition. For $D_1, D_2 \in \mathrm{Div}(G)$, $D_1 \sim D_2$ if and only if $D_1 - D_2$ is in the image of $\Delta : \mathcal{M}(G) \to \mathrm{Div}(G)$.

This equivalence relation is closely related to notion of *chip-firing games* or *dollar games* (see, e.g., [14,10,11,6,25,4]). Given a divisor $D \in \mathrm{Div}(G)$, one can view the integer $D(v)$ as the number of *dollars* assigned to the vertex v. If $D(v) < 0$ then v is said to be *in debt*. A *chip-firing move* consists of choosing a vertex and having it either borrow one dollar from each of its neighbors or give ("fire") one dollar to each of its neighbors. For $D_1, D_2 \in \mathrm{Div}(G)$, $D_1 \sim D_2$ if and only if starting from the configuration D_1 one can reach the configuration D_2 through a sequence of chip-firing moves.

4.2. Chip-firing moves and the energy pairing

For $D \in \mathrm{Div}(G)$ we define $\mathcal{E}_q(D) = \langle D, D\rangle_q$. The following two propositions relate the energy pairing and chip-firing moves, and will be used in the next section.

Proposition 4.1.

(a) *If $E = D + \Delta(f) \in \mathrm{Div}(G)$ for some $f \in \mathcal{M}(G)$, then*

$$\mathcal{E}_q(E) = \mathcal{E}_q(D) + \sum_{v \in V(G)} (D+E)(v) \cdot f(v) - 2\deg(E) \cdot f(q).$$

(b) *If $E = D - \Delta(\chi_A) \in \mathrm{Div}(G)$ for some $A \subseteq V(G)\backslash\{q\}$, then*

$$\mathcal{E}_q(E) = \mathcal{E}_q(D) - \sum_{v \in A} (D+E)(v).$$

If, moreover, E is effective on A (i.e. $E(v) \geqslant 0$ for $v \in A$), then

$$\mathcal{E}_q(E) \leqslant \mathcal{E}_q(D) - \lambda(A) \tag{4.2}$$

where $\lambda(A)$ denotes the size of the $(A, G\backslash A)$-cut (i.e., the number of edges having one end in A and the other end in $V(G)\backslash A$).

Proof. (a) Let $\deg(E) = d$. Then $\deg(D) = d$ as well. We can find a $\mathbb{Q}$-valued function g so that $[E + D - 2d(q)] = Q[g]$. Then

$$\begin{aligned}
\mathcal{E}_q(E) &= \big\langle D + \Delta(f) - d(q), D + \Delta(f) - d(q)\big\rangle \\
&= \mathcal{E}_q(D) + \big\langle D + E - 2d(q), \Delta(f)\big\rangle = \mathcal{E}_q(D) + \big(Q[g]\big)^T L Q[f] \\
&= \mathcal{E}_q(D) + [g]^T Q L Q[f] = \mathcal{E}_q(D) + [g]^T Q[f] \\
&= \mathcal{E}_q(D) + \big(Q[g]\big)^T [f] = \mathcal{E}_q(D) + \big[D + E - 2d(q)\big]^T [f] \\
&= \mathcal{E}_q(D) + \sum_{v \in V(G)} (D+E)(v) \cdot f(v) - 2d \cdot f(q).
\end{aligned}$$

(b) For the first statement let $f = -\chi_A$ in part (a). $E(v) \geqslant 0$ for $v \in A$ means that $D(v) \geqslant \Delta_v(\chi_A) = \mathrm{outdeg}_A(v)$ for $v \in A$. So $\sum_{v\in A}(D+E)(v) \geqslant \sum_{v\in A} \mathrm{outdeg}_A(v) = \lambda(A)$. □

Remark 4.3. Proposition 4.1(a) can be used to give a new solution to the well-known *Pentagon Problem*[3]: "To each vertex of a regular pentagon an integer is assigned in such a way that the sum of all

[3] Problem 3, 27th IMO 1986. We refer the reader to [49,47,45,2,40] for some discussions, solutions, and generalizations of this problem.

five numbers is positive. If three consecutive vertices are assigned the numbers x, y, z respectively, and $y < 0$, then the following operation is allowed: the numbers x, y, z are replaced by $x + y$, $-y$, $z + y$, respectively. Such an operation is performed repeatedly as long as at least one of the five numbers is negative. Determine whether this procedure necessarily comes to an end after a finite number of steps".

To see that this process stops for any n-cycle, let D be the starting configuration and assume that $s = \deg(D) \geqslant 1$. It follows from Proposition 4.1(a) that the quantity

$$\mathbb{E}(D) = \sum_{q \in V(G)} \mathcal{E}_q(D) = \sum_{q,p,v \in V(G)} D(p) j_q(p,v) D(v)$$

goes down by exactly $-2s \cdot D(v) > 0$ after each basic move with $y = D(v) < 0$. Since the energy pairing is positive definite, we always have $\mathbb{E} \geqslant 0$, and thus the procedure will necessarily come to an end after a finite number of steps. Note that this method gives a way to compute the number of steps as well.

Definition. Let $\mathbf{1}$ denote the all-1's divisor. For $D \in \mathrm{Div}(G)$ and $q \in V(G)$, we define $b_q(D) = \langle \mathbf{1}, D \rangle_q$.

Remark 4.4. $b_q(D)$ is the "total potential" induced by the external current source corresponding to the divisor $D - \deg(D)(q) \in \mathrm{Div}^0(G)$.

Proposition 4.5.

(a) *If $E = D + \Delta(f) \in \mathrm{Div}(G)$ for some $f \in \mathcal{M}(G)$, then*

$$b_q(E) = b_q(D) + \sum_{v \in V(G)} \big(f(v) - f(q)\big). \tag{4.6}$$

(b) *If $E = D - \Delta(\chi_A) \in \mathrm{Div}(G)$ for some $A \subseteq V(G) \backslash \{q\}$, then*

$$b_q(E) = b_q(D) - |A|, \tag{4.7}$$

where $|A|$ is the cardinality of the set A. Thus $b_q(\cdot)$ is a monovariant.[4]

Proof. (a) We can find a $\mathbb{Q}$-valued function g so that $[\mathbf{1} - n(q)] = Q[g]$, where $n = |V(G)|$. Then

$$\begin{aligned}
\langle \mathbf{1}, E \rangle_q &= \langle \mathbf{1}, D \rangle_q + \big\langle \mathbf{1}, \Delta(f) \big\rangle_q \\
&= \langle \mathbf{1}, D \rangle_q + \big(Q[g]\big)^T L Q[f] = \langle \mathbf{1}, D \rangle_q + [g]^T Q L Q[f] \\
&= \langle \mathbf{1}, D \rangle_q + [g]^T Q[f] = \langle \mathbf{1}, D \rangle_q + \big(Q[g]\big)^T [f] \\
&= \langle \mathbf{1}, D \rangle_q + \big[\mathbf{1} - n(q)\big]^T [f] \\
&= \langle \mathbf{1}, D \rangle_q + \sum_{v \in V(G)} \big(f(v) - f(q)\big).
\end{aligned}$$

(b) follows from part (a) by setting $f = -\chi_A$. □

The $\mathbb{Q}$-valued function $g : V(G) \to \mathbb{Q}$ in the proof of Proposition 4.5(a) can be computed explicitly, and this gives a useful formula for b_q:

[4] A quantity which either only goes up or only goes down under some process.

Lemma 4.8. *Let $g_q : V(G) \to \mathbb{Q}$ be the unique function such that $\Delta(g_q) = \sum_v(v) - n(q)$ and $g_q(q) = 0$. Then*:

(a) $g_q(v) = \sum_{p \in V(G)} j_q(p, v)$.
(b) *For any divisor $D \in \mathrm{Div}(G)$,*

$$b_q(D) = \sum_v g_q(v)D(v) = \sum_v \sum_p j_q(p, v)D(v). \tag{4.9}$$

In particular, if $D(v) \geqslant 0$ for $v \neq q$, then $b_q(D) \geqslant 0$.

Proof. Part (a) is easy and is left as an exercise. Part (b) follows from Construction 3.1 and the definition of the energy pairing. Alternatively, let $[\mathbf{1} - n(q)] = Q[g_q]$ and $[D - d(q)] = Q[f]$ where $\deg(D) = d$. Then

$$\begin{aligned}\langle \mathbf{1}, D\rangle_q &= \big(Q[g_q]\big)^T L Q[f] \\ &= [g_q]^T Q[f] \\ &= [g_q]^T \big[D - d(q)\big] = \sum_v g_q(v)D(v) \\ &= \sum_v \sum_p j_q(p, v)D(v).\end{aligned}$$

The second statement follows because $j_q(p, v) \geqslant 0$ and $j_q(p, q) = 0$. □

4.3. Reduced divisors

A nice set of representatives for equivalence classes of divisors are given by the "reduced divisors".

Definition. Fix a vertex $q \in V(G)$. A divisor $D \in \mathrm{Div}(G)$ is called *q-reduced*[5] if it satisfies the following two conditions:

(i) $D(v) \geqslant 0$ for all $v \in V(G)\backslash\{q\}$.
(ii) For every non-empty subset $A \subseteq V(G)\backslash\{q\}$, there exists a vertex $v \in A$ such that $D(v) < \mathrm{outdeg}_A(v)$.

In other words, every vertex outside q is nonnegative but simultaneously firing all the vertices in any non-empty subset A of $V(G)$ which is disjoint from q will result in some vertex becoming negative.

The significance of reduced divisors comes primarily from the fact that for every $D \in \mathrm{Div}(G)$, there is a *unique* q-reduced divisor D' such that $D' \sim D$. This basic fact was discovered independently (in different guises) by several different authors (see, e.g., [26,24,41,6]). We give a new proof of this result in Corollary 4.13 below.

We wish to study reduced divisors from a potential-theoretic point of view. Fix a distinguished vertex q and define

$$|D|_q = \big\{E \in \mathrm{Div}(G) \;\big|\; E \sim D,\ E(v) \geqslant 0 \text{ for all } v \neq q\big\}.$$

Lemma 4.10. *For every $D \in \mathrm{Div}(G)$ and any vertex q, the set $|D|_q$ is non-empty.*

[5] Reduced divisors are essentially the same thing as *G-parking functions* [41] or *superstable configurations* [29].

Proof. Pick an ordering $\prec$ on $V(G)$ with the property that q is the first vertex in the ordering and every $v \neq q$ has a neighbor w with $w \prec v$. Starting from the last vertex in the ordering, we can inductively make all vertices other than q effective by replacing D with $D - k\Delta(\chi_w)$ for some neighbor $w \prec v$ and some sufficiently large integer k. □

Lemma 4.11 *(Principle of least action). Let D be a q-reduced divisor. Assume $E \sim D$ and write $D = E + \Delta(f)$.*

(a) *If $E \in |D|_q$, then $f(v) \leqslant f(q)$ for all $v \in V(G)$.*
(b) *If $E + \Delta(g) \in |D|_q$, then $f(v) - f(q) \leqslant g(v) - g(q)$ for all $v \in V(G)$.*

Proof. Part (a) is a consequence of Lemma 3.7. If f does not achieve its global maximum at q, then $A_{\max} \subseteq V(G)\backslash\{q\}$ and for all $v \in A_{\max}$ we have $\Delta_v(f) \geqslant \mathrm{outdeg}_{\mathcal{A}_{\max}}(v)$. Since D is q-reduced, there must be a vertex $u \in A_{\max}$ such that $D(u) < \mathrm{outdeg}_{\mathcal{A}_{\max}}(u)$. But then $E(u) = D(u) - \Delta_u(f) < 0$, contradicting the assumption that $E \in |D|_q$.

For (b), let $E' = E + \Delta(g)$. Then $D = E' + \Delta(f - g)$ and part (a) gives $f(v) - g(v) \leqslant f(q) - g(q)$. □

Theorem 4.12. *Fix $q \in V(G)$ and let $D \in \mathrm{Div}(G)$. Then D is q-reduced if and only if $D \in |D|_q$ and $\mathcal{E}_q(D) < \mathcal{E}_q(D')$ for all $D' \neq D$ in $|D|_q$.*

Proof. If D is q-reduced, then $D \in |D|_q$. Let $E \in |D|_q$. Then $\mathcal{E}_q(D) \leqslant \mathcal{E}_q(E)$; write $D = E + \Delta(f)$ with $f(q) = 0$. By Lemma 4.11(a), we have $f(v) \leqslant 0$. By Proposition 4.1(a), we have

$$\mathcal{E}_q(D) = \mathcal{E}_q(E) + \sum_{v \neq q} (D + E)(v) \cdot f(v) \leqslant \mathcal{E}_q(E).$$

Now assume $D \in |D|_q$ and $\mathcal{E}_q(D) \leqslant \mathcal{E}_q(E)$ for all $E \in |D|_q$ but D is not q-reduced. Then there exists a non-empty set $A \subseteq V(G)\backslash\{q\}$ such that $D_1 = D - \Delta(\chi_A) \in |D|_q$, so Proposition 4.1(b) implies

$$\mathcal{E}_q(D_1) = \mathcal{E}_q(D) - \lambda(A) \leqslant \mathcal{E}_q(D) - 1.$$

It follows that if $\mathcal{E}_q(D_1) = \mathcal{E}_q(D_2) \leqslant \mathcal{E}_q(E)$ for all $E \in |D|_q$, then both D_1 and D_2 are q-reduced. By Lemma 4.11(a), if $D_2 = D_1 + \Delta(f)$ with $f(q) = 0$, then $f(v) \leqslant 0$ for all $v \neq q$. Similarly $-f(v) \leqslant 0$ for all $v \neq q$, so $f = 0$ and $D_1 = D_2$. □

Corollary 4.13. *Fix $q \in V(G)$ and let $D \in \mathrm{Div}(G)$. Then there is a unique q-reduced divisor $D' \in \mathrm{Div}(G)$ linearly equivalent to D.*

Proof. It follows from Lemma 4.10, Proposition 4.1(b), and Lemma 3.5 that we may choose $D' \in \mathrm{Div}(G)$ such that $\mathcal{E}_q(D') \leqslant \mathcal{E}_q(D'')$ for all $D'' \in |D|_q$. By Theorem 4.12, D' is the unique q-reduced divisor linearly equivalent to D. □

An analogue of Theorem 4.12 holds with $\mathcal{E}_q$ replaced by b_q:

Theorem 4.14. *Fix $q \in V(G)$ and let $D \in \mathrm{Div}(G)$. Then D is q-reduced if and only if $D \in |D|_q$ and $b_q(D) < b_q(D')$ for all $D' \neq D$ in $|D|_q$.*

Proof. Note that by Lemma 4.8(b), the function b_q does have a minimum in $|D|_q$. The rest of the proof mirrors the proof of Theorem 4.12.

If D is q-reduced, then $D \in |D|_q$ by definition. Let $E \in |D|_q$ be another divisor. Write $D = E + \Delta(f)$ with $f(q) = 0$. By Lemma 4.11(a) we have $f(v) \leqslant 0$. Now, by Proposition 4.5(a), we have

$$b_q(D) = b_q(E) + \sum_{v \neq q} f(v) \leqslant b_q(E).$$

Now assume $D \in |D|_q$ and $b_q(D) \leqslant b_q(E)$ for all $E \in |D|_q$ but D is not q-reduced. Then there exists a non-empty set $A \subseteq V(G)\backslash\{q\}$ such that $D_1 = D - \Delta(\chi_A) \in |D|_q$. But then Proposition 4.5(b) gives the contradiction

$$b_q(D_1) = b_q(D) - |A| \leqslant b_q(D) - 1.$$

It follows that if $b_q(D_1) = b_q(D_2) \leqslant b_q(E)$ for all $E \in |D|_q$, then both D_1 and D_2 are q-reduced. By Lemma 4.11(a), if $D_2 = D_1 + \Delta(f)$ with $f(q) = 0$, then $f(v) \leqslant 0$ for all $v \neq q$. Similarly $-f(v) \leqslant 0$ for all $v \neq q$, so $f = 0$ and $D_1 = D_2$. □

Remark 4.15. Theorem 4.14 remains true if $b_q(D) = \langle \mathbf{1}, D\rangle_q$ is replaced by $\langle \mathbf{h}, D\rangle_q$ for any "$\mathbb{R}$-divisor" $\mathbf{h}$ (i.e. $h(v) \in \mathbb{R}$), provided that $\mathbf{h}(v) > 0$ for $v \neq q$.

5. Algorithmic aspects of reduced divisors

5.1. Dhar's algorithm

Let D be a divisor on the graph G. In order to check whether or not D is q-reduced using the definition, one needs to check for *all subsets* $A \subseteq V(G)\backslash\{q\}$ whether or not there is a vertex $v \in A$ such that $D(v) < \text{outdeg}_A(v)$. But there is in fact a much more efficient procedure called *Dhar's burning algorithm* (after Dhar [25]).

The idea behind Dhar's algorithm is as follows. Think of the edges of G as being made of a flammable material. A fire starts at vertex q and proceeds along each edge adjacent to q. At each vertex $v \neq q$, there are $D(v)$ firefighters, each of whom can control fires in a single direction (i.e., edge) leading into v. Whenever there are fires approaching v in more than $D(v)$ directions, the fire burns through v and proceeds to burn along all the other edges incident to v. The divisor D is q-reduced iff the fire eventually burns through every vertex of G.

More formally, Dhar's algorithm is stated in Algorithm 1.

Input: A divisor $D \in \text{Div}(G)$, and a vertex $q \in V(G)$.
Output: TRUE if D is q-reduced, and FALSE if D is not q-reduced.

if $D(v) < 0$ for some $v \in V(G)\backslash\{q\}$ **then** output FALSE and Stop.
Let $A_0 = V(G)$ and $v_0 = q$.
for $1 \leqslant i \leqslant n-1$ **do**
  Let $A_i = A_{i-1}\backslash\{v_{i-1}\}$.
  if for all $v \in A_i$, $D(v) \geqslant \text{outdeg}_{A_i}(v)$ **then** output FALSE and Stop.
  else let $v_i \in A_i$ be any vertex with $D(v_i) < \text{outdeg}_{A_i}(v_i)$.
end
Output TRUE.

Algorithm 1: Dhar's burning algorithm.

The complexity of Dhar's algorithm is $O(n^2)$: there are at most n iterations, and at most n inequalities are tested in each iteration.

5.2. The Cori–Le Borgne algorithm

Using a modification of Dhar's burning algorithm, it is possible to obtain an "activity preserving" bijection between q-reduced divisors (of a given degree d) on G and spanning trees of G. This is more or less just a restatement of the work of Cori and Le Borgne in [23] in the language of reduced divisors; however, by using the Cori–Le Borgne algorithm in conjunction with the results of Sections 3 and 4, we are able to obtain new results.

The idea behind the Cori–Le Borgne algorithm is as follows. In our original formulation of Dhar's algorithm, we burned through multiple edges at once. We now use an ordering of $E(G)$ to break ties and implement a "controlled burn" in which only one edge at a time is burnt.

Thus, fix a total order on $E(G)$ and suppose we are given a q-reduced divisor D. We run Dhar's burning algorithm on D, starting with a fire at q. However, any time there are multiple unburnt edges which are eligible to burn, we always choose the *smallest* one. Whenever the fire burns through a vertex v, we *mark* the edge along which the fire traveled just before burning through v. Since D is q-reduced, the fire eventually burns through every vertex of G. The set of marked edges is connected, has cardinality $n-1$, and covers all vertices and thus forms a *spanning tree* T_D of G. We thus obtain an association {reduced divisors} $\rightsquigarrow$ {spanning trees} (Algorithm 2).

The remarkable fact discovered by Cori and Le Borgne is that this association is a *bijection*. The inverse map is also completely explicit and can be described as follows (using the same total order on $E(G)$). Suppose we are given a spanning tree T in G. A controlled burn starts at the vertex q and, as before, any time there are multiple unburnt edges eligible to burn we choose the smallest one. The difference is that now the firefighters at v can control incoming fires in every direction except for those corresponding to edges of T. Thus the fire burns through a vertex $v \neq q$ exactly when it travels along an edge $e \in T$ from some (burnt) vertex w to v. At the moment when v is burned through, we set $D(v)$ equal to $|\{\text{burnt edges adjacent to } v\}| - 1$. Eventually the fire will burn through every vertex and a nonnegative integer $D(v)$ will have been assigned to each vertex $v \neq q$. The value of $D(q)$ is determined by requiring that $\deg(D) = d$. It turns out that the resulting divisor D is q-reduced, so we obtain an association {spanning trees} $\rightsquigarrow$ {reduced divisors} (Algorithm 3) which one checks is *inverse* to Algorithm 2.

Input:
$G = (V, E)$ is graph with a fixed ordering on E,
$q \in V(G)$,
$D = \sum_v a_v(v)$, a q-reduced divisor of degree d.
Output:
T_D a spanning tree of G.

Initialization:
$X = \{q\}$ ("burnt" vertices),
$R = \emptyset$ ("burnt" edges),
$T = \emptyset$ ("marked" edges).

while $X \neq V(G)$ **do**
  $f = \min\{e = \{s,t\} \in E(G) \mid e \notin R,\ s \in X,\ t \notin X\}$,
  let $v \in V(G)\backslash X$ be the vertex incident to f,
  if $a_v = |\{e \text{ incident to } v \mid e \in R\}|$ **then**
    $X \leftarrow X \cup \{v\}$,
    $T \leftarrow T \cup \{f\}$,
  end
  $R \leftarrow R \cup \{f\}$
end
ExtAct $= E\backslash R$,
ExtPass $= R\backslash T$,
Output $T_D = T$.

Algorithm 2: Reduced divisor to spanning tree.

Theorem 5.1. *The association given by Algorithms 2 and 3 is a bijection. More precisely*:

(i) *For any q-reduced divisor D of degree d, Algorithm 2 outputs a spanning tree T_D of G.*
(ii) *For any spanning tree T, Algorithm 3 outputs a q-reduced divisor D_T of degree d on G.*
(iii) *Algorithms 2 and 3 are inverse to one another*: $T_{D_T} = T$ *and* $D_{T_D} = D$.

Moreover, under the bijection furnished by Algorithms 2 and 3:

Input:
$G = (V, E)$ is graph with a fixed ordering on E,
$q \in V(G)$,
T a spanning tree of G.
Output:
$D_T = \sum_v a_v(v)$, a q-reduced divisor of degree d.
Initialization:
$X = \{q\}$ ("burnt" vertices),
$R = \emptyset$ ("burnt" edges).
while $X \neq V(G)$ **do**
  $f = \min\{e = \{s,t\} \in E(G) \mid e \notin R,\ s \in X,\ t \notin X\}$,
  if $f \in T$ **then**
    let $v \in V(G)\backslash X$ be the vertex incident to f,
    $a_v := |\{e \text{ incident to } v \mid e \in R\}|$,
    $X \leftarrow X \cup \{v\}$
  end
  $R \leftarrow R \cup \{f\}$
end
$a_q := d - \sum_{v \neq q} a_v$,
ExtAct $= E\backslash R$,
ExtPass $= R\backslash T$,
Output $D_T = \sum_v a_v(v)$.

Algorithm 3: Spanning tree to reduced divisor.

(iv) *The set R is the same at the end of both algorithms.*
(v) *The externally active edges*[6] *for T are precisely the elements of* ExtAct $= E\backslash R$, *and the externally passive edges for T are precisely the elements of* ExtPass $= R\backslash T$.
(vi) *The degree of $\sum_{v \neq q} a_v(v)$ is equal to $g - ex(T)$, where $g = m - n + 1$. Equivalently, $a_q = d - g + ex(T)$.*

Remark 5.2.

(1) The complexity of both Algorithms 2 and 3 is the same as that of Dhar's algorithm (Algorithm 1), namely $O(n^2)$.
(2) A natural choice for the fixed degree is $d = g$, in which case it follows from Theorem 5.1(vi) that $a_q = ex(T)$.

Remark 5.3. The problem of giving an explicit bijection between reduced divisors (in the guise of G-parking functions) and spanning trees has been studied in several previous works (see, e.g., [23,17,8]). There are also a number of bijections in the literature between *q-critical configurations* and spanning trees (see [26,13,12,24,41,38]). For a fixed vertex q, q-critical configurations provide another set of representatives for equivalence classes of divisors (see, e.g., [10,11]). There is a simple relationship between reduced and critical divisors: D is q-reduced if and only if $K^+ - D$ is q-critical, where $K^+ = \sum_{v \in V(G)} (\deg(v) - 1)(v)$ [6].

5.3. Computing the reduced divisor

Recall that computing the q-reduced divisor equivalent to some divisor D can be viewed as the solution to a linear (Theorem 4.14) or quadratic (Theorem 4.12) integer programming problem. Dhar's algorithm (which runs in time $O(n^2)$) shows that one can efficiently *check* whether a given divisor is

[6] An edge $e \in E\backslash T$ is called *externally active* for T if it is the largest element in the unique cycle contained in $T \cup \{e\}$, and is called *externally passive* for T if it is not externally active. The *external activity* of T is the number of externally active edges for T and is denoted by $ex(T)$.

the solution to the corresponding integer programming problem. Next we show that in fact one can *find* the solution efficiently as well.

Fix a base vertex $q \in V(G)$. Given a divisor $D \in \mathrm{Div}(G)$, Algorithm 4 below efficiently[7] computes the q-reduced divisor $D' \sim D$. The idea behind the algorithm is as follows. Starting with a divisor D, the first step is to replace D with an equivalent divisor whose coefficients are "small". This is accomplished by the simple trick of replacing $[D]$ with $[D] - Q\lfloor L_{(q)}[D]\rfloor$, where L_q is as in Construction 3.1 and $\lfloor\cdot\rfloor$ denotes the coordinate-wise floor function. The second step is to make the divisor effective outside q. This is done by having negative vertices borrow from their neighbors in a controlled way. The third step is to iterate Dhar's algorithm until we reach a q-reduced divisor. More specifically, if D is not yet reduced then by running Dhar's algorithm on D we obtain a subset A of $V(G)\backslash\{q\}$ such that firing all vertices in A once yields a divisor $D - \Delta(\chi_A)$ which is still effective outside q. Replacing D by $D - \Delta(\chi_A)$ and iterating this procedure, one obtains (after finitely many iterations) a q-reduced divisor. Moreover, the number of iterations can be explicitly bounded in terms of the j-function (Section 3.2) using formula (4.9).

A formal statement of the resulting algorithm appears below (Algorithm 4).

Input:
Q is the Laplacian matrix of the graph G,
$D \in \mathrm{Div}(G)$,
$q \in V(G)$.
Output:
$D' \sim D$ the unique q-reduced divisor equivalent to D.

(Step 1)
Find the generalized inverse $L_{(q)}$ of Q, as in Construction 3.1. Compute the divisor $[D'] = [D] - Q\lfloor L_{(q)}[D]\rfloor$.

(Step 2)
while there exists $v \neq q$ with $D'(v) < 0$ **do** $[D'] \leftarrow [D'] + Q[\chi_{\{v\}}]$.

(Step 3)
Let $A_0 = V(G)$, $v_0 = q$, and $i = 1$.
while $i \leqslant n-1$ **do**
  Let $A_i = A_{i-1}\backslash\{v_{i-1}\}$.
  if there exists $v_i \in A_i$ such that $D'(v_i) < \mathrm{outdeg}_{A_i}(v_i)$ **then** $i \leftarrow i+1$.
  else $[D'] \leftarrow [D'] - Q[\chi_{A_i}]$. Reset $i = 1$.
end

Algorithm 4: Finding the reduced divisor.

Correctness of Algorithm 4:

Assume for the moment that the algorithm actually terminates and produces an output. It is easy to see that the output is linearly equivalent to D. Also, the output passes Dhar's algorithm and therefore is q-reduced (in fact, as discussed above, one can view Step 3 as an iterated Dhar's algorithm). Therefore, for the correctness of the algorithm, we only need to show that it terminates (which follows *a posteriori* from the efficiency analysis below).

Efficiency of Algorithm 4:

Proposition 5.4. *If* $[D'] = [D] - Q\lfloor L_{(q)}[D]\rfloor$, *then* $|D'(v)| < \deg(v)$ *for all* $v \neq q$.

Proof. Recall from Construction 3.1 that $Q L_{(q)} = I + R_{(q)}$, where I is the identity matrix and $R_{(q)}$ has -1 entries in q-th row and is zero elsewhere. Therefore $[D] = Q L_{(q)}[D] + \deg(D)\cdot \mathbf{e}_q$, where $\mathbf{e}_q$ is the

[7] "Efficient" in this context means that the running time will be polynomial in m and n with only $\log(\deg(D))$-bit computations involved.

column vector which is 1 in position q and zero elsewhere. Now

$$\begin{aligned}\left[D'\right] &= [D] - Q\left\lfloor L_{(q)}[D]\right\rfloor \\ &= Q\left(L_{(q)}[D] - \left\lfloor L_{(q)}[D]\right\rfloor\right) + \deg(D)\cdot \mathbf{e}_q \\ &= Q\,\mathbf{f} + \deg(D)\cdot \mathbf{e}_q,\end{aligned}$$

where $\mathbf{f} = L_{(q)}[D] - \lfloor L_{(q)}[D]\rfloor$ is a vector with entries in $[0,1)$. It is now easy to show that the absolute values of the entries of $Q\,\mathbf{f}$ are bounded by the degree of the corresponding vertices. □

Remark 5.5. Computing the generalized inverse $L_{(q)}$ takes time at most $O(n^\omega)$, where ω is the exponent for matrix multiplication (currently $\omega = 2.376$ [22]). Notice that this computation is done only once. The second computation in Step 1 can be done using $O(n^2)$ operations (multiplication and addition). For bit complexity, one can check that the denominators appearing in the generalized inverse $L_{(q)}$ are annihilated by the exponent of the Jacobian group. The exponent is bounded above by the number of spanning trees of the graph. If we allow at most c parallel edges then there are at most $c^{n-1}\cdot n^{n-2}$ spanning trees. Moreover, one can also show that the absolute value of the entries of $L_{(q)}$ are bounded above by $R_{\max}$, the maximum effective resistance between any two vertices of the graph. Therefore all integers in the algorithm can be represented in $O(n\cdot \log cn)$ bits.

Now we will use our potential theoretic techniques to bound the number of chip-firing moves in Algorithm 4. As we will see, several different bounds in the literature can be obtained as special cases or corollaries of our general potential theory bound.

Proposition 5.6.

(a) *Let D_1 be the output of Step 1 of Algorithm 4. Then Step 2 of Algorithm 4 terminates in at most $b_q(K^+ - D_1)$ borrowing moves, where $K^+ = \sum_{v\in V(G)}(\deg(v)-1)(v)$.*
(b) *Let D_2 be the output of Step 2 of Algorithm 4. Then Step 3 of Algorithm 4 terminates in at most $b_q(D_2)$ firing moves.*
(c) *Algorithm 4 terminates in fewer than*

$$3\sum_v\sum_p j_q(p,v)\deg(v) \tag{5.7}$$

chip-firing moves.

Proof. (a) By Proposition 5.4, $|D_1(v)| < \deg(v)$ for all $v \neq q$. So for any $v \neq q$ with $D_1(v) < 0$, only one borrowing is needed to make the vertex positive. Moreover, the resulting positive number will be less than $\deg(v)$. This fact, together with Proposition 5.4, guarantees that the output of Step 2 satisfies $0 \leqslant D_1(v) < \deg(v)$ for all $v \neq q$. The result now follows from Lemma 4.8 and Proposition 4.5(b); the value of $b_q(\cdot)$ is at least $\sum_v\sum_p j_q(p,v)D_1(v)$ on the input of Step 2, and is at most $\sum_v\sum_p j_q(p,v)(\deg(v)-1)$. Moreover, with each borrowing $b_q(\cdot)$ increases by 1.

(b) This again follows from Lemma 4.8 and Proposition 4.5(b). Note that $D_2(v) < \deg(v)$, and that no vertex $v \neq q$ can become negative in Step 3.

(c) This follows from parts (a) and (b) and the inequalities

$$b_q\left(K^+ - D_1\right) < 2\sum_v\sum_p j_q(p,v)\deg(v),$$

$$b_q(D_2) < \sum_v\sum_p j_q(p,v)\deg(v). \qquad \square$$

5.4. Comparison to other techniques in the literature

By basic properties of the j-function (see Section 3.2), we have

$$3\sum_v \sum_p j_q(p,v)\deg(v) \leqslant 3(n-1)\sum_v r(v,q)\deg(v).$$

By Proposition 5.6 (specifically (5.7)), it follows that Algorithm 4 terminates in fewer than

$$3(n-1)\sum_v r(v,q)\deg(v) \tag{5.8}$$

chip-firing moves.

The bound (5.8) can be computed in matrix multiplication time $O(n^\omega)$ (currently $\omega = 2.376$) because $j_q(p,v)$ is simply the (p,v)-entry of the matrix $L_{(q)}$ in Construction 3.1.

Remark 5.9. There are several ways to bound the expression in (5.8) in terms of more familiar invariants of the graph. For example:

(1) Let $R_{\max}$ be the maximum effective resistance between vertices of G and let $\Delta_{\max}$ be the maximum degree of a vertex in G. Then (5.8) is bounded above by

$$3(n-1)R_{\max}\sum_{v\neq q}\deg(v),$$

which is, in turn, bounded above by

$$3(n-1)^2 R_{\max}\Delta_{\max}.$$

These estimates give a factor n improvement over the bound for the running time of Algorithm 4 which could be derived using the technique in [29] by Holroyd, Levine, Mészáros, Peres, Propp, and Wilson.

(2) Using Foster's network theorem, one can show that

$$r(v,q) < 3\sum_{v\in V(G)}\big(\deg(v)+1\big)^{-1}$$

(see, e.g., [21, proof of Theorem 6]). So another upper bound for the running time of Algorithm 4 is

$$9(n-1)\sum_{v\in V(G)}\big(\deg(v)+1\big)^{-1}\sum_{v\neq q}\deg(v).$$

This is a good bound when the graph is close (on average) to being regular. If one uses the fact that degree of a vertex is at least the edge-connectivity λ of the graph, one gets a bound of the form $O(n^2m/\lambda)$ for the running time of Algorithm 4. This is (up to constant factors) the bound that one can derive from the techniques of van den Heuvel [28].

(3) Let λ_1 (called the *algebraic connectivity* of G) be the smallest non-zero eigenvalue of Q. Then $r(p,q) \leqslant \frac{2}{\lambda_1}$ for every $p,q \in V(G)$. Indeed, the proof of the Cauchy–Schwarz inequality shows that for any positive semidefinite matrix L with largest eigenvalue η, and for all vectors x and y, $|x^T Ly| \leqslant \eta\|x\|_2\|y\|_2$; if we apply this estimate with $x = y = (p)-(q)$, $L = Q^+$ (cf. Construction 3.2), then $\eta = 1/\lambda_1$, and we obtain

$$r(p,q) = \big\langle (p)-(q), (p)-(q)\big\rangle \leqslant \frac{2}{\lambda_1}.$$

Therefore an upper bound for the running time of Algorithm 4 is

$$\frac{6(n-1)}{\lambda_1}\sum_{v\neq q}\deg(v).$$

This is the bound that one can derive from the techniques of Björner, Lovász and Shor [14] or Chung and Ellis [19].

(4) By Rayleigh's monotonicity law, we have $R_{\max}\leqslant \operatorname{diam}(G)$, where $\operatorname{diam}(G)$ denotes the *diameter* of G. Equality holds if and only if G is a path. In fact R_{max} is much smaller than $\operatorname{diam}(G)$ in a general graph. Another upper bound for the running time of Algorithm 4 is

$$3(n-1)\operatorname{diam}(G)\sum_{v\neq q}\deg(v).$$

This is the bound that one can derive from the techniques of Tardos [46].

Remark 5.10. Items (3) and (4) in the previous remark clarify the relationship between the seemingly different approaches of Tardos and Björner–Lovász–Shor.

6. Some applications of the algorithms

6.1. Bijective matrix-tree theorem

Kirchhoff's celebrated *matrix-tree theorem* is usually formulated as follows. Let G be a (connected) graph. Following the terminology from Construction 3.1, fix an integer $1\leqslant i\leqslant n$ and let Q_i be the invertible $(n-1)\times(n-1)$ matrix obtained from the Laplacian matrix Q of G by deleting i-th row and i-th column from Q.

Theorem 6.1 *(Kirchhoff's matrix-tree theorem). (See [32].) The number of spanning trees in G is equal to* $|\det(Q_i)|$.

Our aim in this section is to give an "efficient bijective" proof of Theorem 6.1. In order to make sense of this goal, it is useful to reformulate Kirchhoff's theorem in a more natural way in terms of the Jacobian group[8] $\operatorname{Jac}(G)=\operatorname{Div}^0(G)/\operatorname{Prin}(G)$, where $\operatorname{Div}^0(G)$ is the subgroup of $\operatorname{Div}(G)$ consisting of divisors of degree zero and $\operatorname{Prin}(G)$ (the group of principal divisors) is the image of the Laplacian operator $\Delta:\mathcal{M}(G)\to\operatorname{Div}(G)$.

By elementary group theory (e.g. the theory of the *Smith normal form*), one sees that $\operatorname{Jac}(G)$ is the torsion part[9] of the cokernel of $Q:\mathbb{Z}^n\to\mathbb{Z}^n$, and the order of $\operatorname{Jac}(G)$ is equal to $|\det(Q_i)|$. We may thus reformulate Kirchhoff's theorem as follows:

Theorem 6.2 *(Kirchhoff's matrix-tree theorem, canonical formulation). The number of spanning trees in G is equal to* $|\operatorname{Jac}(G)|$. *Moreover, there exists an efficiently computable bijection between elements of* $\operatorname{Jac}(G)$ *and spanning trees of G.*

Note that such a bijection cannot be canonical, as that would imply the existence of a distinguished spanning tree in G corresponding to the identity element of $\operatorname{Jac}(G)$, but it is clear (think of the case where G is an n-cycle) that there is in general no distinguished spanning tree. Therefore, one needs to make some choices to write down a bijection.

[8] Although we do not need this here, it is worth mentioning that the energy pairing descends to a non-degenerate $\mathbb{Q}/\mathbb{Z}$-valued bilinear form on $\operatorname{Jac}(G)$; it is called the "monodromy pairing" in [44].

[9] The full cokernel $\operatorname{Pic}(G)=\operatorname{Div}(G)/\operatorname{Prin}(G)$ is isomorphic to $\mathbb{Z}\oplus\operatorname{Jac}(G)$.

Proof of Theorem 6.2. By Corollary 4.13, if we fix a vertex q of G, there is a unique q-reduced divisor representing each class in $\mathrm{Div}^0(G)$. In particular, there is an explicit bijection between $\mathrm{Div}^0(G)$ and the set of q-reduced divisors of degree 0. If in addition we choose a total order on $E(G)$, then the algorithms in Section 5.2 show that there is a bijection between q-reduced divisors of degree 0 and spanning trees of G.

We have shown in Section 5.3 that the unique q-reduced representative for each class in $\mathrm{Div}^0(G)/\mathrm{Prin}(G)$ can be computed efficiently. And in Section 5.2 we proved that the bijection between q-reduced divisors of degree 0 and spanning trees of G is also efficient. □

Kirchhoff's matrix-tree theorem is of course a classical result. The main new contribution here is to observe that the bijections between reduced divisors and spanning trees (as described in Section 5.2 and Remark 5.3), in conjunction with Corollary 4.13, furnish a simple bijective proof of Kirchhoff's theorem, and moreover this bijection is efficiently computable.

6.2. Random spanning trees

The *random spanning tree* problem has been extensively studied in the literature and there are two known types of algorithms: determinant based algorithms (e.g. [27,20,33]) and random walk based algorithms (e.g. [16,1,48,31] and [37, Chapter 4]).

Here we give a new deterministic polynomial time algorithm for choosing a random spanning tree in a graph G.

Although the bound we obtain for the running time of our algorithm does not beat the current best known running time $O(n^\omega)$ of [20], we believe that our algebraic method has some advantages. For example, it is trivial that the output of our algorithm is a uniformly random spanning tree, whereas in other algorithms (e.g. [20]) this fact is non-trivial and requires proof. See Remark 6.3 for another advantage.

The idea behind our algorithm is very simple. Fix a vertex $q \in V(G)$ and a total ordering of $E(G)$. The first step in the algorithm is to compute a presentation of $\mathrm{Jac}(G)$ as a direct sum of cyclic groups; this can be done efficiently by computing the Smith normal form for Q. Once $\mathrm{Jac}(G)$ is presented in this way, it is clear how to select a random element. Having done so, one computes the corresponding q-reduced divisor and then the corresponding spanning tree.

This procedure is formalized in Algorithm 5 below.

Input: A graph G.
Output: A uniformly random spanning tree of G.
(1) Fix a vertex $q \in V(G)$ and a total ordering of $E(G)$.
(2) Compute the Smith normal form of the Laplacian matrix of G to find:
– invariant factors $\{n_1, \ldots, n_s\}$,
– generators $\{\mathbf{g}_1, \ldots, \mathbf{g}_s\}$ for $\mathrm{Jac}(G)$ (thought of as elements of $\mathrm{Div}^0(G)$).
(3) Choose a random integer $0 \leqslant a_i \leqslant n_i - 1$ for $(1 \leqslant i \leqslant s)$.
(4) Compute the divisor $D = \sum_{i=1}^{s} a_i \mathbf{g}_s$.
(5) Use Algorithm 4 to find the unique q-reduced divisor D' equivalent to D.
(6) Use Algorithm 2 to find the spanning tree corresponding to D'.

Algorithm 5: Choosing a uniformly random spanning tree.

To our knowledge, the fastest known Smith normal form algorithm (step (2)) is given in [30] and has running time $(n^{2.697263} \log \| Q \|)^{1+o(1)}$, where $\| Q \|$, for our application, means the maximal degree of a vertex $\Delta_{\max}$. See also [43] for a fast and practical Smith normal form algorithm. For the running time of step (5) see (5.7), (5.8), and Remark 5.9. Step (6) can be done in $O(n^2)$ steps.

Remark 6.3. Note that for repeated sampling of random spanning trees in G, one has to perform steps (1) and (2) of Algorithm 5 only once. Note also that if there are N spanning trees in G, our algorithm uses only $\log_2 N$ random bits for generating each random spanning tree. Thus our algorithm may

have some advantages over existing methods for sampling multiple spanning trees. For example, very few random bits are required in our algorithm to generate pairwise independent spanning trees; to generate k pairwise independent spanning trees, the naive approach would use $k \cdot \log_2 N$ random bits. But one can use standard methods to pick pairwise independent elements of the group using only $O(\log_2 N)$ random bits. Also, it is possible with our method to sample multiple spanning trees according to joint distributions other that the uniform distribution. (We thank Richard Lipton for these observations; see [35].)

6.3. Other applications

We list briefly some other applications of our algorithms:

(1) (The group law attached to chip-firing games) If we fix a vertex $q \in V(G)$, then $\mathrm{Jac}(G)$ induces a group structure on the set of q-reduced divisors (G-parking functions) or q-critical divisors of G. The latter is called *critical group* (or sandpile group) of G. Adding two elements in one of these groups requires first adding the given divisors as elements of $\mathrm{Div}(G)$, and then finding the unique q-reduced or q-critical divisor equivalent to the sum. Our algorithm for finding q-reduced divisors can be used to efficiently compute the group law in these groups. A different approach for performing the group operation is given in [28] using "oil games". The problem of finding a "purely algebraic" method for computing the sum of two elements of the critical group (and analyzing the running time of the resulting algorithm) was posed as an open problem by Chung and Ellis in [19].

(2) (Determining whether the dollar game is winnable) In [6], the authors consider a dollar game played on the vertices of G. Given a divisor D, thought of as a configuration of dollars on G, the goal of the game is to get all the vertices out of debt via borrowing and lending moves, i.e., to find an effective divisor D' linearly equivalent to D. By the proof of Theorem 3.3 in [6], the game is winnable iff the unique q-reduced divisor equivalent to D is effective. As a corollary, once we can efficiently compute the q-reduced divisor associated to a given configuration, we can efficiently decide whether or not there is a winning strategy, and when there is one we can efficiently compute a sequence of winning moves.

These considerations are related to the Riemann–Roch theorem for graphs from [6]. To any divisor $D \in \mathrm{Div}(G)$ one associates an integer $r(D) \geqslant -1$, called the *rank* of D, such that $r(D) \geqslant 0$ iff the unique q-reduced divisor equivalent to D is effective. Our algorithm for finding q-reduced divisors can therefore be used to efficiently check whether or not $r(D) \geqslant 0$. More generally, we can efficiently check whether $r(D) \geqslant c$ for any fixed constant c. It is an open problem to determine whether or not one can compute $r(D)$ itself in polynomial time. For a study of this problem, see [39].

References

[1] D.J. Aldous, The random walk construction of uniform spanning trees and uniform labelled trees, SIAM J. Discrete Math. 3 (4) (1990) 450–465.
[2] N. Alon, I. Krasikov, Y. Peres, Reflection sequences, Amer. Math. Monthly 96 (9) (1989) 820–823.
[3] R. Bacher, P. de la Harpe, T. Nagnibeda, The lattice of integral flows and the lattice of integral cuts on a finite graph, Bull. Soc. Math. France 125 (2) (1997) 167–198.
[4] P. Bak, C. Tang, K. Wiesenfeld, Self-organized criticality, Phys. Rev. A (3) 38 (1) (1988) 364–374.
[5] M. Baker, X. Faber, Metrized graphs, Laplacian operators, and electrical networks, in: Quantum Graphs and Their Applications, in: Contemp. Math., vol. 415, Amer. Math. Soc., Providence, RI, 2006, pp. 15–33.
[6] M. Baker, S. Norine, Riemann–Roch and Abel–Jacobi theory on a finite graph, Adv. Math. 215 (2) (2007) 766–788.
[7] A. Ben-Israel, T.N.E. Greville, Generalized Inverses: Theory and Applications, second ed., CMS Books Math./Ouvrages Math. SMC, vol. 15, Springer-Verlag, New York, 2003.
[8] B.A. Benson, D. Chakrabarty, P. Tetali, G-parking functions, acyclic orientations and spanning trees, Discrete Math. 310 (8) (2010) 1340–1353.
[9] N. Biggs, Algebraic Graph Theory, second ed., Cambridge Math. Lib., Cambridge University Press, Cambridge, 1993.
[10] N. Biggs, Algebraic potential theory on graphs, Bull. Lond. Math. Soc. 29 (6) (1997) 641–682.
[11] N. Biggs, Chip-firing and the critical group of a graph, J. Algebraic Combin. 9 (1) (1999) 25–45.
[12] N. Biggs, The Tutte polynomial as a growth function, J. Algebraic Combin. 10 (2) (1999) 115–133.

[13] N. Biggs, P. Winkler, Chip-firing and the chromatic polynomial, Report LSE-CDAM-97-03, Centre for Discrete and Applicable Mathematics, London School of Economics, London, UK, February 1997.
[14] A. Björner, L. Lovász, P.W. Shor, Chip-firing games on graphs, European J. Combin. 12 (4) (1991) 283–291.
[15] B. Bollobás, Modern Graph Theory, Grad. Texts in Math., vol. 184, Springer-Verlag, New York, 1998.
[16] A. Broder, Generating random spanning trees, in: 30th Annual IEEE Symposium on Foundations of Computer Science, 1989, pp. 442–447.
[17] D. Chebikin, P. Pylyavskyy, A family of bijections between G-parking functions and spanning trees, J. Combin. Theory Ser. A 110 (1) (2005) 31–41.
[18] T. Chinburg, R. Rumely, The capacity pairing, J. Reine Angew. Math. 434 (1993) 1–44.
[19] F. Chung, R.B. Ellis, A chip-firing game and Dirichlet eigenvalues, in: Kleitman and Combinatorics: A Celebration, Cambridge, MA, 1999, Discrete Math. 257 (2–3) (2002) 341–355.
[20] C.J. Colbourn, W.J. Myrvold, E. Neufeld, Two algorithms for unranking arborescences, J. Algorithms 20 (2) (1996) 268–281.
[21] D. Coppersmith, U. Feige, J. Shearer, Random walks on regular and irregular graphs, SIAM J. Discrete Math. 9 (2) (1996) 301–308.
[22] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, in: Computational Algebraic Complexity Editorial, J. Symbolic Comput. 9 (3) (1990) 251–280.
[23] R. Cori, Y. Le Borgne, The sand-pile model and Tutte polynomials, in: Formal Power Series and Algebraic Combinatorics, Scottsdale, AZ, 2001, Adv. in Appl. Math. 30 (1–2) (2003) 44–52.
[24] R. Cori, D. Rossin, B. Salvy, Polynomial ideals for sandpiles and their Gröbner bases, Theoret. Comput. Sci. 276 (1–2) (2002) 1–15.
[25] D. Dhar, Self-organized critical state of sandpile automaton models, Phys. Rev. Lett. 64 (14) (April 1990) 1613–1616.
[26] A. Gabrielov, Abelian avalanches and Tutte polynomials, Phys. A 195 (1–2) (1993) 253–274.
[27] A. Guénoche, Random spanning tree, J. Algorithms 4 (3) (1983) 214–220.
[28] J. van den Heuvel, Algorithmic aspects of a chip-firing game, Combin. Probab. Comput. 10 (6) (2001) 505–529.
[29] A.E. Holroyd, L. Levine, K. Mészáros, Y. Peres, J. Propp, D.B. Wilson, Chip-firing and rotor-routing on directed graphs, in: In and Out of Equilibrium. 2, in: Progr. Probab., vol. 60, Birkhäuser, Basel, 2008, pp. 331–364.
[30] E. Kaltofen, G. Villard, On the complexity of computing determinants, Comput. Complexity 13 (3–4) (2004) 91–130.
[31] J.A. Kelner, A. Madry, Faster generation of random spanning trees, in: 50th Annual Symposium on Foundations of Computer Science, October 2009, pp. 13–21.
[32] G. Kirchhoff, Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme geführt wird, Ann. Phys. Chem. 72 (1847) 497–508.
[33] V.G. Kulkarni, Generating random combinatorial objects, J. Algorithms 11 (2) (1990) 185–207.
[34] L. Levine, J. Propp, What is ... a sandpile?, Notices Amer. Math. Soc. 57 (8) (2010) 976–979.
[35] R. Lipton, A new approach to random spanning trees, blog post, 2009, available at http://rjlipton.wordpress.com/2009/07/15/a-new-approach-to-random-spanning-trees/.
[36] D. Lorenzini, Arithmetical graphs, Math. Ann. 285 (3) (1989) 481–501.
[37] R. Lyons, Y. Peres, Probability on Trees and Networks, Cambridge University Press, 2011, in preparation; Current version available at http://mypage.iu.edu/~rdlyons/.
[38] S.N. Majumdar, D. Dhar, Equivalence between the Abelian sandpile model and the $q \to 0$ limit of the Potts model, Phys. A 185 (1–4) (1992) 129–145.
[39] M. Manjunath, The rank of a divisor on a finite graph: geometry and computation, 2011, preprint available at arXiv: 1111.7251v1.
[40] S. Mozes, Reflection processes on graphs and Weyl groups, J. Combin. Theory Ser. A 53 (1) (1990) 128–142.
[41] A. Postnikov, B. Shapiro, Trees, parking functions, syzygies, and deformations of monomial ideals, Trans. Amer. Math. Soc. 356 (8) (2004) 3109–3142 (electronic).
[42] M. Raynaud, Spécialisation du foncteur de Picard, Inst. Hautes Études Sci. Publ. Math. 38 (1970) 27–76.
[43] D. Saunders, Z. Wan, Smith normal form of dense integer matrices fast algorithms into practice, in: Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation, ISSAC '04, ACM, New York, NY, USA, 2004, pp. 274–281.
[44] F. Shokrieh, The monodromy pairing and discrete logarithm on the Jacobian of finite graphs, J. Math. Cryptol. 4 (1) (2010) 43–56.
[45] S. Smirnov, How do research problems compare with IMO problems?, in: D. Schleicher, M. Lackmann (Eds.), An Invitation to Mathematics, Springer, Berlin, Heidelberg, 2011, pp. 71–83.
[46] G. Tardos, Polynomial bound for a chip firing game on graphs, SIAM J. Discrete Math. 1 (3) (1988) 397–398.
[47] E. Wegert, C. Reiher, Relaxation procedures on graphs, Discrete Appl. Math. 157 (9) (2009) 2207–2216.
[48] D.B. Wilson, Generating random spanning trees more quickly than the cover time, in: Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, PA, 1996, ACM, New York, 1996, pp. 296–303.
[49] P. Winkler, Mathematical Puzzles: A Connoisseur's Collection, A K Peters Ltd., Natick, MA, 2004.