University of Wrocław
Faculty of Mathematics and Computer Science
Institute of Computer Science

Jakub Tarnawski

Walks on undirected graphs and generation of random spanning trees

Wrocław 2014

Abstract

We give an accessible and self-contained introduction to the theory of random walks on undirected graphs. Our focus is on obtaining tight upper bounds on quantitative parameters of random walks, such as the cover time. We apply these bounds to design algorithms for fundamental graph problems. We remark on the long-run properties of random walks and formulate conditions sufficient for these properties to also hold in the short term. We showcase the intriguing interplay between random walks, random spanning trees and electrical networks. Finally, we present the Kelner-Mądry algorithm (FOCS 2009) which builds on these connections to sample a random spanning tree from the uniform distribution in $\widetilde{\mathcal{O}}(m^{3/2})$ time.

Contents

1 Introduction

This thesis is a survey of selected topics in the theory of random walks on undirected graphs, particularly of those of its aspects which are applicable in computer science. After introducing the basic terms used to reason about random walks, we focus mainly on bounding the expected lengths of walks which end once certain conditions are fulfilled, such as the cover time of the graph (see Definition 2.7). We also present their surprising and nontrivial relationships with electrical circuits and random spanning trees. Because of this, the thesis can be alternatively viewed as an attempt to present, *ab initio* and in full detail, the Kelner-Mądry algorithm [KM09], which makes crucial use of all these connections to generate a random spanning tree faster than the cover time of the graph; its description in Chapter 7 is the culmination of the thesis.

The text can hopefully serve as an introduction to the field, as it requires no previous knowledge except for basics of graph theory and probability. We aim to give full, accessible proofs and to make the thesis self-contained. We develop the theory from the ground up and prove all statements leading up to Chapter 7. (The only exception to this are Laplacian solvers (see Theorem 3.4), which we treat as a black box.)

Throughout the thesis we establish all the random-walk-related results used to analyze the running times of algorithms [KM09] and [MST13], which solve the problem of sampling uniformly random spanning trees in undirected graphs (cf. Section 6 for the definition of the problem). In particular, we introduce the notion of *projected graph* (see Section 3.2), which has been conceived for this very purpose. The idea for this thesis was born after we discovered (for ourselves) the *uniformity of expectation* phenomenon, which we discuss at length in Chapter 4, and that it can be used to give alternative (arguably, simpler and more insightful) proofs of the aforementioned results. We hope that reproving them in this new framework will facilitate understanding of the behavior of the random walks that these algorithms utilize.

A list of original results (or new proofs) follows: Lemmas 2.12, 4.1, 4.12 and 4.13 and Propositions 3.10, 3.11 and 4.10.

1.1 Real-world example

To motivate our interest in random walks, let us consider the following real-world example.

Byteasar has a toy maze made of metal. It consists of many crossings, some of which are connected by bidirectional tunnels; we can view the maze as an undirected graph. All tunnels have the same length. The maze is inhabited by Byteasar's pet – a mentally challenged hamster. Byteasar likes to play fetch with it – when the hamster is at a crossing which we will call v, Byteasar drops a ball at another crossing, which we will call u. In order to receive a treat, the hamster must pick the ball up and return with it to v.

As remarked before, the hamster is not terribly bright, even for a hamster. Whenever it finds itself at any crossing, it chooses one of its incident tunnels at random (possibly the one through which it just arrived at the crossing) and runs through that tunnel. It is, however, adept at picking the ball up and it never drops it. Running through one tunnel takes 1 second. Byteasar wonders: what is the expected time before the hamster returns to v with the ball?

It turns out that he can find this out using an ohmmeter! He should connect it to crossings u and v (remember, the maze is made of metal), write down the measured resistance, divide it by the resistance of a single tunnel segment and multiply by twice the number of tunnels in the maze. As we will show in Proposition 3.12, this is his sought value, given in seconds.

The movement of the hamster in the maze is a random walk on the corresponding graph. The expected time that Byteasar is interested in is called the *commute time* between two vertices of this graph (cf. Definition 2.6). We will see in Section 3 that it can be characterized in terms of the *effective resistance* between those vertices in the corresponding electrical network using the

well-known formula $\kappa(u, v) = 2mR_{\text{eff}}(u, v)$. We give a new proof of this fact using the projected graph machinery of Chapter 3 (cf. Proposition 3.11).

1.2 Outline of the thesis

In Chapter 2 we define the random walk as a stochastic process (an example of a Markov chain) and prove its basic properties. We introduce the stationary distribution and the concept of time-reversibility. We then turn to the quantitative parameters of the random walk and their bounds, which will be our main focus in the next two chapters. At the end we give an example application of the theory: we show how to verify connectivity between two vertices of an undirected graph using only logarithmic space.

In Chapter 3 we look at the graph as an electrical network for the first time and introduce the concept of effective resistance, as well as the Laplacian matrix associated with the graph. Next, we establish the notion of the projected graph, used to reason about how random walks behave when we restrict their trajectories to a subgraph. We describe its interplay with effective resistance in Lemma 3.9. Afterwards, we use the developed theory to study the behavior of the walk related to the parameters of looping time and commute time (cf. Section 2.3), as well as prove Lemma 3.17, which is used in the analysis of the algorithm [MST13].

Chapter 4 expands on a property of the random walk which we first noticed in Section 3.3. We generalize this property, which we call *the uniformity principle*, and formally prove it. We then use it to easily reprove results which we earlier obtained using the projected graph, as well as give the first correct proof of Fact 5 of [KM09] (see Section 4.3).

We return to the topic of electrical networks in Chapter 5. We introduce (discrete) harmonic functions and use them to show another connection between electricity and random walks. We also describe how to compute a certain collection of probabilities, which are later utilized by the algorithm of Chapter 7.

The problem of generation of random spanning trees is the following: given a graph G, we want to sample a spanning tree of G so that the probability of picking any tree would be the same. In Chapter 6 the reader will find the proof of a very surprising relationship between random walks and random spanning trees, which gives rise to a simple algorithm for sampling such trees in time $\mathcal{O}(mn)$.

Chapter 7 describes in detail a slightly simplified version of the Kelner-Mądry method [KM09] of sampling random spanning trees, which has runtime $\widetilde{\mathcal{O}}(m^{3/2})$.

In Chapter 8 we remark on related results which did not make it into the thesis, as well as suggest other directions that the reader might be interested in pursuing.

1.3 Background and notation

We assume that the reader is familiar with the basics of graph theory and probability theory. For introductions to these fields, see e.g. [Bol79] and [Ros09].

An alphabetical list of selected terms and symbols is given below. We assume that $G = (V, E)$ is a *connected* graph with n vertices and m edges. In the following $u, v, s, t \in V$ are vertices of G and $V' \subseteq V$ is a subset of vertices.

- $\mathbb{1}$ – all-ones vector,
- (u, v) – an edge, usually undirected,
- $\langle u, v \rangle$ – a directed edge,
- $\langle x, y \rangle$, where x and y are vectors – scalar product of x and y,
- $\partial V'$ – boundary of V', i.e. the set of edges $\{(u, v) : u \in V', v \in V \setminus V'\}$ with exactly one endpoint in V',

- $\gamma_{\text{eff}}(G) = \max_{u,v \in V} R_{\text{eff}}(u,v)$ – effective resistance diameter of G (also used for subsets of vertices, see Definitions 3.14 and 3.16),
- $\varepsilon(v)$ – looping time (see Definition 2.8),
- $\kappa(u,v)$ – commute time (see Definition 2.6),
- $\pi(v)$ – stationary distribution of the random walk (see Section 2.2),
- χ_{st} – vector with 1 on s, -1 on t and 0 on the remaining coordinates (see Definition 3.3),
- boundary edge of V' – an edge in $\partial V'$,
- $\text{cov}(G)$ – cover time (see Definition 2.7),
- $D(G) = \max_{u,v \in V} \text{dist}(u,v)$ – diameter of G (also used for subsets of vertices),
- $d(v)$ – degree of a vertex v in G,
- $\text{dist}(u,v)$ – distance between vertices u and v in G,
- e_v – first-entry edge of vertex v (see Theorem 6.2),
- $E(V')$ – interior of V', i.e. the set of edges $\{(u,v) : u,v \in V'\}$ with both endpoints in V',
- $G[\![V']\!]$ – projected graph of G with respect to V' (see Definition 3.5),
- $H(u,v)$ – hitting time (see Definition 2.4),
- L – Laplacian matrix of G (see Definition 3.1),
- multiplicative ε-approximation of a vector $x \in \mathbb{R}^n$ – a vector $v \in \mathbb{R}^n$ such that $(1-\varepsilon)x_i \leq v_i \leq (1+\varepsilon)x_i$ for all $i = 1, \ldots, n$,
- $\widetilde{\mathcal{O}}(f(n))$ – the set of functions $g(n)$ for which there exists a $c > 0$ such that $g(n) = \mathcal{O}(f(n)\log^c f(n))$ (this notation allows for suppression of logarithmic factors),
- p_{uv} – probability of transition from vertex u to vertex v in one step (see Section 2.1),
- P_n – path graph on n vertices (see Example 2.17),
- $R_{\text{eff}}(u,v)$ – effective resistance between vertices u and v (see Definition 3.3),
- S_n – star graph on n vertices (see Example 2.18),
- T – a spanning tree of G,
- $\mathcal{T}(G)$ – set of all spanning trees T of G,
- X – random walk in G (see Definition 2.1).

Also, for weighted graphs $G = (V, E, w)$ we use the following notation:

- $w(u,v)$ – weight of edge $(u,v) \in E$,
- $d(v)$ – weighted degree of v, including self-loops, i.e. $d(v) = \sum_{u \in V:(u,v) \in E} w(u,v)$,
- $\hat{d}(v)$ – weighted degree of v, excluding self-loops, i.e. $\hat{d}(v) = \sum_{u \in V \setminus \{v\}:(u,v) \in E} w(u,v)$,
- $d(G)$ – weight of G, i.e. $d(G) = \sum_{v \in V} d(v)$.

We sometimes use a superscript to specify the graph we are referring to, as in $d^G(v)$.

2 Random walks

We begin by introducing the concept of the random walk on a graph and several elementary notions. Most definitions and basic results of this chapter are based on [Lov93] and [AKL$^+$79] or are folklore. Our notation is consistent with [MST13].

2.1 Definition of the random walk

We first give an intuitive explanation. Suppose we are standing on a given vertex $v \in V$ of G. One step of the random walk consists of selecting one of v's neighbors at random and moving to it. We repeat this procedure indefinitely. The infinite random sequence of vertices selected this way is called *the random walk on* G, and one concrete sample of this sequence is called a *trajectory* of the random walk.

Definition 2.1. *As the random walk in a graph $G = (V, E)$ we understand the following random process $X = (X_t)_{t\in\mathbb{N}}$. First, a starting vertex $v \in V$ is selected, and we set $X_0 = v$. In each step $t \geq 1$ we choose the next vertex X_t of the walk to be a random neighbor of X_{t-1} (chosen with uniform probability), and we say that the walk has* traversed *the edge $e = (X_{t-1}, X_t) \in E$.*

Usually, v is fixed, but it may also be chosen from a starting probability distribution μ on V. Formally, the stochastic process X depends on G and μ, and constitutes a mapping $X : \Omega \to V^{\mathbb{N}}$, where Ω is the set of states of the underlying probability space. For a fixed point $\omega \in \Omega$, the *trajectory* $X(\omega) : \mathbb{N} \to V$ is a sequence of vertices – a concrete realization of the random walk. As is standard practice with random variables, we will denote a trajectory simply by X when it is clear from the context.

Observe that the random walk X so defined is a finite Markov chain, with the set of states V and transition probability of p_{uv} between two states $u, v \in V$, where

$$p_{uv} = \begin{cases} \frac{1}{d(u)} & \text{if } (u, v) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Indeed, the variable X_t is a function of only X_{t-1} and a random choice of a neighbor of X_{t-1} which is independent from all previous events. In other words, the Markovian property holds:

$$\mathbb{P}(X_t = x_t \mid X_{t-1} = x_{t-1}) = \mathbb{P}(X_t = x_t \mid X_{t-1} = x_{t-1}, \ldots, X_0 = x_0)$$

for all $x_0, \ldots, x_t \in V$.

Note that for the definition to be sound, we must require each vertex to have an outgoing edge (so that the walk does not get "stuck"). Furthermore, if the graph is strongly connected (particularly, if it is undirected and connected), then all states of the corresponding Markov chain are recurrent (see e.g. Fact 2.5).

The above definition can be easily generalized to graphs with multiple edges, and further to weighted graphs (with positive weights on edges). Let $w : E \to \mathbb{R}_+$ be a weight function. When the walk is in a vertex u, the next vertex v is chosen proportionally to $w(u, v)$, i.e., $p_{uv} = \frac{w(u,v)}{\sum_{(u,x)\in E} w(u,x)}$. (Sometimes we will also allow graphs with zero-weight edges if it makes notation more convenient; such edges can be removed without changing any properties of the graph, so the reader may think that they do not occur, or that they are erased as soon as they appear. We allow self-loops in weighted graphs; they too are usually insignificant.) For weighted graphs let us write $d(G) = \sum_{v\in V} d(v)$ to denote the sum of all degrees in G (this includes self-loops). If all weights are 1 and there are no self-loops, then $d(G) = 2m$.

Let us remark in passing that random walks on weighted directed graphs are the same as finite Markov chains: indeed, the state-transition diagram for the corresponding Markov chain can be obtained from any such graph by simply normalizing the weights on edges so that for every $u \in V$ we have $\sum_{(u,x)\in E} w(u, x) = 1$.

To prevent unnecessary clutter in notation, most statements in this thesis will be given for unweighted graphs (unless it is necessary to state them in full generality). They all readily generalize to the weighted case.

Also, from this point on we will restrict our attention only to undirected graphs. Walks on such graphs enjoy many additional properties which make them worth studying, such as having polynomial bounds on their quantitative parameters (which we derive in Section 2.4 and refine throughout the thesis) or being time-reversible (see Proposition 2.3).

2.2 The stationary distribution

Let $\mu_0 = \mu$ be the distribution of the random variable X_0 (the starting distribution), and let μ_1 be the distribution of X_1 (the distribution of where the random walk is after the first step). If it happens that $\mu_0 = \mu_1$, we say that μ is a *stationary distribution* of the random walk on G. Of course, then the distribution of X_t is μ for every $t \in \mathbb{N}$.

It turns out that we can obtain such a distribution by just choosing to start in a vertex v with probability proportional to the degree $d(v)$.

Fact 2.2. *The distribution π given by $\pi(v) = \frac{d(v)}{2m}$ is stationary.*

Proof. Because $\sum_{v \in V} d(v) = 2m$, π is a probability distribution. The probability that the walk is in v after one step is

$$\sum_{u \in V:(u,v) \in E} \pi(u) \cdot \frac{1}{d(u)} = \sum_{u \in V:(u,v) \in E} \frac{1}{2m} = \frac{d(v)}{2m} = \pi(v). \qquad \square$$

We call a walk started from the distribution π a *stationary walk*.

Recall that throughout the thesis we are assuming that G is connected. Under this assumption, it can be shown that the distribution π is the unique stationary distribution of the random walk on G. We do this in Proposition 2.10.

A stationary walk on an undirected graph enjoys the following property.

Proposition 2.3 (time-reversibility). *Let X be a stationary walk. For any sequence of vertices $v_0, \ldots, v_k \in V$ we have*

$$\mathbb{P}(X_0 = v_0, X_1 = v_1, \ldots, X_k = v_k) = \mathbb{P}(X_0 = v_k, X_1 = v_{k-1}, \ldots, X_k = v_0).$$

In other words, any sequence $v_0, \ldots, v_k$ of vertices has the same probability of appearing at a given point of the walk as the reversed sequence $v_k, \ldots, v_0$: if someone showed us a movie recording of the random walk, we would not be able to tell whether it was running forwards or backwards.

Proof.

$$\begin{aligned}
\mathbb{P}(X_0 = v_0, X_1 = v_1, \ldots, X_k = v_k) &= \pi(v_0) \cdot p_{v_0 v_1} \cdot \ldots \cdot p_{v_{k-1} v_k} \\
&= \frac{d(v_0)}{2m} \cdot \frac{1}{d(v_0)} \cdot \frac{1}{d(v_1)} \cdot \ldots \cdot \frac{1}{d(v_{k-1})} \\
&= \frac{d(v_k)}{2m} \cdot \frac{1}{d(v_k)} \cdot \frac{1}{d(v_{k-1})} \cdot \ldots \cdot \frac{1}{d(v_1)} \\
&= \pi(v_k) \cdot p_{v_k v_{k-1}} \cdot \ldots \cdot p_{v_1 v_0} \\
&= \mathbb{P}(X_0 = v_k, X_1 = v_{k-1}, \ldots, X_k = v_0). \qquad \square
\end{aligned}$$

In fact, to establish time-reversibility it suffices to verify for any $u, v \in V$ that $\pi(u)p_{uv} = \pi(v)p_{vu}$. In our unweighted case, both of these quantities are $\frac{1}{2m}$ (provided that $(u, v) \in E$). We thus see that in a stationary walk every edge $(u, v) \in E$ is traversed in every direction with the same frequency. Later in the thesis (particularly in Section 4) we will prove similar uniformity results for walks which do not start from the stationary distribution.

2.3 Quantitative parameters of random walks

So far we have defined the random walk to be infinite in length. However, in reality we often wish to stop the walk as soon as certain conditions are fulfilled. We are interested in studying the expected time that it will take the walk to finish under these circumstances.

For example, suppose that we start the walk in a vertex u. How long will it take before the walk reaches another vertex v?

Definition 2.4. *For two different vertices $u, v \in V$ we define the* hitting time *(sometimes also called* access time*) $H(u, v)$ to be the expected number of steps of a random walk starting in u before it reaches v.*

We start off with a very crude bound for $H(u, v)$. At this point we are only interested in showing that it is always finite.

Fact 2.5. *For all $u, v \in V$, $H(u, v) < \infty$.*

Proof. Let us divide the random walk started at u into phases of length $n - 1$. During each of these phases, regardless of the current vertex at the beginning of the phase, the possibility that v will be visited is at least $\left(\frac{1}{n-1}\right)^{n-1}$, as there exists a path (of length at most $n - 1$) from the current vertex to v, and the probability of following the next edge of this path is at least $\frac{1}{n-1}$ at every step. The expected number of such phases before v is first reached is thus at most $(n-1)^{n-1}$. □

We will usually find it more comfortable to work with the following, more symmetric notion:

Definition 2.6. *For two different vertices $u, v \in V$ we define the* commute time $\kappa(u, v)$ *to be the expected number of steps of a random walk starting in u before it reaches v and then comes back to u.*

Of course, we have $\kappa(u, v) = H(u, v) + H(v, u) = \kappa(v, u)$ by linearity of expectation.

We are particularly interested in the time it will take to visit all vertices of the graph, i.e., cover the graph.

Definition 2.7. *We define the* cover time $\mathrm{cov}(G)$ *of a graph G to be the expected time it takes for a random walk started at a vertex v to visit all vertices of G, maximized over all choices of v.*

Another parameter, which gives us insight into the long-run behavior of the walk, is connected to revisiting a vertex.

Definition 2.8. *For a vertex $v \in V$ we define the* looping time $\varepsilon(v)$ *to be the expected number of steps of a random walk starting in v before it visits v again.*

Let us focus on the looping time for now. Recall that in a stationary walk, for every vertex v the probability that v is visited at any fixed time is the same and equal to $\pi(v)$. In other words, the frequency with which v is visited is $\pi(v)$, and so we would expect $\frac{1}{\pi(v)}$ units of time to pass between two consecutive visits to v. Also, observe that the assumption that the walk is started from the stationary distribution seems insignificant here, as once we reach v and start counting the steps, we do not care anymore from what distribution we arrived at v. We support the correctness of our intuition with the following elegant proof.

Proposition 2.9. *For any $v \in V$ we have $\varepsilon(v) = \frac{1}{\pi(v)} = \frac{2m}{d(v)}$.*

Proof. Imagine the following experiment: run a stationary walk on G and consider τ – the expected time before v is first visited (if we happen to start at v, we walk until it is visited again). We have

$$\tau = \pi(v)\varepsilon(v) + \sum_{u \neq v} \pi(u)H(u,v).$$

Our walk always makes at least one step. If we consider the situation after this first step, the distribution is still stationary and we get

$$\tau = 1 + \pi(v) \cdot 0 + \sum_{u \neq v} \pi(u)H(u,v).$$

This proves that $\pi(v)\varepsilon(v) = 1$. Note that we needed Fact 2.5 to ensure that the right-hand expressions are well-defined (finite). □

As an interesting aside, note that this implies the uniqueness of the stationary distribution.

Proposition 2.10. *If ρ is a stationary distribution of the random walk on G, then $\rho = \pi$.*

Proof. We can repeat the statement and proof of Proposition 2.9 using the stationary distribution ρ and obtain for any $v \in V$

$$\frac{1}{\rho(v)} = \varepsilon(v) = \frac{1}{\pi(v)}$$

since $\varepsilon(v)$ depends only on the graph G. □

Note that we essentially use the connectedness of G in the proof of Proposition 2.9 to ensure that $H(u,v) < \infty$ for all $u \in V$. Indeed, if G fails to be connected, there exist many stationary distributions, e.g., if G has two connected components G_1, G_2 with stationary distributions π_1 and π_2, then any convex combination $\pi = \alpha\pi_1 + (1-\alpha)\pi_2$ with $\alpha \in [0,1]$ is a stationary distribution.

We have thus managed to derive an exact value for the parameter $\varepsilon(v)$. Using much more involved methods, we will also obtain a similar result for the commute time in Proposition 3.12. For now, let us concentrate on upper-bounding the remaining quantities.

2.4 Upper bounds based on graph distances

One of the reasons why random walks on undirected graphs are pleasant to deal with is related to the existence of good (polynomial) upper bounds on commute and cover times. It is something that we do not have for directed graphs, even strongly connected ones, as we see in the following example.

Example 2.11 (cf. Fig. 1). *Consider the strongly connected directed graph $G = (V,E)$ with $V = \{1,\ldots,n\}$ and $E = \{\langle i, i+1\rangle : 1 \leq i < n\} \cup \{\langle i,1\rangle : 2 \leq i \leq n\}$. The expected time for a walk started in vertex 1 to reach vertex n is $\Theta(2^n)$.*

Proof. Denote this expected time by H. Consider what happens after the walk leaves vertex 1: let K be the maximum number of vertex it reaches before it comes back to 1. If $K = n$, we are done after $n-1$ steps, otherwise the walk restarts itself after having made K steps. We can see that

$$\mathbb{P}(K = k) = \begin{cases} 2^{-(k-1)} & \text{if } 2 \leq k < n, \\ 2^{-(n-2)} & \text{if } k = n, \end{cases}$$

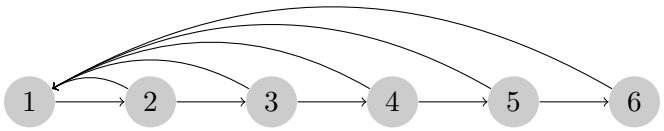

Figure 1: The graph of Example 2.11 with $n = 6$.

and thus

$$\begin{aligned} H &= \sum_{k=2}^{n-1} (k+H)2^{-(k-1)} + (n-1)2^{-(n-2)} \\ &= \sum_{k=1}^{n-2} (k+1)2^{-k} + \left(1 - 2^{-(n-2)}\right) H + (n-1)2^{-(n-2)}, \end{aligned}$$

whence

$$H = 2^{n-2} \sum_{k=1}^{n-2} (k+1)2^{-k} + n - 1$$

which is $\Theta(2^n)$ since $\lim_{n\to\infty} \sum_{k=1}^{n-2} (k+1)2^{-k} = 3$. □

Reaching vertex n was so difficult because the one-way edges $\langle i, 1\rangle$ only hindered our movement; we could not use them in the reverse direction. It turns out that in the absence of such one-way-only obstacles, the hitting time is always polynomially bounded. We will derive such bounds in this section, using the techniques of Aleliunas et al. [AKL+79]. Following their approach, let us first consider the commute time between two adjacent vertices.

Lemma 2.12 (Lemma 2 of [AKL+79]). *Let $(u, v) \in E$ be an edge. Then $\kappa(u, v) \leq 2m$. Moreover, there is equality iff (u, v) is a bridge in G.*

The short argument given in [AKL+79] makes use of an intuitive property of Markov chains related to their long-term behavior which is not trivial to make precise and rigorously prove; we devote Section 4 to it. For now, since we seek to have a complete, self-contained treatment which does not use any unproved intuitions, we give our own proof of this lemma. For the interested reader, we present their argument later as Corollary 4.7.

Proof. We start in u and wait until the walk comes back to u. We call this one loop of the walk. If during this loop v was not visited, we go again, otherwise we finish. The expected time of this entire procedure is obviously $\kappa(u, v)$. On the other hand, it is equal to the expected runtime of the first loop (which is $\varepsilon(u)$) plus the expected runtime of the rest of the procedure (which is 0 if we happened to visit v during the first loop, and $\kappa(u, v)$ otherwise). Hence

$$\kappa(u, v) = \varepsilon(u) + \kappa(u, v) \cdot \mathbb{P}(v \text{ is not visited during first loop}),$$

implying that

$$\kappa(u, v) = \frac{\varepsilon(u)}{\mathbb{P}(v \text{ is visited during first loop})} \leq \frac{\varepsilon(u)}{\mathbb{P}(v \text{ is visited in first step})} = \frac{\frac{2m}{d(u)}}{\frac{1}{d(u)}} = 2m.$$

Also, observe that the equality

$$\mathbb{P}(v \text{ is visited during first loop}) = \mathbb{P}(v \text{ is visited in first step})$$

is satisfied if and only if there is no path from u to v which does not pass through the edge (u, v), i.e., (u, v) is a bridge in G. □

We now extend the bound of Lemma 2.12 to all pairs of vertices.

Lemma 2.13 (Lemma 3 of [AKL+79]). *For any $u, v \in V$ we have $\kappa(u, v) \leq 2m \operatorname{dist}(u, v)$.*

Proof. Let $u = u_0, u_1, \ldots, u_k = v$ be a shortest path between u and v in G. Then

$$\begin{aligned}\kappa(u, v) &= \kappa(u_0, u_k) \\ &\leq H(u_0, u_1) + \ldots + H(u_{k-1}, u_k) + H(u_k, u_{k-1}) + \ldots + H(u_1, u_0) \\ &= \kappa(u_0, u_1) + \ldots + \kappa(u_{k-1}, u_k) \\ &\leq 2mk\end{aligned}$$

by Lemma 2.12. □

In particular, we have $\kappa(u, v) < 2mn$ for all pairs of vertices u, v.

We now turn our attention to the cover time. It is clear that it is lower-bounded by the maximum hitting time $\max_{u,v \in V} H(u, v)$: if we are forced to start at u, visiting all vertices includes visiting v. In consequence we have

$$\operatorname{cov}(G) \geq \max_{u,v \in V} H(u, v) \geq \frac{1}{2} \max_{u,v \in V} \kappa(u, v)$$

where the second inequality follows because for any $u, v \in V$ we have $\frac{1}{2}\kappa(u, v) = \frac{1}{2}(H(u, v) + H(v, u)) \leq \max(H(u, v), H(v, u))$.

Somewhat surprisingly, this lower bound turns out to be tight up to a logarithmic factor, as we will soon demonstrate in Theorem 2.15. We first need a lemma. We state it in a slightly more general form than we need right now, as it will be useful later to prove Lemma 4.12.

Lemma 2.14 (cf. Lemma 2.8 of [Lov93]). *Let $V' \subseteq V$ be a subset of vertices, $s \in V$ – a starting vertex, and let $M = \max_{v \in V'} H(s, v)$ be the maximum hitting time between s and a vertex in V'. Suppose we run a random walk from s until more than half of the vertices from V' are visited. The expected length of this walk is at most $2M$.*

Proof. We assume first that $|V'| = 2k + 1$ is odd. Let α_v be the time when vertex v is first visited. Then the length of the walk, β, is the $(k + 1)$-th largest of the α_v. Hence

$$(k + 1)\beta \leq \sum_{v \in V'} \alpha_v$$

and

$$\mathbb{E}[\beta] \leq \frac{1}{k+1} \sum_{v \in V'} \mathbb{E}[\alpha_v] \leq \frac{1}{k+1} \cdot |V'| \cdot M = \frac{2k+1}{k+1} M < 2M.$$

If $|V'| = 2k$ is even, we proceed analogously and obtain $\frac{2k}{k} M$ at the end. □

Theorem 2.15 (cf. Theorem 2.7 of [Lov93]). *Let $M = \max_{u,v \in V} H(u, v)$ be the maximum hitting time in G. Then $\operatorname{cov}(G) \leq 2M \log n$.*

Proof. Let $s \in V$ be the starting vertex (chosen for us by an adversary). We divide the covering walk into phases. The first phase ends when more than half of vertices have been visited. By Lemma 2.14 applied to $V' = V$ and s, its expected length is at most $2M$. The second round ends when at least half of the remaining vertices have been covered, and the lemma applied to the set of remaining vertices (and the starting vertex being the vertex where the first phase ended) again bounds the expected length of the second phase by $2M$. We proceed in this fashion until all vertices are visited. By linearity of expectation, the cover time of G is bounded by $2M$ times the number of phases, which is clearly $\log n$. □

The constant 2 can be improved: Matthews proved that the cover time is at most MH_n, where H_n is the n-th harmonic number (Theorem 2.6 in [Mat88]).

Summing up, we get

Corollary 2.16. *The following relations hold:*

$$2 \log n \max_{u,v \in V} \kappa(u,v) \geq 2 \log n \max_{u,v \in V} H(u,v) \geq \mathrm{cov}(G) \geq \max_{u,v \in V} H(u,v) \geq \frac{1}{2} \max_{u,v \in V} \kappa(u,v). \qquad \square$$

Both the lower and the upper bound are attained up to a constant, as exhibited by the following two examples.

Example 2.17 (cf. Fig. 2). *The path graph on n vertices is the graph $P_n = (V, E)$ with $V = \{1, \ldots, n\}$ and $E = \{(i, i+1) : 1 \leq i < n\}$. It satisfies $\mathrm{cov}(P_n) = \mathcal{O}(\max_{u,v \in V} \kappa(u,v))$.*

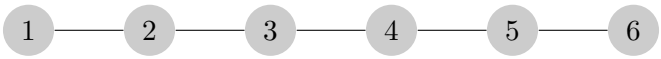

Figure 2: A path graph P_6.

Proof. Let s be any starting vertex. By the time we visit vertices 1 and n, the graph will have been covered, and the expected time that this will take can be bounded by $H(s,1) + H(1,n) = \mathcal{O}(\max_{u,v \in V} \kappa(u,v))$. □

Example 2.18 (cf. Fig. 3). *The star graph on n vertices is the graph $S_n = (V, E)$ with $V = \{1, \ldots, n\}$ and $E = \{(1,i) : 2 \leq i \leq n\}$. It satisfies $\mathrm{cov}(S_n) = \Theta(n \log n) = \Omega(\log n \cdot \max_{u,v \in V} \kappa(u,v))$.*

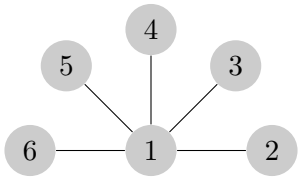

Figure 3: A star graph S_6.

Proof. We first argue that $H(u,v) = \Theta(n)$ for any two vertices $u \neq v$ with $v \neq 1$: from u we first go to 1 in one step (or zero if $u = 1$) and then try to pick v as the next vertex. If we do not succeed, we make two steps and try again; the probability of success is $\frac{1}{n-1}$, so it will take us $n - 1$ tries in expectation. It follows that $\kappa(u,v) = \Theta(n)$ for any $u \neq v$.

As for the cover time of S_n, it is closely related to the so-called coupon collector's problem. It is easy to see that the worst vertex to be starting at is 1. After two steps of the walk, we are back at 1 with two vertices visited. In the next two steps we have a probability of $\frac{n-2}{n-1}$ of visiting a new vertex; if we do not, our situation does not change. Thus the expected time before we visit a third vertex (and go back to 1) is $2 \cdot \frac{n-1}{n-2}$. A fourth new vertex is then visited in expected $2 \cdot \frac{n-1}{n-3}$ steps, and a k-th one – in $2 \cdot \frac{n-1}{n-k+1}$. We get a cover time of

$$\operatorname{cov}(S_n) = -1 + 2\sum_{k=2}^{n} \frac{n-1}{n-k+1} = -1 + 2(n-1)H_{n-1} = \Theta(n \log n),$$

where H_n is the n-th harmonic number (and -1 accounts for not having to return to 1 after the last vertex is visited). □

We now have a good bound on the cover time. Recall that $D(G)$ is the diameter of G.

Corollary 2.19. *For any G we have* $\operatorname{cov}(G) \leq 4mD(G) \log n$.

Proof. By Corollary 2.16 and Lemma 2.13 we get

$$\operatorname{cov}(G) \leq 2 \log n \max_{u,v \in V} \kappa(u,v) \leq 2 \log n \cdot 2m \cdot \max_{u,v \in V} \operatorname{dist}(u,v).$$

□

In particular, we have obtained the following universal bound: $\operatorname{cov}(G) < 4mn \log n$ (using $D(G) < n$). However, if we are inclined to give up $D(G)$ in favor of n in the bound, then we can actually do better, getting rid of the logarithmic factor.

Lemma 2.20 (Theorem 4 of [AKL$^+$79]). *For any G we have* $\operatorname{cov}(G) \leq 2m(n-1)$.

Proof. Fix s to be the starting vertex. Let T be any spanning tree of G. The directed graph obtained from T by replacing each undirected edge with two directed edges is Eulerian; let $u_0, \ldots, u_{2(n-1)}$ be a Eulerian cycle in this graph (with $u_0 = u_{2(n-1)} = s$). It holds that $\{\langle u_0, u_1 \rangle, \ldots, \langle u_{2(n-1)-1}, u_{2(n-1)} \rangle\} = \{\langle u, v \rangle, \langle v, u \rangle : (u,v) \in T\}$, hence

$$\operatorname{cov}(G) \leq \sum_{i=1}^{2(n-1)} H(u_{i-1}, u_i) = \sum_{(u,v) \in T} H(u,v) + H(v,u) = \sum_{(u,v) \in T} \kappa(u,v) \leq (n-1) \cdot 2m$$

by Lemma 2.12. □

Can we also remove the logarithmic factor from the more general bound of Corollary 2.19? The answer is no, as demonstrated by the star graph of Example 2.18. Its diameter is 2 and thus $\operatorname{cov}(S_n) = \Omega(mD(S_n) \log n)$.

We state one more result, which is an interesting consequence of time-reversibility.

Proposition 2.21 (cf. Theorem 2.3 of [CRRS89]). *Let $v_0, \ldots, v_{k-1}, v_k \in V$ be a sequence of vertices with $v_0 = v_k$. Suppose we run a random walk from v_0 until it visits v_1, then v_2, ..., then v_{k-1}, then comes back to $v_k = v_0$. The expected length of this walk is*

$$\frac{1}{2} \sum_{i=0}^{k-1} \kappa(v_i, v_{i+1}).$$

Proof. The length (call it λ) is equal in expectation to the sum of consecutive hitting times:

$$\mathbb{E}[\lambda] = \sum_{i=0}^{k-1} H(v_i, v_{i+1}).$$

On the other hand, for any sequence of vertices α that begins and ends with v_0, the probability that the trajectory begins with α is the same as for the reversed sequence α^{R}. To prove this formally, notice that the random walk beginning in v_0 is the same as the stationary random walk X conditioned on the event that $X_0 = v_0$, and write

$$\mathbb{P}(X \sqsupseteq \alpha \mid X_0 = v_0) = \frac{\mathbb{P}(X \sqsupseteq \alpha)}{\mathbb{P}(X_0 = v_0)} = \frac{\mathbb{P}(X \sqsupseteq \alpha^{\mathrm{R}})}{\mathbb{P}(X_0 = v_0)} = \mathbb{P}(X \sqsupseteq \alpha^{\mathrm{R}} \mid X_0 = v_0)$$

where $\sqsupseteq$ means "begins with" and the middle equality follows from time-reversibility of a stationary walk (Proposition 2.3).

Obviously, such a sequence α contains $v_0, \ldots, v_k$ (in order) iff the reversed sequence contains $v_k, \ldots, v_0$. It follows that the expected time before a walk starting at $v_k = v_0$ visits v_{k-1}, then v_{k-2}, ..., then v_1, then comes back to v_0, is also $\mathbb{E}[\lambda]$ (this is easy to see, for example, from the equation $\mathbb{E}[\lambda] = \sum_{i=1}^{\infty} \mathbb{P}(\lambda \geq i)$). Thus

$$\mathbb{E}[\lambda] + \mathbb{E}[\lambda] = \sum_{i=0}^{k-1} H(v_i, v_{i+1}) + \sum_{i=0}^{k-1} H(v_{i+1}, v_i) = \sum_{i=0}^{k-1} \kappa(v_i, v_{i+1}).$$

□

2.5 Application: undirected *st*-connectivity in logarithmic space

As an application of the fact that the hitting times in an undirected graph are polynomially bounded, we briefly discuss a randomized algorithm ([AKL+79]) which, given an undirected graph G and two vertices s and t, decides whether s and t are connected by a path in G using only $\mathcal{O}(\log n)$ bits of memory.

We assume that we are given read-only access to a stored representation of G so that we are able to determine the vertices adjacent to a given vertex within the bounds of logarithmic space. The algorithm is very simple: we simulate a random walk started at the vertex s for $4mn$ steps. To do this, we only need to store a variable describing the current position of the walk and a counter of steps (whose size is $\mathcal{O}(\log(4mn)) = \mathcal{O}(\log n)$). We return YES if the walk visited the vertex t and NO otherwise.

If the two vertices are not connected, the algorithm with always return NO. Otherwise, the expected number of steps before t is visited is less than $2mn$ by Lemma 2.13, so the probability that t is not visited during a walk of length $4mn$ is less than $\frac{1}{2}$ by Markov's inequality.

The failure probability can be easily diminished by running the random walk longer: for example, if $4mn^2$ steps are allowed, we can bound it by 2^{-n}. (This follows because we can view the walk as having n phases of length $4mn$ each; the probability of not having found t after k phases is less than 2^{-k} by straightforward induction on k.)

It is not easy to construct a deterministic algorithm with the same space complexity: it was only accomplished 26 years later by Reingold [Rei05], and he has been awarded with the Gödel Prize in 2009 for this development. (Another well-known and simple algorithm that solves *st*-connectivity in sublinear space is due to Savitch [Sav70]. It is deterministic, but its space complexity is $\mathcal{O}(\log^2 n)$.)

3 Effective resistances and commute times

In this chapter we will introduce the notion of effective resistance and a tool called the projected graph. Together, they will allow us to provide another characterization of the commute time and obtain better bounds than those of the previous chapter, as well as reason about how a random walk behaves when we look only at its movement inside a (vertex-induced) subgraph of G. In particular, we give a new proof of the well-known identity $\kappa(u,v) = 2mR_{\text{eff}}(u,v)$ in this section (Proposition 3.12).

3.1 Definition of effective resistance

We will now introduce the notion of effective resistance between two vertices in a graph, a powerful tool in modern algorithmic and algebraic graph theory. Originally this notion arose in the context of studying electrical circuits in physics. We first explain its intuitive meaning, then introduce the formal definition using the Laplacian matrix. Our exposition follows [SS08] and [MST13]. For an alternative, more in-depth treatment, the reader may consult [Vis13].

In this section we find it convenient to provide definitions for weighted (undirected) graphs.

Let us consider an undirected graph with positive weights on edges as an electrical network, with a resistor of resistance $\frac{1}{w(u,v)}$ on edge $(u,v) \in E$. Then the effective resistance $R_{\text{eff}}(s,t)$ between two nodes s and t of this network is just the effective electrical resistance between them, i.e., if a current of 1 is injected at s and extracted at t, then $R_{\text{eff}}(s,t)$ is the difference of potentials (voltages) induced between the two nodes.

We will formally define $R_{\text{eff}}(s,t)$ using the concept of the Laplacian matrix associated with a graph. Let us first fix some notation. We define $d(v) = \sum_{(u,v)\in E} w(u,v)$ to be the weighted degree of v including self-loops, and $\hat{d}(v) = \sum_{(u,v)\in E, u\neq v} w(u,v)$ to be the weighted degree of v excluding self-loops.

Definition 3.1. *Let $G = (V,E,w)$ be a weighted undirected graph with n vertices. We define the Laplacian matrix $L = L(G) = (L_{uv})_{u,v\in V}$ associated with G to be an $n \times n$ matrix with real entries, given by the following formula for all $u,v \in V$:*

$$L_{uv} = \begin{cases} \hat{d}(u) & \text{if } u = v, \\ -w(u,v) & \text{if } u \neq v \text{ and } (u,v) \in E, \\ 0 & \text{if } u \neq v \text{ and } (u,v) \notin E. \end{cases}$$

In other words, L is the difference of a diagonal matrix with $d(v)$ on the diagonal minus the weight matrix of G. The Laplacian is a powerful concept with many beautiful properties; here we state only the bare minimum needed to define effective resistance using the matrix L.

Proposition 3.2. *L has the following properties (recall that G is connected):*

(1) the sum of every row (and every column) of L is 0,

(2) for every $x \in \mathbb{R}^V$, $\langle Lx, x\rangle = \sum_{(u,v)\in E} w(u,v)(x_u - x_v)^2 \geq 0$,

(3) the rank of L is $n-1$, the kernel of L being $\operatorname{span}\{\mathbb{1}\} = \{x \in \mathbb{R}^V : (\forall u,v \in V)\ x_u = x_v\}$,

(4) the image of L is $\mathbb{1}^\perp = \{x \in \mathbb{R}^V : \sum_v x_v = 0\}$.

Proof. (1) The sum of row u of L is equal to $\hat{d}(u) - \sum_{(u,v)\in E, v\neq u} w(u,v) = 0$, and L is symmetric.

(2) For an edge $e \in E$ let L^e be the Laplacian of a graph $(V,\{e\})$ on vertex set V with only one edge e (of weight 1). The matrix $L^{(u,v)}$ has nonzero entries only in rows and columns

numbered u and v, therefore $\left\langle L^{(u,v)}x, x\right\rangle = x_u^2 - x_u x_v - x_v x_u + x_v^2 = (x_u - x_v)^2$. Because $L = \sum_{e\in E} w(e)L^e$, we get

$$\langle Lx, x\rangle = \sum_{(u,v)\in E} w(u,v)\left\langle L^{(u,v)}x, x\right\rangle = \sum_{(u,v)\in E} w(u,v)(x_u - x_v)^2.$$

(3) It is clear that $\ker(L) \supseteq \operatorname{span}\{\mathbb{1}\}$: for all u, $(L\mathbb{1})_u = 0$ by (1). To prove the converse $\subseteq$, assume for a contradiction that $Lx = 0$ for a vector $x \in \mathbb{R}^V \setminus \operatorname{span}\{\mathbb{1}\}$. We may treat x as a function from vertices to reals. There exists a pair of vertices for which x assumes different values; since G is connected, there is a path that joins them, and x must change its value along this path at least once. Therefore there must exist an edge $(u, v) \in E$ such that $x_u \neq x_v$. By (2), this means that

$$0 = \langle 0, x\rangle = \langle Lx, x\rangle = \sum_{(u,v)\in E} w(u,v)(x_u - x_v)^2 > 0.$$

(4) The direction $\operatorname{im}(L) \subseteq \mathbb{1}^\perp$ is easy: for any $x \in \mathbb{R}^V$ we have

$$\langle \mathbb{1}, Lx\rangle = \sum_{u,v} L_{uv}x_v = \sum_v x_v \sum_u L_{uv} = \sum_v x_v \cdot 0 = 0$$

by (1). The converse $\supseteq$ now follows from (3). □

Definition 3.3. *Let $s \neq t \in V$ be two different vertices of G. Denote by $\chi_{st} \in \mathbb{R}^V$ a vector with -1 on s, 1 on t and 0 on the remaining coordinates. Suppose that $x \in \mathbb{R}^V$ is a vector satisfying*

$$Lx = \chi_{st}.$$

Then we define the effective resistance between s and t to be

$$R_{\text{eff}}(s,t) = \chi_{st}^{\mathrm{T}} x.$$

To establish the well-definedness of $R_{\text{eff}}(s,t)$, we must show that it does not depend on the choice of x, and that there always exists such an x. The latter follows from Proposition 3.2.(4); to establish the former, note that if $Lx = \chi_{st} = Ly$, then $L(x-y) = 0$, by Proposition 3.2.(3) we have $x - y \in \ker(L) = \operatorname{span}\{\mathbb{1}\}$, and $y = x + c\cdot\mathbb{1}$ for some $c \in \mathbb{R}$. Therefore $\chi_{st}^{\mathrm{T}} y = \chi_{st}^{\mathrm{T}} x + c - c = \chi_{st}^{\mathrm{T}} x$.

Technically, we are now done. However, we feel obligated to demonstrate that the effective resistance so defined really corresponds to the electrical resistance between two points in a network. The rest of this section is devoted mostly to this argument. In the following, we assume that the graph contains no self-loops (as they have no effect neither on the Laplacian nor on the electrical properties of the network) and skip most calculations on matrices.

Let us begin by fixing an arbitrary orientation of every edge in G, and by defining two more matrices associated with G: the *signed edge-vertex incidence matrix* B of size $m \times n$ and the diagonal *edge-weight matrix* W of size $m \times m$ as follows:

$$B_{ev} = \begin{cases} 1 & \text{if } e = \langle \cdot, v\rangle, \\ -1 & \text{if } e = \langle v, \cdot\rangle, \\ 0 & \text{otherwise,} \end{cases} \qquad W_{ef} = \begin{cases} w(e) & \text{if } e = f, \\ 0 & \text{otherwise.} \end{cases}$$

It may be verified that $L = B^{\mathrm{T}}WB$.

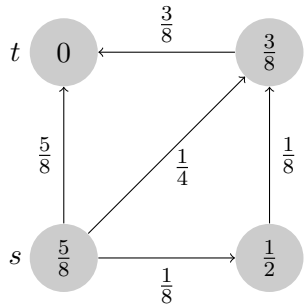

Figure 4: Potentials induced on vertices when unit current is pushed from s to t. The graph is undirected and unweighted: the arrows and labels on edges show the flow of electric current. The potential difference between s and t is $\frac{5}{8} = R_{\text{eff}}(s,t)$.

Now let $i_{\text{ext}} \in \mathbb{R}^V$ be the vector of currents injected (externally) at vertices (in our case, positive unit current will be injected at s and extracted at t, so that $i_{\text{ext}} = \chi_{st}$), $i \in \mathbb{R}^E$ be the vector of currents induced in edges, and $x \in \mathbb{R}^V$ be the vector of potentials induced at vertices.

First, by Kirchhoff's current law, at any node in a circuit, the signed sum of currents flowing into that node is equal to the (signed) current injected at that node (we count currents flowing out via edges as negative currents flowing in). This can be written in matrix notation as $B^{\mathrm{T}} i = i_{\text{ext}}$.

Second, Ohm's law implies that the current flow across an edge is equal to the potential difference between its endpoints divided by its resistance. This can be written as $i = WBx$. We thus get

$$\chi_{st} = i_{\text{ext}} = B^{\mathrm{T}} i = B^{\mathrm{T}} W B x = L x.$$

Now, the induced potential difference between s and t is just $x_s - x_t = \chi_{st}^{\mathrm{T}} x$. See Fig. 4 for an example.

Let us point out that the effective resistance between vertices s and t may be computed (to any required precision ε) by solving the Laplacian system $Lx = \chi_{st}$, which, due to recent breakthroughs, can be done in nearly-linear time. Currently, the simplest Laplacian solver with such runtime is [KOSZ13]; for a thorough treatment of this topic, refer to the monograph [Vis13]. We treat these techniques as a black box and register the following fact without proof.

Theorem 3.4. *If L is the Laplacian of an undirected weighted graph G with m edges, then for any vector $b \in \mathrm{im}(L) \subseteq \mathbb{R}^V$ and any $\varepsilon > 0$ we can find in $\widetilde{\mathcal{O}}(m \log \varepsilon^{-1})$ time a vector $v \in \mathbb{R}^V$ which is a multiplicative ε-approximation of some solution $x \in \mathbb{R}^V$ of $Lx = b$.*

Recall that we call v a multiplicative ε-approximation of x iff $(1-\varepsilon)x_u \le v_u \le (1+\varepsilon)x_u$ for all $u \in V$.

3.2 The projected graph and walks restricted to subgraphs

We will often find it worthwhile to reason about how a trajectory of a random walk looks like when restricted to a (vertex-induced) subgraph of G. More precisely, given a subset of vertices $V' \subseteq V$, we want to obtain a weighted graph $G[\![V']\!]$ on the vertex set V' such that trajectories of random walks in G, but with all vertices not from V' erased, correspond to full trajectories of random walks in $G[\![V']\!]$. We think of this erasure as a kind of projection $V^{\mathbb{N}} \to (V')^{\mathbb{N}}$, and we will call the graph $G[\![V']\!]$ a *projected graph*.

(a) The complete graph K_3 (b) The projected graph $K_3[\![\{1,2\}]\!]$

Figure 5: Projected graph example. Note that $R_{\text{eff}}(1,2) = \frac{2}{3}$ in both graphs.

This notion was first introduced implicitly in the proof of Lemma 6 in [KM09] and made explicit in [MST13] as a tool for showing Lemma 5.5 (which we give as Lemma 3.17). Our exposition in this section is based on [MST13]. We provide definitions for weighted graphs.

Definition 3.5. *For an undirected weighted graph $G = (V, E, w)$ and a nonempty subset of vertices $V' \subseteq V$ we define the* projected graph $G[\![V']\!] = (V', E', w')$ *as a weighted complete undirected graph on V' with $w'(u,v) = d(u)p(u,v)$, where $p(u,v)$ is the probability that v is the first vertex in V' which is hit by a random walk in G starting at u (after it leaves u in the beginning).*

See Fig. 5 for an example.

We first need to show that the weights on our (undirected) edges are well-defined, i.e., $w'(u,v) = w'(v,u)$. This can be seen as a corollary of time-reversibility (Proposition 2.3). In the proof, let us write $(V \setminus V')^*$ to denote the set of all finite sequences of vertices from $V \setminus V'$, and recall that $d(G) = \sum_{v \in V} d(v)$ is the sum of all degrees in G; the stationary distribution in G is $\pi(v) = \frac{d(v)}{d(G)}$. We then have

$$\begin{aligned}
w'(u,v) &= d(u)p(u,v) = d(G)\pi(u)p(u,v) \\
&= d(G) \sum_{\alpha \in (V \setminus V')^*} \mathbb{P}(X_0 = u, X_1 X_2 \dots X_{|\alpha|} = \alpha, X_{|\alpha|+1} = v) \\
&= d(G) \sum_{\alpha \in (V \setminus V')^*} \mathbb{P}(X_0 = v, X_{|\alpha|} \dots X_2 X_1 = \alpha, X_{|\alpha|+1} = u) \\
&= d(G) \sum_{\alpha \in (V \setminus V')^*} \mathbb{P}(X_0 = v, X_1 X_2 \dots X_{|\alpha|} = \alpha, X_{|\alpha|+1} = u) \\
&= d(G)\pi(v)p(v,u) = d(v)p(v,u) = w'(v,u).
\end{aligned}$$

The following statement, which follows directly from the definition, formalizes the idea behind the projected graph.

Proposition 3.6. *Fix a starting vertex $v \in V'$. Let X be the random walk in G starting from v and Y be the random walk in $G[\![V']\!]$ starting from v. Now consider a random sequence $\widetilde{X}$ obtained from X by* projecting *every trajectory to V'; formally, for any $\omega \in \Omega$ let $\widetilde{X}(\omega)$ be the sequence $X(\omega)$ with all vertices not from V' erased. Then $\widetilde{X}$ and Y are identically distributed.*□

Two helpful properties are readily apparent:

Corollary 3.7. *For $\emptyset \neq V'' \subseteq V' \subseteq V$ we have $(G[\![V']\!])[\![V'']\!] = G[\![V'']\!]$.* □

Fact 3.8. *For any $v \in V'$ we have $d^{G[\![V']\!]}(v) = d^G(v)$.*

Proof. We have $d^{G[\![V']\!]}(v) = \sum_{u \in V'} w'(v,u) = d^G(v) \sum_{u \in V'} p(v,u) = d^G(v)$. □

We are now in the position to prove a useful lemma (first given as Lemma 5.4 in [MST13]) which links the notions of the projected graph and of effective resistance.

Lemma 3.9 (projection preserves effective resistance). *Let $G = (V, E, w)$ be an undirected weighted graph, $V' \subseteq V$ be a subset of vertices and $G[\![V']\!]$ be the graph G projected with respect to V'. For any two vertices $u, v \in V'$ it holds that*

$$R_{\text{eff}}^{G[\![V']\!]}(u,v) = R_{\text{eff}}^{G}(u,v).$$

One can see this as a consequence of the fact that the Laplacian of the projected graph is the Schur complement of the block of the Laplacian of G that corresponds to the vertices (variables) $V \setminus V'$ that we eliminate (see e.g. Appendix C.4 in [BV04]). We give the details below.

Proof. Thanks to Corollary 3.7, we only need to prove the result for $V' = V \setminus \{u\}$ with $u \in V$ (which corresponds to eliminating one vertex from the graph). Once we have that, we can enumerate $V \setminus V' = \{u_1, \ldots, u_k\}$ and eliminate one vertex at a time:

$$R_{\text{eff}}^{G} = R_{\text{eff}}^{G[\![V\setminus\{u_1\}]\!]} = R_{\text{eff}}^{(G[\![V\setminus\{u_1\}]\!])[\![V\setminus\{u_1,u_2\}]\!]} = R_{\text{eff}}^{G[\![V\setminus\{u_1,u_2\}]\!]} = \ldots = R_{\text{eff}}^{G[\![V\setminus\{u_1,u_2,\ldots,u_k\}]\!]}.$$

(Formally, we proceed by induction on $|V \setminus V'|$.) So let us set $V' = V \setminus \{u\}$.

Denote $G' = G[\![V']\!]$ and let L, L' be Laplacians of G, G' respectively. It turns out that we can relate L' to L as follows: L' is the Schur complement of the block corresponding to $V \setminus V'$ in the matrix L. In our basic case of $V \setminus V' = \{u\}$ it means that, if we assume that the last row and column of L correspond to the vertex u and write

$$L = \begin{pmatrix} \overline{L} & b \\ b^{\mathrm{T}} & \hat{d}(u) \end{pmatrix}$$

(where $b_v = -w(v,u)$ for $v \in V'$), then $L' = \overline{L} - b(\hat{d}(u))^{-1} b^{\mathrm{T}}$. We prove this for the basic case.

Notice that for any $x, y \in V'$ (possibly $x = y$) we have $p(x,y) = \frac{w(x,y)}{d(x)} + \frac{w(x,u)}{d(x)} \cdot \frac{w(u,y)}{\hat{d}(u)}$ – it is the probability that y is the first vertex other than u that we visit when starting at x, and this happens if either we go directly $x \to y$, or if we go $x \to u$, loop $u \to u$ any number of times, and then go $u \to y$. Hence

$$w'(x,y) = p(x,y)d(x) = w(x,y) + \frac{w(x,u)w(u,y)}{\hat{d}(u)}.$$

First consider $x \neq y \in V'$. We have

$$L'_{xy} = -w'(x,y) = -w(x,y) - \frac{w(x,u)w(u,y)}{\hat{d}(u)} = \left(\overline{L} - \frac{bb^{\mathrm{T}}}{\hat{d}(u)}\right)_{xy}.$$

For the diagonal entries, by Fact 3.8 we get

$$\begin{aligned} L'_{xx} &= \hat{d}^{G'}(x) = d^{G'}(x) - w'(x,x) = d(x) - w'(x,x) \\ &= d(x) - w(x,x) - \frac{w(x,u)^2}{\hat{d}(u)} = \hat{d}(x) - \frac{w(x,u)^2}{\hat{d}(u)} = \left(\overline{L} - \frac{bb^{\mathrm{T}}}{\hat{d}(u)}\right)_{xx}. \end{aligned}$$

We have proved that $L' = \overline{L} - \frac{bb^{\mathrm{T}}}{\hat{d}(u)}$. Now fix vertices $s \neq t \in V'$ and let x be any solution of $Lx = \chi_{st}$. We take x' to be x restricted to V' (with the coordinate x_u dropped). We claim that x' is a solution of $L'x' = \chi'_{st}$. Once we prove this, we will be able to conclude that

$$R^{G'}_{\mathrm{eff}}(s,t) = (\chi'_{st})^{\mathrm{T}} x' = \chi_{st}^{\mathrm{T}} x = R^{G}_{\mathrm{eff}}(s,t).$$

We have

$$Lx = \begin{pmatrix} \overline{L} & b \\ b^{\mathrm{T}} & \hat{d}(u) \end{pmatrix} \begin{pmatrix} x' \\ x_u \end{pmatrix} = \begin{pmatrix} \chi'_{st} \\ 0 \end{pmatrix} = \chi_{st}.$$

The last equality of this system yields

$$x_u = -\frac{b^{\mathrm{T}} x'}{\hat{d}(u)}$$

and the other equalities yield

$$\chi'_{st} = \overline{L}x' + bx_u = \overline{L}x' - \frac{bb^{\mathrm{T}}x'}{\hat{d}(u)} = \left(\overline{L} - \frac{bb^{\mathrm{T}}}{\hat{d}(u)}\right) x' = L'x'.$$

□

3.3 Looping times and commute times

Recall from the discussion in Section 2.3 that in a stationary walk, the frequency with which a vertex u is visited is $\pi(u)$, and every edge is traversed in every direction with the same frequency, which is $\frac{1}{2m}$. We proved in Proposition 2.9 that the expected time between two visits to u is $\varepsilon(u) = \frac{1}{\pi(u)}$. It turns out that every edge is traversed in every direction with the same frequency during such a looping walk. Computing this rigorously will be our first example of an application of the projected graph.

Proposition 3.10. *Suppose we run a random walk from a vertex u until it returns to u, and let $(s,t) \in E$ be any edge of G. Then the expected number of traversals of this edge in the direction $s \to t$ is exactly $\frac{\varepsilon(u)}{2m} = \frac{1}{d(u)}$.*

Proof. If $u = s$, then the edge $u \to t$ may only be traversed during the first step, and the probability of this is $\frac{1}{d(u)}$. If $u = t$, then $s \to u$ may only be traversed during the last step, and the probability of this is also $\frac{1}{d(u)}$ (e.g. by time-reversibility). We may now assume that u, s, t are pairwise different.

We will obtain our result by writing down and solving a system of linear equations. For a vertex $v \in \{u, s, t\}$ let x_v be the expected remaining number of traversals of the edge $s \to t$ in G if we are currently in v. We are looking for the value of x_u.

As we are only interested in fragments of the random walk that touch u, s or t, we will set $V' = \{u, s, t\}$ and consider the projected graph $G[\![V']\!]$. Let w be the weight function of $G[\![V']\!]$. By Fact 3.8, $w(v,u) + w(v,s) + w(v,t) = d(v)$ for any $v \in V'$.

It is easy to construct a system of linear equations describing x_u, x_s, x_t. We must only concentrate on what happens after a visit in s. If we are at vertex s, then the probability that we will move to t in G is $\frac{1}{d(s)}$, and the probability that we will move to t in $G[\![V']\!]$ is $\frac{w(s,t)}{w(u,s)+w(s,s)+w(t,s)} = \frac{w(s,t)}{d(s)}$, of which $\frac{1}{d(s)}$ accounts for moving directly from s to t in G, and

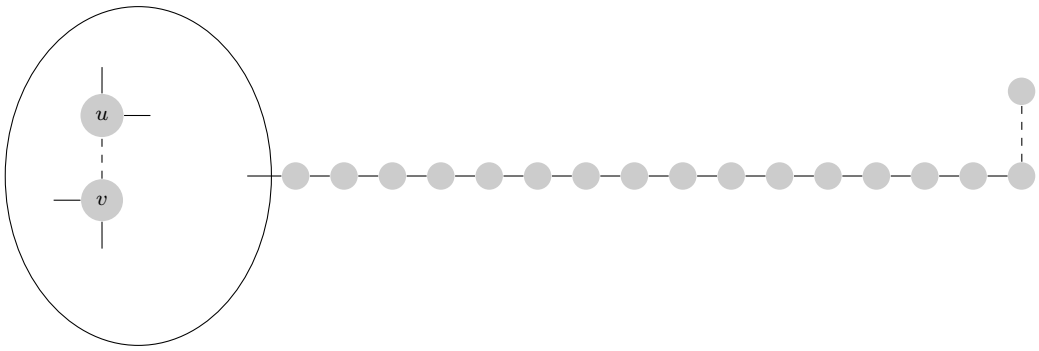

Figure 6: Both dashed edges attract the same amount of traffic on a walk from v to v.

the rest accounts for all possible trajectories of the form $s \to \alpha \to t$, where α is a nonempty sequence of vertices from $V \setminus V'$. We obtain

$$\begin{aligned} x_u &= \frac{w(u,u)\cdot 0 + w(u,s)\cdot x_s + w(u,t)\cdot x_t}{w(u,u)+w(u,s)+w(u,t)}, \\ x_t &= \frac{w(u,t)\cdot 0 + w(t,s)\cdot x_s + w(t,t)\cdot x_t}{w(u,t)+w(t,s)+w(t,t)}, \\ x_s &= \frac{w(u,s)\cdot 0 + w(s,s)\cdot x_s + (w(t,s)-1)\cdot x_t + 1\cdot(x_t+1)}{w(u,s)+w(s,s)+w(t,s)}, \end{aligned}$$

and solving this system for (x_u, x_s, x_t) yields $x_u = \frac{1}{w(u,u)+w(u,s)+w(u,t)} = \frac{1}{d(u)}$. □

Let us remark that this result may seem quite counterintuitive at first (this was certainly the case for the author). For instance, consider a path graph on 3500 vertices (cf. Example 2.17) and a walk from 1 to 1. During this walk, we expect to traverse the edge $(3499, 3500)$ as often as $(2, 3)$. As another example, picture a well-connected sparse graph (like an expander) with a very long path attached to it (see Fig. 6), and let v be a vertex inside the expander; the last edge of the path attracts as much traffic as any edge inside the expander during a walk from v to v.

In the following, we will compute the exact value of the commute time between any two vertices. We will witness the same phenomenon in this case (of every edge contributing the same amount to the expected length of a commute).

Proposition 3.11. *Suppose we run a random walk from a vertex u until it visits another vertex v and then comes back to u, and let $(s,t) \in E$ be any edge of G. Then the expected number of traversals of this edge in the direction $s \to t$ is exactly $R_{\text{eff}}(u,v)$.*

Proof. We proceed along the same lines as the proof of Proposition 3.10. The reader is advised to read it first.

Let us concentrate only on the case when u, v, s, t are pairwise different. Again, we consider the projected graph $G[\![V']\!]$ with $V' = \{u, v, s, t\}$ and weight function w and we build a system of linear equations. This time we will need to know, when visiting s or t, whether v has already been visited. To this end, we introduce the variable x_s^{bef} which denotes the expected remaining number of traversals of the edge $s \to t$ in G if we are currently in s and before our first visit in v, and x_s^{aft}, x_t^{bef}, x_t^{aft} with analogous meanings. We only need to do this for s and t: if we are at u, then v has not been visited yet (else we would finish), and if we are at v, then it is being

visited now. Using the same logic as in the previous proof regarding the traversals of the edge $s \to t$ in $G[\![V']\!]$, we obtain the following:

$$\begin{aligned}
x_u &= \frac{w(u,u)\cdot x_u + w(u,v)\cdot x_v + w(u,s)\cdot x_s^{bef} + w(u,t)\cdot x_t^{bef}}{w(u,u)+w(u,v)+w(u,s)+w(u,t)},\\
x_s^{bef} &= \frac{w(u,s)\cdot x_u + w(s,v)\cdot x_v + w(s,s)\cdot x_s^{bef} + (w(s,t)-1)\cdot x_t^{bef} + 1\cdot(1+x_t^{bef})}{w(u,s)+w(s,v)+w(s,s)+w(s,t)},\\
x_s^{aft} &= \frac{w(u,s)\cdot 0 + w(s,v)\cdot x_v + w(s,s)\cdot x_s^{aft} + (w(s,t)-1)\cdot x_t^{aft} + 1\cdot(1+x_t^{aft})}{w(u,s)+w(s,v)+w(s,s)+w(s,t)},\\
x_t^{bef} &= \frac{w(u,t)\cdot x_u + w(t,v)\cdot x_v + w(s,t)\cdot x_s^{bef} + w(t,t)\cdot x_t^{bef}}{w(u,t)+w(t,v)+w(s,t)+w(t,t)},\\
x_t^{aft} &= \frac{w(u,t)\cdot 0 + w(t,v)\cdot x_v + w(s,t)\cdot x_s^{aft} + w(t,t)\cdot x_t^{aft}}{w(u,t)+w(t,v)+w(s,t)+w(t,t)},\\
x_v &= \frac{w(u,v)\cdot 0 + w(v,v)\cdot x_v + w(s,v)\cdot x_s^{aft} + w(t,v)\cdot x_t^{aft}}{w(u,v)+w(v,v)+w(s,v)+w(t,v)}.
\end{aligned}$$

On the other hand, by solving the Laplacian linear system $Ly = \chi_{uv}$, with L being the Laplacian of $G[\![V']\!]$, and computing $\chi_{st}^{\mathrm{T}} y$ (as in Definition 3.3) we obtain an expression for $R_{\mathrm{eff}}^{G[\![V']\!]}(u,v)$ depending on the values of the weight function w. Now, it may be verified that the expression for x_u computed by solving the above 6×6 system is the same, hence the expected number of traversals of the edge $s \to t$ in G is exactly $R_{\mathrm{eff}}^{G[\![V']\!]}(u,v)$. As $V' = \{u,v,s,t\}$, it may seem that this depends on s or t; however, this is where Lemma 3.9 comes into play. Applying it yields

$$x_u = R_{\mathrm{eff}}^{G[\![V']\!]}(u,v) = R_{\mathrm{eff}}^{G}(u,v). \qquad \square$$

This provides a new proof, without use of renewal theory (see Section 4) or electrical flows, of the following well-known statement:

Proposition 3.12 (Theorem 2.1 of [CRRS89]). *For any two distinct vertices $u, v \in V$ we have*

$$\kappa(u,v) = 2mR_{\mathrm{eff}}(u,v).$$

(If G is weighted, replace $2m$ with $d(G)$.) $\square$

Note that, unlike Lemma 2.13, this provides us with an exact value of $\kappa(u,v)$. By combining these two results we also get $\frac{\kappa(u,v)}{2m} = R_{\mathrm{eff}}(u,v) \le \mathrm{dist}(u,v)$ – an inequality which is clear if one considers the intuitive meaning of $R_{\mathrm{eff}}(u,v)$ as electrical resistance (a shortest path connecting u and v has resistance $\mathrm{dist}(u,v)$).

Let us provide one more interesting corollary.

Corollary 3.13. *The function $R_{\mathrm{eff}} : V \times V \to \mathbb{R}_{\ge 0}$ is a metric on V. (So is κ.)* $\square$

This *effective resistance metric* can in some cases provide deeper insight into the graph's structure than the usual graph-distance metric. In our case, it provides a better (and almost tight) bound on the cover time on the graph.

Definition 3.14. *For a graph G let us define its* effective resistance diameter $\gamma_{\mathrm{eff}}(G)$ *to be the maximum effective resistance between two vertices of G, i.e., $\gamma_{\mathrm{eff}}(G) = \max_{u,v\in V} R_{\mathrm{eff}}(u,v)$.*

Because $R_{\mathrm{eff}}(u,v) \le \mathrm{dist}(u,v)$, we have $\gamma_{\mathrm{eff}}(G) \le D(G)$. The following bound is therefore a strengthening of Corollary 2.19.

Corollary 3.15 (Theorem 2.4 of [CRRS89]). *For any G we have* $\mathrm{cov}(G) \leq 4m\gamma_{\mathrm{eff}}(G)\log n$. *(If G is weighted, replace $2m$ with $d(G)$.)*

Proof. Use Proposition 3.12 and Corollary 2.16 to get

$$\mathrm{cov}(G) \leq 2\log n \max_{u,v\in V} \kappa(u,v) = 2\log n \cdot 2m \cdot \max_{u,v\in V} R_{\mathrm{eff}}(u,v). \qquad \square$$

Note that by Proposition 3.12 and Corollary 2.16 we also get (cf. Lemma 2.2 of [MST13]):

$$\mathrm{cov}(G) \geq \frac{1}{2} \max_{u,v\in V} \kappa(u,v) = m\gamma_{\mathrm{eff}}(G).$$

This means that the gap between the upper and the lower bound is only $4\log n$. Because these bounds are obtained from Corollary 2.16 using the equality $\max_{u,v\in V} \kappa(u,v) = 2m\gamma_{\mathrm{eff}}(G)$, they are still attained up to a constant by the the star graph (see Example 2.18) and the path graph (see Example 2.17), respectively.

Refer to the introductory section of [CRRS89] for a broader discussion of these results.

3.4 Covering a subgraph

To motivate the results of this section, let us consider an algorithm that simulates a random walk in a graph G, and consider the subgraph of G induced by a subset of vertices $V' \subseteq V$. We are told that once it is covered, the algorithm will have a way of "skipping over it" during the rest of the simulation. Before this happens, the simulation will be carried out normally, and we will pay 1 for every step that the random walk takes inside V'. (We do not care about steps taken outside of V' at all, as we are going to account for them differently.) We are interested in the expected cost of the described procedure.

So, suppose we run a random walk in G until all vertices from a subset $V' \subseteq V$ have been visited. We are interested in upper-bounding the number of traversals of edges from $E(V')$ (edges connecting two points in V'). There are two intuitive reasons why we should be able to get a better bound than the general cover time of the graph, which is $\mathcal{O}(m \cdot \gamma_{\mathrm{eff}}(G) \cdot \log n)$: first, we only wait until covering V', not V, so the total length of the walk should be lower if, for example, V' is a cloud of vertices which are close to each other; second, we only pay for traversing edges inside V'.

We can capture the notion of such a cloud by the following natural extension of Definition 3.14.

Definition 3.16. *For a graph G and a subset of its vertices $V' \subseteq V$ we define its* effective resistance diameter $\gamma_{\mathrm{eff}}(V') = \max_{u,v\in V'} R_{\mathrm{eff}}(u,v)$ *to be the maximum effective resistance between two vertices in V'.*

Of course we have $\gamma_{\mathrm{eff}}(G) = \gamma_{\mathrm{eff}}(V)$. Now, it turns out that we can, indeed, replace $\gamma_{\mathrm{eff}}(G)$ in the bound with $\gamma_{\mathrm{eff}}(V')$, which is no larger, and in some cases (e.g., in the algorithms whose analyses use this lemma) substantially smaller. Also, we are able to replace m with the number of edges that have at least one endpoint in V'.

The algorithm sketched above is actually a subroutine of the Kelner-Mądry algorithm for sampling random spanning trees from the uniform distribution [KM09], and proving a variation of the following lemma was the original motivation for introducing the projected graph in both [KM09] and [MST13].

It is perhaps worth noting that the lemma does not follow directly from Corollary 3.15 applied to the subgraph induced by V', because that would work only for a natural random walk in the subgraph, not the (different) random walk induced in the subgraph by the global random walk in G. This is exactly the reason why we utilize the projected graph.

Lemma 3.17 (Lemma 5.5 of [MST13]). *Let $G = (V, E)$ be an unweighted graph and $V' \subseteq V$. Suppose we run a random walk in G from an arbitrary vertex (not necessarily from V') and stop it once all vertices from V' have been visited. Then the expected number of traversals of edges from the set $E(V')$ is*

$$\mathcal{O}((|E(V')| + |\partial V'|) \cdot \gamma_{\text{eff}}(V') \cdot \log n),$$

where $\partial V'$ is the set of edges with exactly one endpoint in V'.

Proof. Let $G[\![V']\!]$ be the projected graph with respect to V'. It follows from Fact 3.8 that

$$d(G[\![V']\!]) = \sum_{v \in V'} d^{G[\![V']\!]}(v) = \sum_{v \in V'} d^G(v) = 2 \cdot |E(V')| + |\partial V'|.$$

Also, by Lemma 3.9 the effective resistances between nodes in V' are the same in $G[\![V']\!]$ as they are in G; in particular, $\gamma_{\text{eff}}^{G[\![V']\!]}(V') = \gamma_{\text{eff}}^{G}(V')$.

Suppose we start the walk in $v \in V$. After some time, during which we do not pay for edge traversals, the walk will hit a vertex of V'. Starting from that moment we can view our walk as a walk in $G[\![V']\!]$ (since we do not care about edges outside of V') and pay for all traversals until V' is covered. Note that this way, if we do not pay for traversing an edge of G, it does not belong to $E(V')$. By Corollary 3.15 we get a bound of

$$\mathcal{O}(d(G[\![V']\!]) \cdot \gamma_{\text{eff}}^{G[\![V']\!]}(V') \cdot \log |V'|) = \mathcal{O}((|E(V')| + |\partial V'|) \cdot \gamma_{\text{eff}}(V') \cdot \log n). \qquad \square$$

Note that we end up "overpaying" a little. We do pay for all traversals of edges from $E(V')$, as we should. However, whenever the edge traversal that we make in $G[\![V']\!]$ does not correspond to traversing the same edge in G, but rather arises from eliminating vertices not from V' from a trajectory of the form $v_1 \to \alpha \to v_2$, where $v_1, v_2 \in V'$ and α is a nonempty sequence of vertices from $V \setminus V'$, then in our proof we pay for one edge traversal, whereas in "reality" we do not. This prompts us to ask whether it is possible to get $|E(V')|$ (which we intuitively expect) in the bound instead of $|E(V')| + |\partial V'|$. The answer is yes – we will show it as Lemma 4.13, but using a different perspective.

4 Uniformity of expectation and phase divisions

The reader will recall that in Section 3.3 we proved two statements regarding the expected number of traversals of any edge e, in any direction, during a loop $u \to u$ or a commute $u \to v \to u$ (Propositions 3.10 and 3.11). We witnessed a surprising phenomenon: this quantity did not at all depend on the edge e (nor on the direction). This behavior – a *uniformity of expectation* – is what we would expect of a stationary walk: at any fixed step of such a walk, the probability of traversing e in any direction (which is $\frac{1}{2m}$) is also independent of e. This cannot be a coincidence!

Indeed, it is not. To better understand the relationship between a stationary walk and, say, a loop from u to u, let us now consider the following intuition. We can separate every trajectory of a stationary walk into phases by cutting it whenever a visit in u occurs. This means that, in terms of distribution, the random walk is a union (concatenation) of an initial segment (whose expected length is bounded) and an infinite sequence of trajectories of a loop $u \to \ldots \to u$. Every such loop is identically distributed, and they are all independent (we say that it is an i.i.d. sequence).

Now, if it were the case that during such a loop, some edge e_1 is traversed more often on average than another edge e_2, then, as the stationary walk can be *almost* partitioned into such loops, this disparity would transfer to the entire stationary walk (a contradiction).

For this reason, we also expect this uniformity to appear in any walk Y such that the stationary walk can be almost partitioned into i.i.d. copies of Y (*phases*). Our second example – the commute $u \to v \to u$ – confirms this prediction. As we will see, there are many other useful examples of such *phase divisions*.

A reader who is content with this argumentation may skip directly to Definition 4.2 and Theorem 4.4. Otherwise, we make it precise and give a rigorous proof in the following section.

We then use this *uniformity of expectation principle* in subsequent sections to obtain results that deepen our understanding of the behavior of the random walk. In particular, we show how to prove results related to the analysis of algorithms [KM09] and [MST13] – one of them in a stronger form than before – using this newfound understanding (and without the use of the projected graph).

4.1 The uniformity principle

Our reasoning may be seen as a special case of results in *renewal theory*, a branch of probability (see e.g. Chapter 7 in [Ros09]). Because of this specialization, we are able to give a more accessible proof than those of the general renewal theorems. We use selected arguments from [Ros09] in the proofs below.

We now need to fix some notation and conventions. In this chapter we will allow random walks to be finite in length, i.e., we may define a random walk to be a prefix of the natural random walk which ends at some random time (for example, once the walk returns to the starting vertex). Also, we will find it more comfortable to think of random walks as sequences of traversed edges, not visited vertices.

Let $u \in V$ be a fixed starting vertex and $e \in E$ a fixed edge of a graph G. For the purposes of this section we define the following:

- U – the random walk started at u,
- X – the stationary random walk,
- P – the random walk that starts from the stationary distribution and ends as soon as u is visited,
- Y_1Y_2, where Y_1 and Y_2 are random walks – a concatenation of the two walks, i.e., a trajectory of Y_1Y_2 is obtained from trajectories of Y_1 and Y_2 by concatenating them (remember that we opt to think of trajectories as sequences of edges),
- p – the steady-state (stationary) probability of traversing the edge e (making an *e-traversal*) in a single step (equal to $\frac{1}{m}$),
- $A_Y(t)$, where Y is a random walk and $t \in \mathbb{N}$ – the number of e-traversals during the first t steps of the random walk Y.

We make the following preliminary remarks:

- the random walks U and X are infinite, and P is (almost surely) finite,
- the random walks X and PU are identically distributed; for simplicity, in the proofs below we will assume that actually $PU = X$ for every possible trajectory,
- for any $t \in \mathbb{N}$ we have $\mathbb{E}[A_X(t)] = pt$, so clearly $\lim_{t\to\infty} \frac{\mathbb{E}[A_X(t)]}{t} = p$.

As t tends to infinity, we expect that the part played in $X = PU$ by P, whose expected length $\mathbb{E}[|P|]$ is bounded, will diminish. Therefore the long-term frequency of traversing the edge e in U should be the same as in $PU = X$. This is proved in the following lemma.

Lemma 4.1. *We have*

$$\lim_{t\to\infty} \frac{\mathbb{E}[A_U(t)]}{t} = p.$$

Proof. Fix $t \in \mathbb{N}$. Our plan is to obtain upper and lower bounds on $\mathbb{E}[A_U(t)]$. We begin with the upper bound.

Let us first notice that $\mathbb{E}[A_U(t)] \leq \mathbb{E}[A_X(|P|+t)]$. Indeed, the number of e-traversals during the first t steps of U is the same as the number of e-traversals during the first t steps of $X = PU$ taken after the first visit to u. But the latter is upper-bounded by the number of e-traversals during the first $|P| + t$ steps of the stationary walk X.

Now to bound $\mathbb{E}[A_X(|P| + t)]$, we notice that the initial segment of X of length $|P| + t$ can be partitioned into a stationary walk of fixed length t and a remainder of length $|P|$. Note that $A_X(|P|+t)-A_X(t)$ is the number of e-traversals during the steps of X numbered $t+1, \ldots, t+|P|$, which can be bounded by just $|P|$. We thus get

$$\begin{aligned}\mathbb{E}[A_U(t)] &\leq \mathbb{E}[A_X(|P| + t)] \\ &= \mathbb{E}[A_X(t)] + \mathbb{E}[A_X(|P| + t) - A_X(t)] \\ &\leq pt + \mathbb{E}[|P|].\end{aligned}$$

Now we turn to the lower bound. Here we use the decomposition $X = PU$ to obtain

$$\begin{aligned}\mathbb{E}[A_U(t)] &= \mathbb{E}[A_X(|P| + t) - A_X(|P|)] \\ &= \mathbb{E}[A_X(|P| + t)] - \mathbb{E}[A_X(|P|)] \\ &\geq \mathbb{E}[A_X(t)] - \mathbb{E}[|P|] \\ &= pt - \mathbb{E}[|P|].\end{aligned}$$

We now have

$$\frac{pt + \mathbb{E}[|P|]}{t} \geq \frac{\mathbb{E}[A_U(t)]}{t} \geq \frac{pt - \mathbb{E}[|P|]}{t}.$$

The left and right terms tend to p as $t \to \infty$. □

We will now wish to divide the infinite random walk U beginning at u into *phases* $U_1, U_2, \ldots$. An example of such a division would be to finish each U_i when u is revisited, thus obtaining a decomposition of the random walk U into loops $u \to u$, each of which has an expected length of $\varepsilon(u)$. A different example would be to wait until another fixed vertex v and then u are visited, thus yielding phases of expected length $\kappa(u, v)$.

The only thing that we require from such a phase division is that it provide a decision procedure that tells us when to finish a phase. We will only allow finishing a phase when the walk is currently in u. Also, for simplicity we will require that this decision procedure be not random, i.e., it must be a deterministic function of the trajectory of the current phase up to the current step. Finishing the phase once u is revisited, or once v is visited and then u is revisited, is such a decision procedure.

This motivates the following definition:

Definition 4.2. *Let U be the infinite random walk starting at u, and let $U_1, U_2, \ldots$ be an infinite sequence of random walks. Let $T_i = |U_1| + \ldots + |U_i|$ for $i \in \mathbb{N}$. We say that $(U_i)_{i\geq 1}$ is a* phase division *of U if it satisfies the following:*

- *there exists a function* $f : (V \times V)^* \to \{0, 1\}$ *such that* $f(\alpha) = 1$ *only if* α *ends with a traversal* $\langle \cdot, u \rangle$,
- *each* U_i *is the following random walk: it begins at* u *and proceeds by copying the consecutive steps of* U*, beginning with step number* $T_{i-1}+1$*, until it becomes true that* $f(\alpha) = 1$*, where* α *is the trajectory of* U_i *up to that point; then it ends,*
- *the expected length of one phase is finite:* $\mathbb{E}[T_1] < \infty$.

It is clear from the definition that $U = U_1U_2U_3...$. Because of the Markovian property of random walks, it holds that $U_2, U_3, \ldots$ are independent and identically distributed copies of U_1. The values $0 = T_0, T_1, T_2, \ldots$ are the times when phases begin and end.

For $t \in \mathbb{N}$, let us denote by $N(t)$ the number of phases that have ended until time t, i.e.,

$$N(t) = \max\{k : t \geq T_k\}.$$

The long-run behavior of the quantity $N(t)$ is described by the following lemma.

Lemma 4.3 (Proposition 7.1 of [Ros09]). *With probability* 1,

$$\lim_{t\to\infty} \frac{N(t)}{t} = \frac{1}{\mathbb{E}[T_1]}.$$

Proof. We have

$$T_{N(t)} \leq t < T_{N(t)+1}$$

and

$$\frac{T_{N(t)}}{N(t)} \leq \frac{t}{N(t)} < \frac{T_{N(t)+1}}{N(t)}.$$

As for the left-hand term, since $\frac{T_{N(t)}}{N(t)} = \frac{|U_1|+...+|U_{N(t)}|}{N(t)}$ is the average of i.i.d. variables $|U_i|$, and $N(t) \to \infty$ as $t \to \infty$, the strong law of large numbers implies that

$$\lim_{t\to\infty} \frac{T_{N(t)}}{N(t)} = \mathbb{E}[T_1]$$

with probability 1.

As for the right-hand term, we can write

$$\frac{T_{N(t)+1}}{N(t)} = \frac{T_{N(t)+1}}{N(t)+1} \cdot \frac{N(t)+1}{N(t)}$$

and because $\lim_{t\to\infty} \frac{N(t)+1}{N(t)} = 1$, we get the same limit by the same argument. The squeeze lemma implies that $\lim_{t\to\infty} \frac{t}{N(t)} = \mathbb{E}[T_1]$. □

Let A_i be the number of e-traversals during the phase U_i. The sequence $(A_i)_{i\geq 1}$ is i.i.d.

We are now ready to state and prove the main result (cf. Propositions 7.3 and 7.4 of [Ros09]).

Theorem 4.4 (uniformity principle). *Let* $(U_i)_{i\geq 1}$ *be a phase division of* U*. The expected number of* e*-traversals during one phase* U_1 *is equal to the steady-state probability of a traversal times the expected length of one phase, i.e.,*

$$\frac{\mathbb{E}[A_1]}{\mathbb{E}[T_1]} = p.$$

Proof. Let $A(t) = A_U(t)$ be the number of e-traversals up to time t, and let $B(t)$ be the number of e-traversals during all phases finished up to time t, i.e.,

$$B(t) = A_1 + A_2 + \ldots + A_{N(t)}.$$

The proof will proceed in three steps.

Claim 1. *With probability one,* $\lim_{t\to\infty} \frac{B(t)}{t} = \frac{\mathbb{E}[A_1]}{\mathbb{E}[T_1]}$.

We write

$$\frac{B(t)}{t} = \frac{A_1 + \ldots + A_{N(t)}}{t} = \frac{A_1 + \ldots + A_{N(t)}}{N(t)} \cdot \frac{N(t)}{t}$$

and note that the first factor tends to $\mathbb{E}[A_1]$ as $t \to \infty$ (and $N(t) \to \infty$) by the strong law of large numbers, while the second factor tends to $\frac{1}{\mathbb{E}[T_1]}$ by Lemma 4.3.

Claim 2. $\lim_{t\to\infty} \frac{\mathbb{E}[B(t)]}{t} = \frac{\mathbb{E}[A_1]}{\mathbb{E}[T_1]}$.

This follows from Claim 1 and the dominated convergence theorem, as $\left|\frac{B(t)}{t}\right| \leq 1$.

Claim 3. $\lim_{t\to\infty} \frac{\mathbb{E}[A(t)]}{t} = \frac{\mathbb{E}[A_1]}{\mathbb{E}[T_1]}$.

The difference between $A(t)$ and $B(t)$ are traversals, up to time t, during the $(N(t)+1)$-th phase. We can thus upper-bound $\mathbb{E}[A(t) - B(t)]$ by just $\mathbb{E}[|U_{N(t)+1}|] = \mathbb{E}[T_1]$, which is finite. Therefore

$$\frac{\mathbb{E}[B(t)]}{t} \leq \frac{\mathbb{E}[A(t)]}{t} \leq \frac{\mathbb{E}[B(t)] + \mathbb{E}[T_1]}{t}$$

and the squeeze lemma together with Claim 2 prove the claim.

The statement now follows from Lemma 4.1 and Claim 3. □

4.2 Applications

We can now easily obtain results such as the following two corollaries. They were proved earlier in the thesis using the projected graph; now we reprove them without using that tool.

Corollary 4.5 (Proposition 3.10). *Suppose we run a random walk from a vertex u until it returns to u, and let $(s,t) \in E$ be any edge of G. Then the expected number of traversals of this edge in the direction $s \to t$ is exactly $\frac{\varepsilon(u)}{2m} = \frac{1}{d(u)}$.*

Proof. We consider a phase division where a phase ends whenever u is visited. By Proposition 2.9 and the uniformity principle we get that the expected number of traversals during a single phase is $\varepsilon(u) \cdot \frac{1}{2m}$. □

Corollary 4.6 (cf. Proposition 3.11). *Suppose we run a random walk from a vertex u until it visits another vertex v and then comes back to u. Then the expected number of traversals of any edge in any direction is the same.*

Proof. We consider a phase division where a phase ends whenever u is visited after v has already been visited. By the uniformity principle we get that the expected number of traversals during a single phase is $\kappa(u,v) \cdot \frac{1}{2m}$. □

Note that, unlike Proposition 3.11, this corollary does not tell us that this expected number of traversals is $R_{\text{eff}}(u,v)$, unless we happen to know *a priori* that $\kappa(u,v) = 2m \cdot R_{\text{eff}}(u,v)$. We actually derived this in Proposition 3.12 as a consequence of Proposition 3.11, but most proofs of this identity do not follow this approach (cf. Theorem 4.1 of [Lov93]), and could be combined with Corollary 4.6 to obtain Proposition 3.11 without use of the projected graph.

Another application of the uniformity principle is the following short proof of Lemma 2.12 given by [AKL$^+$79].

Corollary 4.7 (Lemma 2.12). *Let $(u,v) \in E$ be an edge. Then $\kappa(u,v) \leq 2m$. Moreover, there is equality iff (u,v) is a bridge in G.*

Proof. Consider the random walk started at u which ends when u is visited after v has already been visited. By Corollary 4.6, the expected number of traversals of the edge (u,v) in the direction $u \to v$ during this walk is exactly $\frac{\kappa(u,v)}{2m}$. On the other hand, it is easy to see that this number of traversals is always at most 1, and that its expectation is exactly 1 iff (u,v) is a bridge in G. □

What can we say about the uniformity of expectation between two traversals of an edge (u,v) in the direction $u \to v$? Because a traversal $u \to v$ is an event which can be used to split a trajectory of a walk starting at v into i.i.d. phases, we can indeed show the following:

Proposition 4.8. *The expected number of traversals of any edge in any direction between two consecutive traversals of an edge (u,v) in the direction $u \to v$ is 1.*

Proof. Consider a random walk started at v and its phase division where a phase ends after a $u \to v$ traversal. By the uniformity principle, the expected number of traversals of any edge in any direction during one phase is the same. And we know that the number of traversals of the edge $u \to v$ is always exactly one (since it ends the phase). □

We can interpret this result as follows: if we were forced to "derandomize" some initial segment of the random walk starting at a vertex v, then a Eulerian tour of the digraph $\vec{G}$ (obtained from G by replacing each undirected edge with two directed arcs), which has length $2m$ and passes through every edge in every direction exactly once, would be a good approximation in some respects. Indeed, this is interestingly reflected in the behavior of the *rotor-router mechanism*, a deterministic alternative to the notion of random walk in undirected graphs. In this model, each vertex v maintains a list of its neighbors. One step of a walk which is currently at v involves choosing the first vertex on v's list and moving to it; the entry corresponding to that vertex is then moved to the end of v's list. It turns out [YWB03] that the cover time of G in the rotor-router model is $\Theta(m \cdot D(G))$, and after this time the walk will have stabilized to traversing a fixed Eulerian tour of $\vec{G}$.

Let us give one more short application of Theorem 4.4. Proposition 3.11 may be interpreted as saying that any edge expects a traffic of $R_{\text{eff}}(u,v) + R_{\text{eff}}(v,u)$ during a $u \to v \to u$ commute. We can now show that this formulation extends also to longer cyclical commutes.

Corollary 4.9 (Theorem 2.3 of [CRRS89]). *Let $v_0, \ldots, v_{k-1}, v_k \in V$ be a sequence of vertices with $v_0 = v_k$. Suppose we run a random walk from v_0 until it visits v_1, then v_2, ..., then v_{k-1}, then comes back to $v_k = v_0$. The expected number of traversals of any edge is*

$$\sum_{i=0}^{k-1} R_{\text{eff}}(v_i, v_{i+1}).$$

Proof. Consider the following phase division: end a phase once vertices $v_0, \dots, v_k = v_0$ have been visited (in that order), and apply the uniformity principle to Proposition 2.21, obtaining

$$\frac{1}{m} \cdot \frac{1}{2} \sum_{i=0}^{k-1} \kappa(v_i, v_{i+1}) = \frac{1}{2m} \sum_{i=0}^{k-1} 2mR_{\text{eff}}(v_i, v_{i+1})$$

by Proposition 3.12. □

4.3 Edge traversals until the cover time

We have seen that the uniformity principle applies to walks from u to u and from u to v to u, whose expected lengths (looping times and commute times) serve as important quantitative parameters of the random walk structure of G. What can we say about the cover time in this regard?

Let us start with the negatives. It is not true for any vertex v that the expected number of traversals of an edge e during a covering random walk started in v is independent of e; for instance, if we run a walk on the path graph on n vertices (cf. Example 2.17) from vertex 1, the graph will be covered once vertex n is visited, and so the number of traversals of edge $(n-1, n)$ will be 1, whereas for edge $(1, 2)$ its expectation is $2n - 3$. (This is because the expected traffic on edge $(1, 2)$ during a commute $1 \to n \to 1$ is $2(n-1)$ by Proposition 3.11, and it would be traversed exactly once on the way from n to 1.) We can thus see that the uniformity phenomenon fails to take place here.

Also, recall that if we run a walk from the stationary distribution for t steps, where t is a constant, then any edge will be traversed $\frac{t}{m}$ times in expectation. But it is of paramount importance here that t be a constant. In particular, if we take t to be the (random) time when G is covered, then it is not true that any edge has the same expected number of traversals. For example, if we consider the path graph on 4 vertices, then it is easy to compute that this expectation is $\frac{10}{3}$ for the edge $(1, 2)$ and $\frac{11}{3}$ for the edge $(2, 3)$. Much greater discrepancies may be observed in a graph created by joining a clique of size $\frac{n}{2}$ with a path of the same size: the last vertex on the path is usually the last to be covered, and thus its only adjacent edge is usually only traversed once, whereas the cover time of this graph is $\Omega(n^3)$ (as $m = \Theta(n^2)$ and $\gamma_{\text{eff}} = \Theta(n)$).

Fortunately, we can obtain a positive result here. Note that the expected length of a walk which starts from a vertex s, covers G, and then returns to s can not be much higher than the cover time of G; indeed, the additional time it takes to return to s is bounded by $\max_{v \in V} H(v, s) \leq \text{cov}(G)$, thus the entire walk takes at most $2\,\text{cov}(G)$ expected time. Furthermore, this walk can now be used to define a phase division and invoke the uniformity principle, obtaining the following:

Proposition 4.10. *The expected number of traversals of any edge during a covering random walk is at most* $\frac{2}{m}\,\text{cov}(G)$. □

If we combine this result with Corollary 3.15, we get the following lemma, which along with Lemma 3.17 provides the basis for run-time analysis of the algorithm [MST13]:

Lemma 4.11 (Lemma 5.6 of [MST13]). *Suppose we run a random walk from any starting vertex $s \in V$ until G is covered. Let $E' \subseteq E$ be any subset of edges. The expected number of traversals of edges from E' is*

$$\mathcal{O}\left(\frac{|E'|}{m}\,\text{cov}(G)\right) = \mathcal{O}(|E'| \cdot \gamma_{\text{eff}}(G) \cdot \log n).$$

□

And, if we use Lemma 2.20 instead, we obtain a bound of $\mathcal{O}(|E'| \cdot n)$, which is a slightly stronger version of Fact 5 of [KM09] (cf. also Lemma 7.3). Let us remark that the original proof relied on the misconception that any edge would be traversed the same expected number of times during a stationary walk which stops once G is covered, which we disproved above. This is thus the first correct proof of Fact 5.

4.4 Covering a subgraph again

Let us now recall Section 3.4, where we proved Lemma 3.17 – the other result crucial in the analysis of the random spanning tree algorithms [KM09] and [MST13] – using the projected graph. Now we will present a different proof of this fact, which arguably gives more insight into the behavior of the walk and does not use the notion of projected graph (nor Lemma 3.9). We start with a lemma.

Lemma 4.12. *Let $G = (V, E)$ be an unweighted graph and $V' \subseteq V$. Suppose we run a random walk in G from a vertex $v \in V'$ and stop it on the first visit to v after all vertices from V' have been visited. Then the expected number of steps is*

$$\mathcal{O}(m \cdot \gamma_{\text{eff}}(V') \cdot \log n).$$

Proof. We mimic the proof of Theorem 2.15, with necessary adjustments. We divide the walk into phases (not in the sense of Definition 4.2). The first one ends when more than half of vertices from V' have been visited. By Lemma 2.14 (with $s = v$) we know that its expected length is at most

$$2 \max_{u \in V'} H(v, u) \leq 2 \max_{u \in V'} \kappa(v, u) = 4m \max_{u \in V'} R_{\text{eff}}(v, u) = \mathcal{O}(m \gamma_{\text{eff}}(V'))$$

(we used Proposition 3.12). The second phase ends when at least half of the remaining vertices of V' have been covered, and the lemma applied to that set of vertices (and the starting vertex s being the vertex where the first phase ended) again bounds the expected length of the second phase by the same quantity (it is important here that $s \in V'$, as the first phase must have ended with a visit to a vertex of V'). We proceed in this fashion until V' is covered. By linearity of expectation, the expected length of the walk up to this point is $\mathcal{O}(m \gamma_{\text{eff}}(V'))$ times the number of phases, which is clearly $\log |V'| = \mathcal{O}(\log n)$. It remains to come back to v from the last visited vertex of V', which takes at most $\max_{u \in V'} H(u, v) = \mathcal{O}(m \gamma_{\text{eff}}(V'))$ time in expectation. □

Once we have this, it only remains to reason that because we only pay for traversing edges from the set $E(V')$, we should be able to replace m in the bound of Lemma 4.12 with the size of that set. The tools that we have developed in this chapter allow us to do exactly that.

Lemma 4.13 (cf. Lemma 3.17). *Let $G = (V, E)$ be an unweighted graph and $V' \subseteq V$. Also let $E' \subseteq E(V')$ be any set of edges inside V'. Suppose we run a random walk in G from an arbitrary vertex (not necessarily from V') and stop it once all vertices from V' have been visited. Then the expected number of traversals of edges from the set E' is*

$$\mathcal{O}(|E'| \cdot \gamma_{\text{eff}}(V') \cdot \log n).$$

Proof. First, we wait until the walk reaches a vertex $v \in V'$. We do not pay for any traversals during this time, since $E' \subseteq E(V')$. (We do not care about the distribution of what v will be: since the remainder of the proof works for any $v \in V'$, the bound will follow.)

We consider a phase division of the random walk started at v where a phase ends whenever v is visited after all vertices from V' have been covered. By the uniformity principle we get

that the expected number of traversals of any given edge e during a single phase is $\frac{1}{m}$ times the expected length of one phase, which is $\mathcal{O}(m \cdot \gamma_{\text{eff}}(V') \cdot \log n)$ by Lemma 4.12. By summing over all $e \in E'$ we get the statement. □

Observe that in order to use the uniformity principle to prove this lemma, we needed to divide the random walk into phases which ended at the same vertex where they started. It was necessary to make a random walk (whose expected length we were bounding) run somewhat longer, so that it would return to the starting point after covering V'. But notice that if we did this not using the strategy of Lemma 4.12, but rather by just requiring that the main random walk, starting at an arbitrary vertex $s \in V$, be finished only when it comes back to s (as we did in Proposition 4.10), we would not have succeeded. To see this, consider the example of the random walk on the path graph on n vertices (cf. Example 2.17) starting from vertex 1 and the set $V' = \{n-1, n\}$. A random walk which ends once it covers V' will traverse the edge $e = (n-1, n)$ exactly once, and if we require it to return to $n-1$ afterwards, exactly twice; however, if we require it to come back to 1, the expected number of traversals of e becomes $\Omega(n)$.

We conclude this section by pointing out that the above proof only works for edges inside V', since we were able to skip the initial phase of the walk (before it reached V') without accounting for any traversals. If we happen to start far away from V', some edges (which are also far from V') observe heavy traffic during this initial phase. For this reason, it is not true for an arbitrary edge e that, if we run a random walk from outside of V' and wait until V' is covered, the expected number of traversals of e during this walk is $\mathcal{O}(\gamma_{\text{eff}}(V') \cdot \log n)$. To see this, consider the path graph on n vertices (cf. Example 2.17), $e = (1, 2)$, a walk starting from vertex 1, and $V' = \{n\}$.

5 Random walks and electricity

There are many unexpected relationships between the random-walk structure of a graph and the properties of the electrical network obtained from it. We saw an example of this in Propositions 3.11 and 3.12, where we succeeded in relating the commute time between two vertices to the effective resistance between them. In this chapter we give one more example of such a nontrivial relationship. It will be crucially utilized in the algorithm of Section 7.

For the reader interested in this topic, we recommend the wonderfully easy-to-read book by Doyle and Snell [DS84]. We follow their presentation in the following section.

5.1 Harmonic functions

First, let us first consider a random walk on an undirected graph G, and let s and t be two distinct vertices. Let $q_{st}(v)$ be the probability that the walk will visit s before t if it is currently at v. Clearly we have

$$\begin{aligned} q_{st}(s) &= 1, \\ q_{st}(t) &= 0, \\ q_{st}(v) &= \sum_{u:(v,u)\in E} \frac{1}{d(v)} q_{st}(u) \quad \text{for } v \neq s, t. \end{aligned}$$

In other words, the probability at s is 0, at t it is 1, and at any other vertex v it is the average over neighbors of v. It is our intuition that these are the only conditions that q_{st} has to satisfy, i.e., that the equations above specify a unique function $q_{st} : V \to [0, 1]$. Later in this section we will prove a more general statement.

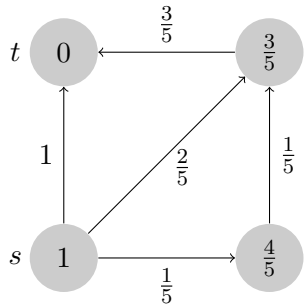

Figure 7: Probabilities that the walk started at a given vertex will hit s before t; equivalently, vertex potentials if we induce a difference of 1 between s and t. The graph is undirected and unweighted: the arrows and labels on edges show the flow of electric current. Total flow from s to t is $\frac{8}{5}$ and thus $R_{\text{eff}}(s,t) = \frac{5}{8}$. See also Fig. 4.

Second, consider an electric network obtained from G by treating every edge as a resistor of resistance 1. Suppose we connect a battery of voltage 1 between s and t (do not try this at home), so that the difference of potentials between s and t will be 1. Let us denote by $x(v)$ the potential at any vertex v, and assume that

$$x(t) = 0.$$

Then we have

$$x(s) = 1.$$

Connecting the battery will cause potential differences to be induced throughout the network. For any vertex $v \neq s, t$, by Kirchhoff's current law, the sum of signed currents flowing through all edges $v \to u$ with $(v, u) \in E$ will be 0. And by Ohm's law, the signed current on an edge $v \to u$ will be $\frac{x(v)-x(u)}{1}$ (since edges have resistance 1). We get that

$$\sum_{u:(v,u)\in E} x(v) - x(u) = 0$$

and so

$$x(v) = \frac{1}{d(v)} \sum_{u:(v,u)\in E} x(u) \quad \text{for } v \neq s, t.$$

See Fig. 7 for an example. We must ask: is x always equal to q_{st} (see above)? The answer is yes. This follows from the fact that they are both *harmonic functions with poles in s and t.*

Definition 5.1. *Let $G = (V, E)$ be a connected undirected graph and let $S \subseteq V$ be a nonempty subset of its vertices. We say that a function $\phi : V \to \mathbb{R}$ is* harmonic with set of poles S *if*

$$\phi(v) = \frac{1}{d(v)} \sum_{u:(u,v)\in E} \phi(u) \quad \textit{for all } v \in V \setminus S.$$

At every vertex $v \in V \setminus S$, a harmonic function ϕ is the average of its values over the neighbors of v. This is a discrete analogue of the mean value property of harmonic functions on $\mathbb{R}^n$. We see that both q_{st} and x are harmonic with set of poles $\{s, t\}$.

Now, once we specify the values $\phi(s)$ for all $s \in S$, there is at most one harmonic function ϕ which assumes these values (finding it is called the *Dirichlet problem*). We show this using the following property, which is analogous to the maximum principle for harmonic functions in analysis:

Fact 5.2 (Maximum Principle). *Let $\phi : V \to \mathbb{R}$ be harmonic with set of poles S. Then the maximum (and minimum) of ϕ are attained on S.*

Proof. Let M be the maximum, and v be such that $\phi(v) = M$. If $v \in S$, we are done. Otherwise, since $\phi(v)$ is the average of $\phi(u)$ over all neighbors u of v, and $\phi(u) \leq M$, it follows that $\phi(u) = M$ for all these neighbors. By "spreading" the value M in this way we will eventually reach a vertex from S, since G is connected and $S \neq \emptyset$. □

Fact 5.3 (Uniqueness Principle). *Let ϕ and ψ be two harmonic functions with pole set S which agree on S, i.e., $\phi(s) = \psi(s)$ for all $s \in S$. Then $\phi = \psi$.*

Proof. It is easy to check that the function $\phi - \psi$ is also harmonic with pole set S, and that it is zero on S. By Fact 5.2 its maximum and minimum are both zero, and so $\phi - \psi$ is identically zero. □

For us this means that the functions q_{st} and x are, indeed, uniquely defined and equal.

5.2 Computing probabilities

We are now interested in the problem of computing the the values $q_{st}(v)$ (see previous section) as fast as possible. Thanks to the electrical connection we have developed, we may do this by computing potentials $x(v)$ in the network obtained from G.

For this, we will use the methods introduced in Section 3.1. To adapt them to our setting, we will relax for now the requirement that $x(s) = 1$ and $x(t) = 0$ (so the voltage of the battery need not be 1), but instead we will assume that we inject a *current* of 1 at s and extract it at t, as we did in Section 3.1. At the end, after obtaining a vector of potentials x, we will scale it linearly to ensure that potentials at s and t are 1 and 0, respectively.

Now, by the same reasoning as in Section 3.1, we can obtain the vector x as a solution to the Laplacian system $Lx = \chi_{st}$. The following is the take-home result of this chapter, and a building block of the algorithm of Chapter 7.

Lemma 5.4 (cf. Lemma 9 of [KM09]). *Let $s, t \in V$ be two distinct vertices of G and let $\varepsilon > 0$. It is possible to compute in time $\widetilde{\mathcal{O}}(m \log \epsilon^{-1})$ a vector $q \in \mathbb{R}^V$ such that for every $v \in V$, the value $q(v)$ is a multiplicative ε-approximation of the probability $q_{st}(v)$ that a random walk started at v will visit s before it visits t.*

Proof. As outlined above, we will solve the Laplacian system $Lx = \chi_{st}$ with a precision of ε using Theorem 3.4. By the discussion in Section 3.1, the vector x so obtained is a vector of potentials induced on vertices if a current of 1 is pushed through the network from s to t. A vector of potentials in an electric network has one degree of freedom, which is reflected in our formalization by the fact that the kernel of L has dimension one (and is equal to $\text{span}\{\mathbb{1}\}$ – see Proposition 3.2.(3)). The vector x can thus be freely modified by adding the same constant to every coordinate. Moreover, the voltage between s and t in this network is $x_s - x_t$; in the network modeled in the previous section, we set it to be 1. We must thus divide all potentials by $x_s - x_t$. We return

$$q(v) = \frac{x_v - x_t}{x_s - x_t}$$

as the answer. It is easy to see that $q(s) = 1$, $q(t) = 0$ and that, by the reasoning of the previous section, the vector q is a multiplicative ε-approximation of q_{st}. □

6 Random spanning trees via random walks

Let us consider the following problem: we are given an undirected graph G and want to sample a spanning tree of G at uniform. More precisely, if $\mathcal{T}(G)$ denotes the set of all spanning trees of G, we wish to develop an algorithm which outputs any spanning tree $T \in \mathcal{T}(G)$ with probability exactly $\frac{1}{|\mathcal{T}(G)|}$.

All known algorithms for solving this problem are based on one of two methods. The first method, which we sketch very briefly, utilizes Kirchhoff's Matrix-Tree Theorem, a statement known for over 160 years:

Theorem 6.1 ([Kir47]). *Let G be an undirected graph (possibly with multiple edges) and L its Laplacian (cf. Section 3.1). The number $|\mathcal{T}(G)|$ of spanning trees of G is equal to any cofactor of L, i.e., the determinant of a matrix obtained from L by deleting any one row and any one column.*

Once we have the capability of computing $|\mathcal{T}(G)|$, we can proceed as follows [Gué83]: pick any edge e and compute $p = \frac{|\mathcal{T}(G\setminus\{e\})|}{|\mathcal{T}(G)|}$, which is the probability that e does not belong to a uniformly random spanning tree of G. Then flip a p-biased coin. With probability p, we decide that e will not be in the spanning tree that we are going to output; we erase e from G and continue. With probability $1-p$, we pick e for the tree; we then contract e in G and continue. This algorithm works in time $\mathcal{O}(mn^3)$ if we use straightforward Gaussian elimination to compute the determinant $|\mathcal{T}(G)|$. As expected, this runtime may be improved: the fastest currently known algorithm utilizing Theorem 6.1, due to Colbourn et al. [CMN96], runs in time $\mathcal{O}(n^\omega)$ with $\omega < 2.376$. We do not pursue this topic further (nor prove Theorem 6.1).

We will concentrate on the second method, which, as the reader will have guessed, is related to random walks. Its cornerstone is the following stunning theorem, discovered independently by Aldous [Ald90] and Broder [Bro89]:

Theorem 6.2. *Let G be an undirected graph. Suppose we run a random walk from an arbitrary starting vertex $s \in V$ and record, for every other vertex $v \in V \setminus \{s\}$, the edge e_v through which the walk first visited v. Then*

$$T = \{e_v : v \in V \setminus \{s\}\}$$

is a uniformly random spanning tree of G.

It will take us a while to prove Theorem 6.2. Before we do, let us notice that it immediately gives rise to an algorithm which just simulates a random walk in G and outputs T. The runtime of this algorithm is proportional to the cover time of G, which we bounded in Corollary 2.19, Lemma 2.20 and Corollary 3.15; the simplest of these bounds is $\mathcal{O}(mn)$.

Corollary 6.3. *It is possible to generate a uniformly random spanning tree of G in time $\mathcal{O}(mn)$.* □

The bound of $\mathcal{O}(mn)$ is unfortunately tight for many graphs. In Chapter 7 we show how to sample a random spanning tree faster than the cover time of the graph.

We devote the rest of this chapter to the proof of Theorem 6.2.

6.1 The chain of rooted trees – proof of Theorem 6.2

We present the proof as in [Ald90], but with more detailed explanations.

To begin with, it is clear that T is a spanning tree – it has $n-1$ edges, is connected and all vertices of G appear as endpoints of its edges. It remains to be seen that for any spanning tree $t \in \mathcal{T}(G)$, the probability that $T = t$ does not depend on t.

It will be instrumental for us to work with *rooted spanning trees* of G. Such a rooted tree S is formed from a regular undirected spanning tree by choosing one vertex to be the root and orienting all edges toward it. Every rooted (spanning) tree S thus corresponds to a pair $(t, v) \in \mathcal{T}(G) \times V$; we denote the root v of S by $r(S)$. For trees obtained as collections of first-visit edges $\{e_v : v \in V \setminus \{s\}\}$, we will assume that they are rooted at s. Note that orienting all edges of such a tree toward s corresponds to orienting every edge e_v to point *away from* v.

Let us consider a stationary walk $X = (X_i)_{i\in\mathbb{Z}}$. For any $i \in \mathbb{Z}$ let us denote by S_i the rooted tree obtained from the walk $(X_i, X_{i+1}, \ldots)$ started at X_i. In other words, S_i is the tree rooted at X_i with edges

$$\{\langle X_{T_v^i}, X_{T_v^i-1}\rangle : v \in V \setminus \{X_i\}\} = \{\langle v, X_{T_v^i-1}\rangle : v \in V \setminus \{X_i\}\} \tag{1}$$

where T_v^i is the time of first visit of X in v since step i. All these edges are oriented toward the root X_i.

We are interested in understanding the random process $(S_i)_{i\in\mathbb{Z}}$. To this end, let us consider two consecutive rooted trees S_0 and S_1 and ask ourselves: if we know one of them, what can we possibly expect the other one to look like (and with what probability)? How do S_0 and S_1 differ?

First, the tree S_0 is rooted at X_0, and S_1 is rooted at X_1. Second, the directed edge $\langle X_1, X_0\rangle = \langle r(S_1), r(S_0)\rangle$ belongs to S_0 but not to S_1, and a directed edge $\langle X_0, w\rangle = \langle r(S_0), w\rangle$, for some neighbor w of $X_0 = r(S_0)$, belongs to S_1 but not to S_0. A good look at (1) should make it clear that these are the only differences.

Suppose we only know S_0; in particular, we do not know X_1. What rooted trees may appear as S_1? Let us modify S_0 by adding an edge and deleting an edge to see what we can get. We know that the only edge in S_1 which is not in S_0 is $\langle r(S_0), w\rangle$ for some neighbor w of $r(S_0)$. Let us add any edge of this form to S_0. Now we need to delete an edge pointing at $r(S_0)$, and by doing so we must destroy the cycle that has appeared. There is exactly one edge on the cycle which points at $r(S_0)$, so we have no choice but to delete it; the tail of the deleted edge, which now has no outgoing edges, becomes the new root. So for any w we have obtained a valid rooted tree S_1; let us denote a tree so created by $F(S_0, w)$. We have shown the following:

$$S_1 = F(S_0, w) \text{ for some neighbor } w \text{ of } r(S_0). \tag{2}$$

(We do not yet claim the converse, i.e., that if $S = F(S', w)$ for some neighbor w of $r(S')$, then it is possible that $S_0 = S'$ and $S_1 = S$.)

Let us now look back in time. What rooted trees may appear as S_0 if we happen to know S_1? Apart from S_1 itself, this will depend only on the vertex X_0. We proceed analogously here: first we add the edge $\langle r(S_1), v\rangle$ to S_1, where v is a neighbor of $r(S_1)$; we know that $r(S_1) = X_1$, and v corresponds to X_0. Now, v should be the root of S_0, so we remove its (only) outgoing edge; also, by doing so, we destroy the cycle that has appeared. We thus get a valid rooted tree S_0, which we denote by $B(S_1, v)$. We have shown the following:

$$S_0 = B(S_1, v) \text{ for some neighbor } v \text{ of } r(S_1). \tag{3}$$

Because of their nature, we expect that B and F are inverse operations in some sense. Indeed, it may be verified that for any rooted tree S' and any neighbor w of $r(S')$, the tree $F(S', w)$ can be transformed back to S' by using the operation B with $v = r(S')$, i.e.,

$$\text{for any } S' \text{ and neighbor } w \text{ of } r(S') \text{ we have } B(F(S', w), r(S')) = S'. \tag{4}$$

Now, our back-in-time perspective is very beneficial in that we can get an exact probability value from it. Indeed: by time-reversibility of the walk X (cf. Proposition 2.3) we have that

$$\mathbb{P}(X_0 = v \mid X_1 = u) = \frac{1}{d(u)}$$

for any $(u, v) \in E$. By (3) this means that, for any rooted tree S,

$$\mathbb{P}(S_0 = B(S, v) \mid S_1 = S) = \frac{1}{d(r(S))} \tag{5}$$

for any neighbor v of $r(S)$ (v acts as X_0). (Note that $d(r(S))$ is the degree of $r(S)$ in G, not in S.) In particular, we see that S_0 only depends on the sequence $(X_1, X_2, \ldots)$ *through* S_1 – equation (5) depends only on S and not directly on $X_1, X_2, \ldots$. Also, the formula obviously remains unchanged if we replace S_0 by S_i and S_1 by S_{i+1} for any $i \in \mathbb{Z}$.

What this means is that the backward sequence $\mathcal{S} = (S_{-i})_{i \in \mathbb{Z}}$ is a Markov chain whose states are all rooted spanning trees of G and whose transition probability matrix is

$$Q(S, S') = \mathbb{P}(S_0 = S' \mid S_1 = S) = \begin{cases} \frac{1}{d(r(S))} & \text{if } S' = B(S, v) \text{ for some } v \text{ with } (r(S), v) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Also, for any given S', the only trees S for which $Q(S, S')$ may be positive are $S = F(S', w)$ for some neighbor w of $r(S')$ (by (2)); and indeed if $S = F(S', w)$, then $Q(S, S')$ is positive and equal to $\frac{1}{d(r(S))}$ by (4) and (5). There are $d(r(S'))$ possible choices of w. Thus we have that

$$\text{for every } S', \quad \sum_S d(r(S)) \cdot Q(S, S') = d(r(S')). \tag{6}$$

As $(X_i)_{i \in \mathbb{Z}}$ is stationary, so is $\mathcal{S}$. Also, let us remark that it is irreducible, i.e., for any two states S^1, S^2 there exists a path of some length t which leads from S^1 to S^2 with positive probability: $Q^t(S^1, S^2) > 0$. This is easy to see if one considers a depth-first traversal of S^2. More precisely, let $r(S^1) = v_0, v_1, \ldots, v_k = r(S^2)$ be any path in G from $r(S^1)$ to $r(S^2)$, and let $r(S^2) = v_k, v_{k+1}, \ldots, v_t = r(S^2)$ be the transcript of a depth-first traversal of S^2 (where every vertex appears twice). Then $S^2 = B(B(\ldots B(B(S^1, v_1), v_2) \ldots, v_{t-1}), v_t)$.

Because $\mathcal{S}$ is a finite irreducible Markov chain, it has a *unique* stationary distribution $\pi^{\mathcal{S}}$ (cf. Corollary 1.17 in [LPW08]). It satisfies

$$\text{for every } S', \quad \sum_S \pi^{\mathcal{S}}(S) \cdot Q(S, S') = \pi^{\mathcal{S}}(S'). \tag{7}$$

Equations (6) and (7) together imply that $\pi^{\mathcal{S}}(\cdot)$ and $d(r(\cdot))$ are proportional, i.e., there exists a normalizing constant $c > 0$ such that for every S,

$$\mathbb{P}(S_0 = S) = \pi^{\mathcal{S}}(S) = c \cdot d(r(S)).$$

Crucially, all rooted spanning trees with the same root have the same the probability of being S_0 – and for a fixed root s, the trees rooted at s are in a 1-1 correspondence with the undirected spanning trees of G.

To conclude, if we fix the starting vertex to be s, then for any tree S rooted at s, the probability

$$\mathbb{P}(S_0 = S \mid X_0 = s) = \frac{\mathbb{P}(S_0 = S)}{\mathbb{P}(X_0 = s)} = \frac{c \cdot d(s)}{\frac{d(s)}{2m}} = 2mc$$

is independent of S. In particular, for any undirected tree $t \in \mathcal{T}(G)$ and starting vertex s, we have that

$$\mathbb{P}(T = t \mid X_0 = s) = \mathbb{P}(S_0 = S \mid X_0 = s) = 2mc$$

where S is t rooted at s. □

7 Fast random spanning tree generation

In this chapter we demonstrate the fast algorithm of Kelner and Mądry (FOCS 2009 [KM09]) for sampling uniformly random spanning trees. The algorithm and its analysis make use of techniques we introduced in Sections 5 and 6, as well as various bounds we developed throughout the thesis, such as Lemmas 2.20, 3.17 and 4.13 and Proposition 4.10. The present chapter can thus be seen as a culmination of the thesis.

There exist multiple versions of the algorithm, differing in running time. The paper [KM09] presents an algorithm with running time $\widetilde{\mathcal{O}}\left(\frac{m^2}{\sqrt{n}}\log\delta^{-1}\right)$ which finds a δ-approximately uniform random spanning tree of G; the error δ arises from using approximate Laplacian solvers (cf. Theorem 3.4). Here, *δ-approximately uniform* means that for any spanning tree T, the probability that T is returned is between $1-\delta$ and $1+\delta$ of uniform. There are three known directions of improvement for this result.

First, in [Mąd11] it is shown how to dispense with the δ-error using a trick due to James Propp and get an exactly uniform random spanning tree in time $\widetilde{\mathcal{O}}\left(\frac{m^2}{\sqrt{n}}\right)$. We incorporate this trick into our presentation.

Second, it is also shown in [KM09] how to improve the algorithm's running time for non-sparse graphs, bringing it down to $\widetilde{\mathcal{O}}(m\sqrt{n}\log\delta^{-1})$ (or $\widetilde{\mathcal{O}}(m\sqrt{n})$ after getting rid of the error). Since the improvement is technical and does not have much to do with random walks, we do not present it here. Instead, we present a version of the algorithm which has complexity $\widetilde{\mathcal{O}}(m^{3/2})$.

Third, in [MST13] Mądry, Straszak and Tarnawski show how to bring this running time further down to $\widetilde{\mathcal{O}}(m^{4/3})$, which is the best known for sparse graphs. They use the effective resistance diameter (cf. Definition 3.14) to better understand the random-walk structure of the graph, employ a more involved clustering scheme (than that of Section 7.2), and they sample the tree in many phases (rather than using just one random walk) to get this improved running time. The runtime analysis of their algorithm uses the projected graph (see Section 3.2) and Lemmas 3.9, 3.17 and 4.11. We do not go into more detail on this here.

The rest of this chapter is devoted to the proof of the following theorem.

Theorem 7.1 ([KM09, Mąd11]). *It is possible to generate a uniformly random spanning tree of G in time $\widetilde{\mathcal{O}}(m^{3/2})$.*

7.1 Outline of the approach

Let us recall the $\mathcal{O}(nm)$ approach of Corollary 6.3: we simulate a random walk starting at any vertex s and register the first-entry edges e_v for all $v \in V \setminus \{s\}$. The tree formed by these edges is uniformly random by Theorem 6.2, and the running time of the algorithm is proportional to the cover time of G.

If we observe a trajectory of the covering random walk, we will usually find that it spends long periods of time walking around regions of G that have already been explored. Such fragments of the walk do not really contribute much to the task of determining the set of first-visit edges e_v – visiting a vertex for the first time is a rather rare event (except in the beginning of the walk). Indeed, from the entire trajectory of the covering walk X, which has average length $\mathrm{cov}(G)$, we only utilize $n-1$ traversals to recover the first-visit edges.

The idea of coping with this inefficiency is the following: what if we could skip over some parts of the walk? For example, if we are inside a subgraph induced by a subset of vertices V', and we already know e_v for all $v \in V'$, then we are only interested in knowing *through which edge we will exit V'*. If we knew this, then we would gladly just fast-forward the walk until that time, since going through the motions inside V' brings us no new information. And Lemma 5.4

suggests that it might be possible to quickly compute a probability distribution on the outgoing edges of V' that would enable such fast-forwarding. We call this idea *shortcutting over* V'.

The plan, then, is to generate, instead of the full trajectory X, a *shortcut* trajectory $\widetilde{X}$ which would be a subsequence of X obtained by erasing some traversals which do not induce new edges e_v (or: long fragments made up of such traversals). It would be ideal if $\widetilde{X}$ was short and if we could generate it fast.

Indeed, it is possible to generate such a shortcut trajectory. We will achieve this by identifying in G a number of *regions* – vertex-induced subgraphs $D_1, \ldots, D_k$. For every region D_i, in order to enable shortcutting the walk over it, we must compute a set of *exit probabilities*. More precisely, we compute, for every $v \in D_i$ and every edge $e \in \partial D_i$ on the boundary of D_i, the probability $P_v(e)$ that e will be the first edge of ∂D_i that the walk will traverse if it begins at v; or, in other words, that e will be the edge through which the walk will exit D_i if it is currently at v.

We then run the random walk simulation. Now, whenever we cover a whole region D_i, we start shortcutting the walk over it. This means that, whenever we find ourselves in a vertex v belonging to a region D_i where all vertices have already been covered by the walk, we sample the exiting edge e according to the probability distribution $P_v = \{P_v(e)\}_{e \in \partial D_i}$, and fast-forward the walk by immediately moving out of D_i through e. We think that all traversals of edges inside D_i that we have fast-forwarded through are ones which are present in X, but absent from $\widetilde{X}$ (they constitute one of the long fragments erased from X that we mentioned earlier). We stop the walk once it has covered all vertices and we output the generated spanning tree.

We proceed to describe the building blocks of the algorithm in detail.

7.2 Ball-growing partitions

The first step of our algorithm will be identifying the regions $D_1, \ldots, D_k$. Let us ask: what are the features of such a collection of regions that we find appealing? In this section we will answer this question by broadly surveying the subroutines of our algorithm and the dependencies of their running times on the qualities of this collection.

First, we will not be able to shortcut the walk over a region D_i before it has been covered; we will pay for all traversals until that time (i.e., they will be in $\widetilde{X}$). Because of this, we would prefer the number of such traversals to be small. But this quantity is exactly what Lemma 4.13 refers to. For the reader's convenience, we repeat it here; also, because the effective-resistance diameter $\gamma_{\text{eff}}(D_i)$ in its statement seems difficult to work with, let us replace it with the graph-distance-based diameter $D(D_i) = \max_{u,v \in D_i} \operatorname{dist}(u, v)$, which upper-bounds $\gamma_{\text{eff}}(D_i)$ (cf. Section 3.3):

Lemma 7.2 (cf. Lemma 4.13). *Let $G = (V, E)$ be an unweighted graph and $V' \subseteq V$. Suppose we run a random walk in G from an arbitrary vertex (not necessarily from V') and stop it once all vertices from V' have been visited. Then the expected number of traversals of edges from the set $E(V')$ is*

$$\mathcal{O}(|E(V')| \cdot D(V') \cdot \log n).$$ □

Now we see that the number of traversals inside D_i before it is covered will be small if we can make $D(D_i)$ small. Let us make this one of our requirements.

Second, there will be edges in G which are not inside any region D_i. We will pay for all traversals of these edges until G is covered (i.e., all traversals from X of these edges will be in $\widetilde{X}$). Therefore it would be best if there were few such edges. Indeed, by Proposition 4.10 combined with Lemma 2.20 we have the following:

Lemma 7.3 (cf. Fact 5 of [KM09]). *Suppose we run a random walk from any starting vertex until G is covered. Let $E' \subseteq E$ be any subset of edges. The expected number of traversals of edges from E' is $\mathcal{O}(|E'| \cdot n)$.* □

It is then reasonable to also require that the number of edges which do not belong to any D_i be small. For brevity, we will call such edges *cut edges* and have them form the set C, i.e.,

$$C = E \setminus (E(D_1) \cup \ldots \cup E(D_k)).$$

Third, we will have to compute the exit probabilities for every D_i. Roughly speaking, we will do it as follows: for every edge $e \in \partial D_i$ we will compute a vector $q_{e\bar{e}} \in \mathbb{R}^{V(D_i)}$ using Lemma 5.4, where $q_{e\bar{e}}(v)$ is the probability that a walk starting from v will traverse edge e before traversing any other edge exiting D_i. As we will do this for every edge $e \in \partial D_i$, we would like the number of such boundary edges of D_i to be small. Because all such edges are cut edges, this reinforces our second requirement that the number of cut edges be small.

This motivates the following definition of a good collection of regions: we will want them to have diameter of at most γ and to contain all but a ϕ-fraction of edges, for some parameters ϕ and γ.

Definition 7.4. *We say that a collection of vertex-induced subgraphs $D_1, \ldots, D_k$ of G is a (ϕ, γ)-decomposition iff:*

(1) the graph-distance-based diameter $D(D_i)$ of each D_i is at most γ,

(2) the total number of cut edges $|C| = |E \setminus (E(D_1) \cup \ldots \cup E(D_k))|$ is at most $\phi \cdot m$.

We are interested in both ϕ and γ as low as possible, but, unsurprisingly, there is an inherent compromise between these two values. Trivially, if we were to require $\gamma = 0$, then we could only get a decomposition with $\phi = 1$; if we only required $\gamma = n$, then we could get $\phi = 0$. The following lemma tells us that we can get a decomposition with $\gamma \approx \phi^{-1}$ in linear time.

Lemma 7.5 (ball-growing partitioning of Leighton and Rao). *For every $\phi = o(1)$ we can find a $\left(\phi, \frac{2 \log m}{\phi}\right)$-decomposition in $\mathcal{O}(m)$ time.*

Proof (following [MST13]). We employ an adaptation of the partitioning procedure from [KM09] that is itself based on the ball-growing technique of Leighton and Rao [LR99].

For a subset of vertices $V' \subseteq V$ let us define the set of neighbors $N(V')$ to be $V' \cup \{u \in V : (u, v) \in E \text{ for some } v \in V'\}$. Also recall that $\partial V'$ denotes the set of edges with exactly one endpoint in V'.

We start with the original G, $i = 1$ and perform the following procedure until G is empty:

- choose any vertex $v \in G$ and set $D := \{v\}$,
- while $|\partial D| > \phi |E(D)|$ do: set $D := N(D)$,
- add D as a new region D_i,
- remove D and all incident edges from G,
- $i := i + 1$.

Obviously $D_1, \ldots, D_k$ is a partition of V. This also implies that all cut edges are on the boundaries of the sets D_i, i.e.

$$C = \partial D_1 \cup \ldots \cup \partial D_k.$$

It is thus easy to see that condition (2) in Definition 7.4 is satisfied:

$$|C| \leq \sum_{i=1}^{k} |\partial D_i| \leq \sum_{i=1}^{k} \phi \cdot |E(D_i)| \leq \phi \cdot m.$$

It remains to show that each piece has a small diameter, i.e., establish condition (1). To this end, note that the radius of D, at the moment it is added as a new piece, is bounded by the number of the while-loop executions. We also see that each execution increases $|E(D)|$ at least by a factor of $1 + \phi$. This means that after s steps we have $|E(D)| > (1 + \phi)^s$. Since

$$(1 + \phi)^{\frac{\log m}{\phi}} \geq m \quad \text{for } \phi < \frac{1}{2},$$

we deduce that the radius of D is bounded by $\frac{\log m}{\phi}$ and thus the diameter of D is at most $\frac{2 \log m}{\phi}$. It is not hard to see that the presented algorithm can be implemented to run in $\mathcal{O}(m)$ time. □

Our algorithm will use Lemma 7.5 to find a decomposition with $\phi = \frac{1}{\sqrt{m}}$, which may be seen as the sweet spot with respect to the time complexities of the algorithm's components. We therefore have the following bounds:

(1) for each i, $D(D_i) \leq 2\sqrt{m} \log m = \widetilde{\mathcal{O}}(\sqrt{m})$,
(2) $|C| \leq \sqrt{m}$.

7.3 Computing exit probabilities

Now that we have our collection of regions, let us concentrate on the problem of computing the exit probabilities $P_v(e)$ for all of them. Recall that for each region D_i, each of its boundary edges $e \in \partial D_i$ and each vertex $v \in D_i$, we need to compute the probability $P_v(e)$ that e will be the edge through which the walk will exit D_i if it is currently at v. We argue that this can be done fast enough for our purposes.

Lemma 7.6 (cf. Lemma 9 of [KM09]). *For any $\varepsilon > 0$, it is possible to compute multiplicative ε-approximations to all exit probabilities $P_v(e)$ in time $\widetilde{\mathcal{O}}(m^{3/2} \log \varepsilon^{-1})$.*

Proof. Consider a single region $D = D_i$ and a boundary edge $e \in \partial D$. We wish to obtain, in one computation, the approximate probabilities $P_v(e)$ for all $v \in D$. Note that $P_v(e)$ is the probability that a random walk started in v will traverse e before it traverses any other edge of ∂D. Also recall that Lemma 5.4 allows us to compute the probability, for any vertices s, t and v, that a random walk started at v will reach s before it reaches t.

To use the lemma in our setting, we will perform a simple reduction. Roughly speaking, we will redirect e to point to a special vertex u, and all other edges of ∂D to point to another special vertex u^*. Formally, we transform the region D into a graph D^e as follows. We take the vertex set $V(D^e)$ to be $V(D) \cup \{u, u^*\}$, where u and u^* are two new vertices. If $e = (v_1, v_2)$ with $v_1 \in D$, then the edge set of D^e will be

$$E(D^e) = E(D) \cup \{(v_1, u)\} \cup \{(w_1, u^*) : (w_1, w_2) \in \partial D \setminus \{e\} \text{ with } w_1 \in D\}.$$

Note that D^e may have parallel edges. It is clear that the reduction achieves its goal, i.e., that for any vertex $v \in D$, the probability that a walk in D^e started at v will reach u before it reaches

u^* is equal to the probability that a walk in G started at v will exit D through e. By Lemma 5.4, an ε-approximation to the vector $\{P_v(e)\}_{v\in D}$ of such probabilities may be computed in time

$$\widetilde{\mathcal{O}}(|E(D^e)|\cdot\log\varepsilon^{-1}) = \widetilde{\mathcal{O}}((|E(D)|+|\partial D|)\cdot\log\varepsilon^{-1}).$$

We can therefore compute these vectors for all regions D_i and all boundary edges $e\in\partial D_i$ in time

$$\sum_{i=1}^{k}\widetilde{\mathcal{O}}(|\partial D_i|\cdot(|E(D_i)|+|\partial D_i|)\cdot\log\varepsilon^{-1}).$$

Because $|\partial D_i|\le|C|\le\sqrt{m}$ for all i, and because $\sum_{i=1}^{k}(|E(D_i)|+|\partial D_i|)\le m+2|C|\le 3m$, this is bounded by $\widetilde{\mathcal{O}}(\sqrt{m}\cdot m\cdot\log\varepsilon^{-1})$. □

7.4 Simulating the shortcut walk – proof of Theorem 7.1

We have now gathered all the necessary ingredients of the algorithm. Let us state it precisely.

We begin by finding a $\left(\sqrt{m},\frac{2\log m}{\sqrt{m}}\right)$-decomposition $D_1,\dots,D_k$ as described in Section 7.2. Recall that we get $D(D_i)=\widetilde{\mathcal{O}}(\sqrt{m})$ and $|C|\le\sqrt{m}$.

Next, we compute approximate exit probabilities $P'_v(e)$ for all regions D_i using Lemma 7.6. For now, let us optimistically assume that this takes time $\widetilde{\mathcal{O}}(m^{3/2})$ and that we compute all exit probabilities exactly, i.e., $P'_v(e)=P_v(e)$. We will show later how to get rid of this assumption.

Before we start simulating the walk, we need to perform some preprocessing. For every vertex v in a region D_i we construct the probability distribution for exiting D from v, i.e., the distribution $\{P_v(e)\}_{e\in\partial D_i}$. (We do this by just copying values over from the vectors $\{P_v(e)\}_{v\in D_i}$ for all $e\in\partial D_i$.) Moreover, we order the boundary edges $\{e_1,\dots,e_{|\partial D_i|}\}=\partial D_i$ arbitrarily and compute prefix sums $S_v(j)=\sum_{l=1}^{j}P_v(e_l)$ for $j=0,\dots,|\partial D_i|$. This takes $\mathcal{O}(\sum_{i=1}^{k}|D_i|\cdot|\partial D_i|)$ time, which is $\mathcal{O}(m^{3/2})$ because $|\partial D_i|\le|C|\le\sqrt{m}$.

Now we run a random walk starting from an arbitrary vertex s until G is covered and register the first-entry edges e_v for all vertices $v\in V\setminus\{s\}$. At every step of the walk, if we are in a vertex v which belongs to a region D_i which has already been covered, we shortcut the walk over D_i. To do this, we sample a uniformly random number $\tau\in[0,1]$ and find by binary search the minimum index $j\in\{1,\dots,|\partial D_i|\}$ satisfying $S_v(j)\ge\tau$. It is clear that this can be done in time $\mathcal{O}(\log m)$ and that by doing so we get a random edge e_j from the distribution P_v. Once we have identified the edge e_j, we fast-forward the walk by moving to the endpoint of e_j which lies outside D_i.

This concludes the description of the simulation procedure. It remains to be shown that the shortcut trajectory $\widetilde{X}$ that we end up simulating is short. There are three types of traversals in $\widetilde{X}$:

(1) traversals of cut edges,

(2) for every i, traversals inside D_i – before it is covered,

(3) for every i, traversals corresponding to shortcutting the walk over D_i – after it is covered.

We deal with each of the types separately. We already have all the tools needed for the analysis.

(1) Because $|C|\le\sqrt{m}$, we can use Lemma 7.3 to upper-bound the number of such traversals by $\mathcal{O}(\sqrt{m}\cdot m\cdot\log n)=\widetilde{\mathcal{O}}(m^{3/2})$. This is true for trajectory X as well as $\widetilde{X}$.

(2) We use Lemma 7.2 to bound the number of such traversals for all i by

$$\sum_{i=1}^{k}\mathcal{O}(|E(D_i)|\cdot D(D_i)\cdot\log n)=\sum_{i=1}^{k}\widetilde{\mathcal{O}}(|E(D_i)|\cdot\sqrt{m})=\widetilde{\mathcal{O}}(m\cdot\sqrt{m}).$$

(3) Here note that every such traversal in $\widetilde{X}$ corresponds to a traversal of a cut edge in X. Indeed, suppose that we were in a vertex $v \in D_i$, picked the exit edge $e = (v_1, v_2) \in \partial D_i$, where $v_1 \in D_i$, and fast-forwarded the walk by jumping to v_2. Then the respective fragment of trajectory X is of the form $v, \ldots, v_1, v_2$ (in particular, it contains the e-traversal $v_1 \to v_2$), whereas in $\widetilde{X}$ we have v, v_2. We conclude that the number of such traversals is at most the number of those of type (1) (in X), which we have already bounded.

This almost finishes the proof. We are left with dealing with the error ε. This is a mostly technical reasoning, with no essentially new ideas.

Roughly speaking, the plan is to treat ε as a variable which we set to some initial value at the beginning. If we run into problems with the current value of ε, we will decrease it, recompute the exit probabilities and try again. Of course, we must pay attention not to skew the distribution of the spanning tree by doing so.

Suppose we have computed the approximate values $P'_v(e)$ for some ε. Let us use them to build prefix sums $S'_v(e)$. Because $|P'_v(e) - P_v(e)| \le \varepsilon P_v(e) \le \varepsilon$, we also have $|S'_v(j) - S_v(j)| \le \varepsilon j \le \varepsilon m$ (for all relevant v, e, j). The following observation is crucial: if we sample a number $\tau \in [0, 1]$ in order to carry out the binary search on S'_v in our shortcutting routine, and it happens that τ is more than εm away from all entries of S'_v, then the edge e_j that we will choose is correct, i.e., is the one that we would have chosen if we were running the binary search on the exact sequence S_v. Indeed, if

$$S'_v(j-1) < \tau \le S'_v(j)$$

and the absolute difference between τ and any element of S'_v is more than εm, then

$$S_v(j-1) \le S'_v(j-1) + \varepsilon m < \tau \le S'_v(j) - \varepsilon m \le S_v(j).$$

We see that if we are lucky enough to only pick numbers τ which are εm-far from elements of S'_v, then we are able to carry out the simulation exactly, even though we do not have access to exact values of P_v. We will call such numbers τ *safe*.

In our algorithm we will keep an accuracy counter a. At the beginning we set $a = 0$ and $\varepsilon = \frac{1}{3m^5 2^a} = \frac{1}{3m^5}$. We compute the exit probabilities and their prefix sums and start simulating the walk. Whenever we sample a random number $\tau \in [0, 1]$ (for the purpose of shortcutting) and perform binary search, we check whether τ is safe. If it is not, we increment a and recompute all exit probabilities (and prefix sums) using the new $\varepsilon = \frac{1}{3m^5 2^a}$. We keep doing this until τ is found to be safe.

To streamline our analysis, we will make another slight modification. Namely, we will only perform at most m^3 shortcuttings; instead of performing any more, we will simulate the walk naively (step-by-step). This will hardly impact our running time, as we rarely expect to make this many steps: recall that the expected number of shortcuttings (steps of type (3)) is $\widetilde{\mathcal{O}}(m^{3/2})$, so by Markov's inequality the probability that we will hit the limit is $o(m^{-1})$, and the expected number of naive steps conditioned on that event is at most $\mathrm{cov}(G) = \mathcal{O}(nm)$. All in all, the expected time that we pay for this modification is $o(n)$.

Since we always pick the same edges that we would have picked if we had access to the exact distributions P_v, we see that the algorithm samples an exactly uniform random spanning tree. It remains to bound the expected time of recomputing the exit probabilities.

Let us denote by A the value of a by the end of the simulation. The quantity that we wish to bound is

$$\mathbb{E}\left[\sum_{a=0}^{A} \widetilde{\mathcal{O}}(m^{3/2} \log(3m^5 2^a))\right] = \mathbb{E}\left[\sum_{a=0}^{A} \widetilde{\mathcal{O}}(m^{3/2}(a+1))\right] = \widetilde{\mathcal{O}}\left(m^{3/2} \sum_{a=0}^{\infty} (a+1)\mathbb{P}(A \ge a)\right).$$

Now for $a \geq 1$, $A \geq a$ if and only if we picked an unsafe τ when using $\varepsilon = \frac{1}{3m^5 2^{a-1}}$. When sampling one number, the probability that it is unsafe may be bounded by $(m+1) \cdot 2m\varepsilon \leq 3m^2\varepsilon$, as all unsafe numbers lay in the union of neighborhoods $\bigcup_{j=0}^{|\partial D_i|}[S'_v(j) - m\varepsilon, S'_v(j) + m\varepsilon]$. By union-bounding over all shortcuttings (there are at most m^3), the probability of picking at least one unsafe τ is at most $m^3 \cdot 3m^2\varepsilon = \frac{1}{2^{a-1}}$. The expected recomputation time becomes

$$\widetilde{\mathcal{O}}\left(m^{3/2} \sum_{a=1}^{\infty} \frac{a+1}{2^{a-1}}\right) = \widetilde{\mathcal{O}}(m^{3/2}),$$

which ends the proof. □

8 Further reading

In the thesis we have discussed in depth a few selected topics related to the behavior of the random walk and to generation of random spanning trees. However, we also missed out on many interesting developments.

- We did not review the modification of the algorithm of Chapter 7 which brings its running time down from $\widetilde{\mathcal{O}}(m^{3/2})$ to $\widetilde{\mathcal{O}}(m\sqrt{n})$ (see [KM09]). This is an improvement for all but sparse graphs.
- It is possible to find a uniformly random spanning tree in time $\widetilde{\mathcal{O}}(m^{4/3})$, which is the best known complexity for sparse graphs; see [MST13].
- We remarked on the connections between random walks and electricity and between random walks and random spanning trees, but not on those between electricity and random spanning trees. In particular, we omitted the following curious fact: for any edge (u, v), the probability that it is contained in a uniformly random spanning tree is equal to $R_{\text{eff}}(u, v)$. See Theorem 4.5 in [Vis13].
- We did not discuss the fact that the distributions μ_i (cf. Section 2.2) converge to the stationary distribution, irrespective of the starting distribution μ_0, as long as the graph is non-bipartite. Also, we discussed quantitative parameters like commute times and cover time, but we said nothing about the mixing rate, which is, roughly speaking, the rate of this convergence. Refer to [LPW08] and [Lov93] for more on this topic.
- One can consider random walks on infinite graphs. In such graphs the walk may not be *recurrent*, i.e. there may be a positive probability that the walk will never return to its starting point. The standard example here is the walk on the d-dimensional grid $\mathbb{Z}^d$. Surprisingly, it turns out that it is recurrent for $d \leq 2$, but not for $d \geq 3$! In studying this, one considers the notion of effective resistance between a vertex and infinity. Refer to the book by Doyle and Snell [DS84] for an account of this dichotomy.
- By generalizing further, one arrives at the notion of a Markov chain (not necessarily finite). Refer to e.g. [LPW08] or [Ros09] for more information on general Markov chains.
- We did not consider the spectral properties of the graph, i.e., ones related to the eigenvalues of its Laplacian matrix. For example, it is possible to bound the cover time of a graph using its spectral gap; see e.g. [BK88].

References

[AKL+79] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *FOCS'79: Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science*, pages 218–223, 1979.

[Ald90] D J Aldous. A random walk construction of uniform spanning trees and uniform labelled trees. *SIAM Journal on Discrete Mathematics*, 3(4):450–465, 1990. URL: `http://www.stat.berkeley.edu/~aldous/Papers/me43.ps.Z`.

[BK88] Andrei Z. Broder and Anna R. Karlin. Bounds on the cover time. *Journal of Theoretical Probability*, 2:101–120, 1988. URL: `http://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-32.pdf`.

[Bol79] Bela Bollobas. *Graph Theory: An Introductory Course*. Springer-Verlag, 1979.

[Bro89] A. Broder. Generating random spanning trees. In *FOCS'89: Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 442–447, 1989. URL: `ftp://gatekeeper.dec.com/pub/dec/SRC/publications/broder/spt.pdf`.

[BV04] S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL: `http://www.stanford.edu/~boyd/cvxbook/`.

[CMN96] Charles J. Colbourn, Wendy J. Myrvold, and Eugene Neufeld. Two algorithms for unranking arborescences. *Journal of Algorithms*, 20(2):268–281, 1996.

[CRRS89] A. K. Chandra, P. Raghavan, W. L. Ruzzo, and R. Smolensky. The electrical resistance of a graph captures its commute and cover times. In *STOC'89: Proceedings of the 21st Annual ACM Symposium on the Theory of Computing*, pages 574–586, 1989. URL: `http://homes.cs.washington.edu/~ruzzo/papers/resist.pdf`.

[DS84] P.G. Doyle and J.L. Snell. *Random walks and electric networks*. Mathematical Association of America, 1984. URL: `http://arxiv.org/abs/math/0001057`.

[Gué83] A. Guénoche. Random spanning tree. *Journal of Algorithms*, 4(3):214–220, 1983.

[Kir47] G. Kirchhoff. Über die Auflösung der Gleichungen auf welche man sei der Untersuchung der Linearen Verteilung galvanischer Ströme geführt wind. *Poggendorgs Ann. Phys. Chem.*, 72:497–508, 1847.

[KM09] Jonathan A. Kelner and Aleksander Mądry. Faster generation of random spanning trees. In *FOCS'09: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009. URL: `http://arxiv.org/abs/0908.1448`.

[KOSZ13] Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. In *STOC'13: Proceedings of the 45th Annual ACM Symposium on the Theory of Computing*, pages 911–920, 2013. URL: `http://arxiv.org/abs/1301.6628`.

[Lov93] L. Lovász. Random walks on graphs: A survey. In *Combinatorics, Paul Erdõs is eighty, Vol. 2*. János Bolyai Mathematical Society, 1993. URL: `http://www.cs.elte.hu/~lovasz/erdos.pdf`.

[LPW08] D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2008. URL: http://pages.uoregon.edu/dlevin/MARKOV/.

[LR99] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999. URL: http://pdf.aminer.org/000/283/379/a_probabilistic_multicommodity_flow_solution_to_circuit_clustering_problems.pdf.

[Mąd11] Aleksander Mądry. *From Graphs to Matrices, and Back: New Techniques for Graph Algorithms*. PhD thesis, Massachusetts Institute of Technology, 2011. URL: http://thl.epfl.ch/madry/docs/thesis.pdf.

[Mat88] Peter Matthews. Covering problems for Brownian motion on spheres. *The Annals of Probability*, 16(1):189–199, 01 1988. doi:10.1214/aop/1176991894.

[MST13] Aleksander Mądry, Damian Straszak, and Jakub Tarnawski. Fast generation of random spanning trees and the effective resistance metric. Preprint, 2013.

[Rei05] Omer Reingold. Undirected st-connectivity in log-space. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 376–385, New York, NY, USA, 2005. ACM. URL: http://www.wisdom.weizmann.ac.il/~/reingold/publications/sl.ps, doi:10.1145/1060590.1060647.

[Ros09] Sheldon M. Ross. *Introduction to Probability Models, Tenth Edition*. Elsevier Science, 2009.

[Sav70] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177 – 192, 1970. doi:10.1016/S0022-0000(70)80006-X.

[SS08] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *STOC'08: Proceedings of the 40th Annual ACM Symposium on the Theory of Computing*, pages 563–568, 2008. URL: http://arxiv.org/abs/0803.0929.

[Vis13] Nisheeth K. Vishnoi. Lx = b. *Foundations and Trends in Theoretical Computer Science*, 8(1-2):1–141, 2013. URL: http://research.microsoft.com/en-us/um/people/nvishno/site/Lxb-Web.pdf.

[YWB03] Vladimir Yanovski, Israel A. Wagner, and Alfred M. Bruckstein. A distributed ant algorithm for efficiently patrolling a network. *Algorithmica*, 37(3):165–186, 2003. doi:10.1007/s00453-003-1030-9.