Faster Algorithms and Graph Structure via Gaussian Elimination

by

Aaron Victor Schild

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Satish Rao, Chair
Professor Nikhil Srivastava
Professor Luca Trevisan

Fall 2019

Faster Algorithms and Graph Structure via Gaussian Elimination

Abstract

Faster Algorithms and Graph Structure via Gaussian Elimination

by

Aaron Victor Schild

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Satish Rao, Chair

Graph partitioning has played an important role in theoretical computer science, particularly in the design of approximation algorithms and metric embeddings. In some of these applications, fundamental tradeoffs in graph partitioning prevented further progress. To overcome these barriers, we consider partitions of certain derived graphs of an undirected graph G obtained by applying Gaussian elimination to the Laplacian matrix of G to eliminate vertices from G. We use this technique and others to obtain new results on the following fronts:

Cheeger's Inequality: Cheeger's inequality shows that any undirected graph G with minimum nonzero normalized Laplacian eigenvalue λ_G has a cut with conductance at most $O(\sqrt{\lambda_G})$. Qualitatively, Cheeger's inequality says that if the relaxation time of a graph is high, there is a cut that certifies this. However, there is a gap in this relationship, as cuts can have conductance as low as $\Theta(\lambda_G)$.

To better approximate the relaxation time of a graph, we consider a more general object. Specifically, instead of bounding the mixing time with cuts, we bound it with cuts in graphs obtained via Gaussian elimination from G. Combinatorially, random walks in these graphs are equivalent in distribution to random walks in G restricted to a subset of its vertices. As a result, all Schur complement cuts have conductance at least $\Omega(\lambda_G)$. We show that unlike with cuts, this inequality is tight up to a constant factor. Specifically, there is a derived graph containing a cut with conductance at most $O(\lambda_G)$.

Oblivious Routing: We show that in any graph, the average length of a flow path in an electrical flow between the endpoints of a random edge is $O(\log^2 n)$. This is a consequence of a more general result which shows that the spectral norm of the entrywise absolute value of the transfer impedance matrix of a graph is $O(\log^2 n)$. This result implies a simple oblivious routing scheme based on electrical flows in the case of transitive graphs.

Random Spanning Tree Sampling: We give an $m^{1+o(1)}\beta^{o(1)}$-time algorithm for generating uniformly random spanning trees in weighted graphs with max-to-min weight ratio β. In

the process, we illustrate how fundamental tradeoffs in graph partitioning can be overcome by eliminating vertices from a graph using Schur complements of the associated Laplacian matrix.

Our starting point is the Aldous-Broder algorithm, which samples a random spanning tree using a random walk. As in prior work, we use fast Laplacian linear system solvers to shortcut the random walk from a vertex v to the boundary of a set of vertices assigned to v called a "shortcutter." We depart from prior work by introducing a new way of employing Laplacian solvers to shortcut the walk. To bound the amount of shortcutting work, we show that most random walk steps occur far away from an unvisited vertex. We apply this observation by charging uses of a shortcutter S to random walk steps in the Schur complement obtained by eliminating all vertices in S that are not assigned to it.

Spectral Sparsification: We introduce a new approach to spectral sparsification that approximates the quadratic form of the pseudoinverse of a graph Laplacian restricted to a subspace. We show that sparsifiers with a near-linear number of edges in the dimension of the subspace exist. Our setting generalizes that of Schur complement sparsifiers. Our approach produces sparsifiers by sampling a uniformly random spanning tree of the input graph and using that tree to guide an edge elimination procedure that contracts, deletes, and reweights edges. In the context of Schur complement sparsifiers, our approach has two benefits over prior work. First, it produces a sparsifier in almost-linear time with no runtime dependence on the desired error. We directly exploit this to compute approximate effective resistances for a small set of vertex pairs in faster time than prior work [33]. Secondly, it yields sparsifiers that are reweighted minors of the input graph. As a result, we give a near-optimal answer to a variant of the Steiner point removal problem.

A key ingredient of our algorithm is a subroutine of independent interest: a near-linear time algorithm that, given a chosen set of vertices, builds a data structure from which we can query a multiplicative approximation to the decrease in the effective resistance between two vertices after identifying all vertices in the chosen set to a single vertex with inverse polynomial additional additive error in near-constant time.

Contents

Acknowledgments

First, I would like to thank Satish Rao and Nikhil Srivastava, who I was extremely fortunate to have as my advisors. I am eternally grateful to them for the freedom that I had to pursue my interests here at Berkeley. Thank you also for steering me back towards the big picture and the right problems. I would be stuck without your help, patience, and generosity.

An enormous number of people have played a huge role in my development. I am particularly indebted to the San Diego Math Circle for getting me excited about math, starting me on the path I am on today, and introducing me to a great group of friends. Thank you to Bernard Chazelle, Moses Charikar, and Philip Klein for introducing me to theory and research. Thank you to Nikhil, Satish, David Aldous, and Luca Trevisan for serving on my committee.

Thank you to everyone that I have worked with in the past and present: Feng Pan, Philip Klein, Eli Fox-Epstein, Christian Sommer, Akshay Ramachandran, Satish, Nikhil, Huan Li, Ravi Kumar, Manish Purohit, Zoya Svitkina, Erik Vee, Amariah Becker, Pak Hay Chan, Lap Chi Lau, Sam Wong, Hong Zhou, Ken-ichi Kawarabayashi, Seri Khoury, Gregory Schwartzman, and Antares Chen. Thank you for sticking with me through long, difficult projects. Thank you for teaching me about math, doing math, writing math, and life. Thank you for exposing me to new areas. Thank you for providing encouraging feedback on my stupidest ideas. Thank you for giving me great problems to work on. Finally, thanks for being so fun to work with.

Thank you Christian for hosting me at Apple. Thank you Manish, Ravi, Zoya, Erik, Morteza, Vahab, Bartek, and Kyle for hosting me at Google. Thank you Feng for hosting me at Los Alamos and to Lap Chi for having me at Waterloo. To all of the anonymous reviewers who have reviewed my papers, thank you!

I am extremely fortunate to have had five very short-seeming years at Berkeley. Thank you to everyone in the Berkeley theory group for making it such a fantastic place to do theory. Thank you Tarun and Richard for listening to me so patiently and for talking to me about optimization.

Thank you to all my friends for making life so good. Thank you to my roomates Cheng, Johnny, Jesse, Aqshems, Jacob, Richard, Alex, Hood, and Yuxin for lots of great nights.

Last, I thank my family, parents, and sister. I would be nowhere without you.

Chapter 1

Introduction

Over the past century, computer science as a field has been interested in obtaining fast algorithms for a myriad of problems and applications. One particularly central area of work has been the study of fast algorithms for graph problems. Graph problems are an important area of study not just due to their immediate applications. Problems on graphs, like the maximum flow problem, are often very general subcases of harder tasks, like linear programming. As a result, better techniques for graph problems have and will continue to be instructive for obtaining better algorithms in general.

In the past decade, many fundamental problems in graph algorithms, like the maximum flow problem [64], have seen dramatically faster algorithms. These improvements began with a near-linear time algorithm for solving special linear systems associated with graphs called Laplacian linear systems [91]. This tool – used as a black box – led to improved algorithms for several problems [30, 64, 72, 23, 46]. For many of these problems, we do not yet have near-linear time algorithms. In this thesis, we investigate the graph-theoretic structure of Laplacian systems in order to obtain better algorithms for some of these problems.

Chapter 2: A Schur Complement Cheeger Inequality

In this chapter, we prove a structural result about graphs which philosophically motivates some of the work of this thesis. Cheeger's Inequality, a classic result in spectral graph theory, links the conductance of cuts that one can find in a graph to the second eigenvalue λ_G of the graph G's normalized Laplacian matrix. It says that all cuts have conductance at least $\lambda_G/2$ and that there exists a cut with conductance at most $\sqrt{2\lambda_G}$. Notice that there is a gap between the upper and lower bounds in this result.

We close this gap by minimizing over a more general set of partitions. Cheeger's Inequality considers the minimum conductance of a cut ∂S defined by a set of vertices S. We instead minimize over pairs of disjoint sets of vertices (S_1, S_2) and consider the conductance of the S_1-S_2 cut in a certain graph H defined using the graph G, which we call the *Schur complement of G onto $S_1 \cup S_2$*. H has vertex set $S_1 \cup S_2$. The edges in G are defined by

computing the Schur complement of the Laplacian matrix of G onto the rows and columns defined by the vertices $S_1 \cup S_2$. H has the combinatorial property that the list of vertices visited by a random walk in H is equivalent in distribution to the list of vertices visited by a random walk in G with vertices outside of $S_1 \cup S_2$ omitted. We call the cut of S_1 in H the *Schur complement cut* of (S_1, S_2) in G. In this section, we show that all Schur complement cuts have conductance at least $\Omega(\lambda_G)$ and that there exists a Schur complement cut with conductance at most $O(\lambda_G)$.

While we do not directly use this result in the rest of this thesis, we do make extensive use of Schur complement cuts in Chapter 4. Intuitively, these cuts are useful in this algorithm due to the fact that they enable lower conductance partitions of graphs.

No coauthor for the result in this chapter, which presents the result originally in [84]. The constant factors in our result were improved in a follow-up work by Miller, Walkington, and Wang [78].

Chapter 3: Oblivious Routing using Electrical Flow

Here, we prove a structural result about electrical flow. We show that in any n-vertex graph, the average length of a flow path in the electrical flow between the endpoints of a random edge is $O(\log^2 n)$. This follows from a more general result about the spectral norm of the entrywise absolute value of the transfer-impedance matrix $BL^\dagger B^T$ of an undirected unweighted graph, where B and L are the incidence and Laplacian matrices of the graph. The spectral norm of this matrix is always at most $O(\log^2 n)$.

This result has a few applications. In edge-transitive unweighted graphs, it implies that electrical flows are $O(\log^2 n)$-competitive oblivious routers. It also implies that in general graphs, electrical flows are $O(\log^2 n)$-competitive ℓ_2-oblivious routers. It is also used in Chapters 4 and 5. The context in which it is used is to accelerate a certain algorithm that iteratively (a) computes an electrical flow and then (b) changes the resistance of some edge by a constant factor, where the edge is chosen using the result of part (a). Our result allows batches of edges to be updated in part (b) instead of just one edge at a time, substantially reducing the runtime of the algorithm. For a more detailed overview of how the result is used, see the introduction of Chapter 5.

This chapter is based on joint work [86] with Satish Rao and Nikhil Srivastava.

Chapter 4: Random Spanning Tree Sampling

We give an $m^{1+o(1)}\beta^{o(1)}$-time algorithm for sampling uniformly random spanning trees in weighted graphs with max-to-min weight ratio β. In the process, we illustrate how fundamental tradeoffs in graph partitioning can be overcome by eliminating vertices from a graph using Schur complements of the associated Laplacian matrix.

Our starting point is the Aldous-Broder algorithm, which samples a random spanning tree using a random walk. As in prior work, we use fast Laplacian linear system solvers to shortcut the random walk from a vertex v to the boundary of a set of vertices assigned to v called a "shortcutter." We depart from prior work by introducing a new way of employing Laplacian solvers to shortcut the walk. To bound the amount of shortcutting work, we show that most random walk steps occur far away from an unvisited vertex. We apply this observation by charging uses of a shortcutter S to random walk steps in the Schur complement obtained by eliminating all vertices in S that are not assigned to it. This charging is possible due to the combinatorial interpretation of Schur complements discussed in the overview to Chapter 2.

No coauthor for the result in this chapter, which presents the result given in [85].

Chapter 5: Spectral Subspace Sparsification

We introduce a new approach to spectral sparsification that approximates the quadratic form of the pseudoinverse of a graph Laplacian restricted to a subspace. We show that sparsifiers with a near-linear number of edges in the dimension of the subspace exist. Our setting generalizes that of Schur complement sparsifiers. Our approach produces sparsifiers by sampling a uniformly random spanning tree of the input graph and using that tree to guide an edge elimination procedure that contracts, deletes, and reweights edges. In the context of Schur complement sparsifiers, our approach has two benefits over prior work. First, it produces a sparsifier in $m^{1+o(1)}$ time in an m-edge graph. We directly exploit this to compute approximate effective resistances for a small set of vertex pairs in faster time than prior work [33], which produced a $(1+\epsilon)$-approximate sparsifier in $\tilde{O}(m+n/\epsilon^2)$ time. Secondly, it yields sparsifiers that are reweighted minors of the input graph. As a result, we give a near-optimal answer to a variant of the Steiner point removal problem.

A key ingredient of our algorithm is a subroutine of independent interest: a near-linear time algorithm that, given a chosen set of vertices, builds a data structure from which we can query a multiplicative approximation to the decrease in the effective resistance between two vertices after identifying all vertices in the chosen set to a single vertex with inverse polynomial additional additive error in near-constant time.

This chapter is based on joint work [68] with Huan Li.

Chapter 2

A Schur Complement Cheeger Inequality

2.1 Introduction

For a set of vertices S, let ϕ_S denote the total weight of edges leaving S divided by the total degree of the vertices in S. Throughout the literature, this quantity is often called the conductance of S. To avoid confusion with electrical conductance, we call this quantity the *fractional conductance* of S. Let ϕ_G denote the minimum fractional conductance of any set S with at most half of the volume (total vertex degree). Let λ_G denote the minimum nonzero eigenvalue of the normalized Laplacian matrix of G. Cheeger's inequality for graphs [5, 4] is as follows:

Theorem 2.1.1 (Cheeger's Inequality)**.** *For any weighted graph G, $\lambda_G/2 \leq \phi_G \leq \sqrt{2\lambda_G}$.*

Cheeger's inequality was originally introduced in the context of manifolds [21]. It is a fundamental primitive in graph partitioning [91, 70] and for upper bounding the mixing time [1] of Markov chains [89]. Motivated by spectral partitioning, much work has been done on higher-order generalizations of Cheeger's inequality [61, 69]. The myriad of applications for Cheeger's inequality and generalizations of it [16, 92], along with the $\Theta(\sqrt{\lambda_G})$ gap between the upper and lower bounds, have led to a long line of work that seeks to improve the quality of the partition found when the spectrum has certain properties (for example, bounded eigenvalue gap [56] or when the graph has special structure [49].)

Here, we get rid of the $\Theta(\sqrt{\lambda_G})$ gap by taking a different approach. Instead of assuming special combinatorial or spectral structure of the input graph to obtain a tighter relationship

[1] Every reversible Markov chain is a random walk on some weighted undirected graph G with vertex set equal to the state space of the Markov chain. The relaxation time of a reversible Markov chain with transition graph G is defined to be $1/\lambda_G$. This quantity is within a $\Theta(\log(\pi_{\min}))$ factor of the mixing time of the chain, where $\pi_{\min}$ is the minimum nonzero entry in the stationary distribution (Theorems 12.3 and 12.4 of [67]).

between fractional conductance and λ_G, we introduce a more general object than graph cuts that enables a tighter approximation to λ_G. Instead of just considering cuts in the given graph G, we consider cuts in certain derived graphs of the input graph obtained by Schur complementing the Laplacian matrix of the graph G onto rows and columns corresponding to a subset of G's vertices. Specifically, pick two disjoint sets of vertices S_1 and S_2, compute the Schur complement of G onto $S_1 \cup S_2$, and look at the cut consisting of all edges between S_1 and S_2 in that Schur complement. Let ρ_G be the minimum fractional conductance of any such cut (defined formally in Section 2.2). We show that the minimum fractional conductance of any such cut is a constant factor approximation to λ_G:

Theorem 2.1.2. *Let G be a weighted graph. Then*

$$\lambda_G/2 \leq \rho_G \leq 25600\lambda_G$$

2.1.1 Effective Resistance Clustering

Our result directly implies a clustering result that relates $1/\lambda_G$ to effective resistances between sets of vertices in the graph G. Think of the weighted graph G as an electrical network, where each edge represents a conductor with electrical conductance equal to its weight. For two sets of vertices S_1 and S_2, obtain a graph H by contracting all vertices in S_1 to a single vertex s_1 and all vertices in S_2 to a single vertex s_2. Let $\texttt{Reff}_G(S_1, S_2)$ denote the effective resistance between the vertices s_1 and s_2 in the graph H. The following is a consequence of our main result:

Theorem 2.1.3. *In any weighted graph G, there are two sets of vertices S_1 and S_2 for which $\texttt{Reff}_G(S_1, S_2) \geq 1/(25600\lambda_G \min(vol_G(S_1), vol_G(S_2)))$. Furthermore, for any pair of sets S_1', S_2', $\texttt{Reff}_G(S_1', S_2') \leq 2/(\lambda_G \min(vol_G(S_1'), vol_G(S_2'))$.*

[19] proved the upper bound present in this result when $|S_1'| = |S_2'| = 1$. We prove Theorem 2.1.3 in Appendix A.1.

2.1.2 Graph Partitioning

Effective resistance in spectral graph theory has been used several times recently (for example [73, 3]) to obtain improved graph partitioning results. $1/\lambda_G$ may not yield a good approximation to the effective resistance between pairs of vertices [19]. For example, on an n-vertex grid graph G, all effective resistances are between $\Omega(1)$ and $O(\log n)$, but $\lambda_G = \Theta(1/n)$. Theorem 2.1.3 closes this gap by considering pairs of *sets* of vertices, not just pairs of vertices.

Cheeger's inequality is the starting point for analysis of spectral partitioning. In some partitioning tasks, cutting the graph does not make sense. For example, spectral partitioning is an important tool in image segmentation [88, 76]. Graph partitioning makes the most sense in image segmentation when one wants to find an object with a sharp boundary. However,

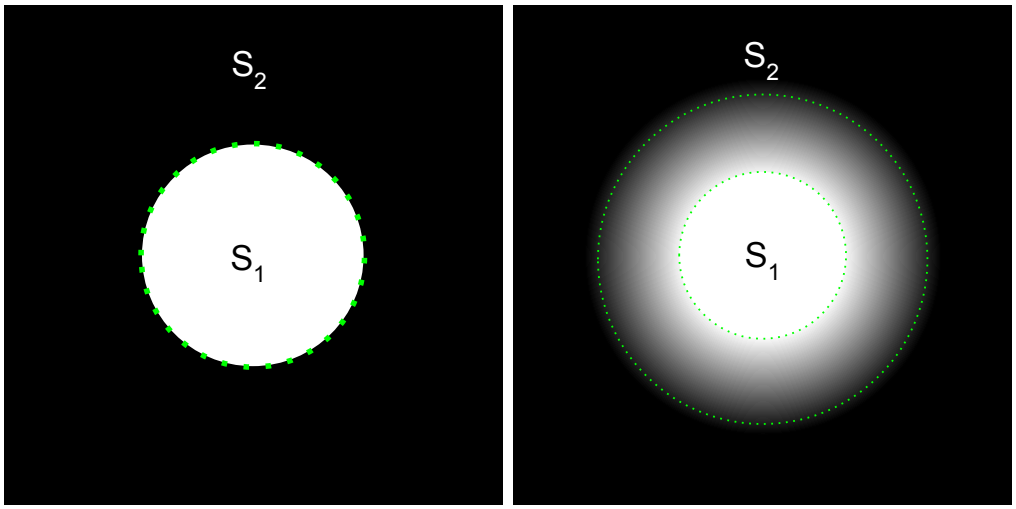

Figure 2.1: Spectral partitioning finds the S_1-S_2 cut in the left image, but may not in the right due to the presence of many equal weight cuts. The minimum fractional conductance Schur complement cut is displayed in both images.

in many images, like the one in Figure 2.1 on the right, objects may have fuzzy boundaries. In these cases, it is not clear which cut an image segmentation algorithm should return.

Considering cuts in Schur complements circumvents this ambiguity. Think of an image as a graph by making a vertex for each pixel and making an edge between adjacent pixels, where the weight on an edge is inversely related to the disparity between the colors of the endpoint pixels for the edge. An optimal segmentation in our setting would consist of the two sets S_1 and S_2 corresponding to pixels on either side of the fuzzy boundary. Computing the Schur complement of the graph onto $S_1 \cup S_2$ eliminates all vertices corresponding to pixels in the boundary.

Some examples in which Cheeger's inequality is not tight illustrate a similar phenomenon in which there are many equally good cuts. For example, let G be an unweighted n-vertex cycle. This is a tight example for the upper bound in Cheeger's inequality, as no cut has fractional conductance smaller than $O(1/n)$ despite the fact that $\lambda_G = \Theta(1/n^2)$. Instead, divide the cycle into four equal-sized quarters and let S_1 and S_2 be two opposing quarters. The Schur complement cut between S_1 and S_2 has fractional conductance at most $O(1/n^2)$, which matches λ_G up to a constant factor.

2.2 Preliminaries

Graph theory: Consider an undirected, connected graph H with edge weights $\{c_e^H\}_{e \in E(H)}$, m edges, and n vertices. Let $V(H)$ and $E(H)$ denote the vertex and edge sets of H re-

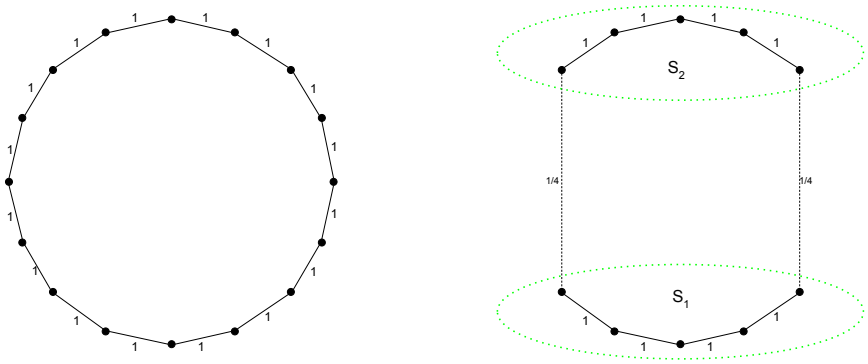

Figure 2.2: A tight example for the upper bound in Cheeger's inequality. The minimum fractional conductance of any cut in this graph is 1/8, while the fractional conductance of the illustrated Schur complement cut on the right is $2(1/4)/(2(1/4)+8(1)) = 1/17 < 1/8$.

spectively. For two sets of vertices $A, B \subseteq V(H)$, let $E_H(A,B)$ denote the set of edges in H incident with one vertex in A and one vertex in B and let $c^H(A,B) := \sum_{e \in E_H(A,B)} c_e^H$. For a set of edges $F \subseteq E(H)$, let $c^H(F) := \sum_{e \in F} c_e^H$. For a set of vertices $A \subseteq V(H)$, let $\partial_H A := E_H(A, V(H) \setminus A)$. For a vertex $v \in V(H)$, let $\partial_H v := \partial_H \{v\}$ denote the edges incident with v in H and let $c_v^H := \sum_{e \in \partial_H v} c_e^H$. For a set of vertices $A \subseteq V(H)$, let $\mathrm{vol}_H(A) := \sum_{v \in A} c_v^H$. When A and B are disjoint, let $H/(A,B)$ denote the graph with all vertices in A identified to one vertex a and all vertices in B identified to one vertex b. Formally, let $H/(A,B)$ be the graph with $V(H/(A,B)) = (V(H) \setminus (A \cup B)) \cup \{a,b\}$, embedding $f : V(H) \to V(H/(A,B))$ with $f(u) := a$ if $u \in A$, $f(u) := b$ if $u \in B$, and $f(u) := u$ otherwise, and edges $\{f(u), f(v)\}$ for all $\{u,v\} \in E(H)$. Let $H/A := H/(A, \emptyset)$.

Laplacians: Let D_H be the $n \times n$ diagonal matrix with rows and columns indexed by vertices in H and $D_H(v,v) = c_v^H$ for all $v \in V(H)$. Let A_H be the adjacency matrix of H; that is the matrix with $A_H(u,v) = c_{uv}^H$ for all $u, v \in V(H)$. Let $L_H := D_H - A_H$ be the Laplacian matrix of H. Let $N_H := D_H^{-1/2} L_H D_H^{-1/2}$ denote the normalized Laplacian matrix of H. For a matrix M, let $M^\dagger$ denote the Moore-Penrose pseudoinverse of M. For subsets A and B of rows and columns of M respectively, let $M[A,B]$ denote the $|A| \times |B|$ submatrix of M restricted to those rows and columns. For a set of vertices $S \in V(H)$, let $\mathbf{1}_S$ denote the indicator vector for the set S. For two vertices $u, v \in \mathbb{R}^n$, let $\chi_{uv} := \mathbf{1}_{\{u\}} - \mathbf{1}_{\{u\}}$. When the graph is clear from context, we omit H from all of the subscripts and superscripts of H. For a vector $x \in \mathbb{R}^n$, let $x_S \in \mathbb{R}^S$ denote the restriction of x to the coordinates in S.

Let λ_H denote the smallest nonzero eigenvalue of N_H. Equivalently,

$$\lambda_H := \min_{x \in \mathbb{R}^n : x^T D_H^{1/2} \mathbf{1}_{V(H)} = 0} \frac{x^T N_H x}{x^T x}$$

For any set of vertices $X \subseteq V(H)$, let

$$L_{\texttt{Schur}(H,X)} := L_H[X,X] - L_H[X, V(H) \setminus X] L_H[V(H) \setminus X, V(H) \setminus X]^{-1} L_H[V(H) \setminus X, X]$$

where brackets denote submatrices with the indexed rows and columns. The following fact applies specifically to Laplacian matrices:

Remark 1 (Fact 2.3.6 of [57])**.** *For any graph H and any $X \subseteq V(H)$, $L_{\texttt{Schur}(H,X)}$ is the Laplacian matrix of an undirected graph.*

Let $\texttt{Schur}(H, X)$ denote the graph referred to in Remark 1. Schur complementation commutes with edge contraction and deletion and is associative:

Theorem 2.2.1 (Lemma 4.1 of [27], statement from [57])**.** *Given H, $S \subseteq V(H)$, and any edge e with both endpoints in S,*

$$\texttt{Schur}(H \setminus e, S) = \texttt{Schur}(H, S) \setminus e$$

and, for any pair of vertices $x, y \in S$,

$$\texttt{Schur}(H/\{x,y\}, S) = \texttt{Schur}(H, S)/\{x,y\}$$

Theorem 2.2.2. *Given H and two sets of vertices $X \subseteq Y \subseteq V(H)$, $\texttt{Schur}(\texttt{Schur}(H, Y), X) = \texttt{Schur}(H, X)$.*

The following property follows from the definition of Schur complements:

Remark 2. *Let H be a graph and $S \subseteq V(H)$. Let $I := \texttt{Schur}(H, S)$. For any $x \in \mathbb{R}^{V(H)}$ that is supported on S with $x^T \mathbf{1}_{V(H)} = 0$,*

$$x^T L_H^\dagger x = x_S^T L_I^\dagger x_S$$

The weight of edges in this graph can be computed using the following folklore fact, which we prove for completeness:

Theorem 2.2.3. *For two disjoint sets $C, D \subseteq V(H)$, let $I := \texttt{Schur}(H, C \cup D)$. Then*

$$c^I(C,D) = \frac{1}{\chi_{cd}^T L_{H/(C,D)}^\dagger \chi_{cd}}$$

Proof. By definition, $c^I(C,D) = c^{I/(C,D)}(\{c\},\{d\})$. By Theorem 2.2.1, $I/(C,D) = \texttt{Schur}(H/(C,D), \{c,d\})$. By Remark 4, $c^{\texttt{Schur}(H/(C,D),\{c,d\})}(\{c\},\{d\}) = \frac{1}{\chi_{cd}^T L_{H/(C,D)}^\dagger \chi_{cd}}$. Combining these equalities gives the desired result. □

We also use the following folklore fact about electrical flows, which we prove for the sake of completeness:

Theorem 2.2.4. *For two vertices $s, t \in V(H)$,*

$$\chi_{st}^T L_H^\dagger \chi_{st} = \frac{1}{\min_{p \in \mathbb{R}^{V(H)}: p_s \leq 0, p_t \geq 1} p^T L_H p}$$

Proof. We first show that

$$\chi_{st}^T L_H^\dagger \chi_{st} = \frac{1}{\min_{p \in \mathbb{R}^{V(H)}: p_s = 0, p_t = 1} p^T L_H p}$$

Taking the gradient of the objective $p^T L_H p$ shows that that the optimal p are the potentials for an electrical flow with flow conservation at all vertices besides s and t. Therefore, p is proportional to $L_H^\dagger \chi_{st} + \gamma \mathbf{1}$ for some $\gamma \in \mathbb{R}$. The constant of proportionality is $\chi_{st}^T L_H^\dagger \chi_{st}$ since the s-t potential drop in p is 1. Therefore,

$$\begin{aligned}\min_{p \in \mathbb{R}^{V(H)}: p_s = 0, p_t = 1} p^T L_H p &= \left(\frac{L_H^\dagger \chi_{st}}{\chi_{st}^T L_H^\dagger \chi_{st}}\right)^T L_H \left(\frac{L_H^\dagger \chi_{st}}{\chi_{st}^T L_H^\dagger \chi_{st}}\right) \\ &= \frac{1}{\chi_{st}^T L_H^\dagger \chi_{st}}\end{aligned}$$

The desired result follows from the fact that in the optimal p, all potentials are between 0 and 1 inclusive. □

Notions of fractional conductance: For a set of vertices $A \subseteq V(H)$, let

$$\phi_A^H := \frac{c^H(\partial_H(A))}{\min(\mathrm{vol}_H(A), \mathrm{vol}_H(V(H) \setminus A))}$$

be the *fractional conductance* of A. Let

$$\phi_H := \min_{A \subseteq V(H): A \neq \emptyset} \phi_A^H$$

be the *fractional conductance* of H.

For two disjoint sets of vertices $A, B \subseteq V(H)$, let $I := \texttt{Schur}(H, A \cup B)$ and

$$\rho_{A,B}^H := \frac{c^I(A, B)}{\min(\mathrm{vol}_I(A), \mathrm{vol}_I(B))}$$

be the *Schur complement fractional conductance* of the pair of sets (A, B). Define the *Schur complement fractional conductance* of the graph H to be

$$\rho_H := \min_{A, B \subseteq V(H): A \cap B = \emptyset, A \neq \emptyset, B \neq \emptyset} \rho_{A,B}^H$$

It will be helpful to deal with the quantities

$$\sigma^H_{A,B} := \frac{c^I(A,B)}{\min(\mathrm{vol}_H(A), \mathrm{vol}_H(B))}$$

and

$$\sigma_H := \min_{A,B \subseteq V(H): A \cap B = \emptyset, A \neq \emptyset, B \neq \emptyset} \sigma^H_{A,B}$$

as well, which we call the *mixed fractional conductances* of (A, B) and H respectively.

The following will be useful in relating $\rho^H_{A,B}$ to $\sigma^H_{A,B}$:

Proposition 2.2.5. *For any two sets $X \subseteq Y \subseteq V(H)$, let $I := \texttt{Schur}(H, Y)$. Then,*

$$vol_I(X) \leq vol_H(X)$$

Proof. It suffices to show this result when $|X| = 1$ because vol is a sum of volumes (degrees) of vertices in the set. Furthermore, by Theorem 2.2.2, it suffices to show the result when $|Y| = |V(H)| - 1$. Let v be the unique vertex in H outside of Y and let u be the unique vertex in X. Then, by definition of the Schur complement,

$$\begin{aligned}
\mathrm{vol}_I(X) &= c^I_u \\
&= \sum_{w \in V(I)} c^I_{uw} \\
&= \sum_{w \in V(I)} \left(c^H_{uw} + \frac{c^H_{uv} c^H_{vw}}{c^H_v} \right) \\
&= \left(\sum_{w \in V(I)} c^H_{uw} \right) + \frac{c^H_{uv}}{c^H_v} \left(\sum_{w \in V(I)} c^H_{vw} \right) \\
&\leq \left(\sum_{w \in V(I)} c^H_{uw} \right) + c^H_{uv} \\
&= c^H_u \\
&= \mathrm{vol}_H(X)
\end{aligned}$$

as desired. □

To prove the upper bound, we given an algorithm for constructing a low fractional conductance Schur complement cut. The following result is helpful for making this algorithm take near-linear time:

Theorem 2.2.6 (Theorem 8.2 of [95]). *Given a graph H, there is a $\tilde{O}(m)$-time algorithm that produces a vector $x \leftarrow$ `ApxFiedler`$(H) \in \mathbb{R}^{V(H)}$ with $x^T D_H^{1/2} \mathbf{1}_{V(H)} = 0$ for which*

$$x^T N_H x \leq 2\lambda_H x^T x$$

2.3 Lower bound

We now show the first inequality in Theorem 2.1.2, which follows from the following lemma by Proposition 2.2.5, which implies that $\sigma_G \leq \rho_G$.

Lemma 2.3.1.

$$\lambda_G \leq 2\sigma_G$$

Proof. We lower bound the Schur complement fractional conductance of any pair of disjoint sets $A, B \subseteq V(G)$. Let $I := \texttt{Schur}(G, A \cup B)$. Let P be the $(A \cup B) \times (A \cup B)$ diagonal matrix with $P(u,u) = c_u^G$ for each $u \in A \cup B$. We start by lower bounding the minimum nonzero eigenvalue λ of the matrix $P^{-1/2} L_I P^{-1/2}$. Let $\lambda_{\max}(M)$ denote the maximum eigenvalue of a symmetric matrix M. By definition of the Moore-Penrose pseudoinverse,

$$1/\lambda = \lambda_{\max}(P^{1/2} L_I^\dagger P^{1/2})$$

By Remark 4,

$$\lambda_{\max}(P^{1/2} L_I^\dagger P^{1/2}) \leq \lambda_{\max}(N_G^\dagger) = 1/\lambda_G$$

Therefore, $\lambda \geq \lambda_G$. We now plug in a test vector. Let

$$z := P^{1/2} \left(\frac{\mathbf{1}_A}{\mathrm{vol}_G(A)} - \frac{\mathbf{1}_B}{\mathrm{vol}_G(B)} \right)$$

$z^T (P^{1/2} \mathbf{1}_{V(I)}) = 0$, so

$$\begin{aligned}
\lambda_G &\leq \lambda \\
&= \min_{x \in \mathbb{R}^{A \cup B} : x^T P^{1/2} \mathbf{1}_{V(I)} = 0} \frac{x^T (P^{-1/2} L_I P^{-1/2}) x}{x^T x} \\
&\leq \frac{z^T (P^{-1/2} L_I P^{-1/2}) z}{z^T z} \\
&= \frac{c^I(A,B) \left((1/\mathrm{vol}_G(A)) + (1/\mathrm{vol}_G(B)) \right)^2}{(\mathrm{vol}_G(A)/\mathrm{vol}_G(A)^2) + (\mathrm{vol}_G(B)/\mathrm{vol}_G(B)^2)} \\
&= \frac{c^I(A,B) \mathrm{vol}_G(A \cup B)}{\mathrm{vol}_G(A) \mathrm{vol}_G(B)} \\
&\leq 2\sigma_{A,B}^G
\end{aligned}$$

□

2.4 Upper bound

We now show the second inequality in Theorem 2.1.2:

Lemma 2.4.1.

$$\rho_G \leq 25600\lambda_G$$

To prove this lemma, we need to find a pair of sets A and B with low Schur complement fractional conductance:

Lemma 2.4.2. *There is a near-linear time algorithm* `SweepCut`(G) *that takes in a graph* G *with* $\lambda_G \leq 1/25600$ *and outputs a pair of nonempty sets* A *and* B *with the following properties:*

- *(Low Schur complement fractional conductance)* $\sigma^G_{A,B} \leq 640\lambda_G$
- *(Large interior)* $\phi^G_A \leq 1/4$ *and* $\phi^G_B \leq 1/4$

We now prove Lemma 2.4.1 given Lemma 2.4.2:

Proof of Lemma 2.4.1 given Lemma 2.4.2. Let $I := \texttt{Schur}(G, A \cup B)$. For any two vertices $u, v \in A \cup B$, $c^I_{uv} \geq c^G_{uv}$. Therefore, $\text{vol}_I(A) \geq 2\sum_{u,v\in A} c^G_{uv}$ and $\text{vol}_I(B) \geq 2\sum_{u,v\in B} c^G_{uv}$. By the "Large interior" guarantee of Lemma 2.4.2, $2\sum_{u,v\in A} c^G_{uv} \geq (3/4)\text{vol}_G(A)$ and $2\sum_{u,v\in B} c^G_{uv} \geq (3/4)\text{vol}_G(B)$. Therefore,

$$\rho^G_{A,B} \leq 4/3\sigma^G_{A,B} \leq 1280\lambda_G$$

by the "Low Schur complement fractional conductance" guarantee when $\lambda_G \leq 1/25600$, as desired. When $\lambda_G > 1/25600$, the lemma is trivially true, as desired. □

Now, we implement `SweepCut`. The standard Cheeger sweep examines all thresholds $q \in \mathbb{R}$ and for each threshold, computes the fractional conductance of the cut $\partial S_{\leq q}$ of edges from vertices with eigenvector coordinate at most q to ones greater than q. Instead, the algorithm `SweepCut` examines all thresholds $q \in \mathbb{R}$ and computes an upper bound (a proxy) for the $\sigma^G_{S_{\leq q/2}, S_{\geq q}}$ for each positive q and $\sigma^G_{S_{\leq q}, S_{\geq q/2}}$ for each negative q. Let $I_q := \texttt{Schur}(G, S_{\geq q} \cup S_{\leq q/2})$ for $q > 0$ and $I_q := \texttt{Schur}(G, S_{\leq q} \cup S_{\geq q/2})$. Let $\kappa_q(y) := \min(q, \max(q/2, y))$ for $q > 0$ and $\kappa_q(y) = \min(q/2, \max(q, y))$ for $q \leq 0$. The proxy is the following quantity, which is defined for a specific shift of the Rayleigh quotient minimizer $y \in \mathbb{R}^{V(G)}$.

$$\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2}) := \frac{4}{q^2} \sum_{e=uv\in E(G)} c^G_e(\kappa_q(y_u) - \kappa_q(y_v))^2$$

for $q > 0$ and

$$\widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2}) := \frac{4}{q^2} \sum_{e=uv\in E(G)} c^G_e(\kappa_q(y_u) - \kappa_q(y_v))^2$$

for $q \leq 0$. We now show that this is indeed an upper bound:

Proposition 2.4.3. *For all $q > 0$,*

$$c^{I_q}(S_{\leq q/2}, S_{\geq q}) \leq \widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$$

For all $q \leq 0$,

$$c^{I_q}(S_{\leq q}, S_{\geq q/2}) \leq \widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2})$$

Proof. We focus on the $q > 0$, as the reasoning for the $q \leq 0$ case is the same. By Theorems 2.2.3 and 2.2.4,

$$c^{I_q}(S_{\leq q/2}, S_{\geq q}) = \min_{p \in \mathbb{R}^{V(G)}: p_a \leq 0 \forall a \in S_{\leq q/2}, p_a \geq 1 \forall a \in S_{\geq q}} p^T L_G p$$

The vector p with $p_a := \frac{2}{q}\kappa_q(y_a) - 1$ for all vertices $a \in V(G)$ is a feasible solution to the above optimization problem with objective value $\widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$. This is the desired result. □

This proxy allows us to relate Schur complement fractional conductances together across different thresholds q in a similar proof to the proof of the upper bound of Cheeger's inequality given in [93]. One complication in our case is that Schur complements for different values of q overlap in their eliminated vertices. Our choice of $\leq q/2$, $\geq q$ plays a key role here (as opposed to ≤ 0, $\geq q$, for example) in ensuring that the overlap is small. We now give the algorithm `SweepCut`:

Algorithm 1: `SweepCut`(G)

Input: A graph G with $\lambda_G \leq 1/25600$

Output: Two sets of vertices A and B satisfying the guarantees of Lemma 2.4.2

$z \leftarrow$ vector with $z^T N_G z \leq 2\lambda_G z^T z$ and $z^T (D_G^{1/2} \mathbf{1}_{V(G)}) = 0$

$x \leftarrow D_G^{-1/2} z$

$y \leftarrow x - \alpha \mathbf{1}_{V(G)}$ for a value α such that $\mathrm{vol}_G(\{v : y_v \leq 0\}) \geq \mathrm{vol}_G(V(G))/2$ and $\mathrm{vol}_G(\{v : y_v \geq 0\}) \geq \mathrm{vol}_G(V(G))/2$

foreach $q \in \mathbb{R}$ **do**

  $S_{\geq q} \leftarrow$ vertices with $y_v \geq q$

  $S_{\leq q} \leftarrow$ vertices with $y_v \leq q$

end

foreach $q > 0$ **do**

  if *(1)* $\widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q}) \leq 640\lambda_G \min(vol_G(S_{\leq q/2}), vol_G(S_{\geq q}))$*, (2)* $c^G(\partial S_{\geq q/2}) \leq 1/4 vol_G(S_{\geq q})$*, and (3)* $\phi_{S_{\geq q}} \leq 1/4$ **then**

    return $(S_{\leq q/2}, S_{\geq q})$

  end

end

foreach $q \leq 0$ **do**

  if *(1)* $\widehat{c}^{I_q}(S_{\geq q/2}, S_{\leq q}) \leq 640\lambda_G \min(vol_G(S_{\geq q/2}), vol_G(S_{\leq q}))$*, (2)* $c^G(\partial S_{\geq q/2}) \leq 1/4 vol_G(S_{\leq q})$*, and (3)* $\phi_{S_{\leq q}} \leq 1/4$ **then**

    return $(S_{\leq q}, S_{\geq q/2})$

  end

end

Our analysis relies on the following key technical result, which we prove in Appendix A.2:

Proposition 2.4.4. *For any* $a, b \in \mathbb{R}$*,*

$$\int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq \leq 10(a-b)^2$$

Proof of Lemma 2.4.2. **Algorithm well-definedness.** We start by showing that `SweepCut` returns a pair of sets. Assume, for the sake of contradiction, that `SweepCut` does not return a pair of sets. Let $I_q := \texttt{Schur}(G, S_{\geq q} \cup S_{\leq q/2})$ for $q > 0$ and $I_q := \texttt{Schur}(G, S_{\leq q} \cup S_{\geq q/2})$ for $q \leq 0$. By the contradiction assumption, for all $q > 0$,

$$\mathrm{vol}_G(S_{\geq q}) \leq \frac{\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})}{640\lambda_G} + 4c^G(\partial S_{\geq q}) + 4c^G(\partial S_{\leq q/2})$$

and for all $q < 0$,

$$\mathrm{vol}_G(S_{\leq q}) \leq \frac{\widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2})}{640\lambda_G} + 4c^G(\partial S_{\leq q}) + 4c^G(\partial S_{\geq q/2})$$

Since $\sum_{v\in V(G)} c_v^G x_v = 0$,

$$\sum_{v\in V(G)} c_v^G x_v^2 \leq \sum_{v\in V(G)} c_v^G y_v^2$$

Now, we bound the positive y_v and negative y_v parts of this sum separately. Negating y shows that it suffices to bound the positive part. Order the vertices in $S_{\geq 0}$ in decreasing order by y_v value. Let v_i be the ith vertex in this ordering, let $k := |S_{\geq 0}|$, $y_{k+1} := 0$, $y_i := y_{v_i}$, $c_i := c_{v_i}^G$, and $S_i := \{v_1, v_2, \ldots, v_i\}$ for each integer $i \in [k]$. Then

$$\begin{aligned}
\sum_{v\in S_{\geq 0}} c_v^G y_v^2 &= \sum_{i=1}^{k} c_i y_i^2 \\
&= \sum_{i=1}^{k} (\mathrm{vol}_G(S_i) - \mathrm{vol}_G(S_{i-1})) y_i^2 \\
&= \sum_{i=1}^{k} \mathrm{vol}_G(S_i)(y_i^2 - y_{i+1}^2) \\
&= 2\int_0^\infty \mathrm{vol}_G(S_{\geq q}) q dq
\end{aligned}$$

By our volume upper bound from above,

$$\begin{aligned}
2\int_0^\infty \mathrm{vol}_G(S_{\geq q}) q dq &\leq 2\int_0^\infty \frac{\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})}{640\lambda_G} q dq + 8\int_0^\infty c^G(\partial S_{\geq q}) q dq + 8\int_0^\infty c^G(\partial S_{\leq q/2}) q dq \\
&= 2\int_0^\infty \frac{\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})}{640\lambda_G} q dq + 8\int_0^\infty c^G(\partial S_{\geq q}) q dq + 8\int_0^\infty c^G(\partial S_{> q/2}) q dq \\
&= 2\int_0^\infty \frac{\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})}{640\lambda_G} q dq + 40\int_0^\infty c^G(\partial S_{\geq q}) q dq
\end{aligned}$$

Substitution and Proposition 2.4.4 show that

$$\begin{aligned}
2\int_0^\infty \mathrm{vol}_G(S_{\geq q}) q dq &\leq 8 \sum_{e=uv\in E(G)} c_e^G \int_0^\infty \left(\frac{(\kappa_q(y_u) - \kappa_q(y_v))^2}{640\lambda_G q} + 5\mathbb{1}_{q\in[y_u, y_v]} q \right) dq \\
&\leq 8 \sum_{e=uv\in E(G)} c_e^G \left(\frac{10}{640\lambda_G} (y_u - y_v)^2 + 5|y_u^2 - y_v^2| \right)
\end{aligned}$$

By Cauchy-Schwarz,

$$
\begin{aligned}
&8 \sum_{e=uv\in E(G)} c_e^G \left(\frac{10}{640\lambda_G}(y_u - y_v)^2 + 5|y_u^2 - y_v^2| \right) \\
&\leq \frac{1}{8\lambda_G} \sum_{e=uv\in E(G)} c_e^G (y_u - y_v)^2 \\
&+ 40 \sqrt{\sum_{e=uv\in E(G)} c_e^G (y_u - y_v)^2} \sqrt{\sum_{e=uv\in E(G)} c_e^G (y_u + y_v)^2} \\
&\leq \frac{1}{4} \sum_{v\in V(G)} c_v^G x_v^2 \\
&+ 80\sqrt{\lambda_G} \sqrt{\sum_{v\in V(G)} c_v^G x_v^2} \sqrt{\sum_{v\in V(G)} c_v^G y_v^2}
\end{aligned}
$$

But since $\sum_{v\in V(G)} c_v^G x_v^2 \leq \sum_{v\in V(G)} c_v^G y_v^2$ and $\lambda_G < 1/25600$,

$$
\frac{1}{4} \sum_{v\in V(G)} c_v^G x_v^2 + 80\sqrt{\lambda_G} \sqrt{\sum_{v\in V(G)} c_v^G x_v^2} \sqrt{\sum_{v\in V(G)} c_v^G y_v^2} < \frac{1}{2} \sum_{v\in V(G)} c_v^G y_v^2
$$

Negating y shows that $\sum_{v\in S_{\leq 0}} c_v^G y_v^2 < 1/2 \sum_{v\in V(G)} c_v^G y_v^2$ as well. But these statements cannot both hold; a contradiction. Therefore, `SweepCut` must output a pair of sets.

Runtime. Computing z takes $\tilde{O}(m)$ time by Theorem 2.2.6. Therefore, it suffices to show that the foreach loops can each be implemented in $O(m)$ time. This implementation is similar to the $O(m)$-time implementation of the Cheeger sweep.

We focus on the first foreach loop, as the second is the same with q negated. First, note that the functions $\phi_{S_{\geq q}}$, $c^G(\partial S_{\geq q/2})$, and $\mathrm{vol}_G(S_{\geq q})$ of q are piecewise constant, with breakpoints at $q = y_u$ and $q = 2y_u$ for each $u \in V(G)$. Furthermore, these functions can be computed for all values in $O(m)$ time using an $O(m)$-time Cheeger sweep for each function.

Therefore, it suffices to compute the value of $\widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$ for all $q \geq 0$ that are local minima in $O(m)$ time. Let $h(q) := \widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$. Notice that the functions $h(q)$ and $h'(q)$ are piecewise quadratic and linear functions of q respectively, with breakpoints at $q = y_u$ and $q = 2y_u$. Using five $O(m)$-time Cheeger sweeps, one can compute the q^2, q and 1 coefficients of $h(q)$ and the q and 1 coefficients of $h'(q)$ between all pairs of consecutive breakpoints. After computing these coefficents, one can compute the value of each function at a point q in $O(1)$ time. Furthermore, given two consecutive breakpoints a and b, one can find all points $q \in (a, b)$ with $h'(q) = 0$ in $O(1)$ time. Each local minimum for h is either a breakpoint or a point with $h'(q) = 0$. Since h and h' have $O(n)$ breakpoints, all local minima can be computed in $O(n)$ time. h can be evaluated at all of these points in $O(n)$ time. Therefore,

all local minima of h can be computed in $O(m)$ time. Since the algorithm does return a q, some local minimum for h also suffices, so this implementation produces the desired result in $O(m)$ time.

Low Schur complement fractional conductance. By Proposition 2.4.3,

$$c^{I_q}(S_{\geq q}, S_{\leq q/2}) \leq \widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})$$

Therefore, $c^{I_q}(S_{\geq q}, S_{\leq q/2}) \leq 640\lambda_G \min(\mathrm{vol}_G(S_{\geq q}), \mathrm{vol}_G(S_{\leq q/2}))$ for $q \geq 0$ by the foreach loop if condition. Repeating this reasoning for $q < 0$ yields the desired result.

Large interior. By definition of α, $\mathrm{vol}_G(S_{\geq q}) \leq \mathrm{vol}_G(S_{\leq q/2})$ for $q > 0$. Since $c^G(\partial S_{\leq q/2}) \leq 1/4\mathrm{vol}_G(S_{\geq q})$, $\phi_{S_{\leq q/2}} \leq 1/4$, as desired.

□

Chapter 3

Oblivious Routing using Electrical Flow

3.1 Introduction

Electrical Flows have have played an important role in several recent advances in graph algorithms — for instance, in the context of exact and approximate maximum flow/minimum cut [23, 63, 72], multicommodity flow [47], oblivious routing [42, 60, 50], graph sparsification [90], and random spanning tree generation [46, 73]. This is due to the emergence of nearly linear time Laplacian solvers for computing them, beginning with the work of Spielman and Teng [91], and also to their well-known connections with random walks. Using them to solve combinatorial problems is not typically immediate, and may be likened to putting a square peg into a round hole: at a high level, many of the traditional problems of computer science are concerned with finding flows in graphs with controlled ℓ_1 and ℓ_∞ norms (corresponding to distance and congestion, respectively), whereas electrical flows minimize the ℓ_2 norm (energy). Reducing one to the other often requires some sort of iterative method for combining many electrical flows with varying demands and graphs.

In this work, we ask the following basic structural question about electrical flows in arbitrary unweighted graphs:

> What is the typical ℓ_1 norm of the unit current electrical flow between two neighboring vertices u, v in a graph?

Recall that the ℓ_1 norm of a unit circulation-free flow is the average distance traveled by the flow, since any such flow $f_{uv} : E \to \mathbb{R}$ may be decomposed as a convex combination of paths which all have the same direction of flow on every edge:

$$f_{uv} = \sum_{i \in \mathcal{P}_{uv}} \alpha_i f_i,$$

where $\mathcal{P}_{uv}$ is the set of simple paths from u to v, and we have

$$\|f_{uv}\|_1 = \sum_{i \in \mathcal{P}_{uv}} \alpha_i \|f_i\|_1 = \sum_{i \in \mathcal{P}_{uv}} \alpha_i \text{length}(f_i).$$

Thus, this question asks when/whether electrical flows in a graph travel a greater distance than shortest paths, and by how much.

3.1.1 Three Examples

To get a feel for this problem, and to set the context for our result and its proof, we begin by presenting three instructive examples. We will use the notation b_v for the indicator vector of a vertex, $b_{uv} = b_u - b_v$ for the signed incidence vector of an edge uv, B for the $m \times n$ signed edge-vertex incidence matrix of a graph (where the edges are oriented arbitrarily), and $L = B^T B$ for the Laplacian matrix. For any pair of vertices u, v, we will use the notation $\Delta(u, v) := \|f_{uv}\|_1 / \text{dist}(u, v)$, where f_{uv} is the unit electrical flow between u and v and dist is the shortest path distance in the graph.

The first example shows that in general $\Delta(u, v)$ can be quite large for the worst-case edge in a graph.

Example 3.1.1 (Parallel Paths)**.** *Consider the graph consisting of a single edge between vertices u and v and $\sqrt{m}$ disjoint parallel paths of length $\sqrt{m}$ with endpoints u and v. Since the effective resistance of the parallel paths taken together is 1, half of the unit flow between u and v will use the paths, assigning a flow of $1/2\sqrt{m}$ to each path, and the other half will traverse the ege uv. Thus, we have $\Delta(u, v) = (\sqrt{m} + 1)/2$. However, notice that for most of the other edges in the graph, Δ is tiny. For instance, for any edge ab near the middle of one of the parallel paths, a $1 - O(1/\sqrt{m})$ fraction of the flow will traverse the single edge, so we will have $\Delta(a, b) = O(1)$.*

On the other hand, $\Delta(u, v)$ is uniformly bounded for every edge in an expander.

Example 3.1.2 (Expander Graphs)**.** *Let G be a constant degree $d-$regular expander graph with transition matrix P satisfying $\|P - J/n\| \leq \lambda$ for some constant λ, where J is the all ones matrix. Letting $Q := P - J/n$ and $E = I - J/n$, we have the power series expansion orthogonal to the all ones vector:*

$$(L/d)^+ = (E - Q)^+ = E + \sum_{k \geq 1} Q^k.$$

Now for every edge uv we calculate the electrical flow across its endpoints:

$$\|BL^+ b_{uv}\|_1 \leq \|B\|_{1\to 1} \|L^+\|_{1\to 1} \|b_{uv}\|_1,$$

where $\|\cdot\|_{1\to 1}$ *is the* $1\to 1$ *operator norm, i.e., maximum column sum of the matrix. Let* $T = O(\log n)$ *be the mixing time of* G*, after which* $\|P^T - J/n\|_{1\to 1} = \|Q^T\|_{1\to 1} \leq 1/n$*. Noting that* $\|B\|_{1\to 1} \leq d$ *and* $Q^k = P^k - J/n$ *and applying the triangle inequality, we obtain:*

$$\|BL^+ b_{uv}\|_1 \leq \frac{2d}{d}\sum_{k=0}^{T}(\|P^k\|_{1\to 1} + \|J/n\|_{1\to 1}) + \|L^+\|_{1\to 1}\cdot\|Q^T\|_{1\to 1}.$$

Since P^k *is a doubly stochastic matrix we have* $\|P^k\|_{1\to 1} = 1$ *for all* k*. Moreover,* $\|L^+\|_{1\to 1} \leq \sqrt{n}\|L^+\| \leq \sqrt{n}/\lambda$*. Combining these facts, we get a final bound of* $\Delta(u,v) = O(\log n)$*, for every edge* $uv \in G$*.*

We remark that bounds similar to (and somewhat more general than) the above were shown in the papers [60, 50] using different techniques.

Finally, we note that there are highly non-expanding graphs for which $\Delta(u,v)$ is also uniformly bounded, which means that expansion does not tell the whole story.

Example 3.1.3 (The Grid)**.** *Let* G *be the* $n\times n$ *two dimensional torus (i.e., grid with sides identified). Then it is shown in [60] that for every edge* $uv \in G$ *we have* $\Delta(u,v) = O(\log n)$*, even though* G *is clearly not an expander. We briefly sketch an argument explaining where this bound comes from. Let* uv *be any horizontal edge in* G*, and let* w *be a vertex in* G *at vertical distance* k *from* u *and* v*. We will show that the potential at* w *in the unit current* uv*-electrical flow is small, in particular that*

$$\phi(w) := b_w^T L^+ b_{uv} = O(1/k^2).$$

First we recall (see, e.g., [17] Chapter IX) that the potential at a vertex w *when* u, v *are fixed to potentials* $-1, 1$ *is:* $2(\mathbb{P}_w(t_v < t_u) - \mathbb{P}_w(t_u < t_v))$*, where* $\mathbb{P}_w$ *is the law of the random walk started at* w *and* t_u *is the first time at which the walk hits* u*. By Ohm's law, this means that:*

$$\phi(w) \leq \left|\frac{2}{\mathrm{Reff}(u,v)}(\mathbb{P}_w(t_v < t_u) - \mathbb{P}_w(t_u < t_v))\right|,$$

where $\mathrm{Reff}(u,v) := b_{uv}^T L^+ b_{uv}$ *is the effective resistance of the edge* uv*. Since the resistance of every edge in the grid is equal to* $1/2$*, we find that* $|\phi(w)| = O(|\mathbb{P}_w(t_v < t_w) - \mathbb{P}_w(t_w < t_v)|)$*.*

We now analyze these probabilities. Roughly speaking, the random walk from w *will take time* $\Omega(k^2)$ *to reach the horizontal line* H *containing* uv*, at which point its horizontal distance (along* H*) from* w *will be distributed as a* k^2*-step random horizontal random walk centered at* w *(since about half of the steps of the random walk up to that point will be horizontal). The difference in probabilities between any two neighboring points in* H *will therefore be at most* $O(1/k^2)$*, which implies the bound on* $|\phi(x)|$*. Consequently, the potential difference across any edge* wx *at distance* k *is at most* $O(1/k^2)$*; since there are* $O(k)$ *edges at distance* k*, the total contribution from such edges is* $O(1/k)$*, and summing over all distances* k *(and repeating the argument for vertical edges) yields a bound of* $O(\log n)$*.*

3.1.2 Our Results

Our first theorem is the following.

Theorem 3.1.4. *If $G = (V, E)$ is an unweighted graph with m edges, then*

$$\sum_{uv \in E} \Delta(u, v) \leq O(m \log^2 n).$$

This theorem formalizes the intuition in the parallel paths example that there cannot be too many edges in a graph for which the electrical flow uses very long paths. A corollary for edge-transitive graphs is that the above bound holds for *every* edge, by symmetry. This generalizes our analysis on the grid (which used very specific properties) to a much broader category which includes all Cayley graphs.

Theorem 3.1.4 is a consequence of a more general result concerning the weighted transfer impedance matrix of a graph. Given a weighted graph $G = (V, E, c)$ with edge weights $c_e \geq 0$, let C be an $m \times m$ diagonal matrix containing the edge weights. Then $L = B^T C B$ is the Laplacian matrix of G and the *weighted transfer impedance matrix* is the $m \times m$ matrix defined as:

$$\Pi = C^{1/2} B L^+ B^T C^{1/2}.$$

It is well-known and easy to see that the entry $(BL^+B)(e, f)$ is the potential difference across the ends of edge e when a unit current is injected across the ends of edge f, and vice versa, and that Π is a projection matrix with trace $n-1$. In particular, the latter fact implies that $\|\Pi\| = 1$, where $\|\cdot\|$ is the spectral norm.

Let $\overline{\Pi}$ be the entrywise absolute value matrix of Π. Our main theorem is:

Theorem 3.1.5. *For an arbitrary weighted graph G,*

$$\|\overline{\Pi}\| = O(\log^2 n)$$

Theorem 3.1.4 follows from Theorem 3.1.5 by plugging in the the all ones vector $u = (1, \ldots, 1)^T$:

$$u^T \overline{\Pi} u = \sum_{e \in E} \|\Pi_e\|_1 = \sum_{e \in E} \Delta(e),$$

where $\Pi_e = BL^+ b_{uv}$ is the row of Π corresponding to $e = uv$, i.e., the electrical flow across the endpoints of e. Since $\|u\|^2 = m$, the spectral norm bound in Theorem 3.1.5 implies that $u^T \overline{\Pi} u \leq O(m \log^2 n)$.

3.1.3 Applications to Oblivious Routing

Oblivious routing refers to the following problem: given a graph G, specify a set of flows $\{f_{uv}\}$ between pairs of vertices u, v so that for any set of demand pairs $(s_1, t_1), \ldots, (s_k, t_k)$, the congestion of the flow obtained by taking the union of $\{f_{s_i t_i}\}_{i \leq k}$ is at most a small factor

(called the competitive ratio) greater than the congestion of the optimal multicommodity flow for the given pairs. This is a well-studied problem with a vast literature which we will not attempt to recount; a landmark result is the optimal theorem of Räcke [82] which shows that there is an oblivious routing scheme with competitive ratio $O(\log n)$ for every graph.

In spite of this optimal result, there has been interest in studying whether simpler schemes achieve good competitive ratios. A particulary simple scheme, studied in [42, 60, 50], is to simply route f_{uv} using the electrical flow. The paper [42] shows that this scheme has a good competitive ratio on any graph when restricted to demands which all share a single source. It was shown in [60, 50] that the competitive ratio of electrical routing on an unweighted graph is exactly equal to $\|\Pi\|_{1\to 1}$, i.e., the maximum of $\Delta(u, v)$ over all edges in a graph. In these papers, it was shown that for grids, hypercubes, and expanders the competitive ratio is $O(\log n)$. Our theorem immediately extends this to all transitive graphs, albeit with a guarantee of $O(\text{polylog}(n))$ rather than $O(\log(n))$.

Corollary 3.1.6. *Electrical Flow Routing achieves a competitive ratio of $O(\log^2 n)$ on every edge-transitive graph.*

Proof. By Theorem 3.1.4 and symmetry, we have that every column sum of $\overline{\Pi}$ is at most $O(\log^2 n)$. By Proposition 1 and Lemma 4 of [60] (or by Theorem 3.1 of [50]), this implies that routing each pair by the electrical flow has a competitive ratio of $O(\log^2 n)$ as an oblivious routing scheme. □

3.1.4 Techniques

Given the expander example above, it may be tempting to attempt to prove Theorem 3.1.4 by decomposing an arbitrary graph into disjoint expanding clusters. However, using such a decomposition would likely require proving that edge electrical flows do not cross between the clusters, which is what we are trying to show in the first place.

We use an alternate scheme inspired by recent Schur-complement based Laplacian solvers. Recall the Schur complement formula for the pseudoinverse of a symmetric block matrix (see e.g. [33] Section 5):

Fact 3.1.7. *If*

$$L = \begin{bmatrix} P & Q \\ Q^T & R \end{bmatrix}$$

for symmetric P, R and R invertible, then:

$$L^+ = Z^T \begin{bmatrix} I & 0 \\ -R^{-1}Q^T & I \end{bmatrix} \begin{bmatrix} \texttt{Schur}(L, P)^+ & 0 \\ 0 & R^{-1} \end{bmatrix} \begin{bmatrix} I & -QR^{-1} \\ 0 & I \end{bmatrix} Z \tag{3.1}$$

where $\texttt{Schur}(L, P) = P - QR^{-1}Q^T$ denotes the Schur complement of L onto P, obtained by eliminating R by partial Gaussian elimination, and Z is the projection orthogonal to the nullspace of L.

The idea is to apply this formula to compute the terms $|b_e^T L^+ b_f|$ by eliminating vertices one-by-one, as in [58], and bounding the original value of $|b_e^T L^+ b_f|$ in terms of the value on small Schur complements. One cannot eliminate arbitrary vertices and get a good bound, though. We use Proposition 3.3.2 to show that there always exists a vertex whose elimination results in a good bound. Since Laplacian matrices with self loops are closed under taking Schur complements the remaining matrix is the Laplacian of a weighted graph as well. Mapping the demand vectors b_e and b_f to the vertex set of this graph and recurring yields the sum of interest.

3.2 Schur Complements, Probabilities, and Energies

In this section we collect some preliminary facts about Schur complements of Laplacians and establish some useful correspondences between electrical potentials and probabilities. We do this so that after recurring on a Schur complement of the graph G that we care about, we can interpret the recursively generated sums that we generate using Fact 3.1.7 in terms of G. We will make frequent use of the fact that for a Laplacian matrix L_G with block L_S, the Schur complement $\texttt{Schur}\,(L_G, L_S)$ is also a Laplacian. For a graph G and subset of vertices S will use the notation $\texttt{Schur}\,(G, S)$ to denote the graph corresponding to $\texttt{Schur}\,(L_G, L_S)$. Since all vectors we will apply pseudoinverses to will be orthogonal to the corresponding nullspaces (the corresponding constant vectors, since all Schur complements will be Laplacians), we will not write the projection Z in Fact 3.1.7 in what follows.

Definition 3.2.1. *Consider a graph G. For any set of vertices $S \subseteq V(G)$ with $|S| \geq 2$, a vertex $v \in S$, and a vertex $x \in V(G)$, let*

$$p_v^{G,S}(x) := \mathbb{P}_x[t_v < t_{S \setminus v}]$$

For an edge $e = \{x, y\} \in E(G)$, let

$$q_v^{G,S}(e) := |p_v^S(x) - p_v^S(y)|$$

where $t_{S'}$ denotes the hitting time to the set S'. Let

$$r_v^{G,S}(e) := \max(p_v^S(x), p_v^S(y), 1/|S|)$$

When G is clear from the context, we omit G from the superscript.

Corollary 3.2.2. *For any set S and any $e = \{x, y\} \in E(G)$, $\sum_{v \in S} r_v^{G,S}(x) \leq 3$.*

Proof. $\{p_v^{G,S}(z)\}_v$ is a distribution for any fixed vertex $z \in V(G)$. Bounding $r_v^{G,S}(e) \leq p_v^{G,S}(x) + p_v^{G,S}(y) + \frac{1}{|S|}$ yields the desired result. □

It is well-known that the above probabilities can be represented as normalized potentials (see, for instance, [17] Chapter IX).

Fact 3.2.3. *Let H be the graph obtained by identifying all vertices of $S\backslash\{v\}$ to one vertex s. Then $p_v^S(x) := \frac{|b_{vs}^T L_H^+ b_{xs}|}{b_{vs}^T L_H^+ b_{vs}}$ for any $x \in V(G)$ and $q_v^S(e) := \frac{|b_{vs}^T L_H^+ b_e|}{b_{vs}^T L_H^+ b_{vs}}$ for any $e \in E(G)$.*

In proving the desired result, it will help to recursively compute Schur complements with respect to certain sets of vertices S. We now relate the Schur complement to the above probabilities, which will be central to our proof; the following proposition is likely to be known but we include it for completeness.

Proposition 3.2.4. *For a set of vertices $S \subseteq V(G)$ and a vertex $x \in V(G)$ possibly not in S, let $b_x \in \mathbb{R}^{V(G)}$ denote the indicator vector of x. Let $b_x^S \in \mathbb{R}^S$ denote the vector with coordinates $b_x^S(v) = p_v^S(x)$ for all $v \in S$. Write L_G as a two-by-two block matrix:*

$$L = \begin{bmatrix} P & Q \\ Q^T & R \end{bmatrix}$$

where P, Q, and R have index pairs $S \times S$, $S \times (V(G) \setminus S)$, and $V(G) \setminus S \times V(G) \setminus S$ respectively. Then

$$b_x^S = M_S b_x$$

where

$$M_S = \begin{bmatrix} I & -QR^{-1} \end{bmatrix}.$$

Proof. If $x \in S$, then b_x^S is the indicator vector of x and x is in the identity block of M_S. Therefore, $b_x^S = M_S b_x$.

If $x \notin S$, then let b_x^c denote the coordinate restriction of b_x to $V(G) \setminus S$. We want to show that $b_x^S = -QR^{-1}b_x^c$. Consider the linear system

$$b_x^c = Rp$$

Let H be the graph obtained by identifying all vertices in S within G to a single vertex s. Then the vector p' with $p'_s = 0$ and $p'_v = p_v$ for all $v \in V(H) \setminus \{s\}$ is a solution to a boundary value problem with $p'_s = 0$ and p'_x having the maximum potential of any vertex. The block matrix Q can be viewed as mapping the potentials p' to a flow proportional to the xs-flows on edges incident with s. By Proposition 2.2 of [71], for example, the incoming flow on edges to s is equal to the probability that an $x \to s$ random walk first visits s by crossing that edge. Grouping edges according to their common endpoints shows that $-QR^{-1}b_x^c$ is a scalar multiple of b_x^S.

However, notice that

$$\mathbf{1}^T(-QR^{-1})b_x^c = (-\mathbf{1}^T Q)R^{-1}b_x^c = \mathbf{1}^T R R^{-1} b_x^c = 1 = \mathbf{1}^T b_x^S$$

so $b_x^S = -QR^{-1}b_x^c$, as desired. □

Once one views the the $q_v^S(e)$s in the above way, it makes sense to discuss the energy of the $q_v^S(e)$s in relation to the probabilities $p_v^S(x)$. It turns out that the total energy contributed

by edges with both endpoints having potential at most p is at most a p fraction of the total energy.

Proposition 3.2.5. *For any $p \in (0,1)$, let F be the set of edges $\{x,y\}$ with $\max_{z\in\{x,y\}} p_v^S(z) \leq p$. Then the total energy of those edges is at most a p fraction of the overall energy. More formally,*

$$\sum_{e\in F} c_e(q_v^S(e))^2 \leq p \sum_{e\in E(G)} c_e(q_v^S(e))^2$$

Proof. Let H be the graph obtained by identifying $S\backslash\{v\}$ to a vertex s in G. By Fact 3.2.3, we can show the desired proposition by proving the following:

$$\sum_{e\in F} c_e(b_{vs}^T L_H^+ b_e)^2 \leq p b_{vs}^T L_H^+ b_{vs}$$

for an arbitrary graph H and the subset of edges $F \subseteq E(H)$ with $\max_{z\in\{x,y\}} |b_{vs}^T L_H^+ b_{zs}| \leq p b_{vs}^T L_H^+ b_{vs}$.

Write the sum on the left side in terms of an integral over sweep cuts. For $p \in (0,1)$, let C_p denote the set of edges cut by the normalized potential p. More precisely, let C_p be the set of edges $\{x,y\}$ with $|b_{vs}^T L_H^+ b_{xs}| \geq p b_{vs}^T L_H^+ b_{vs}$ and $|b_{vs}^T L_H^+ b_{ys}| \leq p b_{vs}^T L_H^+ b_{vs}$. Notice that

$$\begin{aligned}
\sum_{e\in F} c_e(b_{vs}^T L_H^+ b_e)^2 &\leq \int_0^{p b_{vs}^T L_H^+ b_{vs}} \sum_{e\in C_q} c_e |b_{vs}^T L_H^+ b_e| dq \\
&= \int_0^{p b_{vs}^T L_H^+ b_{vs}} 1 dq \\
&\leq p b_{vs}^T L_H^+ b_{vs}
\end{aligned}$$

where the equality follows from the fact that C_q is a threshold cut for the $v-s$ electrical flow and the first inequality follows from splitting the contribution of e to the sum in terms of threshold cuts. This inequality is the desired result. □

Finally, we relate the weighted degrees of of vertices in $\texttt{Schur}(G,S)$ to energies in G with respect to S.

Definition 3.2.6. *Let c_v^H denote the sum of the conductances[1] of edges incident with v in H.*

[1] To avoid confusion, we remind the reader that by conductances we always mean electrical conductances, i.e., weights in the graph, and not conductances in the sense of expansion.

Proposition 3.2.7. *Let G be a graph. Consider a set S and let $H = \texttt{Schur}(G, S)$. Then*

$$c_v^H = \sum_{e \in E(G)} c_e^G (q_v^{G,S}(e))^2$$

Proof. Let I be the graph obtained by identifying $S \setminus \{v\}$ to s in H. Since the effective conductance of parallel edges is the sum of the conductance of those edges, $c_v^H = \frac{1}{b_{vs}^T L_I^+ b_{vs}}$.

By commutativity of Schur complements, I can also be obtained by identifying $S \setminus \{v\}$ in G before eliminating all vertices besides s and v. Let J be the graph obtained by just doing the first step (identifying $S \setminus \{v\}$). By definition of Schur complements,

$$b_{vs}^T L_I^+ b_{vs} = b_{vs}^T L_J^+ b_{vs}$$

By Fact 3.2.3,

$$b_{vs}^T L_J^+ b_{vs} = \frac{1}{\sum_{e \in E(G)} c_e^G (q_v^{G,S}(e))^2}$$

Substitution therefore shows the desired result. □

3.3 Proof of Theorem 3.1.5

We will deduce Theorem 3.1.5 from the following seemingly weaker statement regarding positive test vectors.

Theorem 3.3.1. *Let G be a graph. Then for any vector $w \in \mathbb{R}_{\geq 0}^{E(G)}$,*

$$\sum_{e,f \in E} w_e w_f \sqrt{c_e c_f} |b_e^T L_G^+ b_f| \leq O(\log^2 n) ||w||_2^2$$

Theorem 3.1.5 can be deduced from this by a Perron-Frobenius argument.

Proof of Theorem 3.1.5. Since the matrix $M = |C_G^{1/2} B_G L_G^+ B_G^T C_G^{1/2}|$ has nonnegative entries, there is an eigenvector with maximum eigenvalue with nonnegative coordinates by Perron-Frobenius. Such an eigenvector corresponds to a positive eigenvalue. Theorem 3.3.1 bounds the value of the quadratic form of this eigenvector. In particular, the quadratic form is at most $O(\log^2 n)$ times the ℓ_2 norm squared of the vector, as desired. □

The proof hinges on the following key quantity. Define

$$\texttt{Degree}_S(u) := \frac{(\sum_{e \in E(G)} w_e \sqrt{c_e} q_u^S(e))^2}{\sum_{e \in E(G)} c_e q_u^S(e)^2}$$

The quantity $\texttt{Degree}_S(u)$ may be interpreted as a measure of the sparsity of the vector $(q_u^S(e))_e$, since it is the ratio of the (weighted) ℓ_1^2 norm of this vector and its ℓ_2^2 norm. Note that when $S = V(G)$, $w = \mathbf{1}$, and G is unweighted, $\texttt{Degree}_S(u)$ is simply the degree of u.

There are two parts to the proof: (1) recursively reducing the original problem to a number of problems involving sums of simpler inner products and (2) bounding those sums. The difference between the value of a problem and the subproblem after eliminating u is at most $\texttt{Degree}_S(u)$. We want to show that there always is a choice of u with a small value of $\texttt{Degree}_S(u)$. The following proposition shows this:

Proposition 3.3.2. *For any $\{c_e\}_e$-weighted graph G, set $S \subseteq V(G)$ with $|S| \geq 2$, and nonnegative weights $\{w_e\}_{e \in E(G)}$, the following holds:*

$$\sum_{u \in S} \texttt{Degree}_S(u) \leq O(\log |S|) \sum_{e \in E(G)} w_e^2$$

We now reduce Theorem 3.3.1 to this proposition by picking the vertex u with that minimizes the summand $\texttt{Degree}_S(u)$ of Proposition 3.3.2 and recurring on the Schur complement with u eliminated. The summand of Proposition 3.3.2 is an upper bound on the decrease due to eliminating u.

Proof of Theorem 3.3.1 given Proposition 3.3.2. Define the following:

- $G_0 \leftarrow G$, $c^{(0)} \leftarrow c$, $x_0 \leftarrow \arg\min_{x \in V(G)} \texttt{Degree}_{S_0}(x)$, $S_0 \leftarrow V(G)$, $i \leftarrow 0$.
- While $|V(G_i)| > 2$:
  - $i \leftarrow i + 1$
  - $G_i \leftarrow \texttt{Schur}(G_{i-1}, V(G_{i-1}) \setminus \{x_{i-1}\})$
  - $c^{(i)} \leftarrow$ conductance vector for G_i
  - $S_i \leftarrow V(G_i)$
  - $x_i \leftarrow \arg\min_{x \in V(G_i)} \texttt{Degree}_{S_i}(x)$
- $T \leftarrow i$

Let $L_i \leftarrow L_{G_i}$ and let $m_i = L_{x_i x_i}$. We start by understanding how to express the left hand side of the desired inequality in G_i for all i. For a vertex $x \in V(G)$, let $b_x^{(i)} \in \mathbb{R}^{V(G_i)}$ denote the vector with $b_x^{(i)}(v) = p_v^{G,S_i}(x)$ for all $v \in V(G_i)$. For an edge $\{x, y\} \in V(G)$, let $b_{xy}^{(i)} = b_x^{(i)} - b_y^{(i)}$. Let

$$\mathcal{V}_i := \sum_{e,f \in E(G)} w_e \sqrt{c_e} |b_e^{(i)T} L_i^+ b_f^{(i)}| \sqrt{c_f} w_f$$

We now bound $\mathcal{V}_i$ in terms of $\mathcal{V}_{i+1}$ for all nonnegative integers $i < T$. By Fact 3.1.7 and Proposition 3.2.4,

$$\begin{aligned}
\mathcal{V}_i &= \sum_{e,f \in E(G)} w_e \sqrt{c_e} |b_e^{(i)T} L_i^+ b_f^{(i)}| \sqrt{c_f} w_f \\
&= \sum_{e,f \in E(G)} w_e \sqrt{c_e} |b_e^{(i+1)T} L_{i+1}^+ b_f^{(i+1)} + x_e^{(i)T} \frac{1}{m_i} x_f^{(i)}| \sqrt{c_f} w_f \\
&\leq \mathcal{V}_{i+1} + \sum_{e,f \in E(G)} w_e \sqrt{c_e} |x_e^{(i)T} \frac{1}{m_i} x_f^{(i)}| \sqrt{c_f} w_f
\end{aligned}$$

where $x_e^{(i)} := b_e^{(i)}(x_i)$. Since the $x_e^{(i)}$s are scalars, we can futher simplify the above sum:

$$\begin{aligned}
\sum_{e,f \in E(G)} w_e \sqrt{c_e} |x_e^{(i)T} \frac{1}{m_i} x_f^{(i)}| \sqrt{c_f} w_f &= \frac{1}{m_i} \left(\sum_{e \in E(G)} w_e \sqrt{c_e} |x_e^{(i)}| \right)^2 \\
&= \frac{1}{c_{x_i}^{(i)}} \left(\sum_{e \in E(G)} w_e \sqrt{c_e} x_e^{(i)} \right)^2 \\
&= \frac{\left(\sum_{e \in E(G)} w_e \sqrt{c_e} q_{x_i}^{S_i}(e) \right)^2}{\sum_{e \in E(G)} c_e q_{x_i}^{S_i}(e)^2} \\
&= \texttt{Degree}_{S_i}(x_i)
\end{aligned}$$

where the second-to-last denominator equality follows from Proposition 3.2.7. Since x_i minimizes $\texttt{Degree}_{S_i}(x_i)$, Proposition 3.3.2 with $S \leftarrow S_i$ and $G \leftarrow G$ implies that

$$\begin{aligned}
\texttt{Degree}_{S_i}(x_i) &\leq O\left(\frac{\log n}{|S_i|} ||w||_2^2 \right) \\
&\leq O\left(\frac{\log n}{n - i} ||w||_2^2 \right)
\end{aligned}$$

Plugging this in shows that

$$\mathcal{V}_i \leq \mathcal{V}_{i+1} + O\left(\frac{\log n}{n - i} \right) ||w||_2^2$$

for all $i < T$. Therefore, to bound $\mathcal{V}_0$, it suffices to bound $\mathcal{V}_T$. Let $S_T = \{a, b\}$. Then

$$\begin{aligned}
\mathcal{V}_T &= \sum_{e,f \in E(G)} w_e \sqrt{c_e} |b_e^{(T)T} L_T^+ b_f^{(T)}| \sqrt{c_f} w_f \\
&= \sum_{e,f \in E(G)} w_e \sqrt{c_e} q_{u_i}^{G,S_i}(e) |b_{ab}^T L_T^+ b_{ab}| q_{u_i}^{G,S_i}(f) \sqrt{c_f} w_f \\
&= \sum_{e,f \in E(G)} w_e \sqrt{c_e} \frac{|b_{ab}^T L_T^+ b_e|}{b_{ab}^T L_T^+ b_{ab}} b_{ab}^T L_{T'}^+ b_{ab} \frac{|b_{ab}^T L_T^+ b_f|}{b_{ab}^T L_T^+ b_{ab}} \sqrt{c_f} w_f \\
&\leq ||w||_2^2
\end{aligned}$$

where the last line follows from Cauchy-Schwarz. Therefore,

$$\mathcal{V}_T \leq ||w||_2^2$$

Combining these bounds yields a harmonic sum that proves the desired result. □

Now, we prove Proposition 3.3.2.

Proof. For each vertex $v \in S$ and each integer $i \in [0, \log |S|]$, let $X_v^{(i)} \subseteq E(G)$ denote the set of edges $e = \{x, y\}$ for which $r_v^S(e) \leq 2^{-i}$. Let $T := \log |S|$. For each $0 \leq i < T$, let $Y_v^{(i)} = X_v^{(i)} \setminus X_v^{(i+1)}$. Let $Y_v^{(T)} = X_v^{(T)}$.

For each v and each $i \geq 0$, $X_v^{(0)} = E(G)$, and $X_v^{(i+1)} \subseteq X_v^{(i)}$. Therefore, $\{Y_v^{(i)}\}_{i=0}^T$ is a partition of $E(G)$ for each $v \in S$. By Cauchy-Schwarz,

$$\begin{aligned}
\sum_{u \in S} \texttt{Degree}_S(u) &= \sum_{u \in S} \frac{(\sum_{e \in E(G)} w_e \sqrt{c_e} q_u^S(e))^2}{\sum_{e \in E(G)} c_e q_u^S(e)^2} \\
&\leq \sum_{u \in S} \left(\sum_{e \in E(G)} r_u^S(e) w_e^2 \right) \frac{\sum_{e \in E(G)} c_e q_u^S(e)^2 / r_u^S(e)}{\sum_{e \in E(G)} c_e q_u^S(e)^2}
\end{aligned}$$

By the definition of $X_u^{(i+1)}$,

$$\begin{aligned}
&\sum_{u \in S} \left(\sum_{e \in E(G)} r_u^S(e) w_e^2 \right) \frac{\sum_{e \in E(G)} c_e q_u^S(e)^2 / r_u^S(e)}{\sum_{e \in E(G)} c_e q_u^S(e)^2} \\
&\leq \sum_{u \in S} \left(\sum_{e \in E(G)} r_u^S(e) w_e^2 \right) \left(\sum_{i=0}^{T} 2^{i+1} \frac{\sum_{e \in Y_u^{(i)}} c_e q_u^S(e)^2}{\sum_{e \in E(G)} c_e q_u^S(e)^2} \right)
\end{aligned}$$

By the definition of $X_u^{(i)}$ and Proposition 3.2.5,

$$\sum_{u\in S}\left(\sum_{e\in E(G)} r_u^S(e)w_e^2\right)\left(\sum_{i=0}^{T} 2^{i+1}\frac{\sum_{e\in Y_u^{(i)}} c_e q_u^S(e)^2}{\sum_{e\in E(G)} c_e q_u^S(e)^2}\right) \leq \sum_{u\in S}\left(\sum_{e\in E(G)} r_u^S(e)w_e^2\right)\left(\sum_{i=0}^{T} 2\right)$$
$$\leq \sum_{u\in S}\left(\sum_{e\in E(G)} r_u^S(e)w_e^2\right)(2T+2)$$

By Proposition 3.2.2,

$$\sum_{u\in S}\left(\sum_{e\in E(G)} r_u^S(e)w_e^2\right)(2T+2) \leq (6T+6)\sum_{e\in E(G)} w_e^2$$

Combining these bounds shows that

$$\sum_{u\in S}\texttt{Degree}_S(u) \leq (6T+6)\sum_{e\in E(G)} w_e^2 \leq O(\log|S|)||w||_2^2$$

as desired. □

Chapter 4

Random Spanning Tree Sampling

4.1 Introduction

In this paper, we give the first almost-linear time algorithm for the following problem:

> Given an undirected graph G with weights (conductances) $\{c_e\}_{e \in E(G)}$ on its edges, generate a spanning tree T of G with probability proportional to $\prod_{e \in E(T)} c_e$.

Random spanning tree generation has been studied for a long time [51], has many connections to probability theory (for example [15]), and is a special case of both determinantal point processes [8] and uniform matroid basis sampling [9, 29]. They also have found applications in constructing cut sparsifiers [35, 39] and have played crucial roles in obtaining better approximation algorithms for both the symmetric [36] and asymmetric [12] traveling salesman problem.

The uniform random spanning tree distribution is also one of the simplest examples of a negatively-correlated probability distribution that is nontrivial to sample from. Much work has gone into efficiently sampling from the uniform spanning tree distribution in the past forty years [40]. This work falls into three categories:

- Approaches centered around fast exact computation of effective resistances [40, 27, 55, 28, 43]. The fastest algorithm among these takes $\tilde{O}(n^{\omega})$ time for undirected, weighted graphs [27].
- Approaches that approximate and sparsify the input graph using Schur complements [33, 32]. [33] samples a truly uniform tree in $\tilde{O}(n^{4/3}m^{1/2} + n^2)$ time, while [32] samples a random tree in $\tilde{O}(n^2\delta^{-2})$ time from a distribution with total variation distance δ from the real uniform distribution for undirected, weighted graphs.
- Random-walk based approaches [2, 18, 96, 46, 73]. [73] takes $\tilde{O}(m^{4/3})$ time for undirected, unweighted graphs.

Our main result is an algorithm for sampling a uniformly random spanning tree from a weighted graph with polynomial ratio of maximum to minimum weight in almost-linear time:

Theorem 4.1.1. *Given a graph G with edge weights $\{c_e\}_e$ and $\beta = (\max_{e\in E(G)} c_e)/(\min_{e\in E(G)} c_e)$, a uniformly random spanning tree of G can be sampled in $m^{1+o(1)}\beta^{o(1)}$ time.*

We also give a result whose runtime does not depend on the edge weights, but samples from a distribution that is approximately uniform rather than exactly uniform. However, the runtime dependence on the error is small enough to achieve $1/\text{poly}(n)$ error in almost-linear time, so it suffices for all known applications:

Theorem 4.1.2. *Given a weighted graph G and $\epsilon \in (0, 1)$, a random spanning tree T of G can be sampled from a distribution with total variation distance at most ϵ from the uniform distribution in time $m^{1+o(1)}\epsilon^{-o(1)}$ time.*

Our techniques are based on random walks and are inspired by [2, 18, 46, 73]. Despite this, our runtime guarantees combine the best aspects of all of the former approaches. In particular, our $m^{1+o(1)}\epsilon^{-o(1)}$-time algorithm has no dependence on the edge weights, like the algorithms from the first two categories, but has subquadratic running time on sparse graphs, like the algorithms in the third category.

4.1.1 Other Contributions

We use random walks to generate random spanning trees. The behavior of random walks can be understood through the lens of electrical networks. We prove several new results about electrical flows (for example Lemmas 4.5.4, 4.7.2, and 4.10.14) and find new uses for many prior results (for example [44, 42]). We highlight one of our new results here.

One particularly important quantity for understanding random walks is the effective resistance between two vertices:

Definition 4.1.3 (Effective resistance)**.** *The* energy *of a flow $f \in \mathbb{R}^{E(G)}$ in an undirected graph G with weights $\{c_e\}_{e\in E(G)}$ is*

$$\sum_{e\in E(G)} \frac{f_e^2}{c_e}$$

*For two vertices s, t in a graph G, the G-*effective resistance *between s and t, denoted $\texttt{Reff}_G(s, t)$, is the minimum energy of any $s-t$ flow that sends one unit of flow from s to t.*

We study the robustness of the $s-t$ effective resistance to random changes in the graph G. Specifically, we consider random graphs $H \sim G[F]$ obtained by conditioning on the intersection of a random spanning tree in G with the set $F \subseteq E(G)$, which amounts to sampling a tree T from G, contracting all edges in $E(T) \cap F$, and deleting all edges in $F \setminus E(T)$.

Surprisingly, the $s-t$ effective resistance in H is the same as the $s-t$ effective resistance in G in expectation as long as $G \setminus F$ is connected. However, the $s-t$ effective resistance in H does not concentrate around the effective resistance in G. In particular, s and t could be identified to one another in G, in which case the $s-t$ effective resistance is 0. We show that changing a small number of contractions to deletions makes the effective resistance not much smaller than its original value:

Lemma 4.1.4. *Let G be a graph, $F \subseteq E(G)$, $\epsilon \in (0,1)$, and $s,t \in V(G)$. Sample a uniformly random spanning tree T of G. Then, with high probability, there is a set $F' \subseteq E(T) \cap F$ that depends on T and satisfies both of the following guarantees:*

- *(Effective resistance) Let H' be the graph obtained from G by contracting all edges in $(E(T) \cap F) \setminus F'$ and deleting all edges in $F' \cup (F \setminus E(T))$. Then $\texttt{Reff}_{H'}(s,t) \geq (1-\epsilon)\texttt{Reff}_G(s,t)$*
- *(Size) $|F'| \leq O((\log n)/\epsilon^2)$*

Even better, we show that F' can be computed in almost-linear time. Our algorithm uses a combination of matrix sketching [6, 44] and localization (Chapter 3) that may be of independent interest.

4.2 Algorithm Overview

Our algorithm, like those of [46] and [73], is based on the following beautiful result of Aldous [2] and Broder [18]:

Theorem 4.2.1 (Aldous-Broder)**.** *Pick an arbitrary vertex u_0 and run a random walk starting at u_0 in a weighted graph G. Let T be the set of edges used to visit each vertex besides u_0 for the first time. Then T is a weighted uniformly random spanning tree of G.*

The runtime of Aldous-Broder is the amount of time it takes to visit each vertex for the first time, otherwise known as the *cover time* of G. On one hand, the cover time can be as high as $\Theta(mn)$. On the other hand, Aldous-Broder has the convenient property that only a small number of vertex visits need to be stored. In particular, only $n-1$ visits to vertices add an edge to the sampled tree (the first visits to each vertex besides the starting vertex). This observation motivates the idea of *shortcutting* the random walk.

4.2.1 The shortcutting meta-algorithm

To motivate our algorithm, we classify all existing algorithms based on Aldous-Broder [18, 2, 46, 73] at a very high level. We start by describing Aldous-Broder in a way that is more readily generalizable:

Aldous-Broder

- For each $v \in V(G)$, let $S_v^{(0)} = \{v\}$.

- Pick an arbitrary vertex u_0 and set $u \leftarrow u_0$.
- Until all vertices in G have been visited
  - Sample the first edge that the random walk starting at u uses to exit $S_u^{(0)}$
  - Replace u with the non-$S_u^{(0)}$ endpoint of this edge.
- Return all edges used to visit each vertex besides u_0 for the first time.

We now generalize the above algorithm to allow it to use some number σ_0 of *shortcutters* $\{v\} \subseteq S_v^{(i)} \subseteq V(G)$ for each vertex $v \in V(G)$. If the $\{S_v^{(i)}\}_{i=1}^{\sigma_0}$s are chosen carefully, then instead of running the random walk until it exits $S_v^{(i)}$, one can sample the exiting edge much faster using Laplacian solvers.

Ideally, we could shortcut the random walk directly to the next unvisited vertex in order to minimize the number of wasted visits. Unfortunately, we do not know how to do such shortcutting efficiently. Instead, we use multiple shortcutters per vertex. More shortcutters per vertex means a better approximation to the set of previously visited vertices, which leads to fewer unnecessary random walk steps and a better runtime.

Simple shortcutting meta-algorithm

- For each $v \in V(G)$, let $S_v^{(0)} = \{v\}$ and **pick** shortcutters $\{S_v^{(i)}\}_{i=1}^{\sigma_0}$
- Pick an arbitrary vertex u_0 and set $u \leftarrow u_0$.
- Until all vertices in G have been visited
  - Let $i^* \in \{0, 1, \ldots, \sigma_0\}$ be the maximum value of i for which all vertices in $S_u^{(i)}$ have been visited
  - **Sample** the first edge that the random walk starting at u uses to exit $S_u^{(i^*)}$
  - Replace u with the non-$S_u^{(i^*)}$ endpoint of this edge.
- Return all edges used to visit each vertex besides u_0 for the first time.

To implement the above meta-algorithm, one must make two important choices, each of which is bolded above. Both of these choices only affect the runtime of the meta-algorithm; not its correctness:

- A set of shortcutters for each vertex $v \in V(G)$
- A method for sampling the first exit edge from $S_u^{(i)}$, which we call a *shortcutting method*

The meta-algorithm could also choose an arbitrary starting location u_0, but this choice is not important to any shortcutting-based algorithm.

We now argue that the meta-algorithm correctly samples a uniformly random spanning tree, no matter the choice of the $S_v^{(i)}$s or shortcutting method. First of all, $i = 0$ is always a valid choice for i^*, so i^* exists and the algorithm is well-defined. Since all vertices in $S_v^{(i^*)}$ have been previously visited, using the shortcutter $S_v^{(i^*)}$ does not skip any first visits.

Therefore, by Theorem 4.2.1, the edges returned form a uniformly random spanning tree.

Next, we summarize all algorithms based on this meta-algorithm [2, 18, 46, 73] with σ_0 — the number of shortcutters per vertex — and the choice of shortcutting method. We also list bounds on the runtimes of these algorithms on unweighted graphs to offer context.

While we have not yet discussed what "Offline" and "Online" shortcutting are, we highlight that our shortcutting method is different from that of [46] and [73]. This is one of the key reasons why we are able to obtain a faster algorithm and are able to effectively use more shortcutters per vertex.

4.2.2 Shortcutting methods

The starting point for our improvement is an *online* shortcutting method. This method is based on the following observation:

Key Idea 4.2.2 (Online shortcutting)**.** *For a vertex u and a shortcutter S_u associated with u, the probability that a random walk exits S_u through an edge e can be ϵ-additively approximated for all $e \in \partial S_u$ simultaneously using* one *ϵ-approximate Laplacian system solve on a graph with $|E(S_u) \cup \partial S_u|$ edges.*

When combined with a trick due to Propp [81], one can exactly sample an escape edge in expected $\tilde{O}(|E(S_u) \cup \partial S_u|)$ time with no preprocessing.

We call this method *online* due to its lack of preprocessing and the shortcutting technique used in [46] and [73] *offline* due to its fast query time with high preprocessing time. We summarize the runtime properties of these shortcutting methods for a shortcutter S_u here:

The upside to the online method is that its runtime does not depend quadratically on the size of S_u's boundary. This is a critical barrier to improving the technique of [46] and [73] because it is impossible to obtain balanced cuts with arbitrarily small size that separate a graph into low-radius parts in most metrics. While the high query time for the online method may seem prohibitive initially, it is fine as long as online shortcutting does substantially less work than the random walk would have done to reach the boundary of S_u. In the following path example, it takes the random walk $\Theta(k^2)$ time for the random walk to escape S_u starting at u, but online shortcutting only takes $\tilde{O}(k)$ time to find the escape edge:

4.2.3 Properties of shortcutters

Now, we describe the machinery that allows us to bound the total amount of work. We start with a bound that captures the idea that most random walk steps happen far away from an unvisited vertex. This bound is a weighted generalization of Lemma A.4 given in [73]. We prove it in Appendix B.3.1:

Lemma 4.2.3 (Key result for bounding the number of shortcutter uses)**.** *Consider an arbitrary vertex u_0 in a graph I, an edge $\{u, v\} = f \in E(I)$, and an $R \geq 0$. Let $\mathcal{B}(u, R) \subseteq V(I)$ denote the set of vertices in I with I-effective resistance distance at most R from u. The*

expected number of times that the random walk starting at u_0 traverses f from $u \to v$ before all vertices in $\mathcal{B}(u, R)$ have been visited is at most $\tilde{O}(c_f R)$, where c_f is the conductance of the edge f.

The effective resistance metric[1] appears in the above lemma due to the relationship between random walks and electrical networks. Unlike recent work on sampling random spanning trees ([46, 73]), we apply this lemma in the original graph *and* in other graphs obtained by "eliminating" vertices from the original graph. Specifically, we apply Lemma 4.2.3 to *Schur complements* of the input graph G. In all sections before Section 4.9 — including this one — one does not need to understand how the Schur complement is constructed or its linear algebraic properties. We use the following combinatorial, folklore fact about Schur complements to employ Schur complements as an analysis tool:

Theorem 4.2.4. *Consider a graph I and some set of vertices $S \subseteq V(I)$. Let $J = \texttt{Schur}(I, S)$. Pick a vertex $v \in S$ and generate two sequences of vertices as follows:*

- *Do a random walk in J starting at v and write down the sequence of visited vertices.*
- *Do a random walk in I starting at v and write down the sequence of visited vertices that are also in S.*

These two distributions over sequences are identical.

For a shortcutter S_u, consider the graph $H = \texttt{Schur}(G, (V(G) \setminus S_u) \cup \{u\})$. Each use of the shortcutter S_u can be charged to crossing of at least one edge incident with u in the random walk on H by Theorem 4.2.4. Therefore, to bound the number of times a shortcutter S_u is used over the course of the algorithm, it suffices to bound the total conductance of edges between u and $V(H) \setminus \{u\}$ in H. This motivates one of the key properties of shortcutters, which we call *conductivity* in later sections:

Key Idea 4.2.5 (Shortcutter Schur complement conductance bound)**.** *Let S_C denote a shortcutter for which $S_u^{(i)} = S_C$ for all $u \in C$. Then "on average," the total conductance of the edges between C and $V(G) \setminus S_C$ in the graph $\texttt{Schur}(G, C \cup (V(G) \setminus S_C))$, which we call the* Schur complement conductance *of S_C, is at most $\frac{m^{o(1)}}{\alpha^{i/(\sigma_0+1)} r_{min}}$. α is the ratio between the maximum and minimum effective resistance distance between any two vertices in G and r_{min} is the minimum effective resistance distance between any two vertices in G.*

By "on average," we mean that the shortcutters S_C are organized into $m^{o(1)}$ sets and within each set $\mathcal{C}$, the total Schur complement conductance of the shortcutters $S_C \in \mathcal{C}$ is at most $\frac{m^{o(1)}|\mathcal{C}|}{\alpha^{i/(\sigma_0+1)} r_{min}}$. For the rest of this section, we think of *each* shortcutter as having

[1] The effective resistance $\texttt{Reff}_G$ satisfies the triangle inequality and therefore forms a metric space on the vertices of G

Schur complement conductance at most $\frac{m^{o(1)}}{\alpha^{i/(\sigma_0+1)}r_{min}}$ in order to simplify the description. For a more formal description of the organization of our shortcutters, see Section 4.4.

To bound the number of times S_C is used, Lemma 4.2.3 requires two things:

Sufficient properties for bounding shortcutter uses
(1) A bound on the Schur complement conductance of S_C
(2) A bound on the effective resistance distance to the nearest unvisited vertex outside of S_C

The Schur complement conductance is at most $\frac{m^{o(1)}}{\alpha^{i/(\sigma_0+1)}r_{min}}$ by conductivity. Therefore, we just need to bound the distance to the nearest unvisited vertex. If there was an unvisited vertex within effective resistance distance $\alpha^{(i+1)/(\sigma_0+1)}r_{min}m^{o(1)}$ of C, Lemma 4.2.3 would imply that S_C is only used

$$\left(\alpha^{(i+1)/(\sigma_0+1)}r_{min}m^{o(1)}\right)\left(\frac{m^{o(1)}}{\alpha^{i/(\sigma_0+1)}r_{min}}\right) = m^{o(1)}\alpha^{1/(\sigma_0+1)}$$

times over the course of the shortcutted random walk. To bound the total work done, the following fact suffices:

Key Idea 4.2.6 (Shortcutter overlap)**.** *Each vertex in G is in at most $m^{o(1)}$ shortcutters.*

The above idea implies that the total size of all shortcutters is $O(m^{1+o(1)})$. To use a shortcutter, we apply the online shortcutting method, which takes time proportional to the shortcutter's size (see Table 4.2.2). If each shortcutter is used at most $m^{o(1)}\alpha^{1/(\sigma_0+1)}$ times as described above, the total work due to all shortcutter uses is $(m^{1+o(1)})(m^{o(1)}\alpha^{1/(\sigma_0+1)}) \leq m^{1+o(1)}\alpha^{o(1)}$, as desired.

Therefore, if we can obtain shortcutters with bounds on (1) and (2) that also respect Key Idea 4.2.6, we would have an almost-linear time algorithm for sampling random spanning trees on weighted graphs.

4.2.4 Obtaining shortcutters with Property (1) and small overlap

We have not yet discussed how to actually obtain shortcutters with the desired conductance property. We discuss this in detail in Section 4.5, but we give a summary here for interested readers. We construct $\{S_v^{(i)}\}_{v\in V(G)}$ for each i independently by

- constructing a small number of families of sets that are each well-separated in the effective resistance metric, have distance separation roughly $\alpha^{i/(\sigma_0+1)}r_{min}$, and together cover the graph. These are the *cores* and the construction of these cores is similar to constructions of sparse covers of metric spaces (for example [13]).

- making the shortcutter around each core C be the set of vertices $S_C \subseteq V(G)$ for which a random walk starting at $x \in S_C$ is more likely to hit C before any other core in its family. The sparsity of the cover ensures that the shortcutters satisfy Key Idea 4.2.6, while their well-separatedness ensures Property (1).

4.2.5 Obtaining Property (2) using partial sampling and carving

We now show that when $S_u^{(i)}$ is used, there is an unvisited vertex with effective resistance distance at most $m^{o(1)}\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ from u. Review the shortcutting meta-algorithm. When $S_u^{(i^*)}$ is used, there is some vertex $v \in S_u^{(i^*+1)}$ that the random walk has not yet visited. v, however, may not be close to u. This motivates the following property of a shortcutter $S_u^{(i)}$, which we call *being carved with respect to (a set of vertices) S*:

Key Idea 4.2.7 (Carving)**.** *A shortcutter $S_u^{(i)}$ is* carved with respect to $S \subseteq V(G)$ *if $S_u^{(i)} \cap S$ only consists of vertices that are within effective resistance distance $m^{o(1)}\alpha^{i/(\sigma_0+1)}r_{min}$ of u.*

If $S_u^{(i^*+1)}$ is carved with respect to $V(G)$, then the unvisited vertex v is within distance $m^{o(1)}\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ of u. As a result, there is an unvisited vertex within distance $m^{o(1)}\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ of u, as desired.

It is difficult to directly build shortcutters to make them carved with respect to some set S. Instead, we explore how we could remove vertices from shortcutters so that *all* shortcutters for *all* vertices are carved with respect to $V(G)$. To carve $V(G)$ out of all shortcutters $S_u^{(i)}$, one could just remove all vertices in $S_u^{(i)}$ that are farther than resistance distance $m^{o(1)}\alpha^{i/(\sigma_0+1)}r_{min}$ away from u. Unfortunately, this could remove almost all of the shortcutter in general.

Instead, we compute a *partial sample* of a random spanning tree in order to make it so that each shortcutter does not have to be carved with respect to as many vertices. Specifically, we modify the simple shortcutting meta-algorithm as follows:

Full shortcutting meta-algorithm(one round of partial sampling)

- **Choose** a set $S \subseteq V(G)$ for partial sampling
- For each $v \in V(G)$, let $S_v^{(0)} = \{v\}$ and **pick** shortcutters $\{S_v^{(i)}\}_{i=1}^{\sigma_0}$
- Pick an arbitrary vertex u_0 and set $u \leftarrow u_0$.
- Until all vertices in S have been visited
  - Let $i^* \in \{0, 1, \ldots, \sigma_0\}$ be the maximum value of i for which all vertices in $S \cap S_u^{(i)}$ have been visited
  - **Sample** the first edge that the random walk starting at u uses to exit $S_u^{(i^*)}$
  - Replace u with the non-$S_u^{(i^*)}$ endpoint of this edge.
- Let T' be all edges used to visit each vertex besides u_0 in S for the first time that are in the induced subgraph $F := E(G[S])$

- Condition on the partial sample, which amounts to contracting all edges in $E(T') \cap F$ and deleting all edges of $F \setminus E(T')$ in G

[73] also exploited partial sampling in this way. This algorithm correctly samples the intersection of a random spanning tree of G with $E(G[S])$ because it does not skip any of the first visits to vertices in S and only vertices in S need to be visited in order to determine the edges in $G[S]$ that are in the sample. While this algorithm no longer samples the entirety of a tree, we only need shortcutters to be carved with respect to S rather than all of $V(G)$ in order to show that the total work is $m^{1+o(1)}\alpha^{o(1)}$.

Our algorithm and [73] exploit the full meta-algorithm in multiple rounds. During each round, we pick a set S to condition on, run the meta-algorithm, and repeat until G is a single vertex. At this point, we have sampled a complete spanning tree of G.

We want to choose S to be small enough so that every shortcutter can be carved with respect to S without increasing the Schur complement conductance of those shortcutters too much. As long as all shortcutters are carved with respect to S, the meta-algorithm takes $m^{1+o(1)}\alpha^{o(1)}$ time. However, we also want S to be large enough to make substantial progress.

When $\sigma_0 = 1$, let S be the set of vertices u assigned to the largest shortcutters. By Key Idea 4.2.6, there cannot be too many large shortcutters, which means that S is the union of a small number of clusters with small effective resistance diameter $(\sqrt{\alpha}r_{min})$. Deleting each cluster from each shortcutter S_C with a far-away core C makes S_C carved with respect to S. Furthermore, because the cluster was well-separated from C, its deletion did not increase the Schur complement conductance of S_C much. For more details on this analysis, see Section 4.7.1.

4.2.6 Bounding the number of rounds of partial sampling when $\sigma_0 = 1$

In the previous section for the $\sigma_0 = 1$ case, we saw that conditioning on the induced subgraph of the vertices assigned to the largest shortcutters was a good idea for carving. We now show that computing a partial sample for the induced subgraph of these vertices allows us to make substantial progress. Ideally, one could show that conditioning on the induced subgraph of vertices assigned to the largest shortcutters decreases the size of the graph by a constant fraction. Unfortunately, we do not know how to establish this in general.

To get around this, we do not rebuild shortcutters from scratch after each partial sampling round. Instead, we show that it is possible to make shortcutters $S_u^{(i)'}$ that are contained within the shortcutter $S_u^{(i)}$ from the previous round. It is quite tricky to do this directly, as conditioning on a partial sample can change the metric structure of the graph G. In particular, the conductance of a shortcutter could dramatically increase after conditioning.

To cope with this, we show a concentration inequality that promises the existence of a small set of edges with high probability that, when deleted, restore the conductance of all shortcutters back to their value before conditioning. This result follows from a nontrivial

generalization (Lemma 4.8.2) of Lemma 4.1.4 that we reduce to in Section 4.8, prove in Section 4.10, and find a fast algorithm for in Section 4.11.

Given that $S_u^{(i)'}$ is contained in $S_u^{(i)}$ for all $u \in V(G)$ and all $i \in [\sigma_0]$, conditioning on the vertices with the near-largest shortcutters decreases the maximum size of a remaining shortcutter by a large factor. Therefore, after $O(\log n)$ rounds of conditioning on the induced subgraph of the vertices assigned to the largest shortcutters, no shortcutters are left. At this point, the algorithm is done. Each round takes $m^{1+o(1)}\sqrt{\alpha}$ time in the $\sigma_0 = 1$ case for a total of $\tilde{O}(m^{1+o(1)}\sqrt{\alpha})$ runtime.

4.2.7 Carving and progress when $\sigma_0 > 1$

The bottleneck in the algorithm for the $\sigma_0 = 1$ case is the sampling step. As discussed in Section 4.2.1, using more shortcutters allows us to approximate the set of previously visited vertices better, leading to a better runtime. In particular, the runtime-bounding argument presented earlier, given a carving and conductance bound, shows that using σ_0 shortcutters yields an $m^{1+o(1)}\alpha^{1/(\sigma_0+1)}$-runtime algorithm for sampling a spanning tree.

Unfortunately, carving shortcutters is more complicated when $\sigma_0 > 1$. We need to pick a relatively large set of vertices that can be carved out of *all* shortcutters for *all* vertices simultaneously. To do this, one could start by trying to generalize the strategy in the $\sigma_0 = 1$ case through repetition. Specifically, one could try the following strategy for picking a set S for use in one round of the meta-algorithm with partial sampling:

First attempt at conditioning when $\sigma_0 > 1$

- $S_{\sigma_0+1} \leftarrow V(G)$
- For $i = \sigma_0, \sigma_0 - 1, \ldots, 1$
  - $S_i \leftarrow$ the vertices $u \in S_{i+1}$ with near-maximum size $S_u^{(i)}$ shortcutters; that is within a factor of m^{-1/σ_1} of the maximum.
- Let $S \leftarrow S_1$

This strategy has some benefits. If S_1 could be carved out of all shortcutters, the maximum size of $S_v^{(1)}$ shortcutters for vertices $v \in S_2$ would decrease by a factor of m^{-1/σ_1}.

Before moving on, we elaborate on how conditioning on the induced subgraph of vertices assigned to a shortcutter renders it unnecessary in the future. Start by refining all cores of all $S_v^{(i)}$ shortcutters to obtain σ_0 partitions $\{\mathcal{P}_i\}_{i=1}^{\sigma_0}$ of $V(G)$, with one for each $i \in [\sigma_0]$. Standard ball-growing (for example [66]) ensures that the total conductance of all boundary edges of parts in $\mathcal{P}_i$ is at most $m^{1+o(1)}/(\alpha^{i/(\sigma_0+1)}r_{min})$. Conditioning on the induced subgraph of a part P deletes or contracts all edges in the induced subgraph of P, only leaving P's boundary. Since P's boundary is small, the random walk never needs to use P's shortcutter again because the total number of steps across $\mathcal{P}_i$ boundary edges is at most

$$\frac{m^{1+o(1)}}{\alpha^{i/(\sigma_0+1)}r_{min}}\alpha^{(i+1)/(\sigma_0+1)}r_{min} \leq m^{1+o(1)}\alpha^{o(1)}$$

where the $\alpha^{(i+1)/(\sigma_0+1)} r_{min}$ bound follows from carving. Therefore, conditioning on a part replaces it with its boundary, thus rendering its shortcutter unnecessary.

Now, we go back to analyzing our first attempt at a $\sigma_0 > 1$ algorithm for selecting S. If we could always carve all shortcutters with respect to S_1, conditioning σ_1 times on various S_1s would make all shortcutters for $\mathcal{P}_1$ parts intersecting S_2 irrelevant, thus making it possible to condition on S_2 directly. More generally, if carving were not an issue, every σ_1 rounds of conditioning on S_i would pave the way for one round of conditioning on S_{i+1}. Combining this reasoning for all i implies that we have sampled the entire tree after $\sigma_1^{\sigma_0}$ applications of the meta-algorithm.

Unfortunately, our first attempt does not produce carvable sets S in general because there could be a very large shortcutter with core just outside of some S_i that happens to intersect many of the vertices in S_i. To cope with this, we incorporate a ball-growing type approach that switches to conditioning on this very large but nearby shortcutter if one exists. Once this procedure stops, one can carve the parts assigned to the selected shortcutters out of all other shortcutters because the selected shortcutters are larger than all other shortcutters that the selected parts intersect. For more details, see Section 4.7.

4.2.8 Coping with the fixing lemma in the shortcutting method

In Section 4.2.6, we established that we could obtain containment of shortcutters in past shortcutters if we deleted a small set of "fixing edges" from the graph. However, we cannot actually delete these edges from the graph, as we must do partial sampling in graphs resulting directly from conditioning in order to correctly sample a uniformly random spanning tree.

Instead of deleting these fixing edges from the graph, we just remove their endpoints from the shortcutters and use *offline* shortcutting [73, 46] to make it so that shortcutting to the endpoints of these fixing edges only takes constant time rather than time proportional to the size of the shortcutter. Since there are a small number of fixing edges, the preprocessing time for offline shortcutting is small. Each shortcut to the endpoints of a removed edge takes $\tilde{O}(1)$ time and can be charged to crossing a boundary edge of some core. Constructing the cores using standard ball-growing makes the total conductance of these boundary edges small, so Lemma 4.2.3 can be used to show that the number of such shortcutting steps is small.

While this completes the high-level description of our algorithm, it does not describe all of the contributions of this paper. Along the way, we prove many new results about effective resistance metrics (like Lemma 4.5.4) that may be of independent interest.

4.3 Preliminaries

Graphs

For a (directed or undirected) graph G, let $V(G)$ and $E(G)$ denote its vertex and edge set respectively. n and m refer to the number of vertices and edges respectively of the input graph to our random spanning tree generation algorithm. For a set of edges $F \subseteq E(G)$, $G \setminus F$ denotes the graph obtained by deleting the edges in F from G. For a set of vertices $S \subseteq V(G)$, let G/S be the graph obtained by *identifying* all vertices in S to a single vertex; that is

$$V(G/S) := (V(G) \setminus S) \cup \{s\}$$

and each endpoint of an edge $e \in E(G)$ that is also in S is replaced with s. For a set of edges $F \subseteq E(G)$, let $V(F)$ denote the set of endpoints of edges in F. For an edge $f \in E(G)$, let $G \setminus f$ and G/f denote the graph obtained by deleting and contracting f respectively.

For two sets of vertices $S, S' \in V(G)$, let $E_G(S, S')$ denote the set of edges with one endpoint in S and the other endpoint in S'. Let $G[S] := E_G(S) := E_G(S, S)$. When the graph G is clear from context, we omit it from the subscript. For a graph G with two sets $X, Y \subseteq V(G)$, let $G/(X, Y)$ denote the graph obtained by identifying all vertices in X to one vertex x and all vertices in Y to one vertex y.

For a set of vertices $S \in V(G)$, let $\partial_G S := E_G(S, V(G) \setminus S)$ denote the boundary edges of S. For a singleton set $S = \{w\}$, $\partial_G S$ is abbreviated $\partial_G w$.

In this paper, graphs are sometimes weighted with *conductances* $\{c_e\}_{e \in E(G)}$. For a set $F \subseteq E(G)$, let $c^G(F) := \sum_{e \in F} c_e^G$. Let $r_e^G = 1/c_e^G$. Let $\beta^G := (\max_{e \in E(G)} r_e^G)/(\min_{e \in E(G)} r_e^G)$. When the context is clear, the graph G is omitted from all superscripts in the aforementioned definitions.

Laplacian matrices, electrical flows, and effective resistances

For an undirected graph G with two vertices $s, t \in V(G)$, let $b_{st} \in \mathbb{R}^{V(G)}$ denote the vector with $b_s = 1$, $b_t = -1$, and $b_v = 0$ for $v \neq s, t$. Direct all of the edges of G arbitrarily. Suppose that $e = \{a, b\}$ and is directed from a to b. Define $b_e := b_{ab}$. Define the *Laplacian matrix* of a weighted graph G as

$$\sum_{e \in E(G)} c_e b_e b_e^T$$

This definition is invariant of the orientations of the edges. L_G has nontrivial kernel, but still has a Moore-Penrose pseudoinverse L_G^+. The vector $L_G^+ b_{st}$ is a vector of *potentials* for the *electrical flow* $C_G B_G L_G^+ b_{st}$, where B_G is the $|E(G)| \times |V(G)|$ matrix with rows equal to the vectors b_e for $e \in G$ and C_G is the $|E(G)| \times |E(G)|$ diagonal matrix of edge conductances.

The *effective resistance* between two vertices $s, t \in V(G)$ is the energy of the electrical flow from s to t, which equivalently is

$$\texttt{Reff}_G(s,t) := b_{st}^T L_G^+ b_{st}$$

For an edge $e = \{a, b\} \in E(G)$, let $\texttt{Reff}_G(e) := \texttt{Reff}_G(a, b)$. We use the following folklore fact about effective resistances extensively without reference:

Remark 3. *The vertex set of $V(G)$ is a metric space with respect to the metric $\texttt{Reff}_G$. In particular, $\texttt{Reff}_G$ satisfies the triangle inequality; i.e.*

$$\texttt{Reff}_G(s,t) \leq \texttt{Reff}_G(s,w) + \texttt{Reff}_G(w,t)$$

for any three vertices $s, t, w \in V(G)$.

For a set $S \in V(G)$, define its *effective resistance diameter* to be

$$\max_{u,v \in S} \texttt{Reff}_G(u,v)$$

Often, for clarity, we call this the G-effective resistance diameter of S. Let $r_{min} := \min_{u,v \in V(G)} \texttt{Reff}_G(u,v)$, $r_{max} := \max_{u,v \in V(G)} \texttt{Reff}_G(u,v)$, and $\alpha = \frac{r_{max}}{r_{min}}$. Notice that $\beta \leq \alpha \leq m^2\beta$. Therefore, to obtain an $m^{1+o(1)}\beta^{o(1)}$-time algorithm, it suffices to obtain an $m^{1+o(1)}\alpha^{o(1)}$ time algorithm.

Laplacian solvers

In this paper, we make extensive use of efficient approximate Laplacian solvers [91, 54, 48, 26, 80, 62, 58]:

Theorem 4.3.1 ([26])**.** *There is an $O(m\sqrt{\log n}\log(n/\epsilon))$ time algorithm, that, given a demand vector $d \in \mathbb{R}^{V(G)}$ for some graph G, computes a vector $p \in \mathbb{R}^{V(G)}$ such that*

$$||p - L_G^+ d||_\infty \leq \epsilon$$

with high probability.

Random walks

For a weighted graph G and some vertex $a \in V(G)$, let $\Pr_{a,G}[E]$ denote the probability of an event E over random walks starting at a in the graph G. When the graph is clear from context, we denote this by $\Pr_a[E]$. For a set of vertices $S \in V(G)$, let t_S be the random variable denoting the hitting time to the set S. When S is a singleton $\{b\}$, we abbreviate t_S as t_b.

We use the following fact about random walks extensively:

Theorem 4.3.2 (Proposition 2.2 of [71]). *Let G be a graph with conductances $\{c_e\}_e$. Consider two vertices $s, t \in V(G)$. For a vertex u, let $p_u = b_{st}^T L_G^+ b_{ut}$. Consider an edge $e = \{u, v\}$. Then*

$$p_u c_e = \boldsymbol{E}_s[\text{ number of times } e \text{ is crossed from } u \to v \text{ before } t_t]$$

Low-dimensional embedding of the effective resistance metric

Throughout this paper, we use the fact that the effective resistance metric can be embedded into low-dimensional Euclidean space:

Theorem 4.3.3 ([90]). *With high probability, one can compute an embedding $D : V(G) \to \mathbb{R}^d$ of the vertices of a graph G with $d \leq (\log n)/\epsilon^2$ for which*

$$||D(u) - D(v)||_2^2 \in [(1-\epsilon)\texttt{Reff}(u,v), (1+\epsilon)\texttt{Reff}(u,v)]$$

in near-linear time. Furthermore, for each vertex $u \in V(G)$, $D(u)$ takes $O((\log n)/\epsilon^2)$ time to compute.

Throughout this paper, we use $\epsilon = 1/2$. In most of this paper, we use this result to approximately compute effective resistances. In the appendix and in Section 4.7, we make greater use of this through approximate nearest neighbors. Specifically, we apply *locality-sensitive hashing* for ℓ_2^2:

Definition 4.3.4 (Locality-sensitive hashing and approximate nearest neighbors[11]). *A family $\mathcal{H}$ of functions with domain $\mathbb{R}^d$ is called* (R, cR, p_1, p_2)-sensitive *if for any $p, q \in \mathbb{R}^d$,*

- *If $||p-q||_2^2 \leq R$, then $\Pr_{h\sim\mathcal{H}}[h(q) = h(p)] \geq p_1$.*
- *If $||p-q||_2^2 \geq cR$, then $\Pr_{h\sim\mathcal{H}}[h(q) = h(p)] \leq p_2$.*

Theorem 4.3.5 ([11]). *For any $R > 0$ and dimension d, a $(R, O(c)R, 1/n^{1/c}, 1/n^5)$-sensitive family of hash functions with query time $O(dn^{1/c})$ for $\mathbb{R}^d$ can be computed in almost-linear time.*

Locality sensitive hashing can be used to find approximate nearest neighbors in Euclidean space. In particular, by Theorem 4.3.3, it can be used to find approximate nearest neighbors in effective resistance metrics of graphs with $c \leftarrow \gamma_{\text{ann}} := \log n$:

Theorem 4.3.6 (Fact 2.7 in [11]). *Given a graph G and a set of vertices $S \subseteq V(G)$, there is a data structure D computed by an algorithm $D \leftarrow \texttt{PreprocANN}(G, S)$ with query algorithm $v' \leftarrow \texttt{ANN}_D(v)$. $\texttt{ANN}_D$ takes any vertex $v \in V(G)$ as input and uses the data structure D to return a vertex $v' \in S$ with the following properties with probability at least $1 - 1/n^{10}$:*

- *(Closeness) $\texttt{Reff}_G(v, v') \leq \min_{u\in S} \gamma_{ann} \texttt{Reff}_G(v, u)$*

- *(Preprocessing runtime)* `PreprocANN` *takes* $\tilde{O}(m)$ *time.*
- *(Query runtime)* `ANN` *takes* $\tilde{O}(1)$ *time.*

Basic facts about random spanning trees

Let $T \sim G$ denote the distribution over spanning trees of G with each tree selected with probability proportional to $\prod_{e \in E(T)} c_e$. The following shows that conditioning on a partial sample is equivalent to modifying the input graph:

Theorem 4.3.7 ([73])**.** *Consider a graph G and a set of edges $F \subseteq E(G)$. Fix a spanning tree T_0 of G and let $F_0 := E(T_0) \cap F$. Obtain a graph H of G by contracting all edges in F_0 and deleting all edges in $F \setminus F_0$. Then*

$$\Pr_{T \sim G}[T = T_0 | E(T) \cap F = F_0] = \Pr_{T' \sim H}[T' = T_0 / F_0]$$

For any set $F \subseteq E(G)$, let $H \sim G[F]$ denote the distribution over minors H of G obtained by sampling a tree $T \sim G$, contracting all edges in $F \cap E(T)$, and deleting all edges in $F \setminus E(T)$. We also use the following folklore fact extensively:

Theorem 4.3.8 ([51])**.** *Consider a graph G and an edge $e \in E(G)$. Then*

$$\Pr_{T \sim G}[e \in E(T)] = c_e^G \mathtt{Reff}_G(e)$$

Schur complements

Definition 4.3.9 (Schur complements)**.** *The* Schur complement *of a graph I with respect to a subset of its vertices $S \subseteq V(I)$, denoted $\mathtt{Schur}(I, S)$, is the weighted graph J with $V(J) = S$ with Laplacian matrix*

$$L_J = L_I[S, S] - L_I[S, S^c] L_I[S^c, S^c]^{-1} L_I[S^c, S]$$

where $M[S_0, S_1]$ denotes the submatrix of a matrix M with rows and columns indexed by S_0 and S_1 respectively.

In the above definition, it is not immediately clear that L_J is the Laplacian matrix of a graph, but it turns out to be one. Furthermore, the following associativity property holds:

Remark 4. *For any two disjoint sets of vertices $S_0, S_1 \in V(I)$ for some graph I,*

$$\mathtt{Schur}(\mathtt{Schur}(I, S_0 \cup S_1), S_0) = \mathtt{Schur}(I, S_0)$$

Also, Schur complements commute with edge deletions and contractions in the kept set S:

Remark 5 (Lemma 4.1 of [27]). *Let S be a set of vertices in a graph G and $f \in E_G(S)$. Then,*

$$\textbf{\textit{Schur}}(G \setminus f, S) = \textbf{\textit{Schur}}(G, S) \setminus f$$

and

$$\textbf{\textit{Schur}}(G/f, S) = \textbf{\textit{Schur}}(G, S)/f$$

Schur complements also have the following combinatorial property, which is the only property we use of Schur complements before Section 4.9:

Theorem 4.2.4. *Consider a graph I and some set of vertices $S \subseteq V(I)$. Let $J = \textbf{\textit{Schur}}(I, S)$. Pick a vertex $v \in S$ and generate two sequences of vertices as follows:*

- *Do a random walk in J starting at v and write down the sequence of visited vertices.*
- *Do a random walk in I starting at v and write down the sequence of visited vertices that are also in S.*

These two distributions over sequences are identical.

While this result is likely known, we include a proof for completeness. To prove this, we use the following folklore fact:

Remark 6 (Chapter IX of [17]). *Let $S \subseteq V(G)$ be a set of vertices in a graph G and let $v \in S, x \in V(G)$. Let $H = G/(S \setminus \{v\})$ and let s be the identification of the set $S \setminus \{v\}$ in H. Then*

$$\Pr_x[t_v < t_{S \setminus v}] = \frac{b_{vs}^T L_H^+ b_{xs}}{b_{vs}^T L_H^+ b_{vs}}$$

Proof of Theorem 4.2.4. Let $\{s_i\}_{i \geq 0}$ be the list of vertices visited by a random walk in J starting at v. Let $\{s'_i\}_{i \geq 0}$ be the list of vertices visited by a random walk in I starting at v, with all vertices outside of S omitted. To prove Theorem 4.2.4, it suffices to show that for any sequence $\{\ell_i\}_{i \geq 0}$ of vertices in S and any $j \geq 0$,

$$\Pr[s_{j+1} = \ell_{j+1} | s_i = \ell_i \forall i \in \{0, 1, \ldots, j\}] = \Pr[s'_{j+1} = \ell_{j+1} | s'_i = \ell_i \forall i \in \{0, 1, \ldots, j\}]$$

By the Markov property,

$$\Pr[s_{j+1} = \ell_{j+1} | s_i = \ell_i \forall i \in \{0, 1, \ldots, j\}] = \Pr_{\ell_j, J}[t_{\ell_{j+1}} < t_{S \setminus \{\ell_j, \ell_{j+1}\}}]$$

and

$$\Pr[s'_{j+1} = \ell_{j+1} | s'_i = \ell_i \forall i \in \{0, 1, \ldots, j\}] = \Pr_{\ell_j, I}[t_{\ell_{j+1}} < t_{S \setminus \{\ell_j, \ell_{j+1}\}}]$$

Let $J' = J/(S \setminus \{\ell_j, \ell_{j+1}\})$ and $I' = I/(S \setminus \{\ell_j, \ell_{j+1}\})$. In both graphs, let s be the identification of $S \setminus \{\ell_j, \ell_{j+1}\}$. By Remark 6 applied with $S \leftarrow S \setminus \{\ell_j\}$ and $v \leftarrow \ell_{j+1}$,

$$\Pr_{\ell_j, J}[t_{\ell_{j+1}} < t_{S\setminus\{\ell_j,\ell_{j+1}\}}] = \frac{b_{\ell_j s}^T L_{J'}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{J'}^+ b_{\ell_{j+1} s}}$$

and

$$\Pr_{\ell_j, I}[t_{\ell_{j+1}} < t_{S\setminus\{\ell_j,\ell_{j+1}\}}] = \frac{b_{\ell_j s}^T L_{I'}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{I'}^+ b_{\ell_{j+1} s}}$$

Since Schur complements preserve quadratic forms supported on the kept set,

$$\frac{b_{\ell_j s}^T L_{I'}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{I'}^+ b_{\ell_{j+1} s}} = \frac{b_{\ell_j s}^T L_{\mathtt{Schur}(I', \{s, \ell_j, \ell_{j+1}\})}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{\mathtt{Schur}(I', \{s, \ell_j, \ell_{j+1}\})}^+ b_{\ell_{j+1} s}}$$

By Remark 5,

$$\begin{aligned}
\mathtt{Schur}(I', \{s, \ell_j, \ell_{j+1}\}) &= \mathtt{Schur}(I/(S \setminus \{\ell_j, \ell_{j+1}\}), \{s, \ell_j, \ell_{j+1}\}) \\
&= \mathtt{Schur}(I, S)/(S \setminus \{\ell_j, \ell_{j+1}\}) \\
&= J/(S \setminus \{\ell_j, \ell_{j+1}\}) \\
&= J'
\end{aligned}$$

Therefore,

$$\frac{b_{\ell_j s}^T L_{I'}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{I'}^+ b_{\ell_{j+1} s}} = \frac{b_{\ell_j s}^T L_{J'}^+ b_{\ell_{j+1} s}}{b_{\ell_{j+1} s}^T L_{J'}^+ b_{\ell_{j+1} s}}$$

as desired. □

4.4 Structure of this Chapter and the Proof of Theorem 4.1.1

Now, we formally introduce the concepts that were alluded to in Section 4.2. In the process, we outline the structure of the paper and reduce the main result (Theorem 4.1.1) given the four main components of our algorithm: building shortcutters, selecting vertices to condition on, sampling, and computing a set of fixing edges.

Throughout this section, we use two key parameters: σ_0 and σ_1. These parameters should be thought of as distance and shortcutter size-related parameters respectively. While there are other constants (like the μs, which are all $m^{o(1)}$), these constants are purely determined by proofs in the main sections. Only σ_0 and σ_1 are traded off in order to bound the runtime of the main algorithm `ExactTree`. For more details on parameter values, see Appendix B.7.

4.4.1 Our shortcutting data structure

Recall that in Section 4.2, we stated that no vertices were in more than $m^{o(1)}$ different shortcutters. Here, we organize the shortcutters into a small number of families of disjoint shortcutters, which we call *clans*, in order to achieve this property.

Definition 4.4.1 (Organization of shortcutters)**.** *Consider a graph H obtained as a minor of G. A* cluster *is a set of vertices. In our algorithm, there are three kinds of clusters: parts, cores and shortcutters. We define parts in Definition 4.4.3. A* core *is an arbitrary cluster. A* shortcutter *is a cluster S_C that contains a core C of vertices that are "assigned" to it. A* clan *is a set of (vertex-)disjoint shortcutters. A* horde *is set of clans.*

All hordes in our algorithm satisfy the following invariant:

Invariant 4.4.2. *A horde $\mathcal{H}$ consists of at most $\ell_{\max} \leq m^{o(1)}$ clans.*

Definition 4.4.3 (Covering hordes and overlay partitions)**.** *A horde $\mathcal{H}$ is said to* cover *H if each vertex in H is in the core of some shortcutter in some clan of $\mathcal{H}$.*

Given a collection of covering hordes $\{\mathcal{H}_i\}_{i=1}^{\sigma_0}$, the overlays *$\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ are formed by refining all cores of shortcutters from all clans in $\cup_{j\geq i}\mathcal{H}_j$. More precisely, let χ_i denote the equivalence relation formed by letting $u \sim_{\chi_i} v$ if and only if for all clans $\mathcal{C} \in \cup_{j\geq i}\mathcal{H}_j$, u and v are either (a) both in the same core of $\mathcal{C}$ or (b) both not in any core of $\mathcal{C}$. Let $\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ denote the equivalence classes of χ_i.*

Since all $\mathcal{H}_i$s are covering, each $\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ is a partition of $V(H)$. A part *P is some cluster in $\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ for some $i \in [\sigma_0]$. Each part $P \in \mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ is assigned to a single core C_P of a shortcutter S_P in some clan of $\mathcal{H}_i$.*

Let $\partial\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ denote the set of boundary edges of parts in $\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$.

Organizing shortcutters into clans allows us to define properties that hold for shortcutters in a clan "on average." Now, we define various properties that cores, shortcutters, clans,

and hordes should have. After defining these properties, we summarize their relevance to bounding the shortcutted random walk simulation time in Table 4.4.1.

For each of these definitions, fix a distance scale R. We start by insisting that each core consist of closeby vertices.

Definition 4.4.4 (R-clans)**.** *Call a clan an* R-clan *if each shortcutter's core has* H*-effective resistance diameter at most* R*.*

R may be referred to even if a clan is not an R-clan (i.e. the clan may not have bounded diameter cores).

Each clan contains shortcutters that are relatively similar to one another. This way, our analysis of the shortcutting scheme can focus on the clans within a horde independently. Specifically, a clan $\mathcal{C}$ is said to be *bucketed* if the maximum size of a shortcutter in $\mathcal{C}$ is at most $4m/|\mathcal{C}|$.

Inverting this definition suggests a more convenient definition of the size of a clan.

Definition 4.4.5 (Effective size and bucketing)**.** *The* effective size *of a clan* $\mathcal{C}$*, denoted* $s_{\mathcal{C}}$*, is the following:*

$$s_{\mathcal{C}} := \frac{m}{\max_{S_C \in \mathcal{C}} |E(S_C) \cup \partial S_C|}$$

We say that a clan $\mathcal{C}$ *is* bucketed *if*

$$|\mathcal{C}| \leq 4 s_{\mathcal{C}}$$

Clans also contain shortcutters with the property that using a shortcutter bypasses many random walk steps. Specifically, the *conductance* of a shortcutter is relevant for assessing how many times it is used, as discussed in Section 4.2. For an arbitrary graph H', let $c^{H'}(S_C)$, the conductance of S_C with respect to H', be

$$c^{H'}(S_C) := \sum_{e \in E(C, V(H') \setminus S_C)} c_e^{\texttt{Schur}(H', C \cup (V(H') \setminus S_C))}$$

We define the conductance with respect to H', not H, because we need to delete edges from H in order to maintain the condition that $c^{H'}(S_C)$ is low after conditioning. H' will be a graph obtained by deleting some edges $\texttt{deleted}(\mathcal{C})$ from H:

Definition 4.4.6 (Deletion set and the deletion set condition)**.** *For a clan* $\mathcal{C} \in \mathcal{E}_i$*, maintain a set* $\texttt{deleted}(\mathcal{C})$ *of edges. This set must satisfy the* deletion set condition*, which states that no deleted edge is incident with a nonempty part. Specifically, for any* $P \in \mathcal{P}_i(\mathcal{E})$ *for which* $E(P) \neq \emptyset$*,*

$$\texttt{deleted}(\mathcal{C}) \cap (\partial_H P) = \emptyset$$

The deletion set condition ensures that precomputed random walk steps to the endpoint of a deleted edge cross a boundary edge of some part in $\mathcal{P}_i(\mathcal{E})$. We exploit this in Section 4.6.

The following condition is used to bound the precomputation work during the shortcutting algorithm:

Definition 4.4.7 (Modifiedness)**.** *We say that a clan $\mathcal{C}$ is* τ-modified *if the number of deleted edges is not too high on average:*

$$|\texttt{deleted}(\mathcal{C})| \leq \tau m^{1/\sigma_1} s_{\mathcal{C}}$$

For a clan $\mathcal{C}$, let $H_{\mathcal{C}} := H \setminus \texttt{deleted}(\mathcal{C})$. For a shortcutter $S_C \in \mathcal{C}$, let $c^{\mathcal{C}}(S_C) = c^{H_{\mathcal{C}}}(S_C)$.

Definition 4.4.8 (Conductivity)**.** *A clan $\mathcal{C}$ is* ζ-conductive *if*

$$\sum_{S_C \in \mathcal{C}} c^{\mathcal{C}}(S_C) \leq \frac{\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{R}$$

The ζ-conductive requirement is a way of saying that shortcutters within a clan are large on average. However, we also need a way of saying that they are not too large. If they are too large, the set of vertices that we are conditioning on may intersect too many shortcutters from another clan. View these vertices as being clustered into a small number of low effective resistance diameter balls and focus on each ball C' one at a time. If C' is close to the core of some shortcutter S_C, then C' does not need to be removed from S_C to make S_C carved with respect to C'. Therefore, we only need to delete C' from S_C when C' and C are well-separated. This motivates the notion of *ties*.

Consider an R-clan $\mathcal{C}$ and a shortcutter $S_C \in \mathcal{C}$. Let C' be a cluster with $H_{\mathcal{C}}$-effective resistance diameter at most βR for some $\beta \geq 1$. We say that C' is *tied* to S_C if both of the following hold:

- (Intersection) C' intersects S_C.
- (Well-separatedness) $\min_{u \in C, v \in C'} \texttt{Reff}_{H_{\mathcal{C}}}(u, v) > \beta_0 \beta R$, where $\beta_0 = 100$.

Definition 4.4.9 (Well-spacedness)**.** *An R-clan $\mathcal{C}$ is* well-spaced *if no cluster $C' \subseteq V(H)$ is tied to more than one shortcutter $S_C \in \mathcal{C}$.*

The lack of ties for well-spaced clusters ensures that deleting C' from all shortcutters in $\mathcal{C}$ does not increase the total conductance of shortcutters in $\mathcal{C}$ much.

All of the definitions that we have discussed leading up to this are used to show that conditioning once takes at most $O(m^{1+o(1)}\alpha^{o(1)})$ time. Recall from Section 4.2 that sampling the intersection of a random tree with $E(S)$ for some set of vertices S is supposed to allow us to get rid of some shortcutters because the boundary of S is small.

Definition 4.4.10 (Boundedness)**.** *Say that a clan associated with distance scale R is* κ-bounded *if*

$$\sum_{S_C \in \mathcal{C}} c^H(\partial C) \leq \frac{\kappa m}{R}$$

We now extend our definitions of clans to hordes. A horde $\mathcal{H}$ is an *R-horde* if each clan in $\mathcal{H}$ is an R-clan. A horde $\mathcal{H}$ is *bucketed* if each clan in it is bucketed. A horde is *τ-modified* if each clan in it is τ-modified. A horde $\mathcal{H}$ is *ζ-conductive* if each clan in it is ζ-conductive. A horde is *well-spaced* if each of its clans are well-spaced. A horde is *κ-bounded* if each of its clans is κ-bounded. A horde satisfies the deletion set condition if each of its clans satisfies it.

We now give definitions that are specific to hordes and to collections of hordes. Each vertex needs to have a shortcutter at every distance scale. Since a horde is associated with one distance scale R, each vertex should have a shortcutter in each horde. Now, we define a special collection of hordes called an *empire* with which sampling can be performed:

Definition 4.4.11 (Empires)**.** *An* empire $\mathcal{E}$ *is a set of covering hordes $\{\mathcal{E}_i\}_{i=1}^{\sigma_0}$, with $\mathcal{E}_i$ being an $\mu_{rad}\alpha^{i/(\sigma_0+1)}r_{min}$-horde. Define bucketedness, τ-modifiedness, ζ-conductivity, well-spacing, κ-boundedness, and the deletion set condition for empires as well if these conditions hold for all constituent hordes.*

Now, we show how these properties fit together to bound the runtime of our implementation of the full shortcutting meta-algorithm described in Section 4.2. When the random walk is at a vertex u, the meta-algorithm first finds the maximum $i \in [\sigma_0]$ for which the intersection of a shortcutter S_{P_i} with the set S is covered, where $P_i \in \mathcal{P}_i(\mathcal{E})$ is the unique part containing u. If $E(P_i) = \emptyset$, it does a standard random walk step. Otherwise, it samples whether the random walk hits an endpoint of an edge in $\texttt{deleted}(\mathcal{C})$ before exiting S_{P_i}, where $\mathcal{C}$ is the clan containining S_{P_i}. If so, it uses offline shortcutting to shortcut to $\texttt{deleted}(\mathcal{C})$. Otherwise, it uses online shortcutting to shortcut to the boundary of S_{P_i}.

The above discussion cites three kinds of random walk-related work and one kind of precomputation work. To bound the random walk-related work, we exploit Lemma 4.2.3. Lemma 4.2.3 requires two things: a bound on conductance and a bound on the distance to an unvisited vertex. Work done using a part P_i is charged to the clan containing S_{P_i} as follows:

This table does not discuss the well-spacedness or bucketing conditions. Well-spacedness is used to bound the conductivity increase due to carving in the proof of Lemma 4.4.15, while the bucketing condition is used to bound the number of edges added to $\texttt{deleted}(\mathcal{C})$ in the proof of Lemma 4.4.18.

4.4.2 Creating and maintaining shortcutters

Our algorithm maintains an empire $\mathcal{E}$. Before each conditioning phase, it recomputes shortcutters in order to endow them with properties that are lost after one round of conditioning:

Lemma 4.4.12. *There is an almost-linear time algorithm* `RebuildEmpire`$(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ *that, when given a set of covering hordes* $\{\mathcal{H}_i\}_{i=1}^{\sigma_0}$ *with* $\mathcal{H}_i$ *associated with distance scale* $\alpha^{i/(\sigma_0+1)}r_{min}$ *in a graph* H*, returns an empire* $\mathcal{E} = \{\mathcal{H}'_i\}_{i=1}^{\sigma_0}$ *with the following properties:*

- *(Bucketing)* $\mathcal{E}$ *is bucketed.*
- *(Conductivity) If each horde* $\mathcal{H}_i$ *is* ζ*-conductive, then* $\mathcal{E}$ *is* $(8\log n)\zeta + (16\log n)\mu_{app}$*-conductive.*
- *(Well-spacedness)* $\mathcal{E}$ *is well-spaced.*
- *(Boundedness) If each horde* $\mathcal{H}_i$ *is* κ*-bounded, then* $\mathcal{E}$ *is* $\kappa+\kappa_0$ *bounded, for* $\kappa_0 \leq m^{o(1)}$*.*
- *(Modifiedness and deletion set condition) If each horde* $\mathcal{H}_i$ *is* τ*-modified, then* $\mathcal{E}$ *is* τ*-modified as well. Furthermore, if the deletion set condition is satisfied in each clan of each* $\mathcal{H}_i$*, it continues to be satisfied in* $\mathcal{E}$*.*
- *(Clan growth) The number of clans in* $\mathcal{E}$ *is at most* $\mu_{app}\log n$ *times as high as the number of clans in all of the* $\mathcal{H}_i$*s.*
- *(Containment) For any* $i \in [\sigma_0]$*, consider any part* $P \in \mathcal{P}_i(\mathcal{E})$*. There is a unique part* $Q \in \mathcal{P}_i(\{\mathcal{H}_j\}_j)$ *for which* $P \subseteq Q$*. Furthermore,* $C_P \subseteq C_Q$ *and* $S_P \subseteq S_Q$*.*

Our spanning tree generation algorithm starts by calling `RebuildEmpire` on the set of hordes consisting of one clan, each of which just contains the one shortcutter $V(G)$. These hordes are clearly covering and have $\zeta = 0$, $\kappa = 0$, and $\tau = 0$. `RebuildEmpire` is useful to call on the remnants of empires after conditioning later on in order to achieve the containment property. Containment is essential to our notion of progress, as discussed in Section 4.2.

4.4.3 Selecting parts to condition on

Given an empire $\mathcal{E}$ with respect to a graph H, we can choose a set of vertices S to condition on. The set S is small enough that, when carved out of shortcutters in $\mathcal{E}$, does not increase their conductivity too much. The upside of carving is that each vertex in S is close to the core of any shortcutter that it is in. We now define this precisely:

Definition 4.4.13 (Active parts and carving)**.** *A part* P *is called* active *if it has nonempty interior, i.e.* $E(P) \neq \emptyset$*. A shortcutter is called* active *if any part assigned to it is active.*

A shortcutter S_C *in an* R*-clan has been* carved *with respect to* $S \subseteq V(G)$ *if each vertex* $v \in S \cap S_C$ *is within* H*-effective resistance distance* $\mu_{carve}R$ *of all vertices in* C*. An* R*-clan*

$\mathcal{C}$ in an empire $\mathcal{E}$ has been carved *with respect to S if all of its active shortcutters have been carved with respect to S. An R-horde $\mathcal{H}$ in an empire $\mathcal{E}$ has been* carved *with respect to S if each clan in it has been carved with respect to S. An empire $\mathcal{E}$ has been* carved *with respect to S if each of its hordes has been carved with respect to S.*

The routine `ConditioningVerts` both (a) selects parts $\mathcal{K}$ for conditioning on and (b) removes vertices from the shortcutters of the input empire $\mathcal{E}$ in order to ensure that it is carved with respect to $\cup_{P\in\mathcal{K}}P$. The `ConditioningVerts` subroutine maintains internal state and is the only method that exploits the "Containment" guarantee of Lemma 4.4.12. The "Progress" input condition in the following definition captures the fact that partial sampling eliminates edges in the induced subgraph of the previously chosen parts:

Definition 4.4.14 (`ConditioningVerts` input conditions)**.** *Given an empire $\mathcal{E}$ in a graph H, the algorithm `ConditioningVerts`$(\mathcal{E})$ returns a set of parts $\mathcal{K}$ to condition on and removes vertices from the shortcutters in the empire $\mathcal{E}$ to obtain $\mathcal{E}'$. Let $\mathcal{E}_{prev}$ be the argument supplied to the previous call to `ConditioningVerts`, let $\mathcal{K}_{prev}$:= `ConditioningVerts`$(\mathcal{E}_{prev})$, and let $\mathcal{E}'_{prev}$ be the empire $\mathcal{E}_{prev}$ after being modified by `ConditioningVerts`. Let H_{prev} be the graph in which $\mathcal{E}_{prev}$ lies. The following conditions are the* input conditions *for `ConditioningVerts`:*

- *(Parameters) $\mathcal{E}$ is a bucketed, ζ-conductive, well-spaced, τ-modified, and κ-bounded empire that satisfies the deletion set condition.*
- *(Containment) For any $i \in [\sigma_0]$, consider any part $P \in \mathcal{P}_i(\mathcal{E})$. There is a unique part $Q \in \mathcal{P}_i(\mathcal{E}_{prev})$ for which $P \subseteq Q$. Furthermore, $C_P \subseteq C_Q$ and $S_P \subseteq S_Q$.*
- *(Progress) For each $P \in \mathcal{K}_{prev}$, $E_H(P) = \emptyset$.*

Lemma 4.4.15. *Given an empire $\mathcal{E} = \{\mathcal{E}_i\}_{i=1}^{\sigma_0}$ in a graph H that satisfies the input conditions given in Definition 4.4.14, `ConditioningVerts`$(\mathcal{E})$ returns a set of parts $\mathcal{K}$ to condition on and removes vertices from the shortcutters in the empire $\mathcal{E}$ to obtain $\mathcal{E}'$. Let $S = \cup_{P\in\mathcal{K}}P \subseteq V(H)$. Then the following guarantees are satisfied:*

- *(Conductivity) $\mathcal{E}'$ is a bucketed, τ-modified, $\zeta + 10(\log m)\mu_{app}(\ell_{\max} + \tau)$-conductive, well-spaced, κ-bounded empire that satisfies the deletion set condition.*
- *(Carving) $\mathcal{E}'$ is carved with respect to S.*

4.4.4 Making enough progress during each round of conditioning

In the previous section, we showed that S is small enough to ensure that carving S out of all shortcutters in $\mathcal{E}$ does not increase the conductivity of $\mathcal{E}$ too much. We now show that S is large enough to make a lot of progress. Specifically, we show the following:

Lemma 4.4.16. *Consider a sequence of calls $\mathcal{K}^j \leftarrow$ `ConditioningVerts`$(\mathcal{E}^j)$ that modifies $\mathcal{E}^j$ to obtain $(\mathcal{E}^j)'$. Suppose that H^j is the graph in which $\mathcal{E}^j$ is defined. Suppose that for each $j > 0$, $\mathcal{E} \leftarrow \mathcal{E}^j$, $\mathcal{E}_{prev} \leftarrow \mathcal{E}^{j-1}$, $\mathcal{K}_{prev} \leftarrow \mathcal{K}^{j-1}$ satisfies the input conditions in Definition 4.4.14. Let*

$$j_{final} = (2\sigma_1)^{2\sigma_0}$$

Then $E(H^{j_{final}}) = \emptyset$.

This means that only $(2\sigma_1)^{2\sigma_0} \leq o(\log n)$ rounds of conditioning are necessary to sample a random spanning tree.

4.4.5 Conditioning on the intersection of a random tree with the selected vertices

Now that each shortcutter S_C only intersects vertices to condition on that are close to C, we can make the idea for using online shortcutting in Section 4.2 a reality:

Lemma 4.4.17. *Let $\mathcal{K} \subseteq \cup_{i=1}^{\sigma_0} \mathcal{P}_i(\mathcal{E})$ be a set of parts. Let $F = \cup_{P \in \mathcal{K}} E(P)$ and $S = \cup_{P \in \mathcal{K}} P$. Suppose that the empire $\mathcal{E}$ is ζ-conductive, κ-bounded, τ-modified, satisfies the deletion set condition, and has been carved with respect to S. Then, there is an algorithm* `PartialSample`$(\mathcal{E}, \mathcal{K})$ *that returns the intersection of a random spanning tree T in H with F in $\tilde{O}(((\zeta + \kappa)\mu_{carve} + \tau)\ell_{\max} m^{1+1/\sigma_1} \alpha^{1/(\sigma_0+1)})$ time.*

4.4.6 Fixing shortcutters

After computing $T \cap F \leftarrow$ `PartialSample`, contracting all edges in $F \cap T$ in H, and deleting all edges in $F \setminus T$ from H, $\mathcal{E}$ is no longer an empire with respect to H. In particular, the well-spacedness, ζ-conductivity, and core diameter conditions break down. Well-spacedness and diameter can be fixed by applying `RebuildEmpire`. However, the ζ-conductivity constraint accumulates over an old value. We could recompute the empire from scratch, but that forgoes the containment property that is so important to establishing progress. We deal with this issue by adding edges to `deleted`$(\mathcal{C})$ for each clan $\mathcal{C}$ in $\mathcal{E}$:

Lemma 4.4.18. *Let H be a graph, $\mathcal{E}$ be an empire in H and $\mathcal{K}$ be a set of parts. Let $S = \cup_{P \in \mathcal{K}} P$ and let $F = \cup_{P \in \mathcal{K}} E(P)$. Let $H' \sim H[F]$. Suppose that the following input conditions hold $\mathcal{E}$:*

- *(Bucketing) The empire $\mathcal{E}$ is bucketed.*
- *(Carving) $\mathcal{E}$ is carved with respect to S.*

With high probability over H', `FixShortcutters`$(\mathcal{E}, H', \mathcal{K})$ *adds edges to the deletion set of each clan of $\mathcal{E}$ to obtain a set of covering hordes $\{\mathcal{H}_i'\}$ with the following properties:*

- *(Boundedness) For each i, if $\mathcal{E}_i$ is κ-bounded, then $\mathcal{H}'_i$ is $\ell\kappa$-bounded, where $\ell = \sum_{i=1}^{\sigma_0} |\mathcal{E}_i|$.*
- *(Modifiedness and deletion set condition) For each i, if $\mathcal{E}_i$ is τ-modified and satisfies the deletion set condition, then $\mathcal{H}'_i$ is $\mu_{mod}(\tau+\zeta)$-modified and also satisfies the deletion set condition.*
- *(Conductivity) For each i, if $\mathcal{E}_i$ is ζ-conductive with respect to H, then $\mathcal{H}'_i$ is at most 7ζ-conductive with respect to H'.*

Futhermore, it does so in $m^{1+o(1)}$ time.

4.4.7 An $m^{1+o(1)}\alpha^{o(1)}$ time algorithm for exact random spanning tree generation

We now tie the results from the previous sections together to prove Theorem 4.1.1. We prove this result using the algorithm `ExactTree`, which simply chains the algorithms from the previous sections in order:

Algorithm 2: `ExactTree`(G)

```
H ← G
// the set of hordes which contain one clan consisting of one
   shortcutter (the entire graph)
E ← {{{V(G)}}}_{i=1}^{σ0}
T ← ∅
while E(H) ≠ ∅ do
    E ← RebuildEmpire(E)
    K ← ConditioningVerts(E)
    T ← T ∪ PartialSample(E, K)
    Contract all edges in H added to T and delete all other edges internal to parts of K
    FixShortcutters(E, H, K)
end
return T
```

Most of the effort in proving the above result boils down to checking that all of the input conditions are satisfied for each of the subroutines that `ExactTree` calls.

Proof of Theorem 4.1.1. Invariant 4.4.2. Each of `RebuildEmpire`, `ConditioningVerts`, `PartialSample`, and `FixShortcutters` increases the number of clans by at most a factor of $(\log m)\mu_{\text{app}}$. By Lemma 4.4.16, only $(2\sigma_1)^{2\sigma_0}$ iterations take place. Since there is only one clan initially, the number of clans at the end is at most

$$((\log m)\mu_{\text{app}})^{(2\sigma_1)^{2\sigma_0}} = \ell_{\max} \leq m^{o(1)}$$

as desired.

$\kappa \leq \kappa_{\max}$. Each of the subroutines called in the while loop increases κ by at most a factor of $\ell_{\max}$ and additively by at most $\kappa_0 \leq m^{o(1)}$. Therefore,

$$\begin{aligned}\kappa &\leq (\ell_{\max})^{(2\sigma_1)^{2\sigma_0}} \\ &\leq ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{4\sigma_0}} \\ &= \kappa_{\max} \\ &\leq m^{o(1)}\end{aligned}$$

as desired.

$\tau \leq \tau_{\max}$ and $\zeta \leq \zeta_{\max}$. Each subroutine call increases $\max(\tau, \zeta)$ by a factor of at most $10(\log m)\mu_{\text{app}}$ and additively by at most $10(\log m)\mu_{\text{app}}\ell_{\max}\mu_{\text{mod}}$. Therefore,

$$\begin{aligned}\max(\tau, \zeta) &\leq (10(\log m)\mu_{\text{app}}\ell_{\max}\mu_{\text{mod}})^{(2\sigma_1)^{2\sigma_0}} \\ &\leq ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{8\sigma_0}} \\ &= \max(\tau_{\max}, \zeta_{\max}) \\ &\leq m^{o(1)}\end{aligned}$$

as desired.

Well-definedness. Start with `RebuildEmpire`. At the beginning of the algorithm, $\zeta = 0$, $\kappa = 0$, and all of the deletion sets are empty, so the deletion set condition is satisfied. $\mathcal{E}$ is not an empire when it is supplied to `RebuildEmpire`, but is a set of covering hordes because either (a) this is the first iteration and the cores are all $V(G)$ or (b) Lemma 4.4.18 states that the hordes $\mathcal{H}_i$ are covering. Therefore, `RebuildEmpire`'s input conditions given in Lemma 4.4.12 are always respected.

Next, consider `ConditioningVerts`. The "Parameters" condition is the "Parameters" guarantee from Lemma 4.4.12. The "Containment" condition follows from the "Containment" guarantee of Lemma 4.4.12, along with the fact that `FixShortcutters` only adds to the deletion sets of the clans and `PartialSample` does not change $\mathcal{E}$. Line 8 of `ExactTree` contracts or deletes each edge internal to each part in $\mathcal{K}$. Therefore, the "Progress" condition is satisfied afterwards.

The desired parameter bounds for `PartialSample` are given in the "Boundedness and covering" guarantee of Lemma 4.4.15. The carving condition of Lemma 4.4.17 is the "Carving" guarantee of Lemma 4.4.15.

Finally, deal with `FixShortcutters`. The input conditions for Lemma 4.4.18 are given directly as the "Carving" guarantee of Lemma 4.4.15, the "Bucketing" guarantee of Lemma 4.4.12, and the fact that removing vertices from shortcutters preserves the bucketing guarantee.

Correctness. By Theorem 4.3.7, sampling a random tree in some H is equivalent to partial sampling with $F = \cup_{P \in \mathcal{K}} E(P)$ and sampling a tree in the graph obtained by contracting the chosen edges in F and deleting all others. By Lemma 4.4.17, `PartialSample` returns a valid sample from a uniformly random spanning tree of H intersected with F. Therefore, once $E(H) = \emptyset$, T has been completely sampled and is valid.

Runtime. By Lemma 4.4.16, the while loop runs at most $(2\sigma_1)^{2\sigma_0} \leq m^{o(1)}$ times. `RebuildEmpire`, `ConditioningVerts`, `PartialSample`, and `FixShortcutters` each take $m^{1+o(1)}\alpha^{o(1)+1/(\sigma_0+1)}$ time by Lemmas 4.4.12, 4.4.15, 4.4.17, and 4.4.18 respectively and our bounds on $\ell_{\max}$, $\tau_{\max}$, $\kappa_{\max}$, and $\zeta_{\max}$. Contracting and deleting edges only takes $O(m)$ time. Therefore, the entire algorithm only takes $O(m^{1+o(1)}\alpha^{o(1)+1/(\sigma_0+1)})$ time. Since σ_0 is superconstant, this runtime is $m^{1+o(1)}\alpha^{o(1)}$, as desired. □

4.4.8 An $m^{1+o(1)}\epsilon^{-o(1)}$-time algorithm for generating a random spanning tree from a distribution with total variation distance ϵ from uniform

In Section B.9, we give a simple reduction that proves Theorem 4.1.2 given just Theorem 4.1.1. The reduction samples the intersection of a random tree with a part of the graph with polynomial aspect ratio and smallest resistances. Conditioning on this part of the graph removes the edges with smallest resistance from the graph. A ball-growing-type technique and Theorem 4.1.1 ensures that each round of conditioning eliminates a number of edges from the graph proportional to the amount of work done.

4.5 Computing Shortcutters

In this section, we prove Lemma 4.4.12 by implementing `RebuildEmpire`. `RebuildEmpire` starts by building cores in `CoveringCommunity`. It then builds shortcutters around those cores in `Voronoi`.

4.5.1 Building cores

In order to ensure that `Voronoi` can actually build good shortcutters around the cores, `CoveringCommunity` outputs cores that are organized into well-separated families. We start by giving some relevant definitions.

Definition 4.5.1 (`CoveringCommunity`-related definitions: families and communities)**.** *Consider a graph I. A* family *is a set of clusters $\mathcal{F}$. A* community *is a set of families. An R-*family *is a family of clusters with I-effective resistance diameter at most R. An R-*community *is a community consisting of $R_{\mathcal{F}}$-families for possibly different values $\frac{R}{\mu_{rad}} \leq R_{\mathcal{F}} \leq R$. An (R,γ)-*well-separated family *is an R-family $\mathcal{F}$ with the additional property that the I-effective resistance distance between any two vertices in different clusters in $\mathcal{F}$ is at least γR. A γ-*well-separated community *is a community that consists of $(R_{\mathcal{F}},\gamma)$-well-separated families $\mathcal{F}$.*

Notice that in this definition, γ is constant across all families but $R_{\mathcal{F}}$ is not. Well-separatedness is important for obtaining ζ-conductive shortcutters. However, Lemma 4.4.12 also demands that cores have small total boundary. The boundary size is judged based on a cluster C coming from remains of a former empire. This motivates the definition of X-constraint.

Definition 4.5.2 (`CoveringCommunity`-related definitions: X-constraint and boundedness)**.** *Say that a community is* X-constrained *if all vertices in clusters within families of the community are in X. Say that an X-constrained R-community $\mathcal{D}$ is* κ-bounded *if for any family $\mathcal{F} \in \mathcal{D}$,*

$$\sum_{C \in \mathcal{F}} c^I(\partial C) \leq \frac{\kappa |E_I(X) \cup \partial_I X|}{R}$$

We construct cores using the following result, which is proven in Section B.2.1:

Lemma 4.5.3. *The algorithm $\texttt{CoveringCommunity}_D(X, I, R)$, when given a cluster X, a graph I, a radius R, and a Johnson-Lindenstrauss embedding D of the vertices of $V(I)$, returns an $\mu_{rad}R$-community $\mathcal{D}$ with the following properties:*

- *(Input constraint) $\mathcal{D}$ is X-constrained.*
- *(Covering) Each vertex in X is in some cluster of some family in $\mathcal{D}$.*

- *(Boundedness) Each family $\mathcal{F} \in \mathcal{D}$ satisfies*

$$\sum_{C \in \mathcal{F}} c^I(\partial_I C) \leq \frac{\kappa_0 |E_I(X) \cup \partial_I X|}{R} + c^I(\partial_I X)$$

- *(Well-separatedness) $\mathcal{D}$ is γ_{ds}-well-separated.*
- *(Number of families) $\mathcal{D}$ has at most μ_{app} families.*

Furthermore, `CoveringCommunity` *takes almost-linear time in $|E(X) \cup \partial X|$.*

D is only given as input to `CoveringCommunity` for runtime purposes.

We now briefly discuss how each of these properties relates to properties of cores in Lemma 4.4.12. We apply `CoveringCommunity`(X, I, R) to each core C in the R-clain $\mathcal{C}$ with $I \leftarrow H_{\mathcal{C}}$. The input, covering, and number of families constraints of Lemma 4.5.3 relate to the parts of the containment constraint of Lemma 4.4.12 as it relates to C. The boundedness constraint relates to the boundedness constraint of Lemma 4.5.3. The well-separatedness constraint is used later on to obtain a ζ-conductive clan of shortcutters.

`CoveringCommunity` is very similar to the sparse cover constructions (for example [13]). When growing each cluster, though, (1) consider all nearby vertices for adjacent growth, not just adjacent balls and (2) ball-grow the resulting cluster afterwards to ensure that it has small boundary size. We give this construction in the appendix.

4.5.2 Building the shortcutters

Now we exploit the well-separatedness of the cores to build good shortcutters. We start by showing a result that allows us to translate well-separatedness into a shortcutter conductance upper bound:

Lemma 4.5.4. *Consider a $\gamma_{ds} = 2^{(\log n)^{2/3}}$ well-separated R-family of clusters $\mathcal{F}$ in a graph G. Let $H :=$ `Schur`$(G, \cup_{C \in \mathcal{F}} C)$. Then*

$$\sum_{e \in E(C,C'), C \neq C' \in \mathcal{F}} \frac{\texttt{Reff}_H(e)}{r_e^H} \leq \mu_{app} |\mathcal{F}|$$

We prove this lemma in the appendix. We now give a brief description of the proof. If each cluster is a single vertex, the result is equivalent to Foster's Theorem (Remark 12). One can prove Remark 12 combinatorially by running the Aldous-Broder algorithm on H and writing down the sequence of relevant cluster visits; i.e. visits that end up adding a new edge to the tree. This sequence has the property that no two clusters can alternate more than once; otherwise at least one would be covered. Such sequences have been investigated before; they are *Davenport-Schinzel sequences*. In particular, Davenport-Schinzel sequences

are known to have linear length in the number of letters. This means that only a linear number of edges can be added.

This reasoning generalizes to the case where clusters are not single vertices. If the random walk alternates between two clusters more than $\log_{\gamma_{\text{ds}}} n$ times, it covers one of them with probability at least $1 - 1/n$. Therefore, the sequence of visits is $\log_{\gamma_{\text{ds}}} n$-Davenport-Schinzel. Picking γ_{ds} to be a high enough subpolynomial value gives a linear bound on the length of the sequence. There are some minor complications caused by the fact that a cluster can appear multiple times in a row in the sequence, so it is not truly Davenport-Schinzel. We describe how to cope with this issue in the proof of Lemma 4.5.4.

If we could compute the Schur complement H, we could directly run Aldous-Broder on this Schur complement and sample the intersection of a random tree with all of intracluster edges in $\mathcal{F}$. This is expensive, though. Furthermore, the resulting graph is no longer sparse. Instead, we build shortcutters around each cluster in order to make it so that using a shortcutter effectively simulates one step of the Schur complement random walk in $\mathcal{F}$. Intuitively, we design a shortcutter around each core that has the property that using it takes the random walk from a cluster C to a vertex from which it is more likely to hit some cluster besides C before returning to C. This intuition motivates the following definition:

Definition 4.5.5 (Potential level sets)**.** *Consider a family of clusters $\mathcal{F}$ in a graph G and a cluster $C \in \mathcal{F}$. Let $S_{\mathcal{F}}(p, C)$ denote the cluster of vertices $v \in V(G)$ for which*

$$\Pr_v[t_C < t_{\mathcal{F}\setminus\{C\}}] \geq 1 - p$$

Due to the interpretation of probabilities as electrical potentials, this set can be computed using one Laplacian solve on G with each cluster in $\mathcal{F}$ identified to a vertex:

Remark 7. *For any family $\mathcal{F}$ in a graph G, a cluster $C \in \mathcal{F}$, and a number $p \in (0, 1)$. Then a set $S' \subseteq V(G)$ with $S_{\mathcal{F}}(p - 1/(m\alpha)^4, C) \subseteq S' \subseteq S_{\mathcal{F}}(p + 1/(m\alpha)^4, C)$ can be computed in near-linear time in $|E(G \setminus (\cup_{C' \in \mathcal{F}} C'))|$.*

The upper bound on S_C in the following lemma is used to show that $\mathcal{C}$ is well-spaced; i.e. that the shortcutters are far away from one another. The lower bound is used to show that $\mathcal{C}$ is ζ-conductive for some $\zeta \leq m^{o(1)}\alpha^{o(1)}$; i.e. that using those shortcutters saves a lot of work. We show the following in Section B.2.2:

Lemma 4.5.6. *The algorithm `Voronoi`$(I, \mathcal{F})$ takes a family $\mathcal{F}$ in the graph I and outputs a clan $\mathcal{C}$ in near-linear time in $|E(Z) \cup \partial Z|$ with the property that for each $C \in \mathcal{F}$, there is a shortcutter $S_C \in \mathcal{C}$ with the property that $S_{\mathcal{F}}(1/(8 \log n), C) \subseteq S_C \subseteq S_{\mathcal{F}}(1/8, C)$, where $Z = V(I) \setminus (\cup_{C \in \mathcal{F}} C)$.*

One could satisfy the above lemma just by returning a clan of $S_{\mathcal{F}}(p, C)$s for some constant $p \in [1/(8 \log n), 1/8]$. $S_{\mathcal{F}}(p, C)$ can be computed using one approximate Laplacian system solve. We are not aware of a way to compute all shortcutters for $\mathcal{F}$ efficiently. Instead, `Voronoi` partitions clusters into two megaclusters in $\log n$ different ways so that no two

clusters are in the same part of every partition. Then, it computes $S_{\mathcal{F}}(1/(8\log n), P)$ where P is one side of the partition and intersects all of the partitions to obtain shortcutters. This only requires $O(\log n)$ Laplacian solves. The Laplacian solves are all in the graph with clusters identified to different vertices. This graph has size $|E(Z) \cup \partial Z|$.

As we saw in the Algorithm Overview, these shortcutters are modified many times over the course of the algorithm. The intuitive description of these shortcutters quickly breaks down after any modification. We use Lemma 4.5.4, along with the following proposition, to establish that the clan output by `Voronoi` is ζ-conductive for some reasonable ζ:

Proposition 4.5.7. *Consider a γ-well-separated R-family $\mathcal{F}$ in $Y \subseteq X \subseteq V(H_{\mathcal{C}})$ and let $I := \texttt{Schur}(H_{\mathcal{C}}, \cup_{C\in\mathcal{F}} C)$. Suppose that*

$$c^{\texttt{Schur}(H_{\mathcal{C}}, Y\cup(V(H)\setminus X))}(E(Y, V(H)\setminus X)) \leq \xi$$

For any $C \in \mathcal{F}$, let

$$\Delta_{\mathcal{F}}(C) := \sum_{e\in E_I(C,C'), C'\neq C\in\mathcal{F}} \frac{\texttt{Reff}_I(e)}{r_e^I}$$

Let $\mathcal{F}' := \mathcal{F} \cup \{V(H) \setminus X\}$ and consider any clusters S_C with $S_{\mathcal{F}'}(p, C) \subseteq S_C$ for all $C \in \mathcal{F}$. Then

$$\sum_{C\in\mathcal{F}} c^{\mathcal{C}}(S_C) \leq \left(\sum_{C\in\mathcal{F}} \frac{\Delta_{\mathcal{F}}(C)}{p(\gamma-4)R}\right) + \frac{\xi}{p}$$

A relatively simple argument about electrical potentials and their relationship with effective resistances shows the following, which is used to establish well-spacedness:

Proposition 4.5.8. *Consider a family $\mathcal{F}$ in a graph H. Let $C \in \mathcal{F}$ be a cluster with H-effective resistance diameter R. Consider some S_C for which $C \subseteq S_C \subseteq S_{\mathcal{F}}(p, C)$ for any $p \in (0, 1/2)$. Consider a cluster C' that is tied to C. Then $C' \subseteq S_{\mathcal{F}}(p + 3/10, C)$.*

We prove all of these statements in the appendix.

4.5.3 Tying the parts together

Now, we combine the statements of the previous two subsections into a proof of Lemma 4.4.12. We first implement the algorithm `RebuildEmpire`:

Algorithm 3: `RebuildEmpire`$(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$

1 $D \leftarrow$ factor 2 error Johnson-Lindenstrauss embedding of H into $\mathbb{R}^{C_1 \log n}$ dimensions
2 **foreach** $i \in \{1, 2, \ldots, \sigma_0\}$ **do**
3 $\mathcal{E}_i \leftarrow \emptyset$
4 **foreach** *clan* $\mathcal{C} \in \mathcal{H}_i$ **do**
5 $T \leftarrow \mu_{\text{app}}$
6 $\{\mathcal{C}_{1k}\}_{k=1}^{\log m}, \{\mathcal{C}_{2k}\}_{k=1}^{\log m}, \ldots, \{\mathcal{C}_{Tk}\}_{k=1}^{\log m}$ all are initialized to $\emptyset$
7 **foreach** *shortcutter* $S_C \in \mathcal{C}$ **do**
8 $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_T \leftarrow \emptyset$
9 For each $j \in [T]$, $\mathcal{F}_j \leftarrow j$th family of the community `CoveringCommunity`$_D(C, H_{\mathcal{C}}, \alpha^{i/(\sigma_0+1)} r_{min})$ `// bucketing`
10 **foreach** $j \in [T]$ **do**
11 $\mathcal{F}_{j1} \leftarrow \mathcal{F}_j$
12 **for** $k = 1, 2, \ldots, \log m$ **do**
13 $\mathcal{C}_{jk} \leftarrow \mathcal{C}_{jk} \cup$ the shortcutters in `Voronoi`$(H_{\mathcal{C}}, \mathcal{F}_{jk} \cup \{V(H) \setminus S_C\})$ with size at most 2^k
14 `deleted`$(\mathcal{C}_{jk}) \leftarrow$ `deleted`$(\mathcal{C})$
15 $\mathcal{F}_{j(k+1)} \leftarrow$ the subset of $\mathcal{F}_{jk}$ with shortcutters in `Voronoi`$(H_{\mathcal{C}}, \mathcal{F}_{jk} \cup \{V(H) \setminus S_C\})$ with size greater than 2^k
16 $\mathcal{E}_i \leftarrow \mathcal{E}_i \cup_{j \in [T], k \in [\log m]} \{\mathcal{C}_{jk}\}$
17 **foreach** $i \in \{1, 2, \ldots, \sigma_0\}$ **do**
18 **foreach** $P \in \mathcal{P}_i(\mathcal{E})$ **do**
 `// will justify well-definedness in ''Containment'' analysis`
19 Let Q be the unique part in $\mathcal{P}_i(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ containing P
20 Let C_P be an arbitrary core in an $\mathcal{F}_j$ obtained from the `CoveringCommunity`$(C_Q, H_{\mathcal{C}}, \alpha^{i/(\sigma_0+1)} r_{min})$ call on Line 9
21 Let S_P be the shortcutter in $\mathcal{E}$ assigned to C_P
22 **return** $\{\mathcal{E}_i\}_{i=1}^{\sigma_0}$

We now prove Lemma 4.4.12 given all of the propositions and lemmas in this section.

Lemma 4.4.12. *There is an almost-linear time algorithm* `RebuildEmpire`$(\{\mathcal{H}_i\}_{i=1}^{\sigma_0})$ *that, when given a set of covering hordes* $\{\mathcal{H}_i\}_{i=1}^{\sigma_0}$ *with* $\mathcal{H}_i$ *associated with distance scale* $\alpha^{i/(\sigma_0+1)} r_{min}$ *in a graph* H*, returns an empire* $\mathcal{E} = \{\mathcal{H}'_i\}_{i=1}^{\sigma_0}$ *with the following properties:*

- *(Bucketing)* $\mathcal{E}$ *is bucketed.*

- *(Conductivity) If each horde $\mathcal{H}_i$ is ζ-conductive, then $\mathcal{E}$ is $(8\log n)\zeta + (16\log n)\mu_{app}$-conductive.*
- *(Well-spacedness) $\mathcal{E}$ is well-spaced.*
- *(Boundedness) If each horde $\mathcal{H}_i$ is κ-bounded, then $\mathcal{E}$ is $\kappa+\kappa_0$ bounded, for $\kappa_0 \leq m^{o(1)}$.*
- *(Modifiedness and deletion set condition) If each horde $\mathcal{H}_i$ is τ-modified, then $\mathcal{E}$ is τ-modified as well. Furthermore, if the deletion set condition is satisfied in each clan of each $\mathcal{H}_i$, it continues to be satisfied in $\mathcal{E}$.*
- *(Clan growth) The number of clans in $\mathcal{E}$ is at most $\mu_{app}\log n$ times as high as the number of clans in all of the $\mathcal{H}_i$s.*
- *(Containment) For any $i \in [\sigma_0]$, consider any part $P \in \mathcal{P}_i(\mathcal{E})$. There is a unique part $Q \in \mathcal{P}_i(\{\mathcal{H}_j\}_j)$ for which $P \subseteq Q$. Furthermore, $C_P \subseteq C_Q$ and $S_P \subseteq S_Q$.*

Proof of Lemma 4.4.12. **Radii of clans.** `CoveringCommunity` outputs an $\mu_{\text{rad}}\alpha^{i/(\sigma_0+1)}r_{min}$-community, so each cluster has H effective resistance diameter at most $\mu_{\text{rad}}\alpha^{i/(\sigma_0+1)}r_{min}$, as desired.

Bucketing. Each clan $\mathcal{C}_{jk} \in \mathcal{E}_i$ arises from one clan $\mathcal{C}$ of $\mathcal{H}_i$ and $\mathcal{F}_{jk}$ for that clan $\mathcal{C}$. By Line 13,

$$\max_{S_{C'} \in \mathcal{C}_{jk}} |E(S_{C'}) \cup \partial S_{C'}| \leq 2^k$$

Therefore,

$$s_{\mathcal{C}_{jk}} \geq \frac{m}{2^k}$$

Next, we bound $|\mathcal{C}_{jk}|$. The shortcutters in $\texttt{Voronoi}(H_{\mathcal{C}_{jk}}, \mathcal{F}_{jk})$ are disjoint because the shortcutters $\mathcal{S}_{\mathcal{F}}(1/2-\epsilon, C')$ for $C' \in \mathcal{F}$ are disjoint for any $\epsilon \in (0, 1/2)$. Line 15 ensures that the clusters $\mathcal{F}_{jk}$ are cores of shortcutters with size at least 2^{k-1}. The shortcutters in $\mathcal{C}_{jk}$ are disjoint because `Voronoi` produces disjoint subclusters of S_C and the S_Cs are disjoint by definition of $\mathcal{C}$. This means that

$$|\mathcal{C}_{jk}| \leq m/2^{k-1}$$

Combining inequalities shows that

$$|\mathcal{C}_{jk}| \leq 2s_{\mathcal{C}_{jk}}$$

which implies bucketing.

Conductivity. Suppose that each $\mathcal{H}_i$ is ζ-conductive and consider a clan $\mathcal{C} \in \mathcal{H}_i$. Consider a shortcutter $S_C \in \mathcal{C}$, $j \in [T]$, and $k \in [\sigma_1]$. Let $X \leftarrow S_C$, $Y \leftarrow C$, $\mathcal{F} \leftarrow \mathcal{F}_{jk}$,

$\gamma \leftarrow \gamma_{\text{ds}}$, and $p \leftarrow 1/(8 \log n)$. This is valid because $\mathcal{F}_{jk}$ is a γ_{ds}-well-separated $\alpha^{i/(\sigma_0+1)} r_{min}$-family by Lemma 4.5.3. Therefore, Proposition 4.5.7 applies and shows that

$$\sum_{C' \in \mathcal{F}_{jk}} c^{\mathcal{C}_{jk}}(S_{C'}) \leq \left(\sum_{C' \in \mathcal{F}_{jk}} \frac{2\Delta_{\mathcal{F}_{jk}}(C')}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} \right) + \frac{c^{\mathcal{C}}(S_C)}{p}$$

where the shortcutters $S_{C'}$ are in $\mathcal{C}_{jk}$. By Lemma 4.5.4,

$$\left(\sum_{C' \in \mathcal{F}_{jk}} \frac{2\Delta_{\mathcal{F}_{jk}}(C')}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} \right) + \frac{c^{\mathcal{C}}(S_C)}{p} \leq \frac{2\mu_{\text{app}}|\mathcal{F}_{jk}|}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} + \frac{c^{\mathcal{C}}(S_C)}{p}$$

Summing over all $S_C \in \mathcal{C}$ shows that

$$\sum_{S_{C'} \in \mathcal{C}_{jk}} c^{\mathcal{C}_{jk}}(S_{C'}) \leq \left(\sum_{S_C \in \mathcal{C}} \frac{2\mu_{\text{app}}|\mathcal{F}_{jk}|}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} \right) + \left(\sum_{S_C \in \mathcal{C}} \frac{c^{\mathcal{C}}(S_C)}{p} \right)$$

where $\mathcal{F}_{jk}$ is defined for S_C in the above right hand side summand. Furthermore,

$$\begin{aligned} \left(\sum_{S_C \in \mathcal{C}} \frac{2\mu_{\text{app}}|\mathcal{F}_{jk}|}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} \right) + \left(\sum_{S_C \in \mathcal{C}} \frac{c^{\mathcal{C}}(S_C)}{p} \right) &\leq \frac{2\mu_{\text{app}}|\mathcal{C}_{jk}|}{p\gamma\alpha^{i/(\sigma_0+1)} r_{min}} + \frac{\zeta m^{1/\sigma_1}}{p\alpha^{i/(\sigma_0+1)} r_{min}} s_{\mathcal{C}} \\ &\leq \left(16(\log n)\mu_{\text{app}} + (8 \log n)\zeta m^{1/\sigma_1}\right) s_{\mathcal{C}} / (\alpha^{i/(\sigma_0+1)} r_{min}) \end{aligned}$$

by the bucketing of $\mathcal{C}_{jk}$. This is the desired conductivity statement.

Well-spacedness. Consider any cluster C'' that is tied to $S_{C'} \in \mathcal{C}_{jk}$. Suppose that $S_{C'}$ was generated from the shortcutter S_C. By Lemma 4.5.6, $S_{C'} \subseteq S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(1/8, C')$, where $\mathcal{F}_{jk}$ corresponds to S_C. By Proposition 4.5.8 applied in the graph $H \leftarrow H_{\mathcal{C}}$, $C'' \subseteq S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(3/8, C')$. Since each vertex has potential greater than $1/2$ for only one cluster in $\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}$, $S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(3/8, C')$ does not intersect any other $S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(3/8, C''')$ for $C''' \in \mathcal{F}_{jk} \setminus \{C'\}$. Therefore, C'' cannot intersect any other shortcutter in $\mathcal{C}_{jk}$ for a core in $\mathcal{F}_{jk}$. Furthermore, $S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(3/8, C')$ does not intersect $S_{\mathcal{F}_{jk} \cup \{V(H) \setminus S_C\}}(3/8, V(H) \setminus S_C)$. Therefore, C'' cannot contain any vertices outside of S_C. As a result, it cannot intersect any shortcutters of $\mathcal{C}_{jk}$ that are not for cores in $\mathcal{F}_{jk}$. Combining these two statements shows that C'' can only be tied to one shortcutter $S_{C'} \in \mathcal{C}_{jk}$, which is well-spacedness.

Boundedness. Boundedness follows from the following:

$$\begin{aligned}\sum_{S_{C'}\in\mathcal{C}_{jk}} c^H(\partial_H C') &= \sum_{S_C\in\mathcal{C}}\sum_{C'\in\mathcal{F}_{jk}} c^H(\partial_H C') \\ &\leq \left(\sum_{S_C\in\mathcal{C}} c^H(\partial_H C) + \frac{\kappa_0|E_H(C)\cup\partial_H C|}{\alpha^{i/(\sigma_0+1)}r_{min}}\right) \\ &\leq \left(\sum_{S_C\in\mathcal{C}} c^H(\partial_H C)\right) + \frac{\kappa_0 m}{\alpha^{i/(\sigma_0+1)}r_{min}} \\ &\leq \frac{(\kappa+\kappa_0)m}{\alpha^{i/(\sigma_0+1)}r_{min}}\end{aligned}$$

where the first inequality follows from the "Boundedness" guarantee of Lemma 4.5.3, the second inequality follows from the fact that S_Cs in $\mathcal{C}$ are disjoint, and the third inequality follows from the κ-boundedness of $\mathcal{C}$.

Clan growth. Each clan $\mathcal{C}$ is replaced by clans of the form $\mathcal{C}_{jk}$. There are only $T\log m \leq \mu_{\text{app}}\log n$ clans of this form.

Containment. We start by showing that the part Q defined on Line 19 is well-defined. It suffices to show that $\mathcal{P}_i(\mathcal{E})$ is a refinement of $\mathcal{P}_i(\{\mathcal{H}_j\}_{j=1}^{\sigma_0})$ for all $i\in\{1,2,\ldots,\sigma_0\}$. We show this by induction on decreasing i. Inductively assume that $\mathcal{P}_{i+1}(\mathcal{E})$ is a refinement of $\mathcal{P}_{i+1}(\{\mathcal{H}_j\}_{j=1}^{\sigma_0})$ and consider a part $P\in\mathcal{P}_i(\mathcal{E})$.

Let C be a core in some clan of $\mathcal{H}_i$ that intersects P. By the "Covering" guarantee of Lemma 4.5.3, some core C' in some family output during the call to `CoveringCommunity` on C in Line 9 intersects P. By the "Input constraint" guarantee of Lemma 4.5.3, $C'\subseteq C$. By definition of $\mathcal{P}_i(\mathcal{E})$, $P\subseteq C'$. Therefore, P is contained in all cores C in some clan of $\mathcal{H}_i$ that intersect P. Furthermore, P is contained in a unique part of $\mathcal{P}_{i+1}(\mathcal{E})$ by definition. This part is contained in a unique part of $\mathcal{P}_{i+1}(\{\mathcal{H}_j\}_{j=1}^{\sigma_0})$ by the inductive assumption. Since $\mathcal{P}_i(\mathcal{E})$ is the refinement of all cores in $\mathcal{E}_i$ and $\mathcal{P}_{i+1}(\mathcal{E})$ and P is contained in all cores of $\mathcal{H}_i$ and parts of $\mathcal{P}_{i+1}(\mathcal{E})$ that it intersects, P is contained in a part Q in the refinement $\mathcal{P}_i(\{\mathcal{H}_j\}_{j=1}^{\sigma_0})$. This completes the inductive step and shows that Q is well-defined.

Let C_Q be the core assigned to $\mathcal{Q}$ in $\{\mathcal{H}_j\}_{j=1}^{\sigma_0}$. By the "Covering" guarantee of Lemma 4.5.3 applied to the C_Q call, there is a core $C'\subseteq C_Q$ (by the "Input constraint") with $P\subseteq C'$. Therefore, there exists a choice of C_P on Line 20 and $C_P\subseteq C_Q$.

Now, we show that $S_P\subseteq S_Q$, which is the same as showing that $S_{C_P}\subseteq S_{C_Q}$. Since C_P is in some $\mathcal{F}_k$ created on Line 9 for C_Q, the `Voronoi` call that creates S_{C_P} has the set $V(H)\setminus S_{C_Q}$ in its input family. This means that S_{C_P} cannot intersect $V(H)\setminus S_{C_Q}$, which means that $S_{C_P}\subseteq S_{C_Q}$, as desired.

Modifiedness and the deletion set condition. This follows from the fact that $\texttt{deleted}(\mathcal{C}_{jk}) = \texttt{deleted}(\mathcal{C})$ and the "Containment" guarantee for cores.

Runtime. For each core C of a shortcutter in $\mathcal{C}$, the runtime of `CoveringCommunity` is at most $m^{o(1)}|E(C)\cup\partial C|$ by the runtime condition of Lemma 4.5.3. The runtime of the j

and k loops is at most $m^{o(1)}|E(S_C) \cup \partial S_C|$, by Lemma 4.5.6 and the fact that `Voronoi` takes $V(G) \setminus S_C$ as one input cluster. Since $C \subseteq S_C$ and shortcutters in $\mathcal{C}$ are disjoint, the total work for shortcutters in $\mathcal{C}$ is $m^{1+o(1)}$. Since there are $m^{o(1)}$ clans in the input hordes, the total runtime of `RebuildEmpire` is $m^{1+o(1)}$.

□

4.6 Conditioning on the selected parts

Now, we prove Lemma 4.4.17. Let $S = \cup_{P \in \mathcal{K}} P$. Aldous-Broder only needs to cover S in order to sample the intersection of a random tree with $\cup_{P \in \mathcal{K}} E(P)$. Since $\mathcal{E}$ has been carved with respect to S, each active shortcutter $S_C \in \mathcal{H}_i \in \mathcal{E}$ has one of two statuses at any time:

- (Ready) S_C has been covered and therefore can be used or S_C is not active.
- (Boundary) S_C has not been covered and C is within distance $\mu_{\text{carve}} \alpha^{i/(\sigma_0+1)} r_{min}$ of some unvisited vertex in S.

Now, suppose that a vertex v is in a part P_i in $\mathcal{P}_i(\mathcal{E})$. Look at the maximum level i for which P_i's shortcutter has a "Ready" status. Since P_{i+1}'s shortcutter has a "Boundary" status, v is within distance $\mu_{\text{carve}} \alpha^{(i+1)/(\sigma_0+1)} r_{min}$ of some unvisited vertex in S.

Since P_i's shortcutter is "Ready," it can be used without skipping any first visits to vertices in S. We now clarify what "use" means. Ideally, we could just use online shortcutting to shortcut from P_i to ∂S_{P_i}. This could be too expensive for two reasons:

- $E(P_i) = \emptyset$, in which case S_{P_i} is not active and may be larger than the promised maximum active shortcutter size.
- S_{P_i}'s conductance is low in $H_{\mathcal{C}_i}$ where $\mathcal{C}_i$ is the clan containing S_{P_i}, not in H.

In the first case, no shortcutting is necessary, as ∂S_{P_i} is part of a set of boundary edges with small enough total weight (conductance). In the second case, we precompute the probability that a random walk hits an edge in $\texttt{deleted}(\mathcal{C}_i)$ before ∂S_{P_i}. This is our use of offline shortcutting. We show that the precomputation work is small thanks to τ-modifiedness. If the random walk does hit one of these edges first, shortcutting takes $\tilde{O}(1)$ work and the work can be charged to a traversal over an edge in ∂P_i. Otherwise, the shortcut step can be charged to a step over a small conductance set given by ζ-conductivity.

Our random walk analysis relies heavily on the following lemma, which we prove in the appendix. One can think of this lemma as a softer version of the subgraph cover time bound used by [46]. Unlike [46] and [73], we use the following lemma on many different graphs I corresponding to various Schur complements:

Lemma 4.2.3 (Key result for bounding the number of shortcutter uses)**.** *Consider an arbitrary vertex u_0 in a graph I, an edge $\{u, v\} = f \in E(I)$, and an $R \geq 0$. Let $\mathcal{B}(u, R) \subseteq V(I)$ denote the set of vertices in I with I-effective resistance distance at most R from u. The expected number of times that the random walk starting at u_0 traverses f from $u \to v$ before all vertices in $\mathcal{B}(u, R)$ have been visited is at most $\tilde{O}(c_f R)$, where c_f is the conductance of the edge f.*

We prove this in the appendix.

4.6.1 Our shortcutting method

We now introduce our shortcutting primitive. It relies on two facts about electrical flows, the first of which was used by both Kelner-Madry and Madry-Straszak-Tarnawski:

Theorem 4.6.1 ([46, 73]). *Consider two vertices u, v in a graph I. Let $\boldsymbol{p} \in \mathbb{R}^{V(I)}$ denote the potentials for a $u - v$ electrical flow with p_u and p_v normalized to 1 and 0 respectively. Then for any vertex $w \in V(I)$,*

$$\Pr_w[t_u < t_v] = p_w$$

where t_u is the hitting time to u.

Theorem 4.6.2 (Special case of Theorem 4.3.2). *Consider two vertices $u, v \in V(I)$. For each edge $e \in \partial v$, let f_e denote the unit $u - v$ electrical flow on e. Then for each $e \in \partial v$,*

$$\Pr_u[e \text{ traversed to visit } v \text{ for the first time }] = f_e$$

The first result motivates offline shortcutting, while the second motivates online shortcutting. Efficient Laplacian solvers are approximate rather than exact. A trick due to Propp [81] allows us to get around this issue in expected near-linear time.

Our algorithm maintains a data structure D of shortcutting probabilities that is internal to it and an accuracy parameter ϵ_D, which initially is $1/|X_C|$. Let $\mathcal{C}$ be the clan containing S_C. For each vertex $v \in C$ of a shortcutter S_C and the set $X_C = V(\texttt{deleted}(\mathcal{C})) \cap S_C$, D stores an ϵ_D-approximations q_{vx} for all $x \in X_C$ and q_v to $\Pr_v[t_x < t_{(X_C \setminus \{x\}) \cup \partial S_C}]$ for all $x \in X_C$ and $\Pr_v[t_{\partial S_C} < t_{X_C}]$ respectively. It occasionally recomputes D when it needs higher accuracy in accordance with Propp's trick. Also in accordance with Propp's trick, D represents these probabilities as subintervals of $[0, 1]$, with $r_{vx} = q_{vx} + \sum_{y \text{ before } x} q_{vy}$ for an

arbitrary ordering of X_C.

Algorithm 4: `Shortcut`(S_C, v)

```
   // offline part
1  p ← uniformly random number in [0, 1]
2  x_1, x_2, ..., x_k ← arbitrary ordering of X_C
3  while p is within ε_D distance of some r_vx do
4  |  ε_D ← ε_D/2
5  |  recompute D for S_C
6  if p is in a q_vx interval then
7  |  return an arbitrary edge incident with x
8  else
   |  // p is in the q_v interval, so switch to online
9  |  I ← H[S_C ∪ ∂S_C] \ X_C, with ∂S_C identified to a vertex s
10 |  Compute ε-additive approximations to u − s electrical flows for decreasing ε
   |  repeatedly using Propp's trick to sample an escape edge from I
11 |  return the sampled escape edge
```

The runtime analysis of this algorithm takes place over three parts: preprocessing, runtime to hit X_C, and the runtime to hit ∂S_C:

Lemma 4.6.3. *`Shortcut`(S_C, v) takes as input a shortcutter S_C in a clan $\mathcal{C}$ and a vertex $v \in C$. It samples the first vertex $w \in X_C \cup \partial S_C$ that the random walk starting at v in H hits with the correct probability. The edge is correct if that vertex is outside of S_C. `Shortcut` satisfies the following runtime guarantees:*

- *(Preprocessing) The total work to update D over an arbitrary number of uses of `Shortcut` on S_C is at most $\tilde{O}(|X_C||E(S_C) \cup \partial S_C|)$ in expectation.*
- *(Runtime to hit X_C) If `Shortcut`(S_C, v) returns some vertex in X_C, it took $\tilde{O}(1)$ time to do so, excluding time to update D.*
- *(Runtime to hit ∂S_C) If `Shortcut`(S_C, v) returns some vertex in ∂S_C, it took $\tilde{O}(|E(S_C) \cup \partial S_C|)$ time to do so in expectation.*

Proof. **Correctness.** Consider some $x \in X_C$. p is in the q_{vx} interval with probability exactly $\Pr_v[t_x < t_{(X_C \setminus \{x\}) \cup \partial S_C}]$ by Theorem 4.6.1, so any $x \in X_C$ is sampled with the right probability. ∂S_C is sampled with probability exactly $\Pr_v[t_{\partial S_C} < t_{X_C}]$. Moreover, notice that for any non-S_C endpoint w of an edge in ∂S_C,

$$\Pr_v[t_w < t_{(\partial S_C \setminus \{w\}) \cup X_C} \text{ in } H \mid t_{\partial S_C} < t_{X_C} \text{ in } H\ ] = \Pr_v[t_w < t_{\partial S_C \setminus \{w\}} \text{ in } H \setminus X_C]$$

Since $I = H[S_C \cup \partial S_C] \setminus X_C$, Theorem 4.6.2 implies that we are sampling escape edges with the right probabilities for endpoints w of edges in ∂S_C.

Preprocessing. For a particular value of ϵ_D, the probability of recomputation is at most $2|X_C|\epsilon_D$, as this is a bound on the probability that p is within distance ϵ_D of an r_{vx}. If this happens, Theorem 4.6.1 implies that the q_{ux}s for all $u \in C$ and one x can be computed using one Laplacian solve. Doing this for all vertices in X_C takes $\tilde{O}(|X_C||E(S_C) \cup \partial S_C| \log(1/\epsilon_D))$ time. The expected time is at most

$$\begin{aligned}\sum_{\text{values of } \epsilon_D}^{\infty} 2|X_C|\epsilon_D \tilde{O}(|X_C||E(S_C) \cup \partial S_C| \log(1/\epsilon_D)) &= \sum_{i=0}^{\infty} \frac{i}{2^i} \tilde{O}(|X_C||E(S_C) \cup \partial S_C|) \\ &= \tilde{O}(|X_C||E(S_C) \cup \partial S_C|)\end{aligned}$$

to update D, as desired.

Runtime to hit X_C. If $x \in X_C$ is sampled, the else block does not execute. Everything in the if block takes $\tilde{O}(1)$ time besides updating D, as desired.

Runtime to hit ∂S_C. i Laplacian solves on I with error 2^{-i} are done with probability at most 2^{-i} to compute all of the exit probabilities for v out of S_C. Therefore, the expected work is $\tilde{O}(|E(S_C) \cup \partial S_C|)$, as desired. □

4.6.2 Our implementation of the shortcutting meta-algorithm

Now, we implement `PartialSample`. This algorithm is exactly the same as the shortcutting meta-algorithm given at the beginning of Section 4.2:

Algorithm 5: `PartialSample`$(\mathcal{E}, \mathcal{K})$

$S \leftarrow \cup_{P \in \mathcal{K}} P$
$v \leftarrow$ arbitrary vertex in H
$F \leftarrow \emptyset$
while *there is a vertex in* S *that has not been visited* **do**
  $i \leftarrow$ maximum i for which the shortcutter S_{P_i} for the part $P_i \in \mathcal{P}_i(\mathcal{E})$ containing v has status "Ready"
  if $E(P_i) = \emptyset$ **then**
    $\{v, w\} \leftarrow$ random edge incident with v with probability proportional to conductance
    $e \leftarrow \{v, w\}$
  else
    $e \leftarrow$ `Shortcut`(S_{P_i}, v)
  $w \leftarrow$ the $X_{P_i} \cup \partial S_{P_i}$ endpoint of e
  if *both endpoints of* e *are in* S *and* w *not previously visited* **then**
    $F \leftarrow F \cup \{e\}$
  $v \leftarrow w$
return F

The runtime analysis does the following:

- Preprocessing takes a small amount of time because of τ-modifiedness.
- Random walk steps and shortcut steps to X_{P_i} take $\tilde{O}(1)$ time. We can afford to charge these steps to traversals over boundary edges of $\mathcal{P}_i(\mathcal{E})$ thanks to the deletion set condition.
- Shortcut steps to ∂S_{P_i} can be charged to a step over an edge in the Schur complement obtained by eliminating all vertices internal to S_{P_i} besides X_{P_i}. ζ-conductivity can be used to bound the number of these steps.

Picking the maximum level "Ready" shortcutter allows us to charge these steps to covering the $i+1$th horde, since the above "Boundary" shortcutter contains a closeby uncovered vertex.

Correctness relies on the following fact, which we restate from the overview:

Theorem 4.2.1 (Aldous-Broder)**.** *Pick an arbitrary vertex u_0 and run a random walk starting at u_0 in a weighted graph G. Let T be the set of edges used to visit each vertex besides u_0 for the first time. Then T is a weighted uniformly random spanning tree of G.*

Lemma 4.4.17. *Let $\mathcal{K} \subseteq \cup_{i=1}^{\sigma_0} \mathcal{P}_i(\mathcal{E})$ be a set of parts. Let $F = \cup_{P \in \mathcal{K}} E(P)$ and $S = \cup_{P \in \mathcal{K}} P$. Suppose that the empire $\mathcal{E}$ is ζ-conductive, κ-bounded, τ-modified, satisfies the deletion set condition, and has been carved with respect to S. Then, there is an algorithm* `PartialSample`$(\mathcal{E}, \mathcal{K})$ *that returns the intersection of a random spanning tree T in H with F in $\tilde{O}(((\zeta + \kappa)\mu_{carve} + \tau)\ell_{\max} m^{1+1/\sigma_1} \alpha^{1/(\sigma_0+1)})$ time.*

Proof of Lemma 4.4.17. We start with correctness. We need to sample the intersection of a random tree with $Z = \cup_{P \in \mathcal{K}} E(P)$. By Theorem 4.2.1, it suffices to show that `PartialSample` finds all of the edges used to visit S for the first time, since all edges in Z have both of their endpoints in S.

Since S_{P_i} has a status of "Ready," it is either (1) covered or (2) not necessary to use. If (2) is the case, then $E(P_i) = \emptyset$, which means that a true random walk step out of v is performed. If (1) is the case, then using the shortcutter S_{P_i} will not skip any first visits to vertices in S. Furthermore, $X_{P_i} \cap S$ has been visited, so there is no need to keep track of the correct edge used to visit these vertices. By Lemma 4.6.3, `Shortcut` returns the correct visit edge if the walk exits through ∂S_{P_i}. Therefore, `PartialSample` does not miss any first visits to S, which means that it returns the intersection of a random tree with Z.

Now, we bound the runtime of `PartialSample`. We break this analysis up into a number of different types of steps. First, though, we make an observation that is relevant for all types. v is within distance $\alpha^{(i+1)/(\sigma_0+1)}\alpha^{o(1)} r_{min}$ of some unvisited vertex in S because either (1) $S_{P_{i+1}}$ has status "Boundary" or (2) $i = \sigma_0$, in which case the diameter of the graph is at most $\alpha r_{min} = \alpha^{(i+1)/(\sigma_0+1)} r_{min}$.

If-statement steps. These occur when $E(P_i) = \emptyset$, including the case in which i does not exist. In this case, one random walk step is performed incident with v. This takes $O(\log n)$ time to execute and occurs over an edge of $\partial\mathcal{P}_i(\mathcal{E})$, where $\partial\mathcal{P}_i(\mathcal{E}) := \cup_{P' \in \mathcal{P}_i(\mathcal{E})} \partial P'$.

Apply Lemma 4.2.3 to $I \leftarrow H$, all edges in $\partial\mathcal{P}_i(\mathcal{E})$, $S \leftarrow S$, and $R \leftarrow \mu_{\text{carve}} \alpha^{(i+1)/(\sigma_0+1)} r_{min}$. By the κ-boundedness of $\mathcal{E}$, the total number of steps across edges in $\partial\mathcal{P}_i(\mathcal{E})$ within distance R of an unvisited vertex of S is at most

$$\tilde{O}(c^H(\partial\mathcal{P}_i(\mathcal{E}))R) \leq \frac{\kappa_{\max} m}{\alpha^{i/(\sigma_0+1)} r_{min}} \mu_{\text{carve}} \alpha^{(i+1)/(\sigma_0+1)} r_{min} \leq \mu_{\text{carve}} \kappa_{\max} m \alpha^{1/(\sigma_0+1)}$$

per clan. Summing over all clans yields a bound of $\ell_{\max} \mu_{\text{carve}} \kappa_{\max} m \alpha^{1/(\sigma_0+1)}$, as desired.

Else-statement X_{P_i} steps. Notice that since $E(P_i)$ is nonempty, the deletion set condition applies, which ensures that no edge in `deleted`$(\mathcal{C})$ for any $\mathcal{C}$ is on the boundary of $\mathcal{P}_i$. The random walk, in going from P_i to X_{P_i}, must cross an edge of ∂P_i, as the deletion set condition implies that X_{P_i} is disjoint from ∂P_i. As discussed in the If-statement steps bound, only $\ell_{\max} \mu_{\text{carve}} \kappa_{\max} m \alpha^{1/(\sigma_0+1)}$ steps occur. By the X_{P_i} step condition of Lemma 4.6.3, each of these steps takes $\tilde{O}(1)$ time to execute.

Else-statement ∂S_{P_i} steps. Lemma 4.6.3 says that `Shortcut` returns an edge incident with the first vertex in $X_{P_i} \cup \partial S_{P_i}$ that the random walk visits. In particular, by Theorem 4.2.4, a shortcut step directly to ∂S_{P_i} can be charged to random walk steps across a $C_{P_i} - \partial S_{P_i}$

edge in $I \leftarrow \texttt{Schur}(H, X_{P_i} \cup C_{P_i} \cup (V(H) \setminus S_{P_i}))$. Let $F \leftarrow E_I(C_{P_i}, V(H) \setminus S_{P_i})$, $S \leftarrow V(I) \cap S$, and $R \leftarrow \mu_{\text{carve}} \alpha^{(i+1)/(\sigma_0+1)} \alpha^{o(1)} r_{min}$. By Lemma 4.2.3, the total number of steps that occur across $E_I(C_{P_i}, V(H) \setminus S_{P_i})$ that are within distance R of $S \cap V(I)$ is at most

$$c^I(E_I(C_{P_i}, V(H) \setminus S_{P_i}))R$$

in expectation. We start by arguing that each ∂S_{P_i} shortcut step can be charged to one of these random walk steps. Recall that S_{P_i} is only used when $S \cap S_{P_i}$ is covered. All vertices that were eliminated to obtain I were in S_{P_i}, so all shortcut steps occur within distance R of some vertex in $S \cap V(I)$. Each shortcut step can be charged to at least one step across an edge of $E_I(C_{P_i}, V(H) \setminus S_{P_i})$, as discussed earlier. Therefore, the number of shortcut steps is at most $c^I(E_I(C_{P_i}, V(H) \setminus S_{P_i}))R$.

Now, we bound $c^I(E_I(C_{P_i}, V(H) \setminus S_{P_i}))$. Let $I' = \texttt{Schur}(H \setminus \texttt{deleted}(\mathcal{C}_{P_i}), X_{P_i} \cup C_{P_i} \cup (V(H) \setminus S_{P_i}))$, where $\mathcal{C}_{P_i}$ is the clan that contains the shortcutter S_{P_i}. Each edge of $\texttt{deleted}(\mathcal{C}_{P_i})$ with endpoints in S_{P_i} has both of its (identified) endpoints in X_{P_i}, by definition of X_{P_i}. Therefore, deleting these edges does not affect the conductance of the relevant set:

$$c^I(E_I(C_{P_i}, V(H) \setminus S_{P_i})) = c^{I'}(E_{I'}(C_{P_i}, V(H) \setminus S_{P_i}))$$

Eliminating X_{P_i} also can only increase the conductance of this set. Precisely, let $I'' = \texttt{Schur}(H \setminus \texttt{deleted}(\mathcal{C}_{P_i}), C_{P_i} \cup (V(H) \setminus S_{P_i}))$. Then

$$c^{I'}(E_{I'}(C_{P_i}, V(H) \setminus S_{P_i})) \leq c^{I''}(E_{I''}(C_{P_i}, V(H) \setminus S_{P_i}))$$

This quantity has already been defined. Recall that $H_{\mathcal{C}_{P_i}} = H \setminus \texttt{deleted}(\mathcal{C}_{P_i})$. As a result,

$$c^{I''}(E_{I''}(C_{P_i}, V(H) \setminus S_{P_i})) = c^{\mathcal{C}_{P_i}}(S_{P_i})$$

ζ-conductivity can be used to bound this quantity. In particular, summing these bounds over $\mathcal{C}_{P_i}$ shows that the total number of times shortcutters in $\mathcal{C}_{P_i}$ can be used to travel to their true boundaries is at most

$$\begin{aligned}\tilde{O}\left(\sum_{S_C \in \mathcal{C}_{P_i}} c^{\mathcal{C}_{P_i}}(S_{P_i})R\right) &\leq \tilde{O}\left(\frac{\zeta m^{1/\sigma_1} s_{\mathcal{C}_{P_i}}}{\alpha^{i/(\sigma_0+1)} r_{min}} R\right) \\ &\leq \tilde{O}(\mu_{\text{carve}} \zeta_{\max} s_{\mathcal{C}_{P_i}} m^{1/\sigma_1} \alpha^{1/(\sigma_0+1)})\end{aligned}$$

By Lemma 4.6.3, using any shortcutter $S_C \in \mathcal{C}_{P_i}$ takes at most $\max_{S_C \in \mathcal{C}_{P_i}} |E(S_C) \cup \partial S_C|$ time in expectation. By definition of $s_{\mathcal{C}_{P_i}}$, the total work done using shortcutters in $\mathcal{C}_{P_i}$ is at most

$$(\max_{S_C \in \mathcal{C}_{P_i}} |E(S_C) \cup \partial S_C|)\tilde{O}(\mu_{\text{carve}}\zeta_{\max} s_{\mathcal{C}_{P_i}} m^{1/\sigma_1}\alpha^{1/(\sigma_0+1)}) \leq \mu_{\text{carve}}\zeta_{\max} m^{1+1/\sigma_1}\alpha^{1/(\sigma_0+1)}$$

in expectation. Since $\mathcal{E}$ contains $\ell_{\max}$ clans, the total amount of work due to online shortcut steps is at most $\ell_{\max}\mu_{\text{carve}}\zeta_{\max} m^{1+1/\sigma_1}\alpha^{1/(\sigma_0+1)}$.

Else-statement preprocessing work. Each edge in $\texttt{deleted}(\mathcal{C})$ for any clan $\mathcal{C}$ is incident with at most two shortcutters in $\mathcal{C}$ since shortcutters in a clan are disjoint. By Lemma 4.6.3, an expected $O(1)$ Laplacian solves on a cluster with size at most $\max_{S_C \in \mathcal{C}} |E(S_C) \cup \partial S_C|$ happen for each edge in $\texttt{deleted}(\mathcal{C})$. Therefore, the expected total amount of work is at most

$$(\max_{S_C \in \mathcal{C}} |E(S_C) \cup \partial S_C|)|\texttt{deleted}(\mathcal{C})|$$

Since $\mathcal{C}$ is τ-modified,

$$(\max_{S_C \in \mathcal{C}} |E(S_C) \cup \partial S_C|)|\texttt{deleted}(\mathcal{C})| \leq (\max_{S_C \in \mathcal{C}} |E(S_C) \cup \partial S_C|)(\tau_{\max} m^{1/\sigma_1} s_{\mathcal{C}}) \leq \tau_{\max} m^{1+1/\sigma_1}$$

The desired bound follows from the fact that $\mathcal{E}$ only has $\ell_{\max}$ clans.

Completing the proof. All work that `PartialSample` does falls into one of these four categories for some i. Since there are only σ_0 possible values of i, the total runtime is at most $\tilde{O}(((\zeta_{\max} + \kappa_{\max})\mu_{\text{carve}} + \tau_{\max})\ell_{\max} m^{1+1/\sigma_1}\alpha^{1/(\sigma_0+1)})$, as desired.

□

4.7 Choosing vertices to condition on

In this section, we implement `ConditioningVerts`. This amounts to choosing which parts to condition on when there are multiple levels of shortcutters.

Before doing this, we implement a simple version, `SimpleConditioningVerts`, which works when $\sigma_0 = 1$. This method illustrates the utility of conditioning on parts whose shortcutters are largest. When $\sigma_0 > 1$, we can no longer chose parts whose shortcutters in all hordes are largest. Instead, we find parts whose shortcutters are "locally largest" in all hordes; i.e. they are well-separated from cores of larger shortcutters. In this case, the parts chosen for conditioning can be carved out anyways. This choice makes enough progress because one can show that the sizes of shortcutters for parts chosen decrease by a factor of m^{1/σ_1}. After conditioning σ_1 times, one can condition on parts in a higher-level horde. Since there are σ_0 hordes, the number of rounds required is $\sigma_1^{\sigma_0}$.

4.7.1 Warmup: A version of `ConditioningVerts` for $\sigma_0 = 1$ (one shortcutter per vertex)

In this section, we implement `SimpleConditioningVerts`, which satisfies Lemmas 4.4.15 and 4.4.16 when $\sigma_0 = 1$. `SimpleConditioningVerts` is not used in any way to prove Theorem 4.1.1, but is included to motivate some of the ideas behind `ConditioningVerts`. Recall that using no shortcutters (Aldous-Broder) takes $\tilde{O}(m\alpha)$ time. Replacing `ConditioningVerts` with `SimpleConditioningVerts` results in a $\tilde{O}(m^{1+o(1)}\alpha^{1/2+o(1)})$-time algorithm.

The `SimpleConditioningVerts`$(\mathcal{E})$ routine just takes the input empire $\mathcal{E}$, outputs the parts whose shortcutters have size within an m^{1/σ_1}-factor of the maximum, and "carves" the selected parts out of every shortcutter:

Algorithm 6: `SimpleConditioningVerts`$(\mathcal{E})$, never executed

Data: an empire $\mathcal{E}$ consisting of one $\mu_{\text{rad}} r_{min}\sqrt{\alpha}$-horde $\mathcal{H}$
Result: a set of parts $\mathcal{K}$ to condition on
$\mathcal{X} \leftarrow$ set of parts $P \in \mathcal{P}_1(\mathcal{E})$ that have nonempty $E(P)$ (are active)
$\mathcal{K} \leftarrow$ set of parts $P \in \mathcal{X}$ whose shortcutters S_P have size at least $m^{-1/\sigma_1} \max_{Q\in\mathcal{X}} |E(S_Q) \cup \partial S_Q|$
// carving
foreach *active shortcutter* S_C *in some clan of* $\mathcal{E}$ **do**
    $\mathcal{Z} \leftarrow$ parts in $\mathcal{K}$ with distance greater than $\mu_{\text{carve}} r_{min}\sqrt{\alpha}$ from all vertices in C
    Remove all parts in $\mathcal{Z}$ from S_C
return $\mathcal{K}$

We now analyze this algorithm. We need to show that

- conductivity does not increase much (Lemma 4.4.15)
- enough progress is made (Lemma 4.4.16)

We start with Lemma 4.4.15. The key idea is that conditioning on the parts with largest shortcutters ensures that there are not many of them. Specifically, each of the parts P assigned to a shortcutter S_P is contained in S_P's core C_P, which has low effective resistance diameter ($O(\sqrt{\alpha} r_{min})$). Since the shortcutters come from a small number of clans, each of which consists of disjoint shortcutters, the number of distinct shortcutters for chosen parts is at most $m|\mathcal{E}_1|/$(minimum size of shortcutter for a selected part). Since $\mathcal{E}_1$ only consists of $m^{o(1)}$ clans and all shortcutters for selected parts have size within an m^{1/σ_1}-factor of the maximum across all of $\mathcal{E}_1$, the number of distinct shortcutters for chosen parts is at most $m^{1+o(1)} m^{1/\sigma_1} s_{\mathcal{C}}$ for any clan $\mathcal{C} \in \mathcal{H}_1$. In particular, all parts chosen for conditioning are contained in a small number of low-radius clusters (the cores). The following key proposition finishes the proof:

Proposition 4.7.1. *Let H be a graph, $\mathcal{E}$ be an empire in this graph, and $\mathcal{L}$ be a set of clusters with H-effective resistance diameter at most $\psi \alpha^{i/(\sigma_0+1)} r_{min}$ for some $i \in [\sigma_0]$. Consider a well-spaced clan $\mathcal{C} \in \mathcal{E}_i$. Let $S := \cup_{C \in \mathcal{L}} C$ be the set of vertices in clusters of $\mathcal{L}$.*

Obtain a new clan $\mathcal{C}'$ by deleting all vertices $v \in S$ from shortcutters $S_C \in \mathcal{C}$ for which v is not within H-effective resistance distance $\gamma_{del} \psi \mu_{app} \alpha^{i/(\sigma_0+1)} r_{min}$ of any vertex in C, where $\gamma_{del} = 1000$.

Then

$$\sum_{S_C \in \mathcal{C}'} c^{\mathcal{C}'}(S_C) \leq \frac{\mu_{app}(|\mathcal{L}| + |\texttt{deleted}(\mathcal{C})|)}{\alpha^{i/(\sigma_0+1)} r_{min}} + \sum_{S_C \in \mathcal{C}} c^{\mathcal{C}}(S_C)$$

The proof of this proposition relies on the following lemmas, both of which are proven in the appendix:

Lemma 4.7.2. *Consider a graph H and a set of clusters $\mathcal{D}$, each with effective resistance diameter at most R. Let F be a set of edges in H. Then there is a set of clusters $\mathcal{D}'$ with the following properties:*

- *(Covering) Each vertex in a cluster of $\mathcal{D}$ is in a cluster of $\mathcal{D}'$.*
- *(Diameter) The effective resistance diameter of each cluster in $\mathcal{D}'$ is at most $\mu_{app} R$ in the graph $H \setminus F$.*
- *(Number of clusters) $|\mathcal{D}'| \leq \mu_{app}(|\mathcal{D}| + |F|)$.*

Lemma 4.7.3. *Consider a graph H and two clusters C and S_C, with $C \subseteq S_C$. Let C' be disjoint from C. Additionally, suppose that*

- *The effective resistance diameters of C and C' in H are both at most R.*
- *The effective resistance distance between any pair of points in C and C' in H is at least $\beta_1 R$.*

- $c^H(S_C) \leq \frac{\tau}{R}$.

Then $c^H(S_C \setminus C') \leq \frac{\tau + 1/(\beta_1 - 4)}{R}$.

Proof of Proposition 4.7.1. By Lemma 4.7.2 applied to the clusters $\mathcal{L}$ with deleted edge set $\texttt{deleted}(\mathcal{C})$, S is the union of a set of clusters $\mathcal{L}'$ with $H_{\mathcal{C}}$-effective resistance diameter $\mu_{\text{app}}\psi\alpha^{i/(\sigma_0+1)}r_{min}$. Furthermore, $|\mathcal{L}'| \leq \mu_{\text{app}}(|\mathcal{L}| + |\texttt{deleted}(\mathcal{C})|)$.

Now, consider a cluster $C \in \mathcal{L}'$. To delete all vertices in S from shortcutters with far-away cores, it suffices to delete each of the clusters C from any shortcutter $S_{C'}$ for which the minimum $H_{\mathcal{C}}$ distance between vertices in C' and C is at least $(\gamma_{\text{del}} - 2)\mu_{\text{app}}\psi\alpha^{i/(\sigma_0+1)}r_{min} \geq \beta_0\mu_{\text{app}}\psi\alpha^{i/(\sigma_0+1)}r_{min}$ by the triangle inequality. In this case, C is tied to $S_{C'}$. Since $\mathcal{C}$ is a well-spaced clan, C cannot be tied to any other shortcutter in $\mathcal{C}$. Therefore, deleting C from all shortcutters for whom the cores are at least $H_{\mathcal{C}}$-distance $(\gamma_{\text{del}} - 2)m^{o(1)}\psi\alpha^{i/(\sigma_0+1)}r_{min}$ only modifies one shortcutter $S_{C'}$. Furthermore, the conductance of this shortcutter only increases additively by $1/((\gamma_{\text{del}} - 2)\mu_{\text{app}}\psi\alpha^{i/(\sigma_0+1)}r_{min}) \geq 1/\alpha^{i/(\sigma_0+1)}r_{min}$ by Lemma 4.7.3.

Removing vertices from shortcutters in $\mathcal{C}$ does not destroy its well-spacedness. Therefore, we can apply this reasoning for each of the clusters in $\mathcal{L}'$; incurring a $1/\alpha^{i/(\sigma_0+1)}r_{min}$ conductance increase per cluster deletion. After doing this, we obtain a clan $\mathcal{C}'$ for which

$$\begin{aligned}\sum_{S_C \in \mathcal{C}'} c^{\mathcal{C}'}(S_C) &\leq \frac{|\mathcal{L}'|}{\alpha^{i/(\sigma_0+1)}r_{min}} + \sum_{S_C \in \mathcal{C}} c^{\mathcal{C}}(S_C) \\ &\leq \frac{\mu_{\text{app}}(|\mathcal{L}| + |\texttt{deleted}(\mathcal{C})|)}{\alpha^{i/(\sigma_0+1)}r_{min}} + \sum_{S_C \in \mathcal{C}} c^{\mathcal{C}}(S_C)\end{aligned}$$

as desired. □

Now, we use Proposition 4.7.1 to prove Lemma 4.4.15 for $\sigma_0 = 1$:

Proof of Lemma 4.4.15, with `ConditioningVerts` *replaced by* `SimpleConditioningVerts`. Let $R = \mu_{\text{carve}}\sqrt{\alpha}r_{min}$.

Number of cores containing parts in $\mathcal{K}$. Recall that each part $P \in \mathcal{K}$ is assigned to a shortcutter S_P in the horde $\mathcal{E}_1$ of $\mathcal{E}$. Let C_P denote this core of S_P and let $\mathcal{K}' = \{C_P : \forall P \in \mathcal{K}\}$. We now bound $|\mathcal{K}'|$. Start by bounding the size of the intersection of $\mathcal{K}'$ with the coreset of an arbitrary clan $\mathcal{C}$. $\mathcal{C}$ consists of disjoint shortcutters, so they must have total size at most m. As a result, $\mathcal{K}'$ has size at most

$$\begin{aligned}|\mathcal{K}'| &\le |\mathcal{E}_1| \frac{m}{\min_{C\in\mathcal{K}'} |E(S_C)\cup\partial S_C|} \\ &\le \ell_{\max} \frac{m^{1+1/\sigma_1}}{\max_{C\in\mathcal{K}'} |E(S_C)\cup\partial S_C|} \\ &= \ell_{\max} \frac{m^{1+1/\sigma_1}}{\max_{Q\in\mathcal{X}} |E(S_Q)\cup\partial S_Q|} \\ &\le \ell_{\max} m^{1/\sigma_1} s_{\mathcal{C}}\end{aligned}$$

for any clan $\mathcal{C} \in \mathcal{E}_1$. The first inequality follows from Line 2 of `SimpleConditioningVerts`. This is the desired bound.

Conductivity. Apply Proposition 4.7.1 with $i \leftarrow 1$ and $\mathcal{L} \leftarrow \mathcal{K}'$ on each of the clans in $\mathcal{E}_1$. `SimpleConditioningVerts` does strictly fewer vertex removals than the procedure described in Proposition 4.7.1. As a result, Proposition 4.7.1 implies that the conductance of the shortcutters in a clan $\mathcal{C}$ additively increases by at most $\frac{\mu_{\text{app}}(|\mathcal{K}'|+|\texttt{deleted}(\mathcal{C})|)}{\sqrt{\alpha} r_{min}}$. By the "Number of cores containing parts in $\mathcal{K}$" and τ-modifiedness of $\mathcal{C}$, the conductance increase is at most $\frac{\mu_{\text{app}}(\ell_{\max} m^{1/\sigma_1} + \tau m^{1/\sigma_1}) s_{\mathcal{C}}}{\sqrt{\alpha} r_{min}}$. Therefore, the conductivity of $\mathcal{C}$ in $\mathcal{E}'$ (at the end of `SimpleConditioningVerts`) is at most ζ_1 higher than it was in $\mathcal{E}$, as desired.

Carving. Line 5 of `SimpleConditioningVerts` ensures that the shortcutter S_C has been carved with respect to $\mathcal{K}$. Therefore, all active shortcutters in $\mathcal{E}'$ are carved with respect to $\mathcal{K}$, which means that $\mathcal{E}'$ is carved with respect to $\mathcal{K}$. □

Now, we show that `SimpleConditioningVerts` conditions on a large enough set to make substantial progress. Progress is measured by the maximum size of an active shortcutter:

Proof of Lemma 4.4.16, with `ConditioningVerts` replaced by `SimpleConditioningVerts`. We show that the maximum size of an active shortcutter decreases by a factor of m^{1/σ_1} in between applications of `SimpleConditioningVerts`. If we do this, then no shortcutter is active after σ_1 iterations. Since each part is assigned to a shortcutter, each part P must have $E(P) = \emptyset$ after σ_1 iterations.

Now, consider any part $P \in \mathcal{P}_1(\mathcal{E})$ with an active shortcutter. By the "Containment" input condition, P is contained in a unique $Q \in \mathcal{P}_1(\mathcal{E}_{prev})$. S_P being active implies that S_Q was active. By the "Progress" condition, $Q \notin \mathcal{K}_{prev}$. Therefore, by the conditioning choice that chose $\mathcal{K}_{prev}$,

$$|E(S_Q)\cup\partial S_Q| \le m^{-1/\sigma_1} \max_{\text{previously active parts X}} |E(S_X)\cup\partial S_X|$$

By the "Containment" condition, S_P is smaller than S_Q, so

$$\max_{\text{currently active parts P}} |E(S_P)\cup\partial S_P| \le m^{-1/\sigma_1} \max_{\text{previously active parts X}} |E(S_X)\cup\partial S_X|$$

In particular, the maximum active shortcutter size decreased by a factor of m^{-1/σ_1}, as desired. □

Now, consider what happens when all shortcutters are inactive. By the above proof, this happens after σ_1 conditioning rounds. Since $\mathcal{E}$ is $m^{o(1)}$-bounded at this point and all parts are empty (by inactivity), the total conductance of all edges left over in the graph is at most $m^{1+o(1)}/(\sqrt{\alpha}r_{min})$. Running Aldous-Broder therefore takes $O((m^{1+o(1)}/(\sqrt{\alpha}r_{min}))\alpha r_{min}) = O(m^{1+o(1)}\sqrt{\alpha})$ time. Therefore, the entire algorithm for sampling a random spanning tree from the original graph G takes $O(m^{1+o(1)}\sqrt{\alpha})$ time, as desired.

4.7.2 Generalizing to $\sigma_0 > 1$

We start by reviewing how the main algorithm `ExactTree` with $\sigma_0 = 1$ uses `SimpleConditioningVerts`. `ExactTree` starts by making arbitrary shortcutters using `RebuildEmpire`. Afterwards, it calls `SimpleConditioningVerts` followed by the routines `PartialSample`, conditioning, `FixShortcutters`, and `RebuildEmpire` again.
`SimpleConditioningVerts` chooses all of the parts with near-largest shortcutters for conditioning. Applying the routines `PartialSample`, conditioning, `FixShortcutters`, and `RebuildEmpire` makes the induced subgraphs of the parts with the largest shortcutters empty (inactive), as described by the "Progress" condition. By the "Containment" condition, all remaining shortcutters are smaller than they were previously, so the empire supplied to the second call of `SimpleConditioningVerts` has no shortcutters with size greater than m^{1-1/σ_1}. Specifically, the quantity

$$s_1 := \lfloor \log_{m^{1/\sigma_1}}(\max_{\text{active parts } P \in \mathcal{P}_1(\mathcal{E})} |E(S_P) \cup \partial S_P|) \rfloor$$

strictly decreases during each iteration of the while loop in the `ExactTree` algorithm. $s_1 \leq \sigma_1$ initially, so $s_1 = 0$ after at most σ_1 while loop iterations. At this point, $\mathcal{E}$ has no active parts. As a result, the graph at this point just consists of boundary edges for parts in $\mathcal{P}_1(\mathcal{E})$, at which point just running Aldous-Broder without shortcutting is efficient. Specifically, conditioning on all parts in $\mathcal{E}_1$ paved the way for `ExactTree` to be able to efficiently condition on the entire graph.

Now, we generalize this reasoning to the case in which $\sigma_0 > 1$. Specifically, we design a scheme that is built around the idea of conditioning on parts in $\mathcal{E}_i$ in order to make conditioning on $\mathcal{E}_{i+1}$ efficient:

Key Idea 4.7.4. *`ConditioningVerts` maintains state across multiple calls. Specifically, for all $i \in [\sigma_0]$, it maintains a choice of parts $\mathcal{Q}_i \subseteq \mathcal{P}_i(\mathcal{E})$ that it would like to be able to condition on and a set of parts $\mathcal{R}_i \subseteq P_i(\mathcal{E})$ with $\mathcal{Q}_i \subseteq \mathcal{R}_i$ that are "relevant" to being able to condition on $\mathcal{Q}_{i+1}$. Specifically, if all of the parts in $\mathcal{R}_i$ are inactive, then `ConditioningVerts` can condition on $\mathcal{Q}_{i+1}$. Conditioning on at most σ_1 different choices of $\mathcal{Q}_i$ will make all of the parts in $\mathcal{R}_i$ irrelevant.*

We encourage the reader to delay trying to internalize how exactly the $\mathcal{R}_i$s are constructed. We discuss this in detail in Section 4.7.7. $\mathcal{Q}_i$ can almost be thought of as the set of parts in $\mathcal{R}_i$ with near-maximum shortcutters. We discuss the reasons for saying "almost" in Section 4.7.7. In particular, the $\mathcal{Q}_i$ parts have "locally maximum" shortcutters. We discuss this in more detail in Section 4.7.3.

To get a feel for what one could expect $\mathcal{R}_i$ and $\mathcal{Q}_i$ to be, it helps to think about the $\sigma_0 = 1$ case. In this case, $\mathcal{R}_1 = \mathcal{P}_1(\mathcal{E})$ during each call of `SimpleConditioningVerts`. $\mathcal{Q}_1$ is the set of active parts in $\mathcal{R}_1$ with near-maximum shortcutters. After σ_1 iterations of the while loop of `ExactTree`, all parts in $\mathcal{R}_1$ are inactive, which allows us to condition on $\mathcal{Q}_2$, which is defined to be just one part containing the entire graph. After doing this, all parts in $\mathcal{R}_2$ — which is also defined to be just one part with the entire graph — are inactive. In particular, the graph contains no more edges and we have sampled a complete random spanning tree.

Making progress and intuition for the proof of Lemma 4.4.16

Now that we have some idea for what the $\mathcal{R}_i$s and $\mathcal{Q}_i$s could be, we can talk about our notion of progress. We start by generalizing s_1 to a quantity s_i. Roughly speaking, m^{s_i/σ_1} is the maximum size of a relevant shortcutter in a clan of $\mathcal{E}_i$:

$$s_i := \lfloor \log_{m^{1/\sigma_1}}(\max_{\text{active parts } P \in \mathcal{R}_i} |E(S_P) \cup \partial S_P|) \rfloor$$

This is not how we actually define s_i, but it is a good way to think about it. In particular, the function `DSize` is similar to $\max_{\text{active parts } P \in \mathcal{R}_i} |E(S_P) \cup \partial S_P|$ and we encourage a confused reader to mentally replace `DSize` with $\max_{\text{active parts } P \in \mathcal{R}_i} |E(S_P) \cup \partial S_P|$ with the exception of one place, which we point out in Section 4.7.4. Now, we discuss progress:

Key Idea 4.7.5. *Our notion of progress is that each iteration of the while loop in* `ExactTree` *lexicographically decreases the word* $s_{\sigma_0+1} s_{\sigma_0} s_{\sigma_0-1} s_{\sigma_0-2} \dots s_2 s_1$.

Now, we understand how `ConditioningVerts` could be implemented to guarantee such a lexicographic decrease. Each call to `ConditioningVerts` returns $\mathcal{Q}_k$ for some $k \in [\sigma_0]$ with $s_k > 0$. By the "Progress" input condition in Definition 4.4.14, all parts in $\mathcal{Q}_k$ become inactive before the next call to `ConditioningVerts`. Roughly speaking, as described earlier, $\mathcal{Q}_k$ contains all of the parts with largest shortcutters in $\mathcal{R}_k$. As long as $\mathcal{R}_k$ shrinks (which we show that it does in Section 4.7.7), s_k strictly decreases. All s_i for $i > k$ do not increase by the "Containment" input condition. Therefore, the word $s_{\sigma_0+1} s_{\sigma_0} s_{\sigma_0-1} \dots s_2 s_1$ lexicographically decreased between conditioning rounds.

We now briefly discuss why s_is with $i < k$ could increase. To make progress again, $\mathcal{Q}_k$ needs to be changed. Recall in Key Idea 4.7.4 that the only purpose of $\mathcal{Q}_i$ for $i < k$ is to make it possible to condition on parts in $\mathcal{Q}_k$. Therefore, once $\mathcal{Q}_k$ changes, the $\mathcal{Q}_i$s and $\mathcal{R}_i$s for $i < k$ become useless and need to be chosen anew. In particular, s_i for $i < k$ could increase.

Eventually, $s_{\sigma_0+1}s_{\sigma_0}s_{\sigma_0-1}\dots s_2s_1 = 00\dots00$. At this point, because $\mathcal{R}_{\sigma_0+1}$ consists of one part which is the entire graph, the entire graph is empty and consists of just one vertex. In particular, we have completely sampled a random spanning tree. Since there are only $\sigma_1^{\sigma_0+1}$ possible words, this happens after $\sigma_1^{\sigma_0+1}$ conditioning rounds, suggesting a proof of Lemma 4.4.16.

Conditioning on $\mathcal{Q}_k$

In the previous subsubsection, we needed to find some $\mathcal{Q}_k$ with $s_k > 0$ that could actually be conditioned on. Specifically, we need to find some $\mathcal{Q}_k$ that can be carved out of *all* shortcutters in *all* hordes of $\mathcal{E}$ with increasing the conductivity of any clan too much.

At the beginning of each call to `ConditioningVerts`, the $\mathcal{Q}_{k'}$ selected for conditioning from the previous iteration consists of inactive parts by the "Progress" input condition. Let i^* be the maximum i for which $\mathcal{Q}_{i^*} = \emptyset$. This exists because k' is a valid choice for i^*. If $s_{i^*} > 0$, then $\mathcal{R}_{i^*}$ still contains active parts, which means that we shoud reselect $\mathcal{Q}_{i^*}$ in order to continue rendering parts in $\mathcal{R}_{i^*}$ inactive. This is done using the routine `ExtendHierarchy`. After reselecting $\mathcal{Q}_{i^*}$, `ConditioningVerts` reselects all $\mathcal{Q}_i$s for $i < i^*$ using `ExtendHierarchy`. Then, `ConditioningVerts` returns the parts $\mathcal{Q}_{k^*+1}$, where k^* is the maximum value for which $s_{k^*} = 0$.

While the previous subsubsection illustrated that this algorithm (`ConditioningVerts`) makes enough progress (satisfies Lemma 4.4.16), we still need to demonstrate that $\mathcal{E}$ can be carved with respect to $\mathcal{Q}_{k^*+1}$ without increasing its conductivity too much (by more than $m^{o(1)}$ additively). To do this, it suffices to design the $\mathcal{R}_i$s in a way that respects the following property:

Key Idea 4.7.6. *For each part $P \in \mathcal{P}_i(\mathcal{E})\backslash\mathcal{R}_i$, S_P does not intersect any part $P' \in \cup_{j\le i+1}\mathcal{Q}_j$. In particular, S_P is carved with respect to $\cup_{j\le i+1}\mathcal{Q}_j$.*

Now, we see how this idea enables `ConditioningVerts` to choose $\mathcal{Q}_{k^*+1}$. Each part $P \in \mathcal{P}_{k^*}(\mathcal{E})$ is either in $\mathcal{R}_{k^*}$ or not in $\mathcal{R}_{k^*}$. If $P \in \mathcal{R}_{k^*}$, then P is inactive because $s_{k^*} = 0$. Therefore, its shortcutter S_P does not need to be carved with respect to $\mathcal{Q}_{k^*+1}$ because S_P does not need to be used. If $P \notin \mathcal{R}_{k^*}$, then S_P is carved with respect to $\mathcal{Q}_{k^*+1}$ by Key Idea 4.7.6. For parts $Q \in \mathcal{P}_j(\mathcal{E})$ for $j \ge k^* + 1$, either $Q \notin \mathcal{R}_j(\mathcal{E})$ in which case Key Idea 4.7.6 implies that S_Q is carved, or $Q \in \mathcal{R}_j(\mathcal{E})$ and $\mathcal{Q}_j$ can be carved out of S_Q because $\mathcal{Q}_j$'s shortcutters are bigger than S_Q. In particular, applying Proposition 4.7.1 here implies that carving does not increase the conductivity of S_Q's clan too much (see the `MakeNonedgesPermanent` routine). Therefore, all shortcutters are carved with respect to $\mathcal{Q}_{k^*+1}$. For more details on this argument, see Proposition 4.7.16.

Choosing conditioning hierarchies that respect the key ideas

While we have discussed most of the key properties of conditioning hierarchies that allow `ConditioningVerts` to establish Lemmas 4.4.15 and 4.4.16 (as summarized in the Key

Ideas), it has not been made clear how to actually obtain any of them. We now discuss some insufficient approaches for defining the $\mathcal{Q}_i$s and $\mathcal{R}_i$s. Specifically, we attempt to design `ExtendHierarchy`, which is called on level i^* and below if $s_{i^*} > 0$ in order to get closer to making all of $\mathcal{R}_{i^*}$ inactive. For simplicity, we focus our discussion on the first call to `ConditioningVerts`, in which case $\mathcal{Q}_i = \mathcal{R}_i = \emptyset$ for all $i \leq \sigma_0$ and $\mathcal{Q}_{\sigma_0+1}$ and $\mathcal{R}_{\sigma_0+1}$ consist of just one part that contains the entire graph.

One natural attempt at designing `ExtendHierarchy` is to do the following:

- For each $i \leftarrow \sigma_0, \sigma_0 - 1, \ldots, 1$,
  - Let $\mathcal{R}_i$ be the set of parts with parents in $\mathcal{Q}_{i+1}$.
  - Let $\mathcal{Q}_i$ be the parts $P \in \mathcal{R}_i$ with near-maximum-size shortcutters.

This approach is a simple generalization of the algorithm `SimpleConditioningVerts`. Unfortunately, it does not respect Key Idea 4.7.6 because parts $P \in \mathcal{P}_i(\mathcal{E}) \setminus \mathcal{R}_i$ could have shortcutters S_P that intersect some part in $\mathcal{Q}_{i+1}$. More concretely, this approach does not work because there may be a part $P \in \mathcal{P}_{\sigma_0-2}(\mathcal{E}) \setminus R_{\sigma_0-2}$ for which (a) S_P intersects $\mathcal{Q}_{i+1}$ and (b) S_P is very large compared to the shortcutters of the parts in $\mathcal{Q}_i$. In this case, carving would increase the conductance of S_P too much (see Figure 4.7.1).

We deal with the above issue by allowing $\mathcal{Q}_i$ to not necessarily be contained in $\mathcal{Q}_{i+1}$ and picking $\mathcal{Q}_i$ before $\mathcal{R}_i$ rather than the other way around. In particular, if there is a $P \in \mathcal{P}_i(\mathcal{E})$ with a parent in $\mathcal{R}_{i+1}$ for which (a) S_P intersects $\mathcal{Q}_{i+1}$ and (b) S_P is very large compared to the shortcutters for the default choice of $\mathcal{Q}_i$ described in the first attempt, switch $\mathcal{Q}_i$ to be all parts satisfying (a) that have near-maximum shortcutter size (similar to (b)). Of course, the new choice for $\mathcal{Q}_i$ may be bad for the exact same reasons as the original choice. Luckily, though, this switching procedure can only happen σ_1 times because each switch of $\mathcal{Q}_i$ increases the sizes of the shortcutters considered by an m^{1/σ_1} factor.

The above approach also has the unfortunate property that chosen parts at level j for $j < i$ could make choices at higher levels unusable. To deal with this problem, we would like all parts at lower levels to look as if they are part of $\mathcal{Q}_i$. One way of doing this is to require chosen parts at level $\mathcal{Q}_j$ to be close to $\mathcal{Q}_i$ (see the "Vertical closeness" condition in the definition of conditioning hierarchies). This suggests modifying the algorithm mentioned in the previous paragraph to discuss the effective resistance metric:

- For each $i = \sigma_0, \sigma_0 - 1, \ldots, 1$,
  - $\mathcal{Q}_i \leftarrow$ the set of parts of $\mathcal{P}_i(\mathcal{E})$ with parents in $\mathcal{Q}_{i+1}$
  - While there is a part $P \in \mathcal{P}_i(\mathcal{E})$ with parent in $\mathcal{R}_i$ for which (a) S_P intersects $\mathcal{Q}_{i+1}$ (b) S_P is much larger than shortcutters for parts in $\mathcal{Q}_i$ and (c) P is not too much farther away (say no more than 7 times farther away) from $\mathcal{Q}_{i+1}$ than some part in the current value of $\mathcal{Q}_i$

    - Replace $\mathcal{Q}_i$ with all parts P with parent in $\mathcal{R}_i$ that satisfy (a) and (c) with near-maximum size.
  - $\mathcal{R}_i \leftarrow$ all parts with parents in $\mathcal{R}_i$ that satisfy (a) and (c) for $\mathcal{Q}_i$

The irrelevant parts are far away, so $\mathcal{Q}_{i+1}$ can be carved out of their shortcutters. While this algorithm works for call to `ConditioningVerts`, it happens to violate Key Idea 4.7.5. Luckily, we can fix this issue by observing that the above strategy is part of a more general family of strategies based on a digraph `ConditioningDigraph` at each level i. This digraph is a digraph on parts in $\mathcal{P}_i(\mathcal{E})$ with parents in $\mathcal{R}_{i+1}$. An edge is present in this digraph from $P \to Q$ if and only if (1) P intersects S_Q and (2) Q is not much farther from $\mathcal{Q}_{i+1}$ than P. (1) and (2) are similar to (a) and (c) respectively.

As a result, the second attempt given above can be viewed as doing a BFS in this digraph and making a histogram of sizes of shortcutters for parts at these distances. The above strategy is equivalent to letting $\mathcal{Q}_i$ be the set of parts corresponding to the closest local maximum to the source $\mathcal{Q}_{i+1}$ in this histogram. See Figure 4.7.2 for a visual on the construction of this histogram. This visual suggests a large family of strategies based on finding local maxima in histograms. In `ExtendHierarchy`, we give a simple histogram strategy which picks local maxima each time. This allows us to satisfy Key Idea 4.7.6. However, this strategy also has the property that $\mathcal{R}_i$ shrinks across multiple conditioning rounds. This ensures that Key Idea 4.7.5 is also respected.

There are other complications that we have not adequately addressed in this summary. For example, it is a priori unclear if conditioning digraphs from one call to `ConditioningVerts` relate in any way to conditioning digraphs in previous calls. Luckily, later digraphs are subgraphs of earlier digraphs, thanks to the "Containment" input condition, after modification by the routine `MakeNonedgesPermanent` that does not increase conductivity much.

4.7.3 Conditioning hierarchies: the data structure for representing parts to condition on

In this section, we formally define the data structure used to keep track of sets to condition on: the *conditioning hierarchy*. We again encourage the reader to delay trying to understand exactly how the $\mathcal{Q}_i$s and $\mathcal{R}_i$s are constructed until Section 4.7.7, where `ExtendHierarchy` is introduced.

Definition 4.7.7 (Conditioning hierarchies)**.** *Consider the empire $\mathcal{E}$, a family of sets of* chosen parts $\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}$*, and a family of sets of* relevant parts $\{\mathcal{R}_i\}_{i=1}^{\sigma_0}$*. The pair $\mathcal{CH} = (\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ is called a* conditioning hierarchy *if the following properties are true for all $i \in [\sigma_0]$:*

- *(Horizontal containment) $\mathcal{Q}_i \subseteq \mathcal{R}_i \subseteq \mathcal{P}_i(\mathcal{E})$*
- *(Vertical containment) For every part $P \in \mathcal{R}_i$, there is some $Q \in \mathcal{R}_{i+1}$ for which $P \subseteq Q$.*

- *(Vertical closeness) Each vertex in a part of $\mathcal{Q}_i$ is within distance $(\mu_{carve}/(100\gamma_{temp}\gamma_{ann}^2))\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ of some vertex in a part of $\mathcal{Q}_{i+1}$, where $\gamma_{temp} := 4\gamma_{del}$.*

The "Vertical closeness" property is in some sense a weaker version of the "Vertical containment" property that applies to the chosen parts. While the chosen parts from different hordes are not truly nested, they are close to chosen parts from higher hordes. Recall from the end of the previous section that we do not want the $\mathcal{Q}_i$s to be truly nested, as there may be a much larger shortcutter that intersects $\mathcal{Q}_i$ but whose core is outside of $\mathcal{Q}_i$.

As the conditioning algorithm proceeds, the "Containment" input condition allows us to argue that conditioning hierarchies are in some sense contained within one another:

Definition 4.7.8 (Fitting of conditioning hierarchies)**.** *Consider two conditioning hierarchies $\mathcal{CH} = (\{\mathcal{Q}_k\}_{k=1}^{\sigma_0}, \{\mathcal{R}_k\}_{k=1}^{\sigma_0})$ and $\mathcal{CH}' = (\{\mathcal{Q}'_k\}_{k=1}^{\sigma_0}, \{\mathcal{R}'_k\}_{k=1}^{\sigma_0})$ for two (possibly different) empires $\mathcal{E}$ and $\mathcal{E}'$.*

$\mathcal{CH}$ i-fits within $\mathcal{CH}'$ if both of the following conditions hold for all $j \geq i$:

- *($\mathcal{Q}$-fitting) For all $P \in \mathcal{Q}_j$, there is a $Q \in \mathcal{Q}'_j$ for which $P \subseteq Q$.*
- *($\mathcal{R}$-fitting) For all $P \in \mathcal{R}_j$, there is a $Q \in \mathcal{R}'_j$ for which $P \subseteq Q$.*

$\mathcal{CH}$ completely fits *in $\mathcal{CH}'$ if $\mathcal{CH}$ 1-fits in $\mathcal{CH}'$.*

Next, we define a property of chosen parts that will allow us to carve without substantially increasing conductivity when combined with the "Vertical closeness" property of conditioning hierarchies:

Definition 4.7.9 (Descendant size)**.** *Given an empire $\mathcal{E}$ and a part $P \in \mathcal{P}_i(\mathcal{E})$, the* descendant size *of P, denoted $\texttt{DSize}(\mathcal{E}, P)$, is the maximum number of edges incident with any shortcutter for a descendant part:*

$$\texttt{DSize}(\mathcal{E}, P) := \max_{Q \in \cup_{j \leq i} \mathcal{P}_j(\mathcal{E}) : Q \subseteq P} |E(S_Q) \cup \partial S_Q|$$

When understanding these concepts for the first time, it is helpful to think of defining $\texttt{DSize}(\mathcal{E}, P)$ as $|E(S_P) \cup \partial S_P|$ instead of the maximum over all descendant parts. DSize is used in place of $|E(S_P) \cup \partial S_P|$ for reasons discussed in Section 4.7.4.

Definition 4.7.10 (Locally maximum conditioning hierarchies)**.** *Consider a conditioning hierarchy $\mathcal{CH} = (\{\mathcal{Q}_k\}_{k=1}^{\sigma_0}, \{\mathcal{R}_k\}_{k=1}^{\sigma_0})$ for an empire $\mathcal{E}$. $\mathcal{CH}$ is said to be* locally maximum *if for all $i \in [\sigma_0]$ with $\mathcal{Q}_i \neq \emptyset$,*

$$\min_{P \in \mathcal{Q}_j} \texttt{DSize}(\mathcal{E}, P) \geq m^{-1/\sigma_1} \max_{Q \in \mathcal{R}_j} \texttt{DSize}(\mathcal{E}, Q)$$

`ConditioningVerts` keeps a conditioning hierarchy as continued state, along with a size and distance parameter associated with each $i \in [\sigma_0]$. The size parameter tracks the largest size of a shortcutter for a part in $\mathcal{R}_i$. We discuss the distance parameter in Section 4.7.4. The conditioning hierarchy is initialized with $\mathcal{Q}_i = \emptyset$ for all $i \in [\sigma_0]$ and $\mathcal{R}_i = \emptyset$ except for $i = \sigma_0 + 1$, for which $\mathcal{Q}_{\sigma_0+1} = \{V(G)\}$ and $\mathcal{R}_{\sigma_0+1} = \{V(G)\}$. On a particular call to `ConditioningVerts`, it

- Picks the highest i for which both (1) $\mathcal{Q}_i = \emptyset$ and (2) all parts in $\mathcal{R}_i$ have inactive shortcutters and calls this value i^*
- If a part in $\mathcal{R}_{i^*}$ has an active shortcutter, `ConditioningVerts` uses a ball-growing strategy (see `ExtendHierarchy`) to reselect the chosen and relevant parts for levels i^* and below. After doing this, $\mathcal{Q}_1$ is selected for conditioning.
- Otherwise, $\mathcal{Q}_{i^*+1}$ can be conditioned on (see Proposition 4.7.19)

This process results in a conditioning hierarchy that i^*+1-fits in the previous conditioning hierarchy. As a result, all s_j for $j > i^*$ do not increase.

4.7.4 Conditioning digraphs: the metric used for choosing parts to condition on

At the end of Section 4.7.2, we discussed the fact that picking parts to condition on based on a plot of shortcutter `DSize`s versus "distance" would be a good strategy. We now define `ConditioningDigraph`, from which that distance is defined.

The digraph $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$ is used to obtain $\mathcal{Q}_i$ and $\mathcal{R}_i$ from $\mathcal{Q}_{i+1}$ from $\mathcal{R}_{i+1}$.

Definition 4.7.11 (Conditioning Digraph)**.** *Consider an empire $\mathcal{E}$ defined in a graph H, an index $i \in [\sigma_0]$, and a set of chosen parts $\mathcal{Q} \subseteq \mathcal{P}_i(\mathcal{E})$, and a set of relevant parts $\mathcal{R} \subseteq \mathcal{P}_i(\mathcal{E})$ with $\mathcal{Q} \subseteq \mathcal{R}$. For each part $P \in \mathcal{R}$, pick an arbitrary representative vertex $v_P \in P$. The digraph $I = \texttt{ConditioningDigraph}(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$, called a* temporary conditioning digraph*, is the digraph with vertex set $V(I) = \mathcal{R}$ and a directed edge $(P, Q) \in E(I)$ if and only if all of the following conditions hold, where $D \leftarrow \texttt{PreprocANN}(H, \{v_P\}_{P \in \mathcal{Q}})$:*

- *(Permanent condition) There exists some descendant $Q' \in \cup_{j \le i} \mathcal{P}_j(\mathcal{E})$ of Q for which P intersects $S_{Q'}$.*
- *(Temporary condition) $\texttt{JLReff}_H(v_Q, \texttt{ANN}_D(v_P)) \le 8\gamma_{temp}\gamma_{ann}(\texttt{JLReff}_H(v_P, \texttt{ANN}_D(v_P)) + \alpha^{i/(\sigma_0+1)} r_{min})$ where $\texttt{JLReff}_H(a, b)$ for $a, b \in V(H)$ denotes a multiplicative 2-approximation to $\texttt{Reff}_H(a, b)$ given using Theorem 4.3.3.*

The digraph $J = \texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R})$, called a permanent conditioning digraph*, has vertex set $V(J) = \mathcal{R}$ and a directed edge $(P, Q) \in E(I)$ if and only if the "Permanent Condition" holds for the (P, Q) pair.*

The "Temporary condition" implies the following relevant metric properties thanks to the "Closeness" guarantee of Theorem 4.3.6. We only use ANN in the definition of the temporary condition for runtime purposes:

Corollary 4.7.12 (Temporary condition implications)**.** *Consider two different parts $P, Q \in \mathcal{R}$ and let $P_{\min} := \arg\min_{P' \in \mathcal{Q}} \mathit{Reff}_H(v_P, v_{P'})$. If the pair of parts (P, Q) satisfies the "Temporary condition," then*

$$\mathit{Reff}_H(v_Q, v_{P_{\min}}) \leq 34\gamma_{temp}\gamma_{ann}^2(\mathit{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)}r_{min})$$

If the pair does not satisfy the "Temporary condition," then

$$\mathit{Reff}_H(v_Q, v_{P_{\min}}) \geq \gamma_{temp}(\mathit{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)}r_{min})$$

Proof. **Upper bound.** Since the "Temporary condition" is satisfied,

$$\texttt{JLReff}_H(v_Q, \texttt{ANN}_D(v_P)) \leq 8\gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{JLReff}_H(v_P, \texttt{ANN}_D(v_P)) + \alpha^{i/(\sigma_0+1)}r_{min})$$

Since $\texttt{JLReff}_H$ is a 2-approximation to $\texttt{Reff}_H$,

$$\texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) \leq 32\gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{Reff}_H(v_P, \texttt{ANN}_D(v_P)) + \alpha^{i/(\sigma_0+1)}r_{min})$$

By Theorem 4.3.6,

$$\texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) \leq 32\gamma_{\text{temp}}\gamma_{\text{ann}}^2(\texttt{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)}r_{min})$$

By the triangle inequality and Theorem 4.3.6,

$$\begin{aligned}\texttt{Reff}_H(v_Q, v_{P_{\min}}) &\leq \texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) + \texttt{Reff}_H(\texttt{ANN}_D(v_P), v_P) + \texttt{Reff}_H(v_P, v_{P_{\min}}) \\ &\leq 34\gamma_{\text{temp}}\gamma_{\text{ann}}^2(\texttt{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)}r_{min})\end{aligned}$$

as desired.

Lower bound. Since the "Temporary condition" is not satisfied,

$$\texttt{JLReff}_H(v_Q, \texttt{ANN}_D(v_P)) \geq 8\gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{JLReff}_H(v_P, \texttt{ANN}_D(v_P)) + \alpha^{i/(\sigma_0+1)}r_{min})$$

By the approximation of $\texttt{JLReff}_H$ to $\texttt{Reff}_H$,

$$\texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) \geq 2\gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{Reff}_H(v_P, \texttt{ANN}_D(v_P)) + \alpha^{i/(\sigma_0+1)}r_{min})$$

By definition of $P_{\min}$,

$$\texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) \geq 2\gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)} r_{min})$$

By the triangle inequality and Theorem 4.3.6,

$$\begin{aligned}\texttt{Reff}_H(v_Q, v_{P_{\min}}) &\geq \texttt{Reff}_H(v_Q, \texttt{ANN}_D(v_P)) - \texttt{Reff}_H(\texttt{ANN}_D(v_P), v_P) - \texttt{Reff}_H(v_P, v_{P_{\min}}) \\ &\geq \gamma_{\text{temp}}\gamma_{\text{ann}}(\texttt{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)} r_{min}) \\ &\geq \gamma_{\text{temp}}(\texttt{Reff}_H(v_P, v_{P_{\min}}) + \alpha^{i/(\sigma_0+1)} r_{min})\end{aligned}$$

as desired.

□

Notice that the "Permanent condition" is slightly different from the condition (a) discused in the last subsubsection of Section 4.7.2 in that we are interested in *any* intersection of descendant shortcutters of Q with P, not just S_Q with P. This is done for the same reason that we use `DSize` in place of the actual size of S_Q. Specifically, we use these definitions because they guarantee that any edge in `PermanentConditioningDigraph` for level i is also an edge in `PermanentConditioningDigraph` for higher levels:

Proposition 4.7.13 (Vertical monotonicity)**.** *Consider any empire $\mathcal{E}$, two indices $j < i \in [\sigma_0]$, and two sets of parts $\mathcal{R}_j \subseteq \mathcal{P}_j(\mathcal{E})$ and $\mathcal{R}_i \subseteq \mathcal{P}_i(\mathcal{E})$ with every part in $\mathcal{R}_j$ having an ancestor in $\mathcal{R}_i$. Then for any $(P, Q) \in$ `PermanentConditioningDigraph`$(\mathcal{E}, j, \mathcal{R}_j)$,*

$(P', Q') \in$ `PermanentConditioningDigraph`$(\mathcal{E}, i, \mathcal{R}_i)$, where P' and Q' are the ancestors of P and Q respectively in $\mathcal{P}_i(\mathcal{E})$.

Proof. The permanence condition only gets more restrictive in lower levels. Therefore, if $(P, Q) \in \texttt{PermanentConditioningDigraph}(\mathcal{E}, j, \mathcal{R}_j)$,

the edge $(P', Q') \in \texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R}_i)$, as desired. □

`ExtendHierarchy` uses $I \leftarrow \texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$ to define a directed metric on the set $\mathcal{R}'_{i+1} \subseteq \mathcal{P}_i(\mathcal{E})$ of parts with ancestors in $\mathcal{R}_{i+1}$. Specifically, define a (distance) function d_{dir} on $\mathcal{R}'_{i+1}$ by letting $d_{dir}(Q)$ be the distance from $\mathcal{Q}_{i+1}$ to the parent of Q in $\mathcal{R}_{i+1}$ in the digraph I.

`ExtendHierarchy` could use d_{dir} by picking some distance threshold $j^* + 1$ from $\mathcal{Q}_{i+1}$, letting $\mathcal{R}_i$ be the set of all $P \in \mathcal{R}'_{i+1}$ with $d_{dir}(P) \leq j^* + 1$, and defining $\mathcal{Q}_i$ to be the set of parts $P \in \mathcal{R}_i$ with $d_{dir}(P) \leq j^*$ with near-largest shortcutter size. If it did this, $\mathcal{R}'_{i+1} \setminus \mathcal{R}_i$ would consist of parts that at least distance $\alpha^{(i+1)/(\sigma_0+1)} r_{min}$ from any of the chosen parts $\mathcal{Q}_i$. Even better, the $\mathcal{Q}_i$s are at least γ_{temp} times closer to $\mathcal{Q}_{i+1}$ in the H-effective resistance metric than any part in $\mathcal{R}'_{i+1} \setminus \mathcal{R}_i$. Therefore, the conditioning hierarchy being locally maximum at level $i+1$ shows that deleting all parts of $\mathcal{Q}_i$ from shortcutters for parts in $\mathcal{Q}_{i+1}$ that they have edges to does not increase the conductivity of $\mathcal{E}$ much. The upside of doing this is

that all shortcutters for parts in $\mathcal{R}'_{i+1} \setminus \mathcal{R}_i$ have shortcutters that are carved with respect to $\cup_{j \le i+1} \mathcal{Q}_j$, thus respecting Key Idea 4.7.6. This approach does not quite work across multiple conditioning rounds, but is easily fixed in Section 4.7.7 by using $j^* + (2\sigma_1)^i$ instead of $j^* + 1$.

The previous paragraph suggests that conditioning digraphs can be used to pick good parts to condition on once. To pick good parts to condition on multiple times, we need to show that (a) distances only grow in conditioning digraphs and (b) parts chosen at lower levels cannot intersect shortcutters for parts in higher levels that were declared irrelevant. We discuss concern (a) in Section 4.7.5, while we discuss concern (b) in Section 4.7.7.

Fast computation with conditioning digraphs

We now show that computations involving conditioning digraphs are cheap. These graphs may be very dense, but luckily they are not dense compared to the input graph H:

Proposition 4.7.14. *The conditioning digraphs has two nice computational properties:*

- *(Size)* $\texttt{ConditioningDigraph}(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$ *and* $\texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R})$ *have at most* $m^{1+o(1)}$ *edges.*
- *(Computation time) The graphs* $\texttt{ConditioningDigraph}(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$ *and* $\texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R})$ *can be computed in* $m^{1+o(1)}$ *time.*

Proof. **Size.** It suffices to bound the number of possible intersections between a cluster P and a shortcutter S_Q (property (1) for edge existence). Each intersection must contain a vertex and each vertex is only in one shortcutter in each clan. Since there are only $m^{o(1)}$ clans in $\mathcal{E}$, only $(m^{o(1)})^2 \le m^{o(1)}$ pairwise intersections can involve this vertex. Therefore, there are at most $nm^{o(1)} \le m^{1+o(1)}$ candidate pairs that satisfy (1).

Computation time. For each candidate edge described in the "Size" part, it takes two approximate nearest neighbor calls to check whether then "Temporary condition" is satisfied. By Theorems 4.3.6 and 4.3.3, these queries take $\tilde{O}(1)$ time each, with $\tilde{O}(m)$ preprocessing time up front. Therefore, it only takes $m^{1+o(1)}$ time to compute `ConditioningDigraph`. □

Since this graph has almost-linear size, $\texttt{ConditioningDigraph}(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$ and $\texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R})$-distances from all parts in $\mathcal{Q}$ to each part in $\mathcal{R}$ can be computed in total time $m^{1+o(1)}$.

4.7.5 Maintained state across multiple conditioning rounds (and deleting far-away conditioned vertices) using `MakeNonedgesPermanent`

In this section, we deal with issue (a) raised in the previous section by turning nonedges that do not satisfy the "Temporary condition" into ones that do not satisfy the "Permanent

condition." We start by showing that the permanent conditioning digraph at each level only gets smaller:

Proposition 4.7.15 (Permanent horizontal monotonicity)**.** *Consider two empires $\mathcal{E}'_{prev}$ and $\mathcal{E}$ that satisfy the "Containment" input condition in Definition 4.4.14. Consider some $i \in [\sigma_0]$ and relevant sets $\mathcal{R}'_{prev} \subseteq \mathcal{P}_i(\mathcal{E}'_{prev})$ and $\mathcal{R} \subseteq \mathcal{P}_i(\mathcal{E})$. Suppose that each part $P \in \mathcal{R}$ is contained in some (unique by "Containment") part $Q \in \mathcal{R}'_{prev}$. Let $I'_{prev} :=$ `PermanentConditioningDigraph`$(\mathcal{E}'_{prev}, i, \mathcal{R}'_{prev})$ and $I :=$ `PermanentConditioningDigraph`$(\mathcal{E}, i, \mathcal{R})$. Consider any parts $P, P' \in V(I)$ and let $Q, Q' \in V(I'_{prev})$ be the unique parts that contain P and P' respectively. If $(P, P') \in E(I)$, then $(Q, Q') \in E(I'_{prev})$.*

Proof. Since $(P, P') \in E(I)$, there exists a descendant $W' \in \mathcal{P}_j(\mathcal{E})$ of P' for which P intersects $S_{W'}$ for some $j \leq i$. Let $U' \in \mathcal{P}_j(\mathcal{E}'_{prev})$ be the unique part for which $W' \subseteq U'$. This exists by the "Containment" input condition in Definition 4.4.14. Also by "Containment," $S_{W'} \subseteq S_{U'}$. Therefore, $S_{U'}$ intersects Q since $P \subseteq Q$.

Now, we just need to show that U' is a descendant of Q'; i.e. that $U' \subseteq Q'$. $W' \subseteq P'$ since W' is a descendant of P'. $P' \subseteq Q'$ by definition of Q'. $W' \subseteq U'$ by definition of U', which means that $U' \cap Q' \neq \emptyset$. Since the overlay partitions of any empire form a laminar family of sets, this means that $U' \subseteq Q'$ or that $Q' \subseteq U'$. Since U' is in a lower horde than Q', $U' \subseteq Q'$. Therefore, Q' has a descendant whose shortcutter intersects Q, so $(Q, Q') \in E(I'_{prev})$, as desired. □

Proposition 4.7.15 only reasons about containment of permanent conditioning digraphs between applications of `ConditioningVerts`. In particular, it does not say anything a priori about containment of temporary conditioning digraphs. In general, temporary conditioning digraphs may not be contained within prior conditioning digraphs because conditioning can substantially change the effective resistance metric of a graph. We show that the shortcutters in $\mathcal{E}$ can be modified to make temporary conditioning digraphs into permanent ones. As long as the $\mathcal{Q}$s defining each temporary conditioning digraph are part of a locally maximum conditioning hierarchy, this modification does not substantially increase the conductivity of

$\mathcal{E}$.

Algorithm 7: `MakeNonedgesPermanent`$(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$

```
1  foreach clan C in the horde E_i of E do
2  |  split C into clans C_0, C_1, ..., C_{log m}, where C_j consists of all shortcutters in C with
   |  between 2^j and 2^{j+1} incident edges
3  end
4  I ← ConditioningDigraph(E, i, Q, R)
5  foreach pair (P, Q) with P, Q ∈ R and P intersecting S_{Q'} for some descendant Q' of
   Q do
6  |  if (P, Q) ∉ E(I) then
7  |  |  foreach descendant Q' of Q do
8  |  |  |  S_{Q'} ← S_{Q'} \ P
9  |  |  end
10 |  end
11 end
```

Proposition 4.7.16. *`MakeNonedgesPermanent`$(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$ takes an empire $\mathcal{E}$, an index i, a set of parts $\mathcal{R} \subseteq \mathcal{P}_i(\mathcal{E})$, and a set of parts $\mathcal{Q} \subseteq \mathcal{R}$.*

Suppose that that the locally maximum condition holds; i.e. that

$$\min_{P \in \mathcal{Q}} \texttt{DSize}(\mathcal{E}, P) \geq m^{-1/\sigma_1} \max_{P \in \mathcal{R}} \texttt{DSize}(\mathcal{E}, P)$$

Then `MakeNonedgesPermanent` deletes vertices from shortcutters in $\mathcal{E}$ and splits clans to obtain an empire $\mathcal{E}'$ with the following guarantees:

- *(Permanence) `PermanentConditioningDigraph`$(\mathcal{E}', i, \mathcal{R})$ is a subdigraph of `ConditioningDigraph`$(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$.*
- *(Conductivity and number of clans) If $\mathcal{E}$ is ζ-conductive, then $\mathcal{E}'$ is $\zeta + (\log n)\mu_{app}(\ell_{\max} + \tau)$-conductive. Furthermore, $|\mathcal{E}'_i| \leq O(\log n)|\mathcal{E}_i|$ for all $i \in [\sigma_0]$.*

Furthermore, `MakeNonedgesPermanent` takes almost-linear time.

Proof. **Permanence.** Line 8 removes any intersection between two parts with no edge from one to the other, thus ensuring that any nonedge in `ConditioningDigraph`$(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$, which may satisfy the "Permanent condition," does not satisfy it in $\mathcal{E}'$.

Conductivity and number of clans. Removing shortcutters from a clan cannot decrease its effective size, so $s_{\mathcal{C}_k} \leq s_{\mathcal{C}}$ for all $k \in \{0, 1, \ldots, \log m\}$. Therefore, the number of clans grew by a factor of at most $O(\log n)$.

Now, we bound the increase in conductivity. It suffices to consider a clan $\mathcal{C}_k \in \mathcal{E}_j$ for some $j \leq i$ that contains a shortcutter for a descendant of a part $P_* \in \mathcal{R}$. Otherwise, no shortcutter in $\mathcal{C}_k$ is modified by Line 8, which means that its conductivity does not increase.

We start by characterizing the removals done by Line 8 in a simpler way. Let $\mathcal{N}_z$ be a set of clusters with radius $2^z\alpha^{i/(\sigma_0+1)}r_{min}$ that contain all vertices with H-effective resistance distance at most $2^z\alpha^{i/(\sigma_0+1)}r_{min}$ from $\mathcal{Q}$. Then all vertex removals done by Line 8 are also done by, for all $z \in \{0, 1, \ldots, \log m\}$, removing the clusters in $\mathcal{N}_z$ from shortcutters for cores in $\mathcal{C}_k$ with H-distance greater than $\gamma_{\text{temp}}2^z\alpha^{i/(\sigma_0+1)}r_{min}$ from $\mathcal{N}_z$. This holds by the "Temporary condition" not being satisfied for nonedges of $\texttt{ConditioningDigraph}(\mathcal{E}, i, \mathcal{Q}, \mathcal{R})$ and Corollary 4.7.12.

Now, we bound the effect of deleting $\mathcal{N}_z$ for one $z \in \{0, 1, \ldots, \log m\}$ from shortcutters for γ_{temp}-separated cores in $\mathcal{C}_k$. We start by bounding $|\mathcal{N}_z|$ in terms of $s_{\mathcal{C}_k}$. Start by noticing to bound $|\mathcal{N}_z|$, it suffices to bound the number of $O(\alpha^{i/(\sigma_0+1)}r_{min})$ H-effective resistance diameter clusters needed to cover all parts in $\mathcal{Q}$. By definition of $\texttt{DSize}(\mathcal{E}, P')$, there is a shortcutter with core in $\mathcal{E}_j$ for some $j \le i$ whose size is $\rho = \texttt{DSize}(\mathcal{E}, P')$ that intersects P'. Each clan has at most m/ρ such shortcutters. Furthermore, this core has H-effective resistance diameter $\alpha^{j/(\sigma_0+1)}r_{min} \le \alpha^{i/(\sigma_0+1)}r_{min}$. "Assigning" a part $P' \in \mathcal{Q}$ to the core of the maximum size descendant shortcutter therefore shows that,

$$\begin{aligned}|\mathcal{N}_z| &\le (\sum_{i'=1}^{i} |\mathcal{E}_{i'}|)\frac{m}{\min_{Q\in\mathcal{Q}} \texttt{DSize}(\mathcal{E}, Q)} \\ &\le \frac{\ell_{\max} m}{\min_{Q\in\mathcal{Q}} \texttt{DSize}(\mathcal{E}, Q)}\end{aligned}$$

By the "Locally maximum" input condition,

$$\frac{\ell_{\max} m}{\min_{Q\in\mathcal{Q}} \texttt{DSize}(\mathcal{E}, Q)} \le \frac{\ell_{\max} m^{1+1/\sigma_1}}{\max_{Q\in\mathcal{R}} \texttt{DSize}(\mathcal{E}, Q)}$$

Since $\mathcal{C}_k$ contains a shortcutter for a descendant of a part in $\mathcal{R}$,

$$\frac{\ell_{\max} m^{1+1/\sigma_1}}{\max_{Q\in\mathcal{R}} \texttt{DSize}(\mathcal{E}, Q)} \le \frac{\ell_{\max} m^{1+1/\sigma_1}}{\min_{C\in\mathcal{C}_k} |E(S_C) \cup \partial S_C|}$$

By definition of $\mathcal{C}_k$ on Line 2,

$$\frac{\ell_{\max} m^{1+1/\sigma_1}}{\min_{C\in\mathcal{C}_k} |E(S_C) \cup \partial S_C|} \le \frac{2\ell_{\max} m^{1+1/\sigma_1}}{\max_{C\in\mathcal{C}_k} |E(S_C) \cup \partial S_C|}$$

By definition of $s_{\mathcal{C}_k}$,

$$|\mathcal{N}_z| \le \frac{2\ell_{\max} m^{1+1/\sigma_1}}{\max_{C\in\mathcal{C}_k} |E(S_C) \cup \partial S_C|} \le 2\ell_{\max} m^{1/\sigma_1} s_{\mathcal{C}_k}$$

Therefore, by Proposition 4.7.1 applied to $\mathcal{L} \leftarrow \mathcal{N}_z$ and $\mathcal{C} \leftarrow \mathcal{C}_k$, Line 8 only additively increases the conductivity of $\mathcal{C}_k$ by $\mu_{\text{app}}(2\ell_{\max} + \tau)$. Therefore, summing over each $z \in$

$\{0, 1, 2, \ldots, \log m\}$ shows that the total increase in conductivity of $\mathcal{C}_k$ due to Line 8 is at most $(\log m)\mu_{\text{app}}(2\ell_{\max} + \tau)$, as desired.

Runtime. Looping over intersecting pairs takes $m^{1+o(1)}$ time. Checking the presence of an edge takes constant time. Therefore, this procedure takes $m^{1+o(1)}$ time. □

Now, we are ready to prove the main result of this section. Specifically, applying `MakeNonedgesPermanent` to each temporary conditioning digraph ensures that temporary conditioning digraphs from later applications of `ConditioningVerts` are contained in earlier ones:

Lemma 4.7.17 (Temporary horizontal monotonicity)**.** *Consider some $i \in [\sigma_0]$ and two conditioning hierarchies $\mathcal{CH}_{prev} = (\{\mathcal{Q}_i^{prev}\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i^{prev}\}_{i=1}^{\sigma_0})$ and $\mathcal{CH} = (\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ for the empires $\mathcal{E}_{prev}$ and $\mathcal{E}$ in graphs H_{prev} and H respectively.*

Define an empire $\mathcal{E}'_{prev}$ in H_{prev} that satisfies the following conditions:

- *($\mathcal{E}'_{prev}$ description) $\mathcal{E}'_{prev}$ is obtained by applying* `MakeNonedgesPermanent`$(\mathcal{E}_{prev}, i, \mathcal{Q}_i^{prev}, \mathcal{R}_i^{prev})$ *for each i and letting $\mathcal{E}'_{prev} \leftarrow \mathcal{E}_{prev}$ afterwards.*
- *(Containment) $\mathcal{E}'_{prev}$ and $\mathcal{E}$ satisfy the "Containment" input condition in Definition 4.4.14.*
- *(Locally maximum) $\mathcal{CH}_{prev}$ is locally maximum.*

For each i, let $I :=$ `ConditioningDigraph`$(\mathcal{E}, i, \mathcal{Q}_i, \mathcal{R}_i)$ *and $I_{prev} :=$* `ConditioningDigraph`$(\mathcal{E}_{prev}, i, \mathcal{Q}_i^{prev}, \mathcal{R}_i^{prev})$. *Consider any parts $P, P' \in V(I)$ and let $Q, Q' \in V(I_{prev})$ be the unique parts that contain P and P' respectively. If $(P, P') \in E(I)$, then $(Q, Q') \in E(I_{prev})$.*

Proof. Since edges in `ConditioningDigraph` are more constrained then those in `PermanentConditioningDigraph`, I is a subdigraph of `PermanentConditioningDigraph`$(\mathcal{E}, i, \mathcal{R}_i)$. Therefore,

$$(P, P') \in E(\texttt{PermanentConditioningDigraph}(\mathcal{E}, i, \mathcal{R}_i))$$

By the "Containment" condition of this lemma, Proposition 4.7.15 applies and shows that

$$(Q, Q') \in E(\texttt{PermanentConditioningDigraph}(\mathcal{E}'_{prev}, i, \mathcal{R}_i^{prev}))$$

By the "E'_{prev} description" and "Locally maximum" conditions, Proposition 4.7.16 applies. By the "Permanence" guarantee of Proposition 4.7.16, `PermanentConditioningDigraph`$(\mathcal{E}'_{prev}, i, \mathcal{R}_i^{prev})$ is a subdigraph of I_{prev}, which means that

$$(Q, Q') \in E(I_{prev})$$

as desired. □

4.7.6 `MakeNonedgesPermanent` implies carving as a side effect

In this section, we define *carvable* conditioning hierarchies and show that the empire $\mathcal{E}$ associated with a conditioning hierarchy is carved with respect to the bottom of the hierarchy. In later sections, we will show that `ConditioningVerts` maintains a carvable conditioning hierarchy, which will make it easy to show Lemma 4.4.15.

Definition 4.7.18 (Carvable conditioning hierarchies)**.** *Consider an empire $\mathcal{E}$ and two families of sets $\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}$ and $\{\mathcal{R}_i\}_{i=1}^{\sigma_0}$. We say that the pair $(\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ is* carvable *if satisfies the following properties for all $i \in [\sigma_0]$:*

- *(Horizontal containment) $\mathcal{Q}_i \subseteq \mathcal{R}_i \subseteq \mathcal{P}_i(\mathcal{E})$*
- *(Vertical containment) For every part $P \in \mathcal{R}_i$, there is some $Q \in \mathcal{R}_{i+1}$ for which $P \subseteq Q$.*
- *(Strong vertical closeness) The $\mathcal{R}_{i+1}$ parent of each part in $\mathcal{Q}_i$ is within distance $(2\sigma_1)(4\sigma_1)^i$ of some part in $\mathcal{Q}_{i+1}$ in $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$.*
- *(Locally maximum) If $\mathcal{Q}_i \neq \emptyset$, $\min_{P \in \mathcal{Q}_i} \texttt{DSize}(\mathcal{E}, P) \geq m^{-1/\sigma_1} \max_{Q \in \mathcal{R}_i} \texttt{DSize}(\mathcal{E}, Q)$.*
- *(Irrelevant parts are far away) Any part $P \in \mathcal{P}_i(\mathcal{E}) \setminus \mathcal{R}_i$ with parent $Q \in \mathcal{R}_{i+1}$ has the property that Q is at least distance $(4\sigma_1)^i$ from any parent of a part in $\mathcal{Q}_i$ in the digraph $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$.*

The above definition does not specifically call the $(\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ pair a conditioning hierarchy. We now show that it is. This allows us to call such a pair of families of sets a *carvable conditioning hierarchy* for $\mathcal{E}$:

Remark 8. *A carvable pair of families of sets $(\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ for $\mathcal{E}$ is a locally maximum conditioning hierarchy.*

Proof. **Horizontal containment, vertical containment, and locally maximum.** All of these conditions are exactly the same as the definitions for conditioning hierarchies.

Vertical closeness. Consider the shortest path $P = P_0 \to P_1 \to \ldots \to P_t = Q$ in $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$ from a part $P \in \mathcal{Q}_{i+1}$ to the parent of a part $Q \in \mathcal{Q}_i$. By Corollary 4.7.12 and the triangle inequality, all vertices in Q are at most distance $(34\gamma_{\text{temp}}\gamma_{\text{ann}}^2)^t \alpha^{(i+1)/(\sigma_0+1)} r_{min}$ from $\mathcal{Q}_{i+1}$ in the effective resistance metric of H, the graph associated with $\mathcal{E}$. By the "Strong vertical closeness" condition, $t \leq (2\sigma_1)(4\sigma_1)^i$. Since $(34\gamma_{\text{temp}}\gamma_{\text{ann}}^2)^{(2\sigma_1)(4\sigma_1)^i} \leq (\mu_{\text{carve}}/(100\gamma_{\text{temp}}\gamma_{\text{ann}}^2))$, all vertices in Q are at most $(\mu_{\text{carve}}/(100\gamma_{\text{temp}}\gamma_{\text{ann}}^2))\alpha^{(i+1)/(\sigma_0+1)} r_{min}$-distance away from $P \in \mathcal{Q}_{i+1}$, as desired. □

Now, we show that applying `MakeNonedgesPermanent` for each $i \in [\sigma_0]$ does the required amount of carving:

Proposition 4.7.19. *Consider the empire $\mathcal{E}$ and a carvable conditioning hierarchy $\mathcal{CH} = (\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ for $\mathcal{E}$. Obtain an empire $\mathcal{E}'$ by applying* `MakeNonedgesPermanent`$(\mathcal{E}, i, \mathcal{Q}_i, \mathcal{R}_i)$ *for each $i \in [\sigma_0]$ and letting $\mathcal{E}'$ be $\mathcal{E}$ at the end. $\mathcal{E}'$ has the following property:*

- *(Carving) For each active part P, let $Q \in \mathcal{P}_{i^*}(\mathcal{E})$ be the smallest ancestor of P (possibly P) for which $Q \in \mathcal{R}_{i^*}$. Then S_P is carved with respect to $\cup_{j \le i^*} \mathcal{Q}_j$.*

Proof. **Well-definedness.** We just need to make sure that the input conditions for `MakeNonedgesPermanent` are satisfied. $\mathcal{Q}_i \subseteq \mathcal{R}_i$ because $\mathcal{CH}$ is a conditioning hierarchy. Since $\mathcal{CH}$ is locally maximum,

$$\min_{P \in \mathcal{Q}_i} \texttt{DSize}(\mathcal{E}, P) \ge m^{-1/\sigma_1} \max_{P \in \mathcal{R}_i} \texttt{DSize}(\mathcal{E}, P)$$

for all $i \in [\sigma_0]$. This is the remaining input condition to `MakeNonedgesPermanent`.

Carving. Let $i' \le i^*$ be the value for which $P \in \mathcal{P}_{i'}(\mathcal{E})$. Break the reasoning up into two cases:

$\underline{i' < i^*}$. By the "Irrelevant parts are far away" condition of carvable conditioning hierarchies, the distance from any part in $\mathcal{Q}_{i^*}$ to Q in `PermanentConditioningDigraph`$(\mathcal{E}', i^*, \mathcal{R}_{i^*})$ is at least $(4\sigma_1)^{i^*-1}$. By "Strong vertical closeness" for all $j < i^*$ and Proposition 4.7.13, the distance from $\mathcal{Q}_{i^*}$ to any $\mathcal{P}_{i^*}(\mathcal{E})$-ancestor of a part in $\mathcal{Q}_j$ is at most

$$(2\sigma_1)((4\sigma_1)^{i^*-2} + (4\sigma_1)^{i^*-3} + \ldots + 1) < (4\sigma_1)^{i^*-1} - 1$$

In particular, no $\mathcal{P}_{i^*}(\mathcal{E})$-ancestor of a part in $\mathcal{Q}_j$ for any $j \le i^*$ has an edge to Q in `PermanentConditioningDigraph`$(\mathcal{E}', i^*, \mathcal{R}_{i^*})$. Therefore, since the "Permanent condition" is not satisfied for nonedges of this graph and $P \subseteq Q$, S_P does not intersect any part in $\cup_{j \le i^*} \mathcal{Q}_j$, which is the desired carving property.

$\underline{i' = i^*}$. In this case, $P = Q \in \mathcal{R}_{i^*}$. Consider any $X \in \mathcal{Q}_j$ for some $j \le i^*$ that intersects S_P and let Y be its ancestor in $\mathcal{P}_{i^*}(\mathcal{E})$. By "Vertical containment," $Y \in \mathcal{R}_{i^*}$. Let $Y_{\min} = \arg\min_{Y' \in \mathcal{Q}_{i^*}} \texttt{Reff}_H(v_Y, v_{Y'})$. By "Vertical closeness" and the triangle inequality,

$$\texttt{Reff}_H(v_Y, v_{Y_{\min}}) \le (\mu_{\text{carve}}/(100\gamma_{\text{temp}}\gamma_{\text{ann}}^2))\alpha^{(i^*)/(\sigma_0+1)} r_{min}$$

By construction of $\mathcal{E}'$, Lemma 4.7.17 applies. Therefore, edges in `PermanentConditioningDigraph`$(\mathcal{E}', i^*, \mathcal{R}_{i^*})$ also satisfy the "Temporary condition," which means that

$$\texttt{Reff}_H(v_P, v_{Y_{\min}}) \le (34\gamma_{\text{temp}}\gamma_{\text{ann}}^2)(\texttt{Reff}_H(v_Y, v_{Y_{\min}}) + \alpha^{(i^*)/(\sigma_0+1)} r_{min})$$

by Corollary 4.7.12. By the triangle inequality and the previous two inequalities,

$$\begin{aligned}\texttt{Reff}_H(v_P, v_Y) &\le \texttt{Reff}_H(v_P, v_{Y_{\min}}) + \texttt{Reff}_H(v_{Y_{\min}}, v_Y) \\ &\le (\mu_{\text{carve}}/2)\alpha^{(i^*)/(\sigma_0+1)} r_{min}\end{aligned}$$

Therefore, by $\alpha^{(i^*)/(\sigma_0+1)}r_{min}$-boundedness of $\mathcal{E}_{i^*}$ and the triangle inequality, all vertices of X are within distance $\mu_{\text{carve}}\alpha^{(i^*)/(\sigma_0+1)}r_{min}$ of P, which means that S_P is carved with respect to $\cup_{j\le i^*}\mathcal{Q}_j$, as desired. □

4.7.7 Deciding on parts to condition on from $\mathcal{P}_i(\mathcal{E})$ given a choice from $\mathcal{P}_{i+1}(\mathcal{E})$

In the previous section, we defined carvable conditioning hierarchies. Proposition 4.7.19 shows that `MakeNonedgesPermanent` makes $\mathcal{E}$ carved with respect to the lowest $\mathcal{Q}_i$ with active shortcutters. In this section, we give an algorithm, `ExtendHierarchy`, which adds one level to the bottom of a carvable conditioning hierarchy. `ExtendHierarchy` extends carvable conditioning hierarchies while ensuring that if we can condition on $\mathcal{Q}_i$, we make a substantial amount of progress. We now define conditioning hierarchies for which substantial progress can be made at any level:

Definition 4.7.20 (Advanceable conditioning hierarchies)**.** *Consider an empire $\mathcal{E}$, a conditioning hierarchy $\mathcal{CH} = (\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ for $\mathcal{E}$, and a tuple of* distance indices *$(j_i)_{i=1}^{\sigma_0}$. $\mathcal{CH}$ is* advanceable for $(j_i)_{i=1}^{\sigma_0}$ *if the following holds for all $i \in [\sigma_0]$:*

- *(All local maxima) Let $s_i := \lfloor \log_{m^{1/\sigma_1}}(\max_{P\in\mathcal{R}_i} \texttt{DSize}(\mathcal{E}, P))\rfloor$. Then $\mathcal{Q}_i$ contains all parts $Q \in \mathcal{R}_i$ whose $\mathcal{P}_{i+1}(\mathcal{E})$ parents are within distance $j_i(4\sigma_1)^i$ of some part in $\mathcal{Q}_{i+1}$ in*

  $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$ *with* $\texttt{DSize}(\mathcal{E}, Q) \ge m^{s_i/\sigma_1}$.

- *(Relevant set bound) When $\mathcal{Q}_{i+1} \neq \emptyset$, $\mathcal{R}_i$ contains the set of parts whose $\mathcal{P}_{i+1}(\mathcal{E})$ parents are within distance $(j_i+1)(4\sigma_1)^i$ of $\mathcal{Q}_{i+1}$ in* $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$.

The "All local maxima" condition, when coupled with the "Progress" input condition for `ConditioningVerts`, shows that conditioning on $\mathcal{Q}_i$ replaces the word of sizes $s_{\sigma_0+1}s_{\sigma_0}\dots s_1$ with a lexicographically smaller word, as desired in Key Idea 4.7.5. The "Relevant set bound" is used to show that relevant sets only get smaller in the output of `ExtendHierarchy`.

We now give an algorithm `ExtendHierarchy` that adds an additional level to a carvable conditioning hierarchy that is advanceable for some tuple of indices that is lexicographically smaller than the previous tuple. The intuition behind `ExtendHierarchy` is that the distance index j^* returned has the property that parts just beyond that distance (within distance $(j^*+1)(4\sigma_1)^i$) do not have larger shortcutters. As a result, after "Temporary condition"

nonedges are converted into "Permanent condition" nonedges, the shortcutters with parts that are farther than distance $(j^*+1)(4\sigma_1)^i$ respect Key Idea 4.7.6.

Algorithm 8: `ExtendHierarchy`$(\mathcal{E}, i, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1}, j_{prev})$

1 $I \leftarrow$ `ConditioningDigraph`$(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$
2 **foreach** $j \in \{0, 1, 2, \ldots, 2\sigma_1\}$ **do**
3 $\quad \mathcal{Q}'_j \leftarrow$ the set of parts in $\mathcal{P}_i(\mathcal{E})$ with $\mathcal{P}_{i+1}(\mathcal{E})$ parents that are in $\mathcal{R}_{i+1}$ and have distance at most $j(4\sigma_1)^i$ from $\mathcal{Q}_{i+1}$ in I
4 $\quad s_j \leftarrow \lfloor \log_{m^{1/\sigma_1}}(\max_{Q \in \mathcal{Q}'_j} \texttt{DSize}(\mathcal{E}, Q)) \rfloor$
5 **end**
`// selection rule: pick the maximum local maximum closer than the previous one`
6 $j^* \leftarrow \arg\max_{j:(1) s_{j+1}=s_j \text{ and } (2) j \le j_{prev}} j$
`// condition on all parts with shortcutters with size in the` s_{j^*} `bucket`
7 **return** *(j^*, s_{j^*}, all parts Q in $\mathcal{Q}'_{j^*}$ with* `DSize`$(\mathcal{E}, Q) \ge m^{s_{j^*}/\sigma_1}$*, all parts Q in $\mathcal{Q}'_{j^*+1}$)*

Proposition 4.7.21. *`ExtendHierarchy`$(\mathcal{E}, i, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1}, j_{prev})$ takes in an empire $\mathcal{E}$, an index i, and $\mathcal{Q}_{i+1}$ and $\mathcal{R}_{i+1}$ for a carvable conditioning hierarchy $\mathcal{CH} = (\{\mathcal{Q}_k\}_{k=1}^{\sigma_0}, \{\mathcal{R}_k\}_{k=1}^{\sigma_0})$ that is advanceable with respect to some distance indices $(j_k)_{k=1}^{\sigma_0}$ with $j_i = j_{prev}$. If*

- *$j_{prev} \ge \sigma_1$*
- *$\mathcal{Q}_k = \emptyset$ for all $k \le i$*

then `ExtendHierarchy`$(\mathcal{E}, i, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1}, j_{prev})$ returns j'_i, $s_{j'_i}$, $\mathcal{Q}_i$, and $\mathcal{R}_i$ that, when added to $\mathcal{CH}$, make it have the following properties:

- *(Carvable) $\mathcal{CH}$ is a carvable conditioning hierarchy for $\mathcal{E}$.*
- *(Advanceable) $\mathcal{CH}$ is advanceable for the distance indices $(j'_k)_{k=1}^{\sigma_0}$, where $j'_k = j_k$ for all $k > i$ and*

$$j_i - (s_{j_i} - s_{j'_i}) - 1 \le j'_i \le j_i$$

- *(Strong relevant set bound) $\mathcal{R}_i$ **is** the set of parts whose $\mathcal{P}_{i+1}(\mathcal{E})$ parents are within distance $(j'_i+1)(4\sigma_1)^i$ of $\mathcal{Q}_{i+1}$ in `ConditioningDigraph`$(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$.*

Proof. **Well-definedness.** It suffices to show that the optimization problem on Line 6 is feasible. Since $\mathcal{Q}'_j \subseteq \mathcal{Q}'_{j+1}$ for all j, $s_j \le s_{j+1}$ for all j. Since $j_{prev} > \sigma_1 + 2$ and the s_js are integers between 0 and σ_1 inclusive, there must exist a $j \le j_{prev}$ that satisfies condition (1) of the optimization problem on Line 6. Therefore, the algorithm is well-defined.

Carvable conditioning hierarchy.

Horizontal containment. Follows from the fact that $\mathcal{Q}'_j \subseteq \mathcal{Q}'_{j+1}$ for all j.

Vertical containment. All of the parts in $\mathcal{Q}'_{j^*+1}$ have parents in $\mathcal{R}_{i+1}$.
Strong vertical closeness. The algorithm only considers $j \le 2\sigma_1$.
Locally maximum. This is the definition of s_{j^*} and $\mathcal{Q}_{j^*}$.
Irrelevant parts are far away. All parts in $\mathcal{P}_i(\mathcal{E}) \setminus \mathcal{R}_i$ have parents in $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}_{i+1}, \mathcal{R}_{i+1})$ with distance greater than $(j^*+1)(4\sigma_1)^i$ from $\mathcal{Q}_{i+1}$, while all parts in $\mathcal{Q}_i$ have distance at most $j^*(4\sigma_1)^i$ from $\mathcal{Q}_{i+1}$. Therefore, by the triangle inequality, the distance from parents of $\mathcal{Q}_i$ to any parent of a part in $\mathcal{P}_i(\mathcal{E}) \setminus \mathcal{R}_i$ is at least $(4\sigma_1)^i$ as long as the parent is in $\mathcal{R}_{i+1}$. Otherwise, the desired bound follows from the fact that $\mathcal{CH}$ satisfied "Irrelevant parts are far away" on higher levels along with Proposition 4.7.13.

Advanceable.

j'_i bound. In the output, $j'_i \leftarrow j^*$, so we just need to bound j^*. $j^* \le j_{prev}$ by condition (2) for j^*. For all j, $s_{j+1} \ge s_j$. This combined with condition (1) shows that for all $j > j^*$, $s_{j+1} \ge s_j + 1$. Therefore, $s_{j_{prev}} - s_{j^*} \ge j_{prev} - j^* - 1$. Rearrangement gives the desired lower bound.

All local maxima. By definition in the output, $\mathcal{Q}_{j'_i}$ contains all parts P with $\texttt{DSize}(\mathcal{E}, P) \ge m^{s_{j'_i}/\sigma_1}$. This is the desired property by Line 4.

Strong relevant set bound. This is the definition of $\mathcal{Q}'_{j'_i+1}$. □

4.7.8 ConditioningVerts

Now, we implement `ConditioningVerts`, which returns parts to condition on in order to satisfy Lemmas 4.4.15 and 4.4.16. `ConditioningVerts` maintains a carvable conditioning hierarchy that is advanceable for distance indices $(j_i)_{i=1}^{\sigma_0}$. While maintaining such a hierarchy is relatively simple (just get rid of anything outside of the promised distances/sizes), many lower level $\mathcal{Q}_i$s will become empty. To make more progress, `ConditioningVerts` applies `ExtendHierarchy` to refill the lower $\mathcal{Q}_i$s with parts for smaller shortcutters than before.

Algorithm 9: `ConditioningVerts`$(\mathcal{E})$ initialization

```
   // Initialization; only occurs on first ConditioningVerts call
 1 Q_{σ0+1} ← {V(G)}
 2 R_{σ0+1} ← {V(G)}
 3 s_{σ0+1} ← σ1
 4 j_{σ0+1} ← 2σ1
 5 for i = {1, 2, ..., σ0} do
 6 |   Q_i ← ∅
 7 |   R_i ← ∅
 8 |   s_i ← σ1
 9 |   j_i ← 2σ1
10 end
   // End initialization
```

Algorithm 10: `ConditioningVerts`$(\mathcal{E})$ each execution

```
 1 for i = {σ0, σ0 − 1, ..., 2, 1} do
 2 |   Q_i ← refinement of Q_i by P_i(E), with all parts with shortcutters of size either (a)
   |   below m^{s_i/σ1} or (b) parent distance to Q_{i+1} in
   |   ConditioningDigraph(E, i + 1, Q_{i+1}, R_{i+1}) farther than j_i(4σ1)^i removed
 3 |   R_i ← refinement of R_i by P_i(E)
 4 end
   // claim:  always exists
 5 i* ← maximum i for which Q_i = ∅
 6 if s_{i*} = 0 then
   |   // can condition on i* + 1 directly, as no i* shortcutters are active
 7 |   Postprocess(E)
 8 |   return Q_{i*+1}
 9 else
10 |   Set all j_is to 2σ1 for i < i*
11 |   for i = i*, i* − 1, ..., 1 do
12 |   |   (j_i, s_i, Q_i, R_i) ← ExtendHierarchy(E, i, Q_{i+1}, R_{i+1}, j_i)
13 |   end
14 |   Postprocess(E)
15 |   return Q_1
16 end
```

Algorithm 11: `Postprocess`$(\mathcal{E})$, which shares state with `ConditioningVerts`

```
1 for i = {1, 2, ..., σ0} do
2 |   MakeNonedgesPermanent(E, i, Q_i, R_i)
3 end
```

We start by checking that this algorithm is well-defined. In doing so, we also describe the conditioning hierarchy throughout the algorithm:

Proposition 4.7.22. *`ConditioningVerts` is well-defined. More precisely, all of the input conditions for algorithms called are met.*

Furthermore, $\mathcal{CH} = (\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ is a carvable conditioning hierarchy that is advanceable for the distance indices $(j_i)_{i=1}^{\sigma_0}$ between the end of the first for loop and the `Postprocess` call.

Proof. We inductively show that call k to `ConditioningVerts` is well-defined. In particular, we show that i^* always exists and that the input conditions to `ExtendHierarchy`, `MakeNonedgesPermanent`, and Proposition 4.7.19 are satisfied. In this induction, we also show that the carvable and advanceable conditions are maintained.

Base case. During the first call to `ConditioningVerts`, $i^* = \sigma_0$ because everything besides Q_{σ_0+1} is initialized to the empty set. The input conditions to `ExtendHierarchy` for each i are satisfied because $j_i = 2\sigma_1 \geq \sigma_1$ and the "Carvable" and "Advanceable" guarantees from the level $i+1$ application of `ExtendHierarchy`. The "Carvable" guarantee ensures that the input conditions for both `MakeNonedgesPermanent` and Proposition 4.7.19 are satisfied. This completes the proof that the first call to `ConditioningVerts` is well-defined and that $\mathcal{CH}$ is carvable and advanceable in the desired line range.

Call k for $k > 1$ i^* existence. Consider the set $\mathcal{Q}_{i_{prev}}$ returned during the $(k-1)$th call to `ConditioningVerts`. By the "Progress" input condition to `ConditioningVerts`, every part $P \in \mathcal{Q}_{i_{prev}}$ has the property that $E(P) = \emptyset$ at the beginning of the kth call to `ConditioningVerts`. In particular, Line 2 eliminates P from $\mathcal{Q}_{i_{prev}}$. In particular, $\mathcal{Q}_{i_{prev}} = \emptyset$ after Line 2. In particular, the optimization problem defining i^* is feasible, so i^* exists.

Call k for $k > 1$ `ExtendHierarchy` j_{prev} feasibility. It suffices to show that $j_i \geq \sigma_1 + s_i$ after each execution of `ConditioningVerts`, because $s_i \geq 0$ for all i. When j_i is set to $2\sigma_1$, $j_i \geq \sigma_1 + s_i$ holds trivially. Only `ExtendHierarchy` changes j_i to some other value. When it does this during the lth call to `ConditioningVerts` for $l < k$,

$$j_i^l \geq j_i^{l-1} - ((s_i^{l-1} - 1) - s_i^l) - 1 \geq j_i^{l-1} - (s_i^{l-1} - s_i^l)$$

by Proposition 4.7.21 inductively working in earlier `ConditioningVerts` calls. The superscript denotes the value after the lth call to `ConditioningVerts`. The $(s_i^{l-1} - 1)$ bound holds because when $\mathcal{Q}_i$ becomes empty, s_i decreases. By the inductive hypothesis,

$$j_i^{l-1} \geq \sigma_1 + s_i^{l-1}$$

Conbining these two inequalities shows that $j_i^l \geq \sigma_1 + s_i^l$. This completes the inductive step and shows that $j_i \geq \sigma_1 + s_i \geq \sigma_1$ before the kth call to `ConditioningVerts`. Therefore, the j_{prev} argument supplied to `ExtendHierarchy` is always at least $\sigma_1 + s_i^{l-1} \geq \sigma_1$, as desired.

Call k for $k > 1$ carvable. Suppose that during call $k-1$ for $k > 1$ to `ConditioningVerts`, $\mathcal{CH}$ is a carvable conditioning hierarchy just before `Postprocess`. After `Postprocess` from call $k-1$, $\mathcal{CH}$ has not changed, as Propositions 4.7.16 and 4.7.19 imply that `Postprocess`

does not modify $\mathcal{CH}$. Therefore, $\mathcal{CH}$ is still a locally maximum conditioning hierarchy at this point that satisfies the "Irrelevant parts are far away" condition. The conditions of Lemma 4.7.17 are satisfied, so distances in each temporary conditioning digraph do not decrease between the end of call $k-1$ and the beginning of call k. Therefore, the "Irrelevant parts are far away" condition still applies. Line 2 maintains "Horizontal containment," "Vertical containment," and "Locally maximum" while restoring "Strong vertical closeness."

Therefore, $(\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ is a carvable conditioning hiearchy at the end of the first for loop. Since `ExtendHierarchy` maintains carvable conditioning hierarchies, $(\{\mathcal{Q}_i\}_{i=1}^{\sigma_0}, \{\mathcal{R}_i\}_{i=1}^{\sigma_0})$ is a carvable conditioning hierarchy when `Postprocess` is called during the kth call to `ConditioningVerts`. This completes the inductive step.

Call k for $k > 1$ advanceable. Suppose that during call $k-1$ for $k > 1$ to `ConditioningVerts`, $\mathcal{CH}$ is an advanceable conditioning hierarchy for indices $(j_i)_{i=1}^{\sigma_0}$. By Lemma 4.7.17, conditioning digraph distances only increase, which means that the "Relevant set bound" continues to at the beginning of `ConditioningVerts` call k. Line 2 restores the "All local maxima" property. These are both of the conditions for $\mathcal{CH}$ being advanceable, so $\mathcal{CH}$ is advanceable for $(j_i)_{i=1}^{\sigma_0}$ after the first for loop. `ExtendHierarchy` maintains advanceability. This completes the inductive step.

Call k for $k > 1$ `ExtendHierarchy`. We have already shown that $\mathcal{CH}$ is carvable and advanceable before the for loop containing Line 12. Furthermore, we have already shown that $j_i \geq \sigma_1$. These are all of the conditions for `ExtendHierarchy`.

Call k for $k > 1$ `MakeNonedgesPermanent`. $\mathcal{CH}$ is carvable when it is supplied to `Postprocess`. This is a sufficient input condition for `MakeNonedgesPermanent`.

□

4.7.9 Proof of Lemmas 4.4.15 (conductivity not increased much) and 4.4.16 (enough progress is made)

Now, we are finally ready to show that `ConditioningVerts` satisfies Lemmas 4.4.15 and 4.4.16.

Lemma 4.4.15. *Given an empire $\mathcal{E} = \{\mathcal{E}_i\}_{i=1}^{\sigma_0}$ in a graph H that satisfies the input conditions given in Definition 4.4.14, `ConditioningVerts`$(\mathcal{E})$ returns a set of parts $\mathcal{K}$ to condition on and removes vertices from the shortcutters in the empire $\mathcal{E}$ to obtain $\mathcal{E}'$. Let $S = \cup_{P\in\mathcal{K}} P \subseteq V(H)$. Then the following guarantees are satisfied:*

- *(Conductivity) $\mathcal{E}'$ is a bucketed, τ-modified, $\zeta + 10(\log m)\mu_{app}(\ell_{\max} + \tau)$-conductive, well-spaced, κ-bounded empire that satisfies the deletion set condition.*
- *(Carving) $\mathcal{E}'$ is carved with respect to S.*

Proof of Lemma 4.4.15. **Conductivity and clan count.** By Proposition 4.7.22, the input to both `MakeNonedgesPermanent` is a carvable conditioning hierarchy. Therefore, the "Conductivity" guarantee of Proposition 4.7.16 shows the desired conductivity increase. These

propositions also show that the number of levels only increases by a factor of $O(\log n)$. No other parameters increase, as desired.

Carving. Proposition 4.7.19 implies that all shortcutters in $\mathcal{E}'$ are carved with respect to $\mathcal{Q}_1$. When $s_{i^*} = 0$, all parts in $\mathcal{R}_{i^*}$ have inactive shortcutters, so they do not need to be carved.

Now, consider any part $P \in \mathcal{P}_k(\mathcal{E}) \setminus \mathcal{R}_k$ for some $k \leq i^*$ with $E(P) \neq \emptyset$. Let Q be the ancestor of P in $\mathcal{R}_j$ for minimum j. Since $E(P) \neq \emptyset$ and $s_{i^*} = 0$, $j > i^*$, which means that S_P is carved with respect to $\mathcal{Q}_{i^*+1}$ by Proposition 4.7.19.

The shortcutters in $\cup_{\ell \geq i^*+1} \mathcal{E}_\ell$ are carved with respect to vertices in parts of $\mathcal{Q}_{i^*+1}$ by Proposition 4.7.19. Therefore, all shortcutters are carved with respect to $\mathcal{Q}_{i^*+1}$ if $s_{i^*} = 0$, as desired. □

The proof of Lemma 4.4.16 shows that enough progress is made by showing that each conditioning round lexicographically decreases the word $s_{\sigma_0} s_{\sigma_0 - 1} \dots s_2 s_1$:

Lemma 4.4.16. *Consider a sequence of calls $\mathcal{K}^j \leftarrow$ `ConditioningVerts`$(\mathcal{E}^j)$ that modifies $\mathcal{E}^j$ to obtain $(\mathcal{E}^j)'$. Suppose that H^j is the graph in which $\mathcal{E}^j$ is defined. Suppose that for each $j > 0$, $\mathcal{E} \leftarrow \mathcal{E}^j$, $\mathcal{E}_{prev} \leftarrow \mathcal{E}^{j-1}$, $\mathcal{K}_{prev} \leftarrow \mathcal{K}^{j-1}$ satisfies the input conditions in Definition 4.4.14. Let*

$$j_{final} = (2\sigma_1)^{2\sigma_0}$$

Then $E(H^{j_{final}}) = \emptyset$.

Proof of Lemma 4.4.16. By the "Relevant set bound," Lemma 4.7.17, and the "Strong relevant set bound," $\mathcal{R}_i$ decreases (all parts replaced with subsets) during an interval of iterations in which $i^* \leq i$. If $s_{i^*} > 0$, then s_{i^*} strictly decreases by the "All local maxima" condition defining the s_is. Furthermore, the s_is for $i > i^*$ do not increase by the "Containment" input condition. In particular, the new word is lexicographically smaller than the old one.

Therefore, we just need to consider the case in which $s_{i^*} = 0$. In this case, the "Progress" input condition implies that s_{i^*+1} decreases in the next call to `ConditioningVerts`. By "Containment," s_i for $i > i^* + 1$ does not increase. Therefore, the new word is again lexicographically smaller than the old one. These are all of the cases. The desired bound on the number of conditioning rounds follows from the fact that there are only $\sigma_1^{\sigma_0}$ possible words.

□

Algorithm	Shortcutting method	σ_0	Runtime
[2, 18]	N\A	0	$O(mn)$
[46]	Offline	1	$O(m\sqrt{n})$
[73]	Offline	2	$O(m^{4/3})$
This paper	Online	$\Theta\left(\frac{\log\log n}{\log\log\log n}\right)$	$m^{1+O(\frac{\log\log\log n}{\log\log n})}$ $= m^{1+o(1)}$

Table 4.2.1: Shortcutting methods, number of shortcutters, and runtimes for prior algorithms

Shortcutting method	Preprocessing	Query
Online	None	$\tilde{O}(\lvert E(S_u) \cup \partial S_u \rvert)$
Offline	$\tilde{O}(\lvert E(S_u) \cup \partial S_u \rvert \lvert \partial S_u \rvert)$	$\tilde{O}(1)$

Table 4.2.2: Shortcutting methods and their runtimes

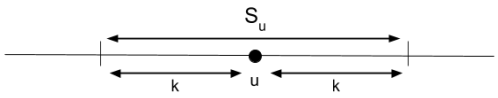

Figure 4.2.1: How online shortcutting saves over the random walk.

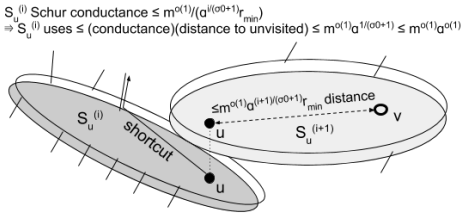

Figure 4.2.2: Bounding the number of times $S_u^{(i)}$ is used.

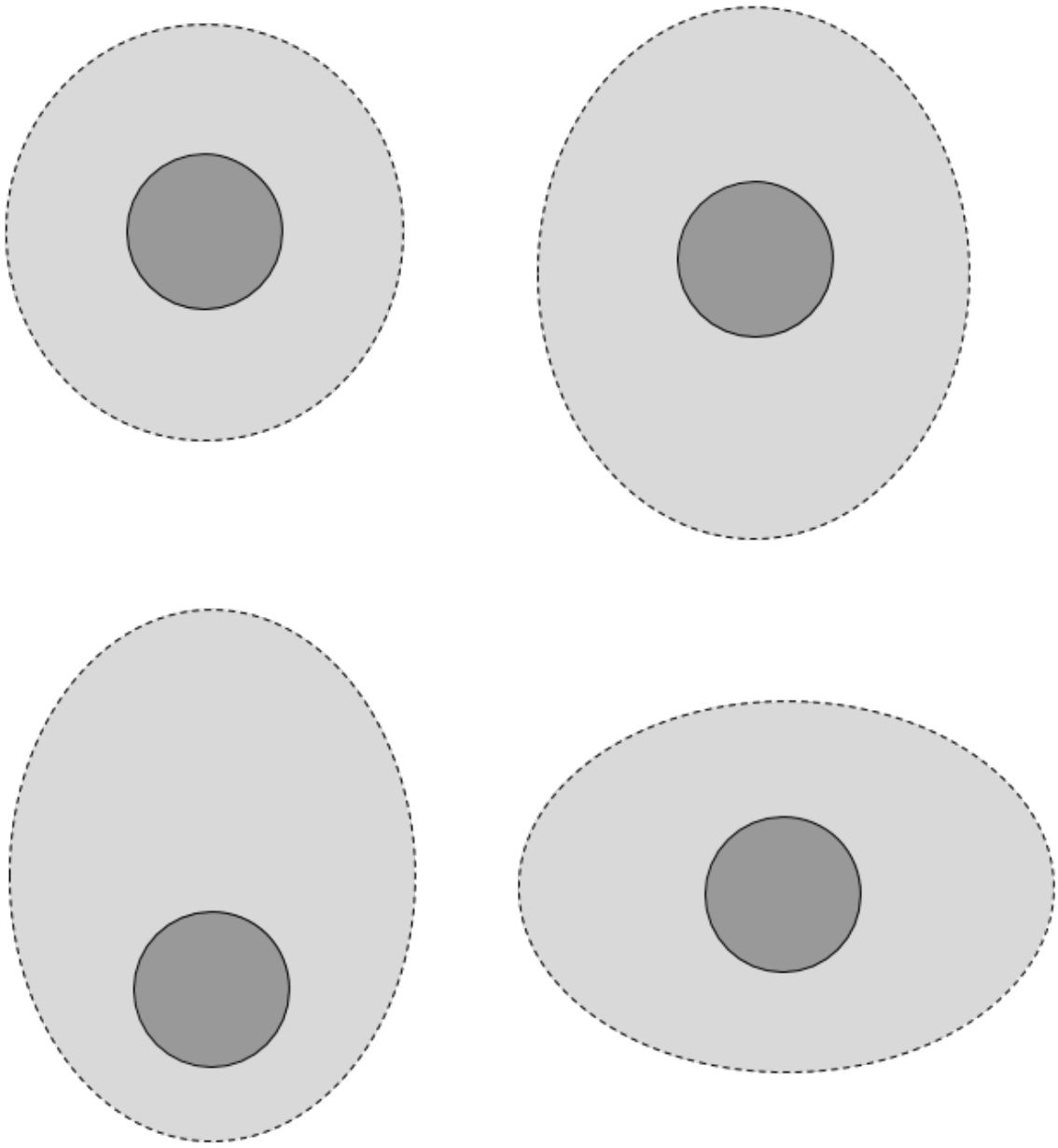

Figure 4.4.1: Visualizing shortcutters and cores within a clan. Each shortcutter (light gray) contains a core (dark gray) of vertices for whom the shortcutter is usable.

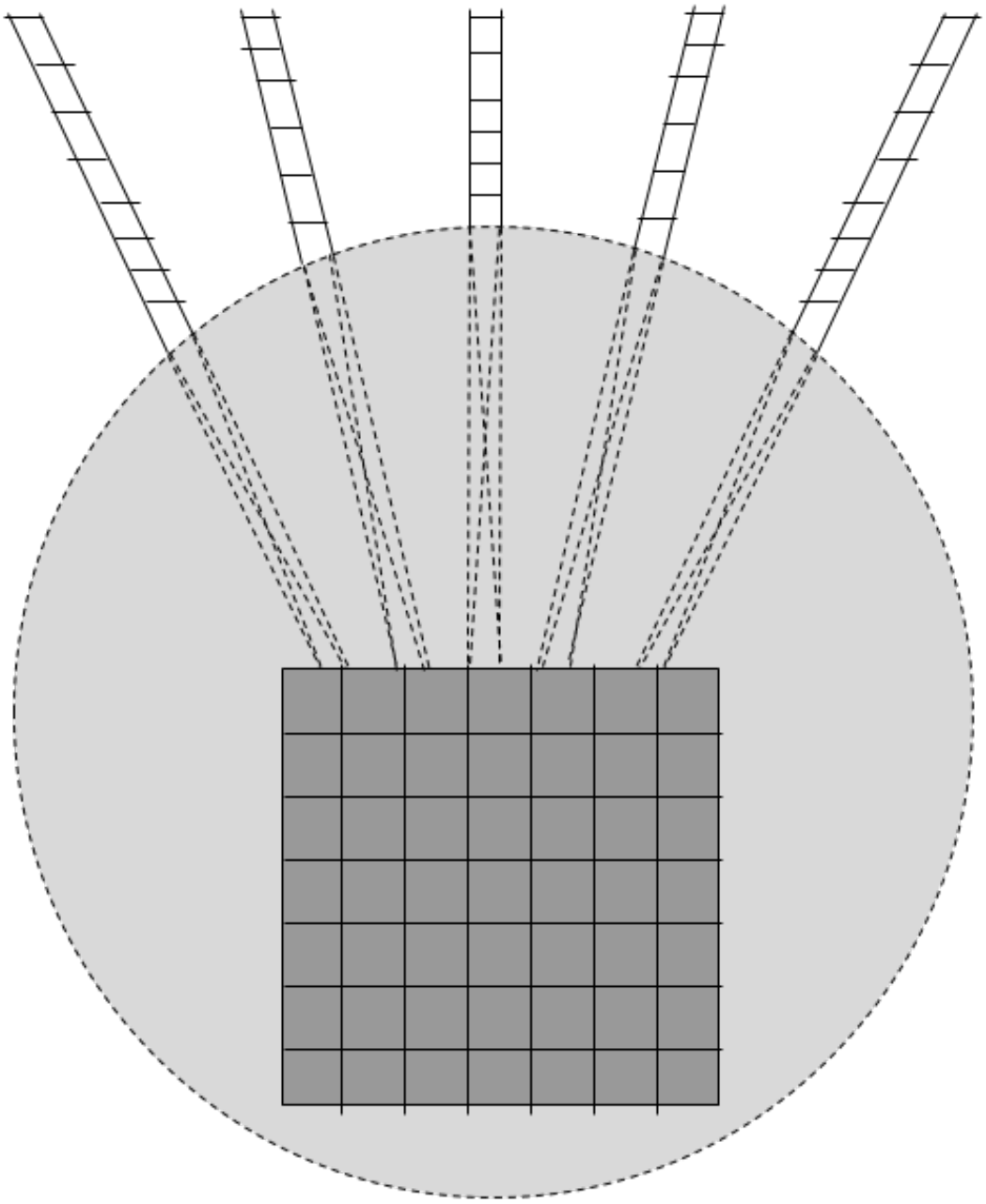

Figure 4.4.2: The conductivity of the light gray shortcutter is determined by the conductance of the dashed edges, which are obtained by Schur complementing all vertices in the shortcutter with its core (dark gray) removed. The conductance of these edges is relevant for assessing the quality of a shortcutter because (1) doing a random walk on a Schur complement is equivalent to doing a random walk on the original graph and removing all eliminated vertices from the visit list and (2) Lemma 4.2.3.

Type of work	Edge to charge to	Reason for charging
Walk across ∂P_i	P_i boundary edge	Normal random walk step
Shortcut to $\texttt{deleted}(\mathcal{C})$	P_i boundary edge	Deletion set condition
Shortcut to ∂S_{P_i}	An edge in $\texttt{Schur}(H_{\mathcal{C}}, C_{P_i} \cup V(H) \setminus S_{P_i})$	Theorem 4.2.4

Type of work	Conductance bound for work charged to a clan $\mathcal{C}$	Distance to unvisited vertex
Walk across ∂P_i	$\kappa m/(\alpha^{i/(\sigma_0+1)} r_{min})$ (boundedness)	$\alpha^{(i+1)/(\sigma_0+1)} r_{min}$ (carving, R-clan)
Shortcut to $\texttt{deleted}(\mathcal{C})$	$\kappa m/(\alpha^{i/(\sigma_0+1)} r_{min})$ (boundedness)	$\alpha^{(i+1)/(\sigma_0+1)} r_{min}$ (carving, R-clan)
Shortcut to ∂S_{P_i}	$\zeta m^{1/\sigma_1} s_{\mathcal{C}}/(\alpha^{i/(\sigma_0+1)} r_{min})$ (conductivity)	$\alpha^{(i+1)/(\sigma_0+1)} r_{min}$ (carving, R-clan)

Type of work	Total steps	Work per step	Total work
Walk across ∂P_i	$\kappa m \alpha^{1/(\sigma_0+1)}$ (Lemma 4.2.3)	$O(1)$	$\leq m^{1+o(1)} \alpha^{o(1)}$
Shortcut to $\texttt{deleted}(\mathcal{C})$	$\kappa m \alpha^{1/(\sigma_0+1)}$ (Lemma 4.2.3)	$\tilde{O}(1)$ (offline)	$\leq m^{1+o(1)} \alpha^{o(1)}$
Shortcut to ∂S_{P_i}	$\zeta m^{1/\sigma_1} s_{\mathcal{C}} \alpha^{1/(\sigma_0+1)}$ (Lemma 4.2.3)	$\tilde{O}(\frac{m}{s_{\mathcal{C}}})$ (online)	$\leq m^{1+o(1)} \alpha^{o(1)}$
Precomputation	$\tau m^{1/\sigma_1} s_{\mathcal{C}}$ (modifiedness)	$\tilde{O}(\frac{m}{s_{\mathcal{C}}})$	$\leq m^{1+o(1)}$

Table 4.4.1: Accounting for work during each sampling round

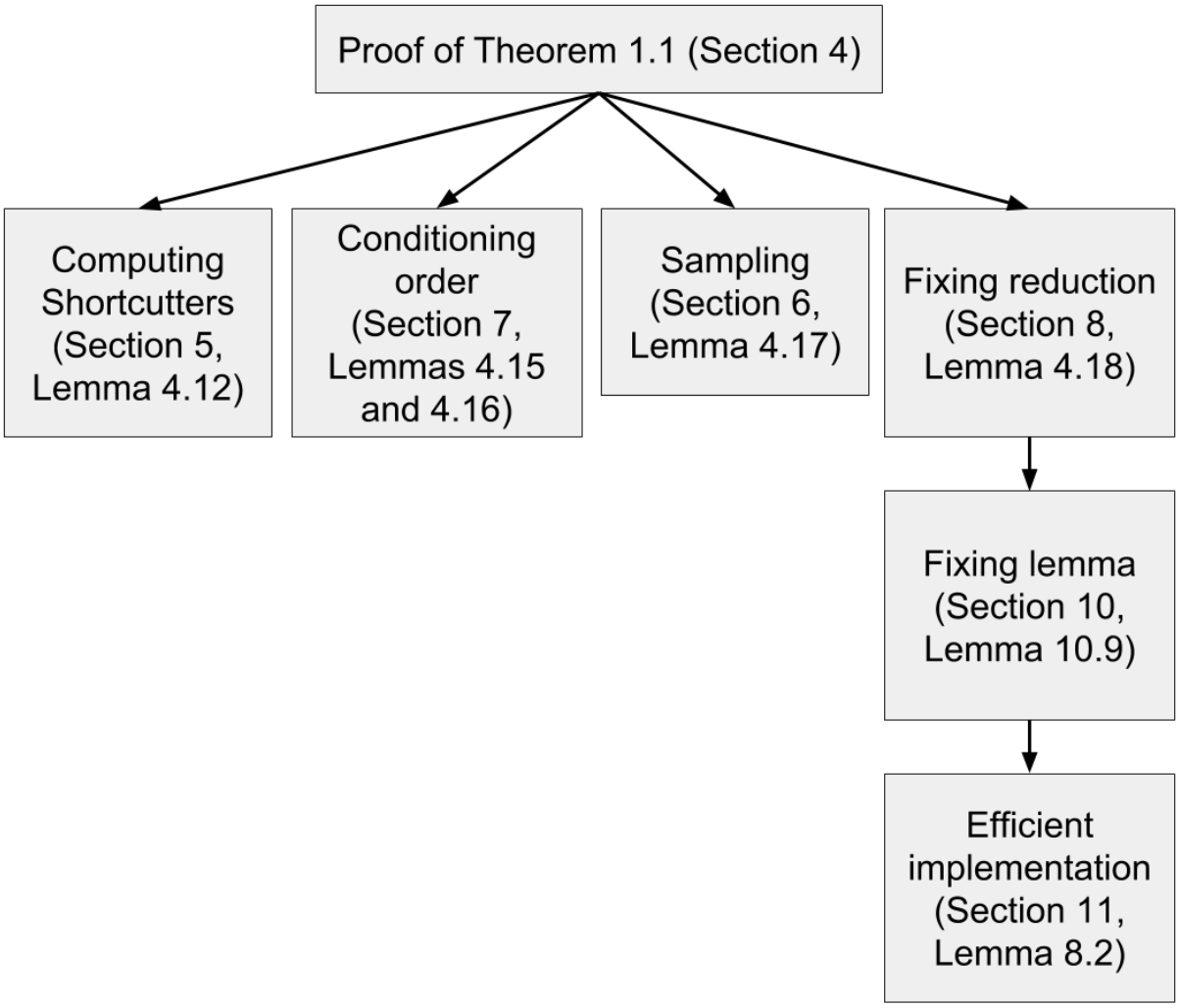

Figure 4.4.3: Organization of the paper. Each section only depends on its children in the tree.

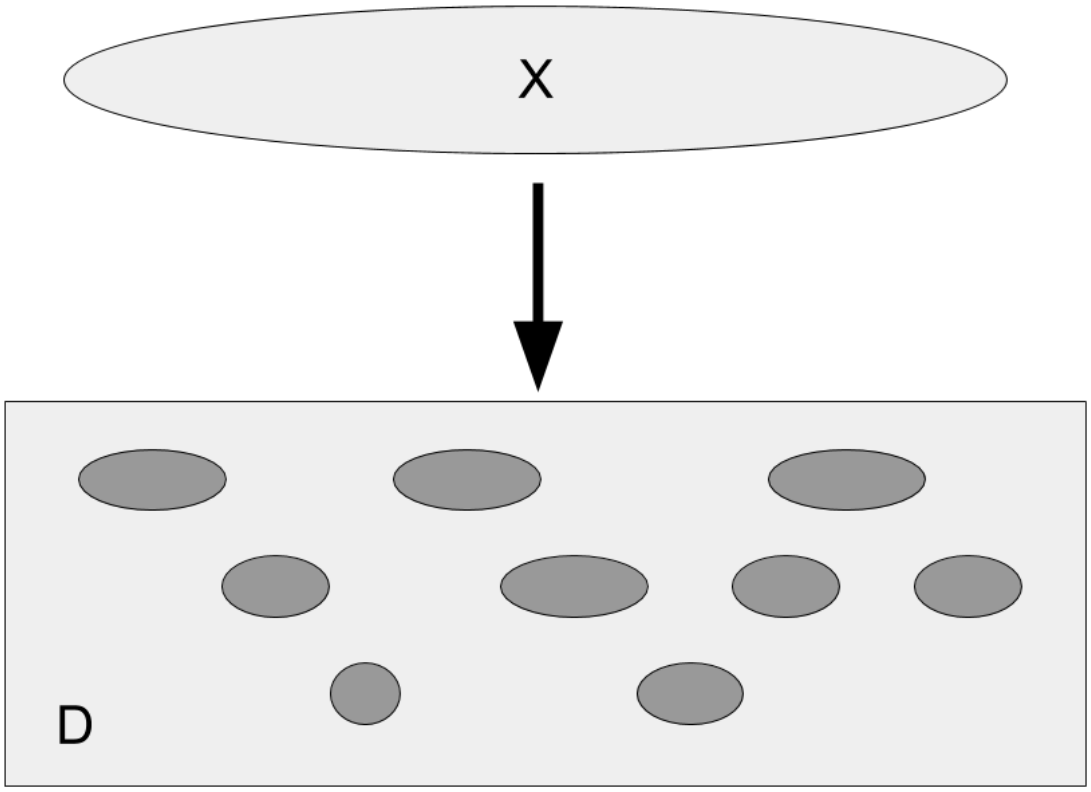

Figure 4.5.1: `CoveringCommunity` splits the set X into a small number of well-separated families of clusters in the effective resistance metric. `CoveringCommunity` is applied to each core of each clan independently. This splits each clan into $m^{o(1)}$ clans.

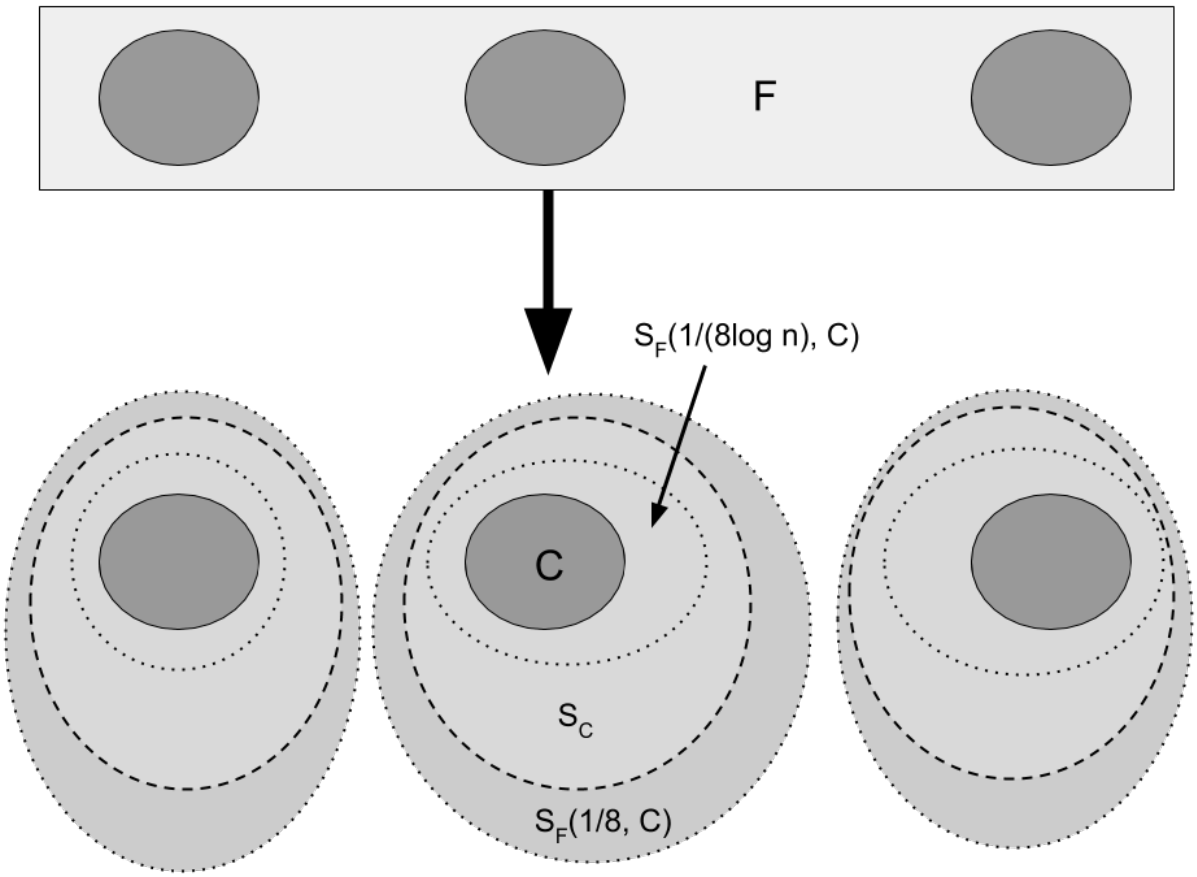

Figure 4.5.2: `Voronoi` produces shortcutters S_C that contain $S_{\mathcal{F}}(1/(8\log n), C)$ and are contained within $S_{\mathcal{F}}(1/8, C)$. This containment guarantee, along with the well-separatedness of $\mathcal{F}$, is translated into a bound on conductivity using Proposition 4.5.7 and Lemma 4.5.4.

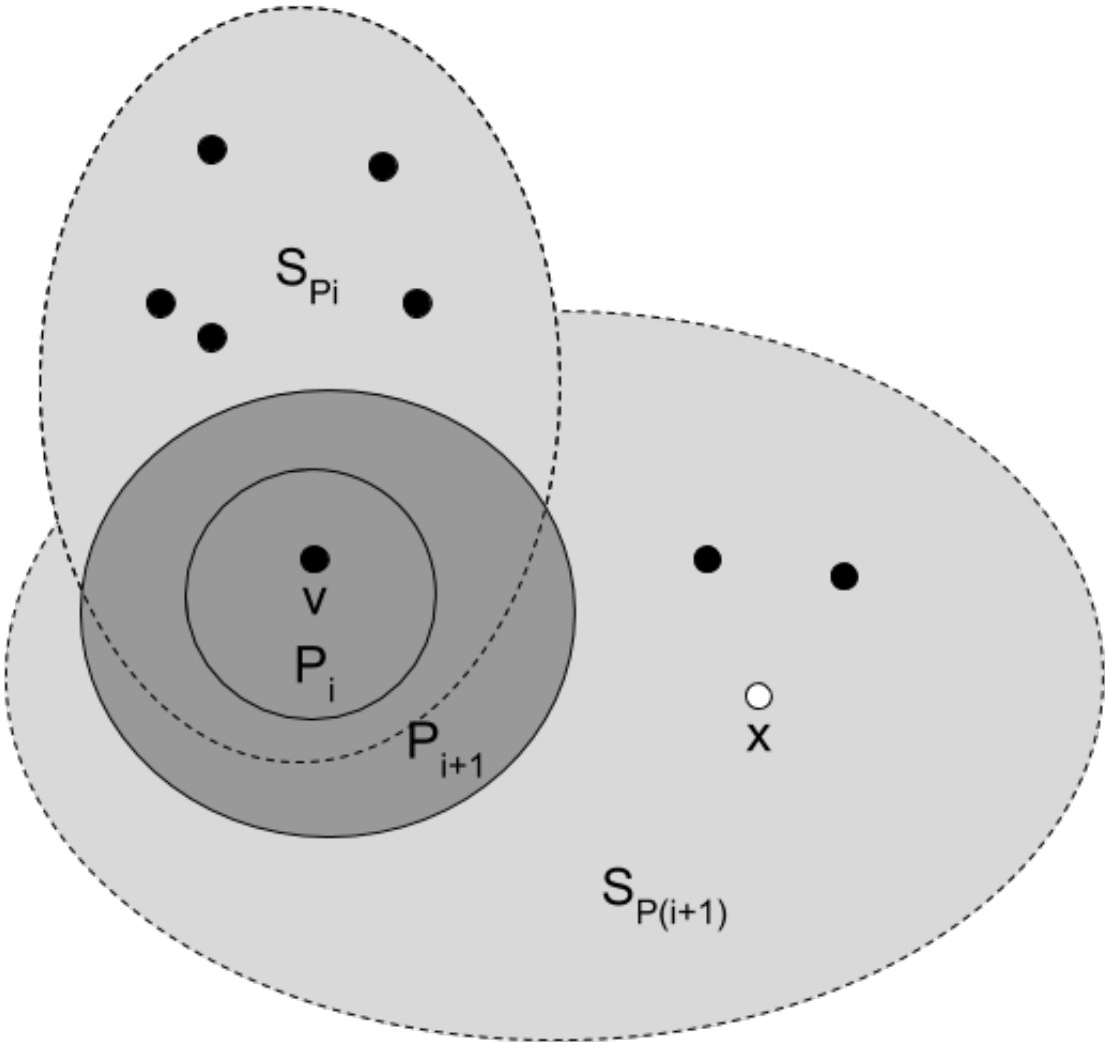

Figure 4.6.1: Runtime analysis of the shortcutted random walk in `PartialSample`. When S_{P_i} is used to shortcut a random walk starting at v, there is guaranteed to be an unvisited vertex x in $S_{P_{i+1}}$. By carving, x is within distance $\mu_{\text{carve}}\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ of v. Therefore, $R = \mu_{\text{carve}}\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ can be used when applying Lemma 4.2.3. $\alpha^{(i+1)/(\sigma_0+1)}r_{min}$ partially cancels with an $\alpha^{i/(\sigma_0+1)}r_{min}$ in the denominator of both the boundedness and conductivity of $\mathcal{E}_i$.

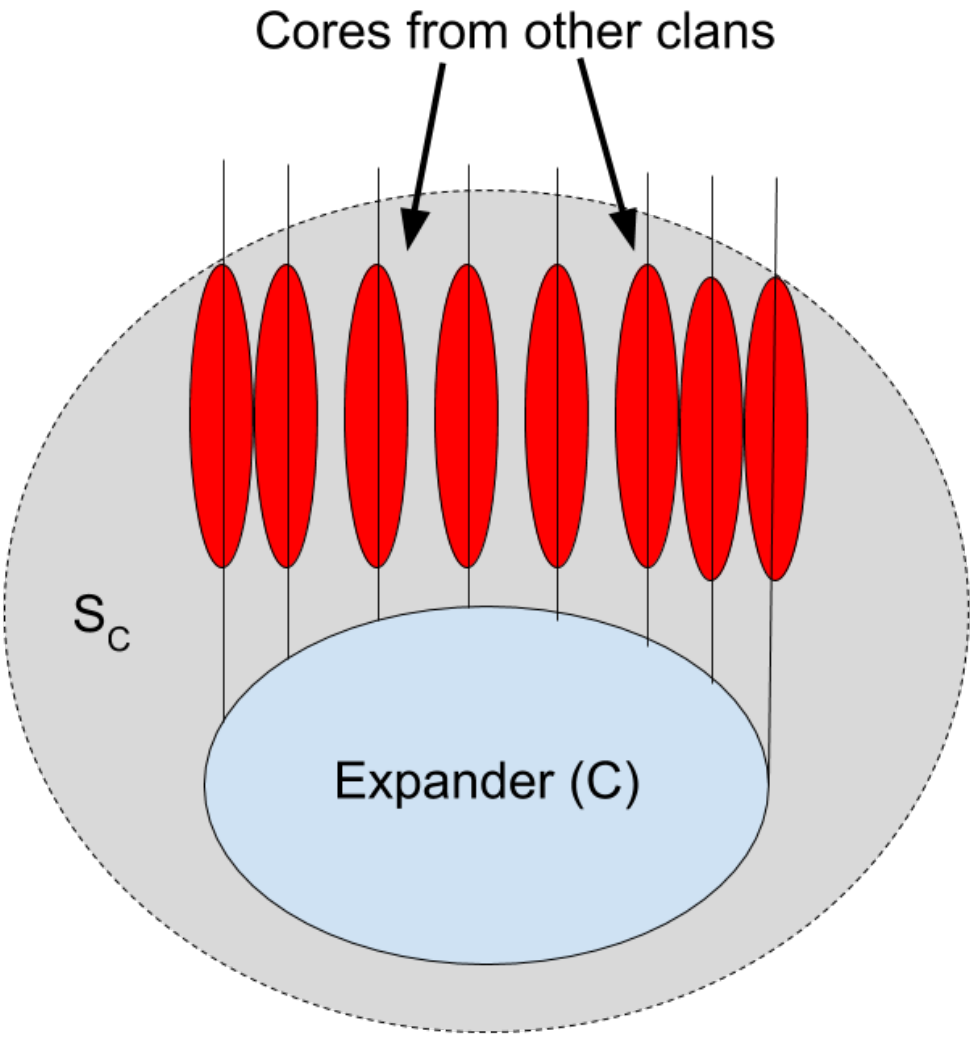

Figure 4.7.1: The situation that `SimpleConditioningVerts` is trying to avoid. Suppose that the shortcutter S_C makes up a large fraction of the graph and that `SimpleConditioningVerts` were to select the red cores to condition on first. Applying the algorithm in Lemma 4.4.15 could cause the conductance of S_C to increase untenably. Intuitively, this is because the shortcutter S_C will take the random walk from C to each of the deleted cores at least once. This is bad, though, as the deleted cores could correspond to very small shortcutters. As a result, there could be $\Theta(n)$ of them. S_C contains $\Theta(m)$ edges, so the work just due to this shortcutter is $\Theta(mn)$, which is quadratic! If one conditions on parts in C before the red cores, this issue does not come up because the large size of S_C ensures that there are not many of them.

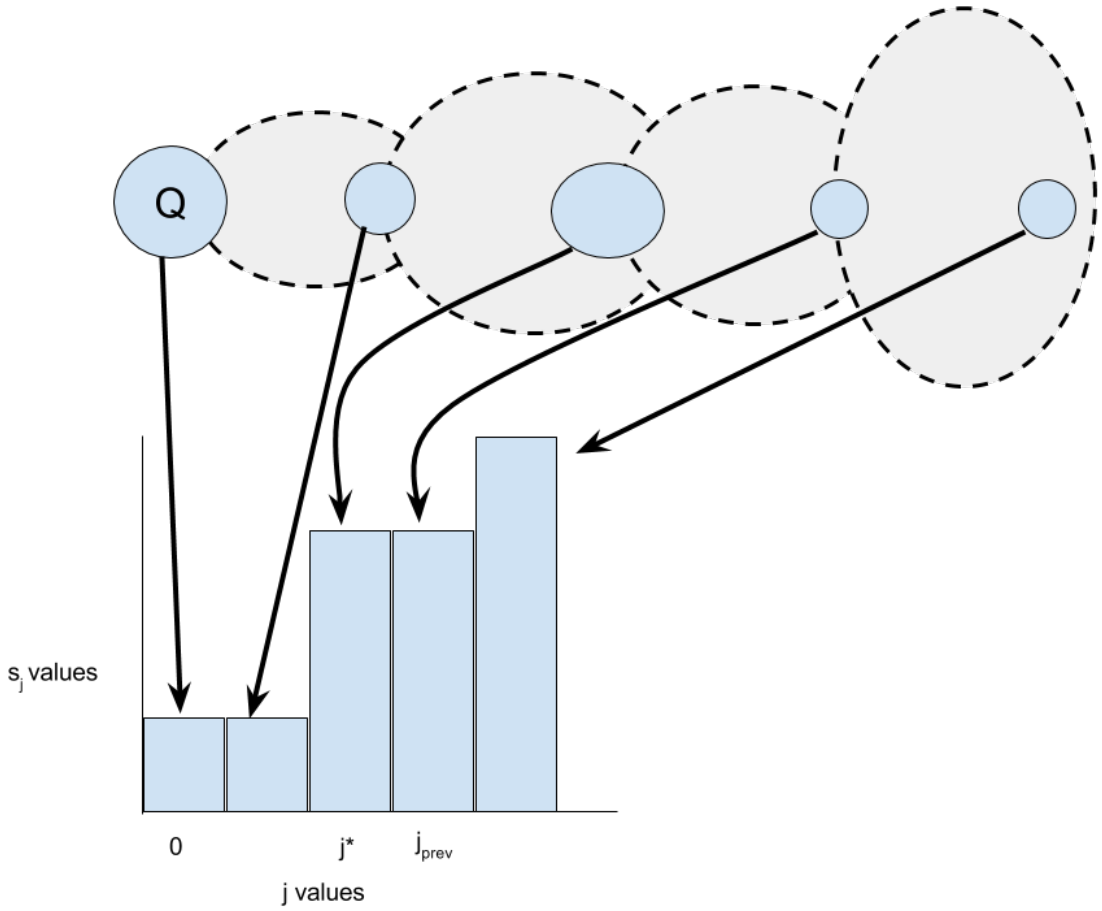

Figure 4.7.2: Constructing s_js from parts and their shortcutters in `ExtendHierarchy`. Our choice of j^* allows us to get rid of all parts that are closer to $\mathcal{Q}$ than j^* that have near-maximum size, thus bringing us one step closer to being able to condition on Q (one out of at most σ_1 steps.)

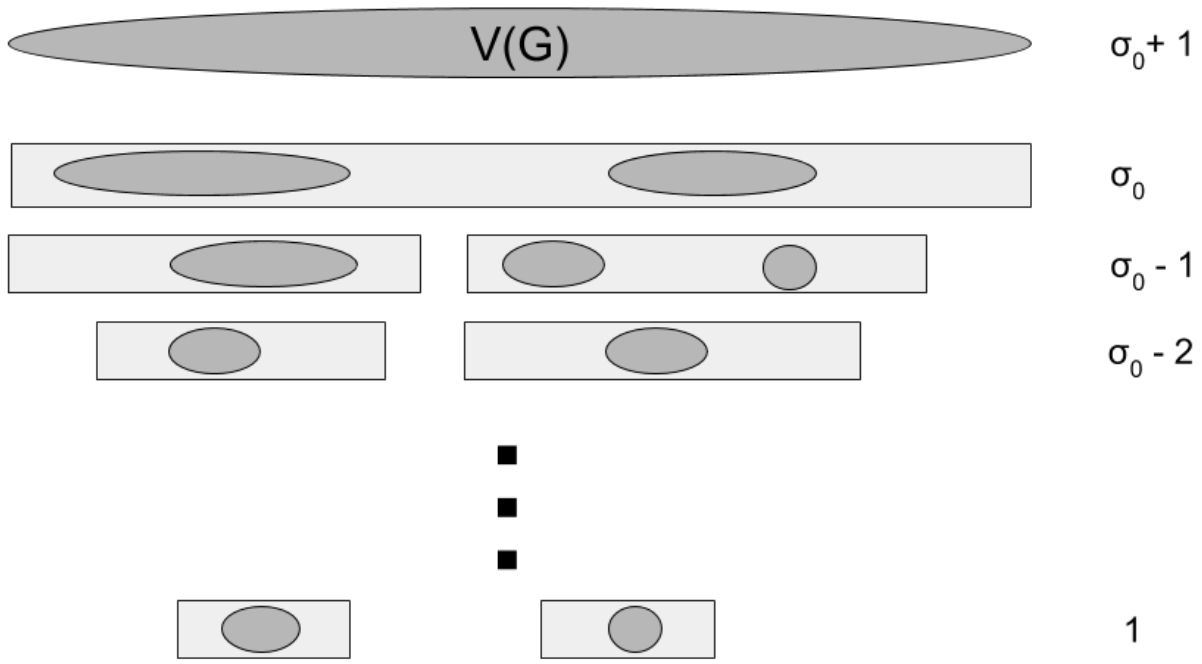

Figure 4.7.3: Typical $\mathcal{Q}_i$s (dark gray ovals) and $\mathcal{R}_i$s (light gray rectangle). Notice that the $\mathcal{R}_i$s are laminar and that they contain the $\mathcal{Q}_i$ parts from the level above.

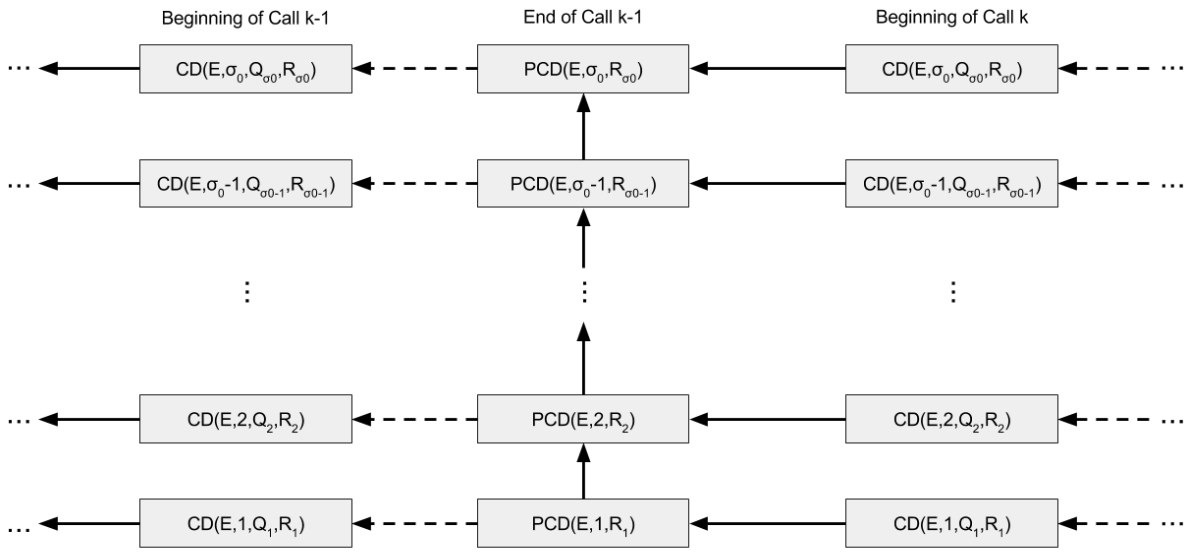

Figure 4.7.4: Containment relationships between conditioning digraphs. CD and PCD abbreviate `ConditioningDigraph` and `PermanentConditioningDigraph` respectively. A conditioning digraph I_0 is *contained* in another conditioning digraph I_1 if (a) each vertex (part) $P \in V(I_0)$ is entirely contained in a unique part $Q \in V(I_1)$ and (b) for any edge $(P, P') \in E(I_0)$, $(Q, Q') \in E(I_1)$, where Q and Q' are the unique containing parts for P and P'. Arrows point towards the containing graph. The dashed horizontal edges are present due to Lemma 4.7.17 and calls to `MakeNonedgesPermanent`. The solid horizontal edges are present due to Proposition 4.7.15. The vertical edges are present thanks to Proposition 4.7.13.

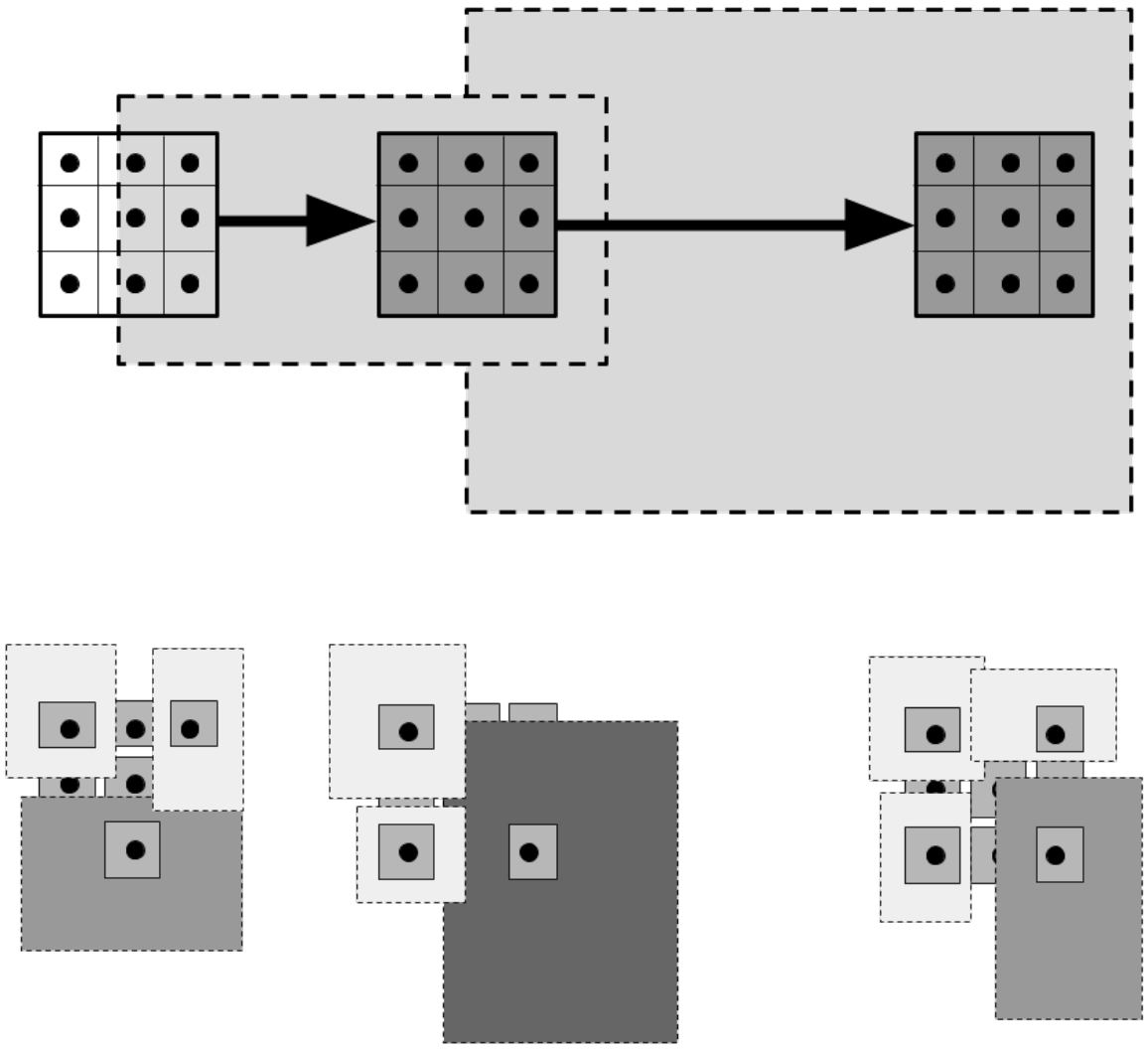

Figure 4.7.5: Picking parts to condition on in `ExtendHierarchy`. The top diagram depicts parts in $\mathcal{P}_{i+1}(\mathcal{E})$ (thick outline) with their shortcutters. All of the thick-outlined parts are in $\mathcal{R}$ and the leftmost one is also in $\mathcal{Q}$. The thick arrows depict directed edges in the graph $\texttt{ConditioningDigraph}(\mathcal{E}, i+1, \mathcal{Q}, \mathcal{R})$. The bottom diagram depicts parts in $\mathcal{P}_i(\mathcal{E})$. The three thick-outlined parts from left to right contain the thin-outlined parts in $\mathcal{Q}'_1, \mathcal{Q}'_2$, and $\mathcal{Q}'_3$ respectively. The bottom diagram depicts the shortcutters for some of those parts (the thin-outlined, light-gray dashed regions). The medium gray shortcutters determine s_j for each $\mathcal{Q}'_j$. The darkest shortcutter motivates the choice of $j^* = 2$ in this example.

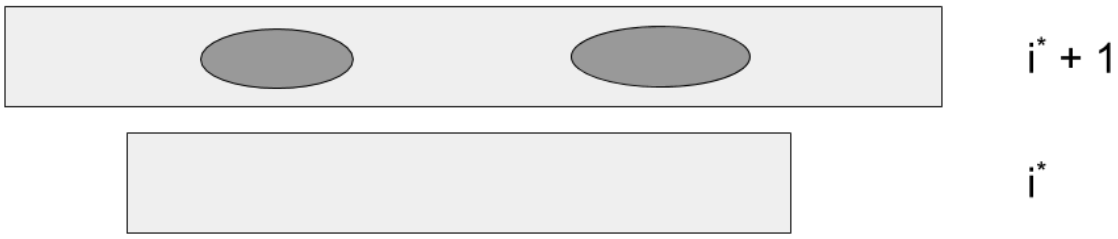

Figure 4.7.6: In the above diagram, $\mathcal{Q}_{i^*}$ is empty. Since $s_{i^*} = 0$ and $\mathcal{Q}_{i^*}$ contains all of the parts in $\mathcal{R}_{i^*}$ with nearly-largest shortcutters, all parts in the relevant set have size 0 (inactive) shortcutters. Therefore, any shortcutter in level i^* or below is either irrelevant or inactive. This means that carving is only required on levels $i^* + 1$ and higher, which has already been achieved.

4.8 Fixing reduction

In this section, we reduce Lemma 4.4.18 to a simpler result (Lemma 4.8.2).

Definition 4.8.1 (Schur complement conductance and degree)**.** *Consider a graph I. Let $J := \textbf{\textit{Schur}}(I, S \cup S')$, $J' := \textbf{\textit{Schur}}(I/S, S \cup S')$,*

$$c^I(S, S') := c^J(E_J(S, S'))$$

and

$$\Delta^I(S, S') := \sum_{e \in E_{J'}(S,S')} \textbf{\textit{Reff}}_{J'}(e) c_e^{J'}$$

Lemma 4.8.2. *There is an algorithm $F' \leftarrow \textbf{\textit{FastFix}}(I, J, D, S, S', F, \epsilon)$ that takes as input a graph I, a random graph J that is a valid sample from the distribution $I[F]$, a set $D \subseteq E(I)$, $S, S' \subseteq V(G)$, $F \subseteq E(I) \setminus D$, and an accuracy parameter $\epsilon \in (0, 1)$. With high probability on J, it outputs a set $F' \subseteq F$ with two properties:*

- *(Conductance) $c^{J \setminus D \setminus F'}(S, S') \leq (1 + \epsilon) c^{I \setminus D}(S, S')$*
- *(Size) $|F'| \leq \mu_{con}(\Delta^{I \setminus D}(S, S') + \Delta^{I \setminus D}(S', S) + |D|)\epsilon^{-3}$*

Furthermore, $\textbf{\textit{FastFix}}$ takes $m^{1+o(1)}$ time.

When trying to understand the statement of Lemma 4.8.2 for the first time, it is helpful to think about the case when $D = \emptyset$. The $D = \emptyset$ case is very similar to the general case. It is also helpful to disregard the J argument to `FastFix`. This argument is only provided for runtime purposes, since sampling from a tree is only efficient in our case due to `PartialSample`. Our reduction only uses $\epsilon = \Omega(1)$.

We prove this result in Section 4.10 and implement the almost-linear time algorithm for producing F' in Section 4.11. We illustrate many of the ideas behind the analysis of `FastFix` in Section 4.10.1.

4.8.1 Reduction from Lemma 4.4.18 to Lemma 4.8.2

Let $\mathcal{L}_i$ be the set of clusters with H-effective resistance diameter at most $\alpha^{i/(\sigma_0+1)} r_{min} \mu_{\text{carve}}$ for which $S \subseteq \cup_{C \in \mathcal{L}_i} C$. These sets are guaranteed to exist by the "Carved" condition of Lemma 4.4.18. Each of these sets of clusters maps to another set of clusters with $H_{\mathcal{C}}$-effective resistance diameter at most $\alpha^{i/(\sigma_0+1)} r_{min} \mu_{\text{carve}}$ for any clan $\mathcal{C} \in \mathcal{E}_i$. Call this set $\mathcal{L}_i^{\mathcal{C}}$. $\mathcal{L}_i^{\mathcal{C}}$ is not much larger than $\mathcal{L}_i$ by Lemma 4.7.2. We restate Lemma 4.7.2 here to emphasize that one does not need to read Section 4.7 before reading this section. It is proven in the appendix:

Lemma 4.7.2. *Consider a graph H and a set of clusters $\mathcal{D}$, each with effective resistance diameter at most R. Let F be a set of edges in H. Then there is a set of clusters $\mathcal{D}'$ with the following properties:*

- *(Covering) Each vertex in a cluster of $\mathcal{D}$ is in a cluster of $\mathcal{D}'$.*
- *(Diameter) The effective resistance diameter of each cluster in $\mathcal{D}'$ is at most $\mu_{app}R$ in the graph $H \setminus F$.*
- *(Number of clusters) $|\mathcal{D}'| \leq \mu_{app}(|\mathcal{D}| + |F|)$.*

We can now focus on the clans $\mathcal{C}$ in $\mathcal{E}_i$ independently. Consider each shortcutter $S_C \in \mathcal{C}$. S_C intersects some clusters in $\mathcal{L}_i^{\mathcal{C}}$. If clusters in $\mathcal{L}_i^{\mathcal{C}}$ contain a path from C to ∂S_C and that path is contracted after conditioning, the conductance of S_C becomes infinite. We want to avoid this situation. To do this, we could try to apply Lemma 4.8.2 with $S \leftarrow C$, $S' \leftarrow \partial S_C$, $I \leftarrow H$, $D \leftarrow \texttt{deleted}(\mathcal{C})$, $F \leftarrow F$, and $\epsilon \leftarrow 1/2$. While this ensures that the conductance of S_C does not increase much after deleting F', $\Delta^{H\setminus D}(S, S')$ could be very large. Our goal is for F' to have average size $m^{o(1)}$, where the average is over the shortcutters in $\mathcal{C}$.

To achieve this, instead of directly tempering S_C's conductance, it is useful to consider fixing the conductance between C and all clusters in $\mathcal{L}_i^{\mathcal{C}}$ that intersect S_C. One slight issue with this is that some of these clusters may be very close to C, resulting in the initial $C - \mathcal{L}_i^{\mathcal{C}}$ conductance being much higher than $1/\alpha^{i/(\sigma_0+1)}r_{min}$. This can be alleviated by fixing the conductance between C and the subclusters in $\mathcal{L}_i^{\mathcal{C}}$ restricted to $S_C \setminus S_{\{C,V(H)\setminus S_C\}}(1/4, C)$. $S_{\{C,V(H)\setminus S_C\}}(1/4, C)$ serves as "buffer space" between C and the restrictions of $\mathcal{L}_i^{\mathcal{C}}$, ensuring that the initial conductance is at most $\frac{m^{o(1)}}{\alpha^{i/(\sigma_0+1)}r_{min}}$. The fact that $\mathcal{L}_i^{\mathcal{C}}$ consists of a relatively small set of clusters ensures that $\Delta^{H\setminus D}(C, \cup_{C'\in\mathcal{L}_i^{\mathcal{C}}} C')$ is small. This ensures that there is a small set F' that, when deleted, nearly reverses the increase in the $C - \mathcal{L}_i^{\mathcal{C}}$ conductance due to conditioning.

However, we need to reverse the increase in S_C's conductance, not the $C-\mathcal{L}_i^{\mathcal{C}}$ conductance. S_C's conductance, though, can essentially be upper bounded by the sum of the $C - \mathcal{L}_i^{\mathcal{C}}$ conductance and the conductance of edges in the Schur complement with respect to $C \cup \mathcal{L}_i^{\mathcal{C}}$ that go directly from the boundary of the buffer zone $S_{\{C,V(H)\setminus S_C\}}(1/4, C)$ and ∂S_C. The latter conductance is not affected by conditioning, because all edges of F that are not in the buffer zone are covered by clusters in $\mathcal{L}_i^{\mathcal{C}}$. We have already argued that the former conductance is restored by deleting F'. Therefore, S_C's conductance only increases by a small constant factor over what it used to be.

We formalize the above intuition by implementing the reduction in `FixShortcutters`

using `FastFix`.

Algorithm 12: `FixShortcutters`$(\mathcal{E}, H', \mathcal{K})$

```
foreach i ∈ [σ_0] do
    foreach clan C ∈ E_i do
        S' ← ∅
        K_C ← the parts in K that intersect some S_C ∈ C
        foreach part P ∈ K_C do
            S' ← S' ∪ (P with all S_{C,V(H)\S_C}(1/4, C)s for cores C of shortcutters
            S_C ∈ C removed)
        deleted(C) ← deleted(C) ∪ FastFix(H, H', deleted(C), ∪_{S_C∈C} C, S', F, 1/4)
return E
```

Now, we prove that this algorithm has the desired effect. Before doing so, we state some useful facts about Schur complements and conductances that we prove in the appendix.

Lemma 4.8.3. *Consider any three disjoint sets of vertices $S_0, S_1, S_2 \subseteq V(I)$ and $S_0' \subseteq S_0$. Let $J = \texttt{Schur}(I, S_0 \cup S_1 \cup S_2)$ and $J' = \texttt{Schur}(I, S_0' \cup S_1 \cup S_2)$. Then*

$$c^{J'}(E_{J'}(S_0', S_1)) \leq c^J(E_J(S_0, S_1))$$

Lemma 4.8.4. *Consider any two disjoint sets $S_0, S_1 \subseteq V(I)$ with $S_0' \subseteq S_0$. Then $c^I(S_0', S_1) \leq c^I(S_0, S_1)$.*

Lemma 4.8.5. *For any cluster S_C in a graph I and any $p \in (0, 1)$,*

$$c^I(C, V(I) \setminus S_{\{C, V(I)\setminus S_C\}}(p, C)) \leq \frac{c^I(C, V(I) \setminus S_C)}{p}$$

Lemma 4.8.6. *Consider a graph I with two clusters C_1 and C_2 with two properties:*

- *The I-effective resistance diameters of C_1 and C_2 are both at most R.*
- *The minimum effective resistance between a vertex in C_1 and a vertex in C_2 is at least γR for $\gamma > 4$.*

Let J be the graph with C_1 and C_2 identified to s and t respectively. Then $\texttt{Reff}_J(s, t) \geq (\gamma - 4)R$.

Lemma 4.4.18. *Let H be a graph, $\mathcal{E}$ be an empire in H and $\mathcal{K}$ be a set of parts. Let $S = \cup_{P\in\mathcal{K}} P$ and let $F = \cup_{P\in\mathcal{K}} E(P)$. Let $H' \sim H[F]$. Suppose that the following input conditions hold $\mathcal{E}$:*

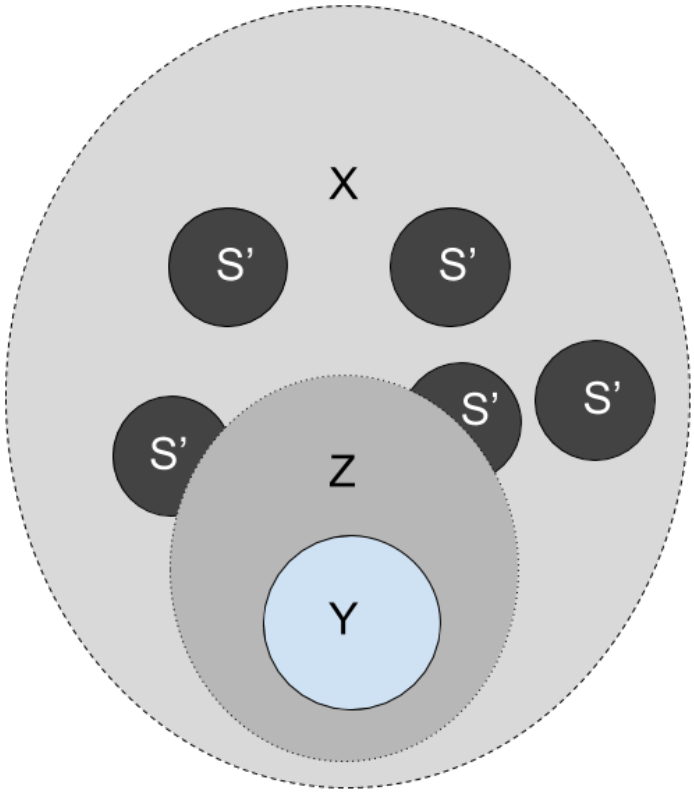

Figure 4.8.1: The reduction for one shortcutter in some clan. All edges of F are between two vertices of S', two vertices of Z, or one vertex in S' and one in Z. The edges between two vertices in S' are irrelevant, as Z is primarily responsible for certifying that X has low conductance after conditioning. The edges between one vertex in S' and one vertex in Z do not affect the $(V(H) \setminus X) - Z$ direct conductance. The edges in F with both edges in Z can be in F'.

- *(Bucketing) The empire $\mathcal{E}$ is bucketed.*
- *(Carving) $\mathcal{E}$ is carved with respect to S.*

With high probability over H', `FixShortcutters`$(\mathcal{E}, H', \mathcal{K})$ adds edges to the deletion set of each clan of $\mathcal{E}$ to obtain a set of covering hordes $\{\mathcal{H}'_i\}$ with the following properties:

- *(Boundedness) For each i, if $\mathcal{E}_i$ is κ-bounded, then $\mathcal{H}'_i$ is $\ell\kappa$-bounded, where $\ell = \sum_{i=1}^{\sigma_0} |\mathcal{E}_i|$.*
- *(Modifiedness and deletion set condition) For each i, if $\mathcal{E}_i$ is τ-modified and satisfies the deletion set condition, then $\mathcal{H}'_i$ is $\mu_{mod}(\tau+\zeta)$-modified and also satisfies the deletion set condition.*
- *(Conductivity) For each i, if $\mathcal{E}_i$ is ζ-conductive with respect to H, then $\mathcal{H}'_i$ is at most 7ζ-conductive with respect to H'.*

Futhermore, it does so in $m^{1+o(1)}$ time.

Proof of Lemma 4.4.18 given Lemma 4.8.2. **Boundedness.** No part in $\mathcal{K} \cap (\cup_{k\leq i}\mathcal{P}_k(\mathcal{E}))$ intersects the boundary of a core C in some clan of $\mathcal{E}_i$, as C is part of the refinement used to define $\mathcal{P}_k(\mathcal{E})$. Therefore, each H'-boundary edge of the image C' of C in H' is either a boundary edge of some part in $\mathcal{K} \cap (\cup_{k>i}\mathcal{P}_k(\mathcal{E}))$ or is a boundary edge of C. The total conductance of such edges is at most $\ell\kappa m/(\alpha^{i/(\sigma_0+1)}r_{min})$, as desired.

Conductivity. Consider each clan $\mathcal{C}$ in isolation and consider the set S' generated for the clan $\mathcal{C}$. Let $X = \cup_{S_{C'}\in\mathcal{C}} S_{C'}$ and $Y = \cup_{S_{C'}\in\mathcal{C}} C'$.

We start by bounding the $Y - S'$ conductance before conditioning. This is depicted in Figure 4.8.2. By definition of S' and the fact that the S_Cs are vertex disjoint, S' does not intersect $Z = S_{\{Y,V(H)\setminus X\}}(1/4, Y)$. Therefore, by Lemma 4.8.4 with $S_0 \leftarrow S', S'_0 \leftarrow V(H)\setminus Z$, and $S_1 \leftarrow Y$,

$$c^{H_C}(S', Y) \leq c^{H_C}(V(H) \setminus Z, Y)$$

By Lemma 4.8.5 with $C \leftarrow Y$, $S_C \leftarrow X$ and $p \leftarrow 1/4$ and the definition of Z,

$$c^{H_C}(V(H) \setminus Z, Y) \leq 4c^{H_C}(X, Y)$$

Since $\mathcal{C}$ was ζ-conductive before conditioning,

$$c^{H_C}(X, Y) \leq \frac{\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{\alpha^{i/(\sigma_0+1)} r_{min}}$$

Combining these inequalities shows that

$$c^{H_C}(S', Y) \leq \frac{4\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{\alpha^{i/(\sigma_0+1)} r_{min}}$$

Let $H'_{\mathcal{C}}$ be the graph obtained by deleting $\texttt{deleted}(\mathcal{C})$ from H' after adding the `FastFix` edges. By the "Conductance" guarantee of Lemma 4.8.2,

$$c^{H'_{\mathcal{C}}}(S', Y) \leq \frac{5\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{\alpha^{i/(\sigma_0+1)} r_{min}}$$

We have now finished bounding the $Y - S'$ conductance after conditioning. Now, we bound the $Y - (V(H) \setminus X)$ conductance after conditioning. This is depicted in Figure 4.8.3. By definition, $Y \subseteq Z$. Notice that all edges in F have endpoints in $S' \cup Z$. As a result, the direct Z to $V(H) \setminus X$ conductance does not depend on F:

$$c^{\texttt{Schur}(H'_{\mathcal{C}}, Z \cup S' \cup (V(H) \setminus X))}(E(Z, V(H) \cup X)) = c^{\texttt{Schur}(H_{\mathcal{C}}, Z \cup S' \cup (V(H) \setminus X))}(E(Z, V(H) \cup X))$$

The graph subscript for E is eliminated here for clarity, as the graph is the same as the Schur complement given in the superscript. Apply Lemma 4.8.3 with $I \leftarrow H'_{\mathcal{C}}$, $S_0 \leftarrow Z$, $S'_0 \leftarrow Y$, $S_1 \leftarrow V(H) \setminus X$, and $S_2 \leftarrow S'$ to conclude that

$$\begin{aligned} c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))}(E(Y, V(H) \cup X)) &\leq c^{\texttt{Schur}(H'_{\mathcal{C}}, Z \cup S' \cup (V(H) \setminus X))}(E(Z, V(H) \cup X)) \\ &= c^{\texttt{Schur}(H_{\mathcal{C}}, Z \cup S' \cup (V(H) \setminus X))}(E(Z, V(H) \cup X)) \\ &\leq \frac{4\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{3\alpha^{i/(\sigma_0+1)} r_{min}} \end{aligned}$$

where the last inequality follows from the fact that $X \setminus Z = S_{\{Y, V(H) \setminus X\}}(3/4, V(H) \setminus X)$, Lemma 4.8.5, and the ζ-conductivity of $\mathcal{C}$ in the graph $H_{\mathcal{C}}$.

Now, we have bounds on the direct $Y - V(H) \setminus X$ and $Y - S'$ conductances in $H'_{\mathcal{C}}$. We now eliminate S' to obtain the desired bound on the $Y - (V(H) \setminus X)$ conductance in $H'_{\mathcal{C}}$. This is depicted in Figure 4.8.4. Start by applying Lemma 4.8.3 with $I \leftarrow H'_{\mathcal{C}}$, $S_0 \leftarrow (V(H) \setminus X) \cup S'$, $S'_0 \leftarrow V(H) \setminus X$, $S_1 \leftarrow Y$, and $S_2 \leftarrow \emptyset$ to show that

$$\begin{aligned} c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup (V(H) \setminus X))}(E(Y, V(H) \setminus X)) &\leq c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))}(E(Y, S' \cup (V(H) \setminus X))) \\ &\leq c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))}(E(Y, S')) \\ &+ c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))}(E(Y, (V(H) \setminus X))) \\ &\leq c^{H'_{\mathcal{C}}}(Y, S') + c^{\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))}(E(Y, (V(H) \setminus X))) \\ &\leq \frac{19\zeta m^{1/\sigma_1} s_{\mathcal{C}}}{3\alpha^{i/(\sigma_0+1)} r_{min}} \end{aligned}$$

This is precisely saying that $\mathcal{C}$ is $\frac{19\zeta}{3}$-conductive after running `FixShortcutters`, as desired.

Modifiedness and deletion set condition. We start with the deletion set condition. Consider any part $P \in \mathcal{P}_i(\mathcal{E})$ for some $i \in [\sigma_0]$. By the laminarity of the overlay partition, either $P \subseteq Q$ for some $Q \in \mathcal{K}$ or P does not intersect any part in $\mathcal{K}$. In the former case, conditioning on all parts in $\mathcal{K}$ makes $E_{H'}(P) = \emptyset$ and the deletion set condition does not need to apply for P. In the latter case, F does not intersect $E_H(P) \cup \partial_H P$. Since `FastFix` adds a subset of F to each $\texttt{deleted}(\mathcal{C}')$ for any clan $\mathcal{C}'$ in the empire $\mathcal{E}$, the deletion set condition remains satisfied for the part P, as desired.

Now, we move onto modifiedness. To apply Lemma 4.8.2, we just need to bound $\Delta^{H_\mathcal{C}}(Y, S')$, $\Delta^{H_\mathcal{C}}(S', Y)$, and $|D|$. The reasoning for the first two bounds is very similar. By the modifiedness of the input, $|D| \leq \tau m^{1/\sigma_1} s_\mathcal{C}$, so it suffices to bound the first two quantities.

We start with bounding $\Delta^{H_\mathcal{C}}(S', Y)$. Consider the graph $H_\mathcal{C}/S'$ with the vertices S' identified to one vertex s. By Rayleigh monotonicity, cores of shortcutters in $\mathcal{C}$ have $H_\mathcal{C}/S'$ effective resistance diameter at most $\mu_{\text{rad}} \alpha^{i/(\sigma_0+1)} r_{min} \leq \mu_{\text{carve}} \mu_{\text{app}} \alpha^{i/(\sigma_0+1)} r_{min} := R$. Break the cores of shortcutters in $\mathcal{C}$ up into two sets $\mathcal{C}_{near}$ and $\mathcal{C}_{far}$, with these sets being the cores in $\mathcal{C}$ with and without respectively a vertex with $H_\mathcal{C}/S'$ effective resistance distance from s at most $3R$. We bound the contribution to the degree of $\mathcal{C}_{near}$ and $\mathcal{C}_{far}$ independently.

First, we bound the $\mathcal{C}_{near}$ contribution. Every vertex in $\mathcal{C}_{near}$ clusters is within $H_\mathcal{C}/S'$ effective resistance distance $4R$ of s by the triangle inequality. Recall that Z separates the $\mathcal{C}_{near}$ clusters from s. Therefore, by Lemma 4.8.3, Lemma 4.8.5 with $p = 1/4$, and ζ-conductivity before conditioning,

$$c^{H_\mathcal{C}/S'}(s, \cup_{C \in \mathcal{C}_{near}} C) \leq \frac{4\zeta m^{1/\sigma_1} s_\mathcal{C}}{\alpha^{i/(\sigma_0+1)} r_{min}}$$

Let $K = \texttt{Schur}(H_\mathcal{C}/S', \{s\} \cup (\cup_{C \in \mathcal{C}} C))$. Then,

$$\begin{aligned}\sum_{e \in E_K(s, \cup_{C \in \mathcal{C}_{near}} C)} \texttt{Reff}_K(e) c_e^K &\leq 4R c^K(E_K(s, \cup_{C \in \mathcal{C}_{near}} C)) \\ &\leq 16 \mu_{\text{carve}} \zeta m^{1/\sigma_1} s_\mathcal{C}\end{aligned}$$

Now, we bound the $\mathcal{C}_{far}$ contribution. To do this, we exploit Lemma 4.8.6 on each cluster. In particular, for each cluster $C \in \mathcal{C}_{far}$ with a vertex at distance exactly γ_C from s, Lemma 4.8.6 implies that

$$c^K(s, C) \leq \frac{1}{(\gamma_C - 4)R}$$

By the triangle inequality, all vertices have K-effective resistance distance at most $(\gamma_C + 1)R$ from s. Therefore, they contribute at most $(\gamma_C + 1)/(\gamma_C - 4) \geq (5+1)/(5-4) = 6$ to the degree each. By bucketing, there are at most $4s_\mathcal{C}$ clusters in $\mathcal{C}$. Therefore,

$$\sum_{e \in E_K(s, \cup_{C \in \mathcal{C}_{far}} C)} \texttt{Reff}_K(e) c_e^K \leq 24 s_{\mathcal{C}}$$

We have now shown that

$$\Delta^{H_{\mathcal{C}}}(S', Y) \leq (24 + 24\zeta \mu_{\text{carve}} m^{1/\sigma_1}) s_{\mathcal{C}}$$

Next, we bound $\Delta^{H_{\mathcal{C}}}(Y, S')$. Since the shortcutters of $\mathcal{C}$ are carved with respect to S' and all parts in S' intersect some shortcutter of $\mathcal{C}$ (by Line 4), S' can be written as a union of $|\mathcal{C}|$ clusters with H-effective resistance diameter at most $\mu_{\text{carve}} \alpha^{i/(\sigma_0+1)} r_{min}$ given by a $\mu_{\text{carve}} \alpha^{i/(\sigma_0+1)} r_{min}$ H-effective resistance ball around each core in $\mathcal{C}$. Let $\mathcal{L}$ be this set. By the bucketing of $\mathcal{C}$, $|\mathcal{L}| \leq 4 s_{\mathcal{C}}$. By Lemma 4.7.2, the clusters in $\mathcal{L}$ can be covered by a set $\mathcal{L}'$ with size $|\mathcal{L}'| \leq \mu_{\text{app}}(|\mathcal{L}| + |\texttt{deleted}(\mathcal{C})|) \leq \mu_{\text{app}}(4 + \tau m^{1/\sigma_1}) s_{\mathcal{C}}$ of clusters with $H_{\mathcal{C}}$ effective resistance diameter at most $\mu_{\text{app}} \mu_{\text{carve}} \alpha^{i/(\sigma_0+1)} r_{min} \leq R$.

Split $\mathcal{L}'$ into two sets $\mathcal{L}_{near}$ and $\mathcal{L}_{far}$, based on whether or not they contain a vertex within $H_{\mathcal{C}}/Y$ distance $3R$ of the identification t of Y in $H_{\mathcal{C}}/Y$. Applying the same argument as above with s replaced with t and the core set of $\mathcal{C}$ replaced with $\mathcal{L}'$ shows that

$$\Delta^{H_{\mathcal{C}}}(Y, S') \leq 24|\mathcal{L}_{far}| + 24\mu_{\text{app}}\mu_{\text{carve}}(\zeta m^{1/\sigma_1} s_{\mathcal{C}})$$

As discussed after defining $\mathcal{L}'$,

$$|L_{far}| \leq |\mathcal{L}'| \leq \mu_{\text{app}}(4 + \tau m^{1/\sigma_1}) s_{\mathcal{C}}$$

Plugging this bound in shows that $\Delta^{H_{\mathcal{C}}/Y}(Y, S') \leq 100\mu_{\text{app}}(\mu_{\text{carve}}\zeta + \tau) m^{1/\sigma_1} s_{\mathcal{C}}$. By Lemma 4.8.2, $\mathcal{C}$ is $200\mu_{\text{app}}\mu_{\text{carve}}\mu_{\text{con}}(\zeta+\tau) = \mu_{\text{mod}}(\zeta+\tau)$-modified after applying `FixShortcutters`, as desired.

Runtime. `FixShortcutters` does a linear amount of work and one call to `FastFix` for each clan. Since there are at most $\ell_{\max} \leq m^{o(1)}$ clans, the total runtime is almost-linear.

□

4.9 Conductance concentration inequality (fixing lemma) preliminaries

Now, we give preliminaries for Lemmas 4.10.1 and 4.8.2.

4.9.1 Linear algebraic descriptions of conductances and rank one updates

For the rest of the paper, it will be helpful to think about the quantities $c^I(S, S')$ and $\Delta^I(S, S')$ linear-algebraically. We start with $c^I(S, S')$:

Proposition 4.9.1.

$$c^I(S, S') = \frac{1}{b_{ss'}^T L_{I/(S,S')}^+ b_{ss'}}$$

Proof. Since all edges accounted for in $c^I(S, S')$ go directly between S and S' in $\texttt{Schur}(I, S \cup S')$, $c^{I/(S,S')}(s, s') = c^I(S, S')$. $c^{I/(S,S')}(s, s')$ is the conductance of the one edge between s and s' in the graph $\texttt{Schur}(I/(S, S'), \{s, s'\})$. The conductance of this edge is the reciprocal of its resistance. By commutativity of Schur complements with identification, this edge's resistance is $b_{ss'}^T L_{I/(S,S')}^+ b_{ss'}$, as desired. □

Now, we interpret $\Delta^I(S, S')$ linear-algebraically. As discussed in Section 4.3, the effective resistance between vertices x and y in a graph I is just $b_{xy}^T L_I^+ b_{xy}$. Therefore, we just need to interpret condunctances between a vertex s and a set of vertices S'. Proposition 4.9.1 gives us a way of thinking about the total conductance of edges between s and S', so it suffices to describe the normalized conductances. It turns out that these conductances can be computed using one electrical flow:

Proposition 4.9.2. *Consider a graph I with a vertex s and a set $S' \subseteq V(I)$. Let $J \leftarrow \texttt{Schur}(I, \{s\} \cup S')$. Let $K = I/S'$ with S' identified to a vertex s'. For each vertex $w \in S'$,*

$$\frac{c_{sw}^J}{c^J(s, S')} = \sum_{e \in \partial_I w} (b_{ss'}^T L_K^+ b_e) c_e^K$$

Proof. By Theorem 4.3.2,

$$\sum_{e \in \partial_I w} (b_{ss'}^T L_K^+ b_e) c_e^K = \Pr_s[t_w < t_{S' \setminus \{w\}}]$$

The random walk in the above expression is done in the graph I. By Theorem 4.2.4,

$$\Pr_s[t_w < t_{S' \setminus \{w\}}] = \frac{c_{sw}^J}{c^J(s, S')}$$

Combining these equalities yields the desired result. □

We now characterize how conductances and effective resistances change after deleting single edges.

Definition 4.9.3 (Leverage and nonleverage scores)**.** *For any graph I and $e \in E(I)$, let*

$$\texttt{lev}_I(e) := c_e^I \texttt{Reff}_I(e)$$

and

$$\texttt{nonlev}_I(e) := 1 - c_e^I \texttt{Reff}_I(e)$$

The following bounds follow immediately from the Sherman-Morrison rank one update formula ([87]).

Proposition 4.9.4. *Let I be a graph. Consider two vertices $s, t \in V(I)$, an edge f, and a demand vector $d \in \mathbb{R}^{V(I)}$. Then*

$$b_{st}^T L_{I\setminus f}^+ d = b_{st}^T L_I^+ d + \frac{(b_{st}^T L_I^+ b_f)(b_f^T L_I^+ d)}{r_f(1 - \texttt{lev}_I(f))}$$

Proposition 4.9.5. *Let I be a graph. Consider two vertices $s, t \in V(I)$, an edge f, and a demand vector $d \in \mathbb{R}^{V(I)}$. Then*

$$b_{st}^T L_{I/f}^+ d = b_{st}^T L_I^+ d - \frac{(b_{st}^T L_I^+ b_f)(b_f^T L_I^+ d)}{r_f \texttt{lev}_I(f)}$$

4.9.2 Splitting edges

In Propositions 4.9.4 and 4.9.5, the dependencies on $\texttt{nonlev}_I(f)$ and $\texttt{lev}_I(f)$ are inconvenient. One can mitigate this by *splitting* edges in one of two ways. When an edge has low leverage score, it should be split *in series* by replacing it with two edges that have half the resistance. When an edge has high leverage score, it should be split *in parallel* by replacing it with two edges that have double the resistance. For an edge $e \in E(I)$, define $(J, \{e_1, e_2\}) \leftarrow \texttt{Split}(I, e)$ to be the routine that does the following:

- If $\texttt{lev}_I(e) \leq 1/2$, let J be the graph with e replaced by a path of two edges e_1 and e_2, each with conductance $2c_e$.
- If $\texttt{lev}_I(e) > 1/2$, let J be the graph with e replaced by two parallel edges e_1 and e_2, each with conductance $c_e/2$.

Let $(I', F') \leftarrow \texttt{Split}(I, F)$ be the graph-set pair that results from splitting all edges in F. For any $F \subseteq E(I)$, the distribution over graphs $I' \sim I[[F]]$ is obtained by splitting all edges in F and conditioning on an arbitrary copy for each edge. The purpose of doing this is as follows:

Proposition 4.9.6. *For an arbitrary subset $F \subseteq E(I)$, let $(I', F') \leftarrow \texttt{Split}(I, F)$. For all copies f_i, $i \in \{1, 2\}$ of edges $f \in F$ in I',*

$$\texttt{lev}_{I'}(f_i) \in [1/4, 3/4]$$

Proof. Splitting an edge $e \neq f \in F$ does not change the leverage score of f, so it suffices to show the desired proposition when $F = \{f\}$. When $\texttt{lev}_I(f) \geq \frac{1}{2}$, splitting it into two parallel copies does not change its effective resistance but doubles its resistance. Therefore, $\frac{1}{4} \leq \texttt{lev}_{I'}(f_i) \leq \frac{1}{2}$. When $\texttt{lev}_I(e) \leq \frac{1}{2}$, subdividing f to a copy f_i makes

$$\texttt{lev}_{I'}(f_i) = \frac{1}{2}\texttt{lev}_I(f) + \frac{1}{2}$$

Since $0 \leq \texttt{lev}_I(f) \leq \frac{1}{2}$, the desired inequality follows. □

Furthermore, doing this does not change the random spanning tree distribution:

Proposition 4.9.7. *Sampling a random spanning tree $T \sim J$ for $(J, \{e_1, e_2\}) \leftarrow \texttt{Split}(I, e)$ is equivalent to doing the following:*

- *Sample a random tree $T \sim I$.*
- *If $\texttt{lev}_I(e) \leq 1/2$, do the following:*
  - *If $e \in T$, return a tree T' obtained by replacing e with e_1 and e_2.*
  - *If $e \notin T$, return a tree T' obtained by adding e_1 with probability $1/2$ and e_2 otherwise.*
- *If $\texttt{lev}_I(e) \geq 1/2$, do the following:*
  - *If $e \in T$, return a tree T' obtained by replacing e with e_1 with probability $1/2$ and e_2 otherwise.*
  - *If $e \notin T$, return $T' \leftarrow T$.*

Proof. Consider each of the two cases separately:

Leverage score below 1/2. In this case, we need to show that (a) splitting e in series into edges e_1 and e_2 and sampling a tree from the resulting graph J is equivalent to (b) sampling a tree from I, splitting e in series if e is in the tree, and adding one of e_1, e_2 to the tree otherwise with probability $1/2$ for each. It suffices to show that each spanning tree T of J has the same probability of being generated through Procedures (a) and (b). Suppose first that $e_1, e_2 \in E(T)$. T is generated with probability proportional to $c^J_{e_1} c^J_{e_2} \prod_{f \neq e_1, e_2 \in E(T)} c^J_f = 4(c^I_e)^2 \prod_{f \neq e_1, e_2 \in E(T)} c^I_f$ using Procedure (a) and with probability proportional to $c^I_e \prod_{f \neq e_1, e_2 \in E(T)} c^I_f$ using Procedure (b). Now, suppose that one of e_1, e_2 is not in T and that $e_1 \in E(T)$ without loss of generality. Procedure (a) generates T with probability proportional to $(c^J_{e_1}) \prod_{f \neq e_1, e_2 \in E(T)} c^J_f = 2c^I_e \prod_{f \neq e_1, e_2 \in E(T)} c^I_f$. Procedure (b)

generates T with probability proportional to $(1/2)\prod_{f\neq e_1,e_2\in E(T)} c_f^I$. In both cases, Procedure (a) generates trees with weight $4c_e^I$ times the weight that Procedure (b) uses. Since the weights used for the Procedure (a) and (b) are proportional, Procedures (a) and (b) generate trees from the same distribution.

Leverage score above 1/2. We need to show that (c) splitting e in parallel into edges e_1 and e_2 and sampling from the resulting graph J is equivalent to (d) sampling a tree from I and replacing e with e_1 or e_2 with probability $1/2$ for each if e is in the tree. First, consider the case in which one of e_1, e_2 is in T. Without loss of generality, suppose that $e_1 \in E(T)$. T comes from Procedure (c) with probability proportional to $c_{e_1}^J \prod_{f\neq e_1,e_2\in E(T)} c_f^J = (c_e^I/2)\prod_{f\neq e_1,e_2\in E(T)} c_f^I$. T comes from Procedure (d) with probability proportional to $(1/2)c_e^I \prod_{f\neq e_1,e_2\in E(T)} c_f^I$. Now, suppose that $e_1, e_2 \notin E(T)$. In both procedures, T is generated with probability proportional to $\prod_{f\in E(T)} c_f^I$. The weights for Procedures (c) and (d) are equal, so Procedures (c) and (d) generate trees from the same distribution, as desired. □

We use the above proposition to show that computing a graph from the distribution $H[F]$ (the second algorithm below) is equivalent to computing a graph using a more incremental strategy that is particularly amenable to analysis using rank 1 updates (the first algorithm below):

Proposition 4.9.8. *Consider a graph H and a set $F \subseteq E(H)$. Consider any algorithm of the following form:*

- *Initialize $H_0 \leftarrow H$ and $k \leftarrow 0$*
- *While F is not empty*
  - *Pick a random edge $f_k \sim \mathcal{D}_k$, where $\mathcal{D}_k$ is an arbitrary distribution over F that only depends on the contractions/deletions of the previous edges $f_0, f_1, \ldots, f_{k-1}$*
  - *Let $H_{k+1} \sim H_k[[f_k]]$*
  - *Remove f_k from F if a self-loop/leaf is created; otherwise replace f_k in F with the remaining copy*
  - *Increment k*

Then H_k is equivalent in distribution to H'_k in the following algorithm for all $k \geq 0$, which only requires sampling the intersection of T_0 with F, not all of T_0:

- *Sample $T_0 \sim H$, $H'_0 \leftarrow H$, and set $k \leftarrow 0$*
- *While F is not empty*
  - *Pick a random edge $f_k \sim \mathcal{D}_k$ and let $(J, \{f^{(0)}, f^{(1)}\}) \leftarrow$ `Split`(H'_k, f_k).*

- *Let T_{k+1} be the random spanning tree of J obtained by setting $T \leftarrow T_k$ and $e \leftarrow f_k$ in Proposition 4.9.7*
- *Let H'_{k+1} be the subgraph of J with $f^{(0)}$ contracted if $f^{(0)} \in E(T_{k+1})$ and $f^{(0)}$ deleted otherwise*
- *Increment k*

Proof. We prove this fact by induction on k. For $k = 0$, clearly $H_0 = H'_0$. H_k is obtained from H_{k-1} by sampling the intersection of a random spanning tree in the graph J obtained by doing $(J, \{f^{(0)}, f^{(1)}\}) \leftarrow \texttt{Split}(H_{k-1}, f_{k-1})$. By Proposition 4.9.7, this is equivalent to sampling a random tree in H_{k-1} and splitting it randomly as described in Proposition 4.9.7.

By the inductive hypothesis, H_{k-1} and H'_{k-1} have the same distribution. Furthermore, for $k > 0$, notice that f_{k-1} is obtained from the same distribution $\mathcal{D}_{k-1}$. These replacements, when coupled with the previous paragraph, yield the second algorithm. Therefore, H_k and H'_k are equivalent in distribution, completing the induction step. □

Proposition 4.9.9. *Consider a graph H and a set $F \subseteq E(H)$. Sampling $H' \sim H[F]$ is equivalent to any strategy of the following form:*

- *Initialize $H_0 \leftarrow H$ and $k \leftarrow 0$*
- *While F is not empty*
  - *Pick a random edge $f_k \sim \mathcal{D}_k$, where $\mathcal{D}_k$ is an arbitrary distribution over F that only depends on the contractions/deletions of the previous edges $f_0, f_1, \ldots, f_{k-1}$*
  - *Let $H_{k+1} \sim H_k[[f_k]]$*
  - *Remove f_k from F if a self-loop/leaf is created; otherwise replace f_k in F with the remaining copy*
  - *Increment k*

Proof. By Proposition 4.9.8, this algorithm is the same as the second algorithm in Proposition 4.9.8. The final output of the second algorithm given in Proposition 4.9.8 is equivalent to sampling from $H[F]$ in distribution. Therefore, the algorithm given in Proposition 4.9.9 is equivalent to sampling from $H[F]$ in distribution, as desired. □

4.9.3 Concentration inequalities

The following is relevant for the slow version of Lemma 4.8.2:

Theorem 4.9.10 (Theorem 16 of [24])**.** *Let X be the martingale satisfying $|X_{i+1} - X_i| \leq c_i$ for all i specifying martingale increments. Then*

$$\Pr[|X_n - \boldsymbol{E}[X_n]| \geq \lambda] \leq 2e^{-\frac{\lambda^2}{2\sum_{i=1}^{n} c_i^2}}$$

where n is the total number of martingale increments.

To speed up the algorithm and prove Lemma 4.8.2, we exploit the following additional concentration inequality:

Theorem 4.9.11 (Theorem 18 in [24]). *Let X be a martingale satisfying:*

- *$Var(X_i|X_{i-1}) \leq \sigma_i^2$ for all i.*
- *$|X_i - X_{i-1}| \leq M$ for all i.*

Then

$$\Pr[|X_n - \boldsymbol{E}[X_n]| \geq \lambda] \leq e^{-\frac{\lambda^2}{2(\sum_{i=1}^n \sigma_i^2 + M\lambda/3)}}$$

4.9.4 [42] bound for source-sharing demand vectors

We use the following bound to bound changes in normalized potentials of vertices due to conditioning:

Theorem 4.9.12 (Lemma 3.1 of [42], with n replaced by τ). *Let G be a graph and consider three vertices $s_1, s_2, t \in V(G)$. Then for any positive integer τ*

$$\sum_{f \in E(G)} \frac{|b_{s_1t}^T L_G^+ b_f||b_f^T L_G^+ b_{s_2t}|}{r_f} \leq (8 \log \tau)|b_{s_1t}^T L_G^+ b_{s_2t}| + \frac{1}{\tau}(b_{s_1t}^T L_G^+ b_{s_1t} + b_{s_2t}^T L_G^+ b_{s_2t})$$

4.9.5 Matrix sketching

The `FastFix` algorithm needs to compute ℓ_p norms of vectors whose entries would naively require as many as $\Theta(m)$ Laplacian solves to compute. We show that these vector norms can be approximated with $O(\text{polylog}(n))$ Laplacian solves using matrix sketching:

Theorem 4.9.13 ([44], Theorem 3 stated for ℓ_p rather than just ℓ_1). *An efficiently computable, polylog(d)-space linear sketch exists for all ℓ_p norms exists for all $p \in (0, 2]$. That is, given a $d \in \mathbb{Z}_{\geq 1}$, $\delta \in (0, 1)$, $p \in (0, 2]$, and $\epsilon \in (0, 1)$, there is a matrix $C = \texttt{SketchMatrix}(d, \delta, p, \epsilon) \in \mathbb{R}^{l \times d}$ and an algorithm $\texttt{RecoverNorm}(s, d, \delta, p, \epsilon)$ with the following properties:*

- *(Approximation) For any vector $v \in \mathbb{R}^d$, with probability at least $1 - \delta$ over the randomness of `SketchMatrix`, the value $r = \texttt{RecoverNorm}(Cv, d, \delta, p, \epsilon)$ is as follows:*

$$(1 - \epsilon)||v||_p \leq r \leq (1 + \epsilon)||v||_p$$

- *$l = c/\epsilon^2 \log(1/\delta)$ for some constant $c > 1$*
- *(Runtime) `SketchMatrix` and `RecoverNorm` take $O(ld)$ and poly(l) time respectively.*

4.9.6 Localization bound

The `FastFix` algorithm needs to be able to reuse low-contribution edges for many iterations. We exploit the following bound to do this:

Theorem 4.9.14 (Restatement of Theorem 3.3.1)**.** *Let G be a graph. Then for any vector $w \in \mathbb{R}^{E(G)}_{\geq 0}$,*

$$\sum_{e,f \in E(G)} w_e w_f \frac{|b_e^T L_G^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq O(\log^2 n)||w||_2^2$$

4.10 Conductance concentration inequality

In this section, we prove the following inefficient analogue of Lemma 4.8.2:

Lemma 4.10.1. *Given a graph I, a random graph J that is a valid sample from the distribution $I[F]$, a set $D \subseteq E(I)$, $S, S' \subseteq V(G)$, $F \subseteq E(I) \setminus D$, and an accuracy parameter $\epsilon \in (0,1)$. With high probability on J, there is a set $F' \subseteq F$ with two properties:*

- *(Conductance)* $c^{J \setminus D \setminus F'}(S, S') \leq (1+\epsilon) c^{I \setminus D}(S, S')$
- *(Size)* $|F'| \leq \tilde{O}(\epsilon^{-3}(\Delta^{I \setminus D}(S, S') + \Delta^{I \setminus D}(S', S) + |D|))$

The proof conditions on one edge in F at a time. In principle, doing this could drastically change the conductance between S and S'. Luckily, it does not change much in expectation if one conditions on a random edge in F. While the conductance does not change much in expectation, it could change in absolute amount. In Proposition 4.10.5, we show that there always exists an edge an edge that, if conditioned on, does not change the $S - S'$ conductance much. Using these statements in concert with Azuma's Inequality shows that the $S - S'$ conductance does not change by more than a $(1+\epsilon)$ factor with high probability if one conditions on all but a very small number of edges in F. Letting these edges be F' proves the desired result.

To prove Lemma 4.10.1, we will need to use Azuma's Inequality to control a number of other quantities besides the $S - S'$ conductance to show that one can always find the edges described in Proposition 4.10.5. As a result, we choose to discuss a much simpler result first, which may be of independent interest.

4.10.1 Warmup: Controlling the effective resistance between two vertices

In this subsection, we prove Lemma 4.10.1 with $S = \{s\}$ and $S' = \{t\}$ each being one vertex and D being empty. This is also Lemma 4.1.4, which we restate here with notation that agrees with Lemma 4.10.1:

Lemma 4.10.2 (Restatement of Lemma 4.1.4)**.** *Let I be a graph, $F \subseteq E(I)$, $\epsilon \in (0,1)$, and $s, t \in V(I)$. Sample a graph $J \sim I[F]$. Then, with high probability, there is a set $F' \leftarrow$ `SpecialFix`$(I, s, t, F, \epsilon) \subseteq F$ that satisfies both of the following guarantees:*

- *(Effective resistance)* $b_{st}^T L_{J \setminus F'}^+ b_{st} \geq (1-\epsilon)(b_{st}^T L_I^+ b_{st})$
- *(Size)* $|F'| \leq O((\log n)/\epsilon^2)$

While this special case is not directly relevant to the proof of Lemma 4.10.1, it offers a simpler way of demonstrating most of the key ideas.

`SpecialFix` splits and conditions on the edge with lowest energy until there are $O((\log n)/\epsilon^2)$ edges left. At this point, the algorithm returns the remaining edges in F to be in F'.

Algorithm 13: `SpecialFix`(I, s, t, F, ϵ), never executed

Data: the graph I, $s, t \in V(I)$, arbitrary $F \subseteq E(I)$, $\epsilon \in (0, 1)$
Output: the subset $F' \subseteq F$ of edges that should be deleted

$J \leftarrow I$
while *F has more than $9(\log n)/\epsilon^2$ non-leaf/loop edges* **do**
  $f \leftarrow$ the non-leaf/loop edge in $E(J)$ that minimizes $(b_{st}^T L_J^+ b_f)^2 c_f$
  $(J, \{f_1, f_2\}) \leftarrow$ `Split`(J, f)
  $J \sim J[f_1]$
  $F \leftarrow (F \setminus \{f\}) \cup \{f_2\}$
return *the non-leaf/loop edges of F*

Proof of Lemma 4.10.2. Let $F^0 = F$ and for all $i \geq 0$, let F^{i+1} be the set of non-leaf/loop edges left over after conditioning on $|F^i|/8$ edges in F^i. When a non-leaf/loop edge is split and conditioned on, it has a probability of at least $1/4$ of resulting in an additional loop or leaf. Furthermore, the number of non-leaf/loop edges in F never increases. Therefore, since all conditionings are independent of each other, $|F^{i+1}| \leq 63|F^i|/64$ with high probability as long as $|F^i| \geq \log n$ by Chernoff bounds. Therefore, there are only $\log n$ F^is before the algorithm terminates.

Let J^i be the graph J with $F = F^i$. We now show that

$$b_{st}^T L_{J^{i+1}}^+ b_{st} \geq (1 - \frac{10\sqrt{\log n}}{\sqrt{|F^i|}})(b_{st}^T L_{J^i}^+ b_{st})$$

with high probability. Let J_j be the graph immediately before the jth iteration after J^i and let f_j be the edge conditioned on during this iteration to obtain J_{j+1}. Let F_j be the value of f before conditioning on f_j. By Propositions 4.9.4 and 4.9.5,

$$\begin{aligned}
\mathbf{E}_{J_{j+1}}[b_{st}^T L_{J_{j+1}}^+ b_{st}] &= \mathtt{lev}_{J_j}(f_j) \left(b_{st}^T L_{J_j}^+ b_{st} - \frac{(b_{st}^T L_{J_j}^+ b_{f_j})^2}{r_{f_j} \mathtt{lev}_{J_j}(f_j)} \right) \\
&\quad + \mathtt{nonlev}_{J_j}(f_j) \left(b_{st}^T L_{J_j}^+ b_{st} + \frac{(b_{st}^T L_{J_j}^+ b_{f_j})^2}{r_{f_j} \mathtt{nonlev}_{J_j}(f_j)} \right) \\
&= b_{st}^T L_{J_j}^+ b_{st}
\end{aligned}$$

so $X_j := b_{st}^T L_{J_j}^+ b_{st}$ is a martingale. We now show that our choice of f_j makes the martingale Lipschitz. Proposition 4.9.6 implies that $\mathtt{lev}_{J_j}(f_j) \geq 1/4$ and that $\mathtt{nonlev}_{J_j} \geq 1/4$. By Propositions 4.9.4 and 4.9.5,

$$|b_{st}^T L_{J_{j+1}}^+ b_{st} - b_{st}^T L_{J_j}^+ b_{st}| \leq 4\frac{(b_{st}^T L_{J_j}^+ b_{f_j})^2}{r_{f_j}}$$

This is the energy contribution of f_j to the overall $s-t$ energy. Therefore,

$$|b_{st}^T L_{J_{j+1}}^+ b_{st} - b_{st}^T L_{J_j}^+ b_{st}| \leq \frac{4}{|F_j|} b_{st}^T L_{J_j}^+ b_{st}$$

Since $|F^i|/8$ conditionings occur during J^i and J^{i+1}, $|F_j| \geq 7|F^i|/8$ for all j and

$$b_{st}^T L_{J_{j+1}}^+ b_{st} \leq \left(1 + \frac{32}{7|F^i|}\right)^{|F^i|/8} b_{st}^T L_{J_j}^+ b_{st} \leq e b_{st}^T L_{J_j}^+ b_{st}$$

We now bootstrap this bound using Azuma's Inequality to obtain a much better bound on the change in the $s-t$ resistance. In particular,

$$|b_{st}^T L_{J_{j+1}}^+ b_{st} - b_{st}^T L_{J_j}^+ b_{st}| \leq \frac{4e}{|F_j|} b_{st}^T L_{J^i}^+ b_{st}$$

Therefore, by Theorem 4.9.10,

$$\Pr[|b_{st}^T L_{J^{i+1}}^+ b_{st} - b_{st}^T L_{J^i}^+ b_{st}| \geq \frac{10\sqrt{\log n}}{\sqrt{|F^i|}} b_{st}^T L_{J^i} b_{st}] \leq 1/\text{poly}(n)$$

Therefore, with high probability, the change is at most a $(1-\epsilon)$ factor unless $|F^i| \leq (\log n)/\epsilon^2$. Deleting these edges can only increase the $s-t$ resistance. Conditioning on F' instead of deleting would have given a graph J_{final} equivalent to sampling $J_{final} \sim I[F]$ by Proposition 4.9.7. This completes the proof. □

4.10.2 Generalizing the warmup

By Proposition 4.9.1, for any graph H with disjoint sets of vertices $S, S' \subseteq V(H)$, $1/c^H(S,S') = b_{ss'}^T L_{H'}^+ b_{ss'}$, where H' is a graph obtained from H by identifying S to s and S' to s'. Therefore, to show that $c^H(S,S')$ does not increase by more than a $(1+\epsilon)$ factor, it suffices to show that $b_{ss'} L_{H'}^+ b_{ss'}$ does not decrease by more than a $(1-\epsilon)$ factor, where H is obtained from $I \setminus D$ by conditioning on a partial sample from a random spanning tree in I.

This quantity is similar to the quantity controlled during the warmup. There are two initial obstacles to directly using the approach in the warmup:

- The tree being sampled is sampled in a different graph $(H \cup D)$ from the graph in which the quantity $b_{ss'}^T L_{H'}^+ b_{ss'}$ is defined. This discrepancy causes $b_{ss'}^T L_{H'}^+ b_{ss'}$ to not be a martingale under conditioning.

- Conditioning could cause the quantity in the "Size" upper bound to change.

To get around these issues, we show that there exists an edge that changes $b_{ss'}^T L_{H'}^+ b_{ss'}$, $\Delta^H(S, S')$, and $\Delta^H(S', S)$ by a very small factor in expectation. Ideally, there would be an edge to condition on that changes $b_{ss'}^T L_{H'}^+ b_{ss'}$ and the "Size" bound in expectation by at most a $(1 + (\Delta^H(S, S') + \Delta^H(S', S) + |D|)/|F|^2)$ factor in expectation. If this is the case, then conditioning on all but $(\Delta^H(S, S') + \Delta^H(S', S) + |D|)\epsilon^{-1}$ edges changes the expectation by at most a $(1+\epsilon)$ factor in expectation. When combined with martingale concentration, this shows that the size bound and the conductance change by at most a factor of $(1 + O(\epsilon))$ with high probability.

Unfortunately, such an increase is hard to obtain. For example, the expected change in $b_{ss'}^T L_{H'}^+ b_{ss'}$ due to conditioning on an edge f can be written as a product of two terms:

- The change in f's leverage score due to identifying S to s, identifying S' to s', and deleting all edges in D.
- The energy of the $b_{ss'}$ electrical flow in H' on f.

The second quantity is at most $O(1/|F|)b_{ss'}^T L_{H'}^+ b_{ss'}$ for most edges in F due to the fact that the average energy on an edge is small. The first quantity is trickier to bound and sometimes cannot be bounded at all. For example, if F solely consists of edges with endpoints in S, the change in leverage score upon identifying S to s is for all edges in F is high (constant). Luckily, though, the endpoints of these edges are very close to s in the $L_{H'}^+ b_{ss'}$ potential embedding.

This phenomenon holds in general. In particular, Lemma 4.10.12 can be used to show that the total leverage score decrease of all edges with an endpoint that has $s - s'$ normalized potential at least p away from s and s' when S and S' are identified is at most $O((\Delta^H(S', S) + \Delta^H(S, S'))/p)$. The total leverage score increase is at most $|D|$, so the total leverage score change is at most the sum of these two quantities. Therefore, if a large fraction of the edges in F have an endpoint with normalized $s - s'$ potential that is between p and $1 - p$, there is an edge $f \in F$, which, when conditioned on, causes an

$$1 + O\left(\left(\frac{|D| + (\Delta^H(S', S) + \Delta^H(S, S'))/p}{\epsilon}\right)\left(\frac{b_{ss'}^T L_{H'}^+ b_{ss'}}{|F|}\right)\right)$$
$$\leq 1 + O\left(\frac{(|D| + \Delta^H(S', S) + \Delta^H(S, S'))}{p|F|^2}\right)$$

factor change in the "Size" bound and $b_{ss'}^T L_{H'}^+ b_{ss'}$ in expectation. This differs from the desired bound due to the factor of $1/p$.

This bound is good enough, though, to allow conditioning on all but roughly $\frac{(|D| + \Delta^H(S', S) + \Delta^H(S, S'))}{p\epsilon}$ edges before the "Size" bound or $b_{ss'}^T L_{H'}^+ b_{ss'}$ changes by more than an $(1 + \epsilon)$ factor. Once this many edges are left that have maximum normalized endpoint potential at most $O(p)$, defer conditioning on these edges until the very end of the algorithm.

Deferring conditioning on these edges works for the following reasons.

- The total normalized potential of such edges is at most $\frac{(|D|+\Delta^H(S',S)+\Delta^H(S,S'))}{p\epsilon}O(p) = O((|D| + \Delta^H(S', S) + \Delta^H(S, S'))\epsilon^{-1})$.
- If we can show that the total normalized potential of deferred edges only increases by a $(1 + \epsilon)$ factor over the course of conditioning on the rest of F, the total normalized potential of all deferred edges is at most $O((|D| + \Delta^H(S', S) + \Delta^H(S, S'))\epsilon^{-1})$ at the end of the algorithm.
- By Markov's inequality, at most $O((|D| + \Delta^H(S', S) + \Delta^H(S, S'))\epsilon^{-2})$ of the deferred edges have any endpoint with normalized potential in the interval $[\epsilon, 1 - \epsilon]$. Conditioning on the rest of the edges therefore decreases the $s - s'$ resistance in H' by at most a factor of $(1 - O(\epsilon))$ by Proposition 4.10.20. Letting F' be the remaining $O((|D| + \Delta^H(S', S) + \Delta^H(S, S'))\epsilon^{-2})$ edges yields the desired result by Rayleigh monotonicity.

4.10.3 Reducing the fixing lemma to the existence of stable oracles

We prove Lemma 4.10.1 using the algorithm `Fix` with the oracle `SlowOracle`. This algorithm picks edges to condition on in F one by one in the graph I. It selects edges that do not change the following quantities very much:

- $\Delta^{I\setminus D}(S, S')$
- $\Delta^{I\setminus D}(S', S)$
- $c^{I\setminus D}(S, S')$
- Various sums of potential drops

The key idea is that there always is an edge that does not change any of these quantities very much. Unlike in the special case when S and S' are single vertices, conditioning might decrease the effective resistance in expectation. We show that the same choice of edge that does not change these quantities also leads to a very small change in the expectation.

Stability implies concentration

We now formalize the concept of low-change edges using *stable functions*:

Definition 4.10.3 (Stable functions)**.** *Call a function g on graphs* electrical *if g is preserved under two operations:*

- *edge subdivision; that is replacing an edge e with a path of length two consisting of edges with resistance $r_e/2$*

- *edge splitting; that is replacing an edge e with two parallel copies with resistance $2r_e$*

A set of edges $X \subseteq F$ is called a (ρ_L, ρ_E, δ)-stable *subset of F for an electrical function g if*

- $|g(H/f) - g(H)| \leq \frac{\rho_L}{|F|} g(H) + \delta$ *for all edges* $f \in X$
- $|g(H \backslash f) - g(H)| \leq \frac{\rho_L}{|F|} g(H) + \delta$ *for all* $f \in X$
- $\frac{1}{|X|} \sum_{f \in X} |g(H) - \boldsymbol{E}_{H' \sim H[f]}[g(H')]| \leq \frac{\rho_E}{|F|^2} g(H) + \delta$

An set X is called (ρ_L, ρ_E)-stable *for g if it is $(\rho_L, \rho_E, 0)$-stable for g.*

Definition 4.10.4 (Multiplicative functions)**.** *A function $h : \mathbb{R}^d_{>0} \to \mathbb{R}_{>0}$ is called* γ-multlipschitz *if the function $h'(x_1, \ldots, x_d) := \log h(\exp(x_1), \exp(x_2), \ldots, \exp(x_d))$ is γ-Lipschitz in the ℓ_1 norm.*

We encapsulate our use of standard concentration inequalities into one proposition. This proposition is used to show that conditioning on a long sequence of edges that are stable for the functions $\{x_a\}_{a=1}^{\ell_1}$ and $\{y_b\}_{b=1}^{\ell_2}$ does not change the value of any of these functions by a large amount with high probability. This proposition allows the stability to depend on the x_a functions, which one should think of as being related to $\Delta^{I \backslash D}(S, S')$.

Proposition 4.10.5. *Consider two collections of electrical functions $\{x_a\}_{a=1}^{\ell_1}$ and $\{y_b\}_{b=1}^{\ell_2}$. Fix a graph $I_0 = I$ and a set $F_0 = F \subseteq E(I)$. Consider two sequences of random graphs $\{I_i\}_i$, $\{I'_i\}_i$, edges $\{f_i\}_i$, and edge sets $\{F_i\}_i$, $\{X_i\}_i$ with the following properties for all i:*

- $I_{i+1} \sim I_i[[f_i]]$ *for a uniformly random edge* $f_i \in X_i$
- $F_{i+1} := F_i \cap E(I_{i+1})$
- $X_i \subseteq F_i$
- X_i *is* $(I'_i, \rho_L, \rho_i, 0)$*-stable for all functions x_a and* $(I'_i, F'_i) := \texttt{Split}(I_i, F_i)$
- X_i *is* $(I'_i, \rho_L, \rho_i, \delta)$*-stable for all functions y_b*
- $|F_i| \geq \sigma$
- $\rho_i := \rho_E(\{x_a(H_i)\}_a)$*; that is ρ_i is only a function of the values of the x_as on H_i.*
- ρ_E *is a γ-multlipschitz function.*

Let $\tau_E := \max_i(\rho_i/|X_i|)$. Then with high probability, all of the following bounds apply for all i:

- *For all* $w \in \{x_a\}_a$,

$$|w(I) - w(I_i)| \leq \tilde{O}\left(\tau_E + \frac{\sqrt{\gamma\ell_1}\rho_L^{3/2}}{\sqrt{\sigma}}\right) w(H)$$

- *For all* $w \in \{y_b\}_b$,

$$|w(I) - w(I_i)| \leq \tilde{O}\left(\tau_E + \frac{\sqrt{\gamma\ell_1}\rho_L^{3/2}}{\sqrt{\sigma}}\right) w(H) + \tilde{O}(\rho_L\gamma\ell_1|F|\delta)$$

Before proving this, notice that $f_i \in F_{i+1}$ if and only if the edge set of I_i doesn't change, i.e. one of the following two conditions holds:

- $\texttt{lev}_{I_i}(f_i) \leq 1/2$ and f_i is contracted to form I_{i+1}
- $\texttt{lev}_{I_i}(f_i) > 1/2$ and f_i is deleted to form I_{i+1}

The proof of Proposition 4.10.5 uses Azuma's Inequality.

Proof of Proposition 4.10.5. Break up the sequence of edges to condition on into a small number of subsequences with guaranteed small changes in $x_a(I)$ and $y_b(I)$ values. Then, apply Theorem 4.9.10 to each subsequence. Specifically, let $i_0 = 0$ and i_{j+1} be the maximum value for which $|F_{i_{j+1}}| \geq |F_{i_j}|(1 - 1/(16\gamma\ell_1\rho_L))$. Concentration bounds will be shown for each subsequence of is between i_j and i_{j+1}.

We start with bounds on $|i_{j+1} - i_j|$. Consider the random variable $|F_i|$. By the fourth property of stable edges, $\mathbf{E}[|F_{i+1}||F_i] \leq |F_i| - \frac{1}{4}$. In particular for any $z > i_j$,

$$\mathbf{E}[|F_z||F_{i_j}, i_j] \leq |F_{i_j}| - \frac{1}{4}(z - i_j)$$

$X_i = |F_i| - \mathbf{E}[|F_i|]$ is a martingale with $c_i = 1$ for all i, so by Theorem 4.9.10

$$|F_z| \leq |F_{i_j}| - \frac{1}{4}(z - i_j) + \sqrt{(z - i_j)\log n}$$

with probability at least $1 - \frac{1}{\text{poly(n)}}$. As long as $z - i_j > 8\log n$, the above inequality implies that

$$|z - i_j| \leq 8(|F_{i_j}| - |F_z|)$$

For $z = i_{j+1}$, $|i_{j+1} - i_j| \leq \frac{1}{2\gamma\ell_1\rho_L}|F_{i_j}|$ with high probability.

By the first two properties of stable edges and the fact that $|F_i|$ is nondecreasing,

$$x_a(I_i) \le x_a(I_{i_j}) \left(1 + \frac{\rho_L}{|F_{i_{j+1}}|}\right)^{i_{j+1}-i_j}$$
$$\le \left(1 + \frac{1}{\gamma\ell_1}\right) x_a(I_{i_j})$$

and $x_a(I_i) \ge (1 - \frac{1}{\gamma\ell_1})x_a(I_{i_j})$ for all $i \in [i_j, i_{j+1}]$ and all a with high probability. By the multlipschitzness of ρ_E,

$$\rho_i \le (1 + 1/(\gamma\ell_1))^{\gamma\ell_1}\rho_{i_j} \le e\rho_{i_j}$$

for all $i \in [i_j, i_{j+1}]$ with high probability. Similarly,

$$y_b(I_i) \le (1 + 1/(\gamma\ell_1))(y_b(I_{i_j}) + |F|\delta)$$

for all $i \in [i_j, i_{j+1}]$ and b with high probability.

We now use these worst-case bounds on x_as and y_bs within each interval to get a better bound using the expectation. By the third property of stable edges and the definition of τ_E,

$$|\mathbf{E}[w(I_{i_{j+1}})|I_{i_j}, i_j] - w(I_{i_j})| \le (i_{j+1} - i_j)\frac{\tau_E}{|F_{i_{j+1}}|}e(w(I_{i_j}) + |F|\delta) + |F|\delta$$
$$\le 2e\frac{\tau_E}{\gamma\ell_1\rho_L}w(I_{i_j}) + O(|F|\delta)$$

for functions $w \in \{y_b\}_b$. For functions $w \in \{x_a\}_a$, a similar bound holds:

$$|\mathbf{E}[w(I_{i_{j+1}})|I_{i_j}, i_j] - w(I_{i_j})| \le (i_{j+1} - i_j)\frac{\tau_E}{|F_{i_{j+1}}|}ew(I_{i_j})$$
$$\le 2e\frac{\tau_E}{\gamma\ell_1\rho_L}w(I_{i_j})$$

Therefore, by Theorem 4.9.10,

$$|w(I_{i_{j+1}}) - w(I_{i_j})| \le 2e\frac{\tau_E}{\gamma\ell_1\rho_L}w(I_{i_j}) + 2e\sqrt{(\log n)\frac{|F_{i_j}|}{2\gamma\ell_1\rho_L}}\frac{\rho_L}{|F_i|}w(I_{i_j}) + \tilde{O}(|F|\delta)$$

with high probability for $w \in \{y_b\}_b$, with analogous statements for x_a with $\delta = 0$ (see proposition guarantees). Since $|F_{i_{j+1}+1}| \le |F_{i_j}|(1 - \frac{1}{16\rho_L\gamma\ell_1})$, $j \le 16\rho_L\gamma\ell_1(\log n)$. Since $|F_i| \ge \sigma$ for all i, the total change is

$$\tilde{O}\left(\tau_e + \frac{\rho_L^{3/2}\sqrt{\gamma\ell_1}}{\sqrt{\sigma}}\right) w(I) + \tilde{O}(\rho_L\gamma\ell_1|F|\delta)$$

as desired. □

Stable oracles and their use in proving the fixing lemmas

In this section, we exploit Proposition 4.10.5 to reduce Lemmas 4.10.1 and 4.8.2 to the construction of a certain oracle. This oracle outputs a set of edges remains stable for the main objective $b_{ss'}^T L_H^+ b_{ss'}$, the "Size" bound, and an auxilliary objective that bounds the endpoint potentials of two sets of "deferred" edges.

Before defining the oracle, we define a proxy for the degree function, called the *normalized degree*, that will be more convenient to control directly than $\Delta^H(X,Y)$:

Definition 4.10.6 (Normalized degree)**.** *For a graph H with disjoint vertex sets X, Y, let*

$$\delta_{X,Y}(H) := \frac{\Delta^H(X,Y)}{c^H(X,Y)}$$

It is easier to control due to the following representation, which involves quantities whose change can be easily analyzed using rank 1 updates:

Remark 9. *$\delta_{X,Y}(H)$ can be written in a different way:*

$$\delta_{X,Y}(H) = \sum_{w\in Y}\sum_{e\in\partial_H w}(b_{xw}^T L_{H/X}^+ b_{xw})\frac{b_{xy}^T L_{H/(X\cup Y)}^+ b_e}{r_e}$$

where x and y are the identifications of X and Y respectively in $H/(X\cup Y)$ and e is oriented towards w.

Proof. By Proposition 4.9.2, for each $w \in Y$,

$$\sum_{e\in\partial_H w}\frac{b_{xy}^T L_{H/(X\cup Y)}^+ b_e}{r_e} = \frac{c_{xw}^{\mathtt{Schur}(H/X,X\cup Y)}}{\sum_{w'\in Y} c_{xw'}^{\mathtt{Schur}(H/X,X\cup Y)}}$$

Since Schur complement conductances do not depend on edges with endpoints that were not eliminated,

$$\sum_{w'\in Y} c_{xw'}^{\mathtt{Schur}(H/X,X\cup Y)} = c^H(X,Y)$$

Combining these equalities shows that

$$\sum_{w\in Y}(b_{xw}^T L_{H/X}^+ b_{xw})\sum_{e\in\partial_H w}\frac{b_{xy}^T L_{H/(X\cup Y)}^+ b_e}{r_e} = \frac{\Delta^H(X,Y)}{c^H(X,Y)}$$

as desired. □

Now, we define an oracle that generalizes the concept of picking the minimum energy edge that was so important to `SpecialFix`:

Definition 4.10.7 (Stable oracles)**.** *An* $(\rho, K(z))$-stable oracle*, is a function*

$Z \leftarrow \texttt{Oracle}(I, S, S', D, A, B, W)$ *that takes in a graph* I*, two disjoint sets of vertices* $S, S' \subseteq V(I)$*, a set of* deleted *edges* $D \subseteq E(I)$*, two sets of* deferred *edges* $A, B \subseteq E(I)$*, and a set of edges* W *to condition on.* K *is allowed to be a function of some parameter* z*. This oracle is given inputs that satisfy the following conditions:*

- *(Bounded leverage score difference) For all* $e \in W$*,* $|\texttt{lev}_{I\setminus D}(e) - \texttt{lev}_{I/(S,S')}(e)| \leq 1/16$
- *(Narrow potential neighborhood) There is a* $p \leq 1/4$ *for which* ***one*** *of the following conditions holds:*
  - *(*s *narrow potential neighborhood)* $\frac{b_{ss'}^T L^+_{(I\setminus D)/(S,S')}((b_{su}+b_{sv})/2)}{b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}} \in [p, 2p]$ *for all* $\{u, v\} \in W$*.*
  - *(*s' *narrow potential neighborhood)* $\frac{b_{ss'}^T L^+_{(I\setminus D)/(S,S')}((b_{us'}+b_{vs'})/2)}{b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}} \in [p, 2p]$ *for all* $\{u, v\} \in W$*.*

It outputs a set $Z \subseteq E(I)$ *of edges to condition on.*

Let $I_0 = I$ *and for each* $i > 0$*, obtain* I_i *by picking a uniformly random edge* $f_{i-1} \in Z$*, letting* $I_i \leftarrow I_{i-1}[[f_{i-1}]]$*, and removing* f_{i-1} *from* Z*.*

`Oracle` *satisfies the following stability-related properties with high probability for all* $i < K(|W|)$*:*

- *(Size of* Z*)* $|Z| \geq |W|/(\log^4 n)$
- *(Leverage score stability)*
  - *(Upper leverage score stability)* $|\texttt{lev}_{I_i\setminus D}(f_i) - \texttt{lev}_{I\setminus D}(f_i)| \leq 1/16$
  - *(Lower leverage score stability)* $|\texttt{lev}_{I_i/(S,S')}(f_i) - \texttt{lev}_{I/(S,S')}(f_i)| \leq 1/16$
- *(Midpoint potential stability)*
  - *(*s *midpoint potential stability) Let* $f_i = \{u_i, v_i\}$*. Then*

$$\frac{b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')}(b_{su_i} + b_{sv_i})/2}{b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{ss'}} \geq p/2$$

- *(s' midpoint potential stability) Let $f_i = \{u_i, v_i\}$. Then*

$$\frac{b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')}(b_{u_i s'} + b_{v_i s'})/2}{b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{ss'}} \geq p/2$$

- *($S - S'$-normalized degree change stability)*
  - *($S - S'$ conductance term stability)*

$$\sum_{w \in S'} (b_{sw}^T L^+_{(I_i \setminus D)/S} b_{sw}) \sum_{e \in \partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i \setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \leq \frac{\rho}{|W|} \delta_{S,S'}(I \setminus D)$$

  - *($S' - S$ conductance term stability)*

$$\sum_{w \in S} (b_{s'w}^T L^+_{(I_i \setminus D)/S'} b_{s'w}) \sum_{e \in \partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i \setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \leq \frac{\rho}{|W|} \delta_{S',S}(I \setminus D)$$

  - *($S - S'$ energy term stability)*

$$\sum_{w \in S'} \frac{(b_{sw}^T L^+_{(I_i \setminus D)/S} b_{f_i})^2}{r_{f_i}} \sum_{e \in \partial_{I_i} w} \frac{b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_e}{r_e} \leq \frac{\rho}{|W|} \delta_{S,S'}(I \setminus D)$$

  - *($S' - S$ energy term stability)*

$$\sum_{w \in S} \frac{(b_{s'w}^T L^+_{(I_i \setminus D)/S'} b_{f_i})^2}{r_{f_i}} \sum_{e \in \partial_{I_i} w} \frac{b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_e}{r_e} \leq \frac{\rho}{|W|} \delta_{S',S}(I \setminus D)$$

- *(Deferred endpoint potential change stability)*

$$\left(\sum_{\{u,v\} \in A} \frac{|b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i \setminus D)/(S,S')} (b_{su} + b_{sv})|}{r_{f_i}} \right)$$
$$+ \left(\sum_{\{u,v\} \in B} \frac{|b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i \setminus D)/(S,S')} (b_{us'} + b_{vs'})|}{r_{f_i}} \right)$$
$$\leq \frac{\rho}{|W|} \left(\sum_{\{u,v\} \in A} b_{ss'}^T L^+_{(I \setminus D)/(S,S')} (b_{su} + b_{sv}) + \sum_{\{u,v\} \in B} b_{ss'}^T L^+_{(I \setminus D)/(S,S')} (b_{us'} + b_{vs'}) \right) + \frac{r_{min}}{n^4}$$

- *(Main objective change stability)*

$$\frac{(b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i})^2}{r_{f_i}} \leq \frac{\rho}{|W|} b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}$$

$|W|$ in each of the above guarantees refers to the original size of W.

We now use this oracle to prove the following result, which will later be used to show Lemmas 4.10.1 and 4.8.2:

Lemma 4.10.8. *Given a (ρ, K)-stable oracle `Oracle`, there is an algorithm `Fix`$(I, J, S, S', \epsilon, F, D)$ that takes in a graph I, a random graph J that is a valid sample from the distribution $I[F]$, a set $D \subseteq E(I)$ of deleted edges, $S, S' \subseteq V(G)$, $F \subseteq E(I) \setminus D$, and an accuracy parameter $\epsilon \in (0, 1)$. With high probability on J, there is a set $F' \leftarrow$ `Fix`$(I, J, S, S', \epsilon, F, D) \subseteq F$ with two properties:*

- *(Conductance) $c^{J\setminus D\setminus F'}(S, S') \leq (1+\epsilon)c^{I\setminus D}(S, S')$*
- *(Size) $|F'| \leq \tilde{O}(\rho^3\epsilon^{-3}(\Delta^{I\setminus D}(S, S') + \Delta^{I\setminus D}(S', S) + |D|))$*

Futhermore, `Fix` takes $\tilde{O}(m + (\max_{z\leq|F|} \frac{z\rho^3}{K(z)} + \log^2 n)(m + \mathcal{T}(\texttt{Oracle})))$ time, where $\mathcal{T}(\texttt{Oracle})$ is the runtime of `Oracle`.

The runtime in the above guarantee corresponds to running `Oracle` $\frac{|F|\rho^3}{K}$ times. Later in this section, we implement a simple $(O(\text{polylog}(n)), 1)$-stable oracle called `SlowOracle`. This oracle is sufficient to prove Lemma 4.10.1. However, $\mathcal{T}(\texttt{SlowOracle}) = \Theta(m^2)$. Furthermore, $K = 1$. As a result, the runtime of the algorithm could be as high as $\Theta(m^3)$. To reduce this runtime, we exploit Theorem 5.4.1 and sketching techniques to obtain `FastOracle`. Sketching allows us to make $\mathcal{T}(\texttt{FastOracle}) = m^{1+o(1)}$ and Theorem 5.4.1 allows us to reuse Z for many iterations. Specifically, `FastOracle` is an $(m^{o(1)}, zm^{-o(1)})$-stable oracle. Therefore, plugging in `FastOracle` to `Fix` proves Lemma 4.8.2.

To prove Lemma 4.10.8, proceed as described in Section 4.10.2. At any given point, bucket edges in F by their maximum normalized endpoint $S - S'$ potential. Let $[p, 2p]$ be the potential range with the most edges of F in the corresponding bucket. If this bucket W has more than $\rho^3((\Delta^{I\setminus D}(S, S') + \Delta^{I\setminus D}(S', S))/p + |D|)$, call `Oracle` to produce a subset $Z \subseteq W$ and split and condition on a uniformly random sample of K elements from Z. Otherwise, "defer" conditioning on all edges of Z by adding them to A or B, depending on whether they are closer to S or S' respectively in the potential embedding. Once all remaining edges of F are in either A or B, delete the ones with an endpoint with normalized potential in the interval $[\epsilon, 1-\epsilon]$ and condition on all others.

We now give pseudocode that implements the above description:

Algorithm 14: `Fix`$(I, J, S, S', \epsilon, F, D)$

Data: two disjoint sets of vertices S and S', the graph I, the random sample $J \sim I[F]$, $\epsilon \in (0,1)$, $F, D \subseteq E(G)$
Result: the set F'

`// edges whose endpoint potentials we want to control`
$A, B \leftarrow \emptyset$
while $|F|$ *is not empty* **do**
  `// identifications` s, s'
  $I' \leftarrow (I \setminus D)/(S, S')$
  foreach $i \leftarrow 0, 1, \ldots$ **do**
    $X_i \leftarrow \{\{u,v\} \in F : \frac{b_{ss'}^T L_{I'}^+ (b_{su}+b_{sv})/2}{b_{ss'}^T L_{I'}^+ b_{ss'}} \in (2^{-i-2}, 2^{-i-1}]\}$
    $Y_i \leftarrow \{\{u,v\} \in F : \frac{b_{ss'}^T L_{I'}^+ (b_{us'}+b_{vs'})/2}{b_{ss'}^T L_{I'}^+ b_{ss'}} \in [2^{-i-2}, 2^{-i-1})\}$
  end
  $i_{\max} \leftarrow 2 \log n$
  $X_{\text{low}} \leftarrow \cup_{i > i_{\max}} X_i$
  $Y_{\text{low}} \leftarrow \cup_{i > i_{\max}} Y_i$
  $W \leftarrow \arg\max_{W \in \{X_0, Y_0, \ldots, X_{i_{\max}}, Y_{i_{\max}}, X_{\text{low}}, Y_{\text{low}}\}} |W|$
  $i \leftarrow$ index of W as X_i or Y_i, with $i \leftarrow i_{\max} + 1$ if W is low
  $p \leftarrow 2^{-i-2}$
  if $|W| \leq 10000 (\log n) \xi_{buckets}^2 \rho^3 \epsilon^{-2} (\Delta^{I \setminus D}(S, S') + \Delta^{I \setminus D}(S', S) + |D|) 2^i$ **then**
    `// defer all edges in` W
    $F \leftarrow F \setminus W$
    Add edges of W to A if $W = X_i$, otherwise add W to B
  else
    Remove all edges e from W for which $\texttt{lev}_{I \setminus D}(e) - \texttt{lev}_{I/(S,S')}(e) > 1/32$ using Johnson-Lindenstrauss with $\epsilon = 1/64$
    $Z \leftarrow$ `Oracle`(I, S, S', D, A, B, W)
    $Z' \leftarrow$ a uniformly random subset of Z with size $K(|W|)$
    `// can be implemented using` J `in` $O(|Z'|)$ `time`
    $I \leftarrow I[[Z']]$
    $F \leftarrow F \cap E(I)$
  end
end
$F' \leftarrow$ set of edges in $A \cup B$ that have some endpoint with normalized I' potential between $\epsilon/2$ and $1 - \epsilon/2$
return F'

Our analysis of this algorithm centers around applying Proposition 4.10.5. Specifically, Proposition 4.10.5 is applied to show that the electrical functions

- $\delta_{S,S'}(H \setminus D)$
- $\delta_{S',S}(H \setminus D)$
- $b_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_{ss'}$
- $\left(\sum_{u\in V(A)} b_{su}^T L^+_{(H\setminus D)/(S,S')} b_{ss'}\right) + \left(\sum_{v\in V(B)} b_{vs'}^T L^+_{(H\setminus D)/(S,S')} b_{ss'}\right)$

do not change much over the course of conditioning on many edges. We focus our discussion of intuition on the first two functions, since the others are easier. The first two functions are very similar, so we focus our discussion on the first one. Furthermore, bounding ρ_E is in Proposition 4.10.5 is more difficult than bounding ρ_L, so we focus on that.

To bound expected changes in $\delta_{S,S'}(H \setminus D)$, it helps to define a quantity related to the discrepancy between leverage scores in H (the graph a tree is being sampled in) and $(H \setminus D)/(S, S')$ (the graph in which quantities of interest are defined):

Definition 4.10.9 (Leverage score change)**.** *Consider an edge e in a graph H obtained by identifying vertices and deleting edges in a graph G. Define*

$$\texttt{levcng}_{G\to H}(e) = \frac{r_e - b_e^T L_G^+ b_e}{r_e - b_e^T L_H^+ b_e} - \frac{b_e^T L_G^+ b_e}{b_e^T L_H^+ b_e} = \frac{r_e(b_e^T L_H^+ b_e - b_e^T L_G^+ b_e)}{(r_e - b_e^T L_H^+ b_e) b_e^T L_H^+ b_e}$$

It is also helpful to define the following maximum energy fraction, which is used to define a generalization of the fact that the effective resistance is the sum of energies on all edges:

Definition 4.10.10 (Maximum energy fraction)**.** *For a graph H, a set $Y \subseteq V(H)$, a vertex $x_0 \notin Y$, and some edge $f \in E(X)$, let*

$$\alpha^H_{x_0,Y}(f) = \max_{w\in Y} \frac{(b_{x_0w}^T L_H^+ b_f)^2}{(b_{x_0w}^T L_H^+ b_{x_0w}) r_f}$$

Remark 10. *When $\gamma \le \texttt{lev}_H(e) \le 1 - \gamma$ for some $\gamma \in [0, 1/2]$,*

$$|\texttt{lev}_G(e) - \texttt{lev}_H(e)| \le |\texttt{levcng}_{G\to H}(e)| \le 2|\texttt{lev}_G(e) - \texttt{lev}_H(e)|/\gamma$$

Consider any edge $f \in E(H) \setminus D$. One can write the expected change in $\delta_{S,S'}(H \setminus D)$ after conditioning using rank-one updates:

$$\mathbf{E}_{H'\sim H[f]}[\delta_{S,S'}(H'\setminus D)] - \delta_{S,S'}(H\setminus D)$$
$$= -\texttt{levcng}_{H\to(H\setminus D)/S}(f)\sum_{w\in S'}\sum_{e\in\partial_H w}\left(\frac{(b_{sw}^T L^+_{(H\setminus D)/S}b_f)^2}{r_f}\right)\left(\frac{b_{ss'}^T L^+_{(H\setminus D)/(S,S')}b_e}{r_e}\right)$$
$$-\texttt{levcng}_{H\to(H\setminus D)/(S,S')}(f)\sum_{w\in S'}\sum_{e\in\partial_H w}\left(b_{sw}^T L^+_{(H\setminus D)/S}b_{sw}\right)\left(\frac{(b_{ss'}^T L^+_{(H\setminus D)/(S,S')}b_f)(b_f^T L^+_{(H\setminus D)/(S,S')}b_e)}{r_f r_e}\right)$$
$$+\left(\frac{\texttt{lev}_H(f)}{\texttt{lev}_{(H\setminus D)/S}(f)\texttt{lev}_{(H\setminus D)/(S,S')}(f)}+\frac{1-\texttt{lev}_H(f)}{(1-\texttt{lev}_{(H\setminus D)/S}(f))(1-\texttt{lev}_{(H\setminus D)/(S,S')}(f))}\right)$$
$$\sum_{w\in S'}\sum_{e\in\partial_H w}\left(\frac{(b_{sw}^T L^+_{(H\setminus D)/S}b_f)^2}{r_f}\right)\left(\frac{(b_{ss'}^T L^+_{(H\setminus D)/(S,S')}b_f)(b_f^T L^+_{(H\setminus D)/(S,S')}b_e)}{r_f r_e}\right)$$

By the triangle inequality and the definition of $\alpha_{s,S'}^{(H\setminus D)/S}(f)$,

$$|\mathbf{E}_{H'\sim H[f]}[\delta_{S,S'}(H'\setminus D)] - \delta_{S,S'}(H\setminus D)|$$
$$\le |\texttt{levcng}_{H\to(H\setminus D)/S}(f)|\sum_{w\in S'}\sum_{e\in\partial_H w}\left(\frac{(b_{sw}^T L^+_{(H\setminus D)/S}b_f)^2}{r_f}\right)\left(\frac{b_{ss'}^T L^+_{(H\setminus D)/(S\cup S')}b_e}{r_e}\right)$$
$$|\texttt{levcng}_{H\to(H\setminus D)/(S,S')}(f)|\sum_{w\in S'}\sum_{e\in\partial_H w}\left(b_{sw}^T L^+_{(H\setminus D)/S}b_{sw}\right)\left(\frac{|b_{ss'}^T L^+_{(H\setminus D)/(S,S')}b_f||b_f^T L^+_{(H\setminus D)/(S,S')}b_e|}{r_f r_e}\right)$$
$$+\left(\frac{\texttt{lev}_H(f)}{\texttt{lev}_{(H\setminus D)/S}(f)\texttt{lev}_{(H\setminus D)/(S,S')}(f)}+\frac{1-\texttt{lev}_H(f)}{(1-\texttt{lev}_{(H\setminus D)/S}(f))(1-\texttt{lev}_{(H\setminus D)/(S,S')}(f))}\right)$$
$$\alpha_{s,S'}^{(H\setminus D)/S}(f)\sum_{w\in S'}\sum_{e\in\partial_H w}\left(b_{sw}^T L^+_{(H\setminus D)/S}b_{sw}\right)\left(\frac{|b_{ss'}^T L^+_{(H\setminus D)/(S,S')}b_f||b_f^T L^+_{(H\setminus D)/(S,S')}b_e|}{r_f r_e}\right)$$

The stable oracle guarantees can be used on all of the quantities in the above sum. The "$S - S'$ normalized degree change stability" guarantees bound the double sums in each of the three above terms. The "Leverage score stability" guarantees bound the leverage score quantity in the second order term. Initially, the `levcng` and α quantities may seem trickier to bound. Luckily, we can prove bounds on their average values in terms of $\Delta^{H\setminus D}(U_S, S') + \Delta^{H\setminus D}(U_{S'}, S)$, where U_S and U'_S are the sets of vertices with normalized potential less than $1-p$ and greater than p respectively, with s and s' assigned to 0 and 1 respectively. These Δ quantities are in turn bounded using the "Midpoint potential stability" guarantee, along with the following:

Proposition 4.10.11. *Consider two disjoint sets of vertices X and Y in a graph G. Let $G' = G/(X \cup Y)$, with x and y the identifications of X and Y respectively. Let Z be the set of vertices v with electrical potential $p_v \leq \gamma$ for some $\gamma \in (0,1)$ with boundary conditions $p_x = 0$ and $p_y = 1$. Then*

$$\Delta^G(Z,Y) \leq \frac{1}{1-\gamma}\Delta^G(X,Y)$$

where z is the identification of Z in G/Z.

This bound on Δ is then used in conjunction with the following to bound `levcng` and α:

Lemma 4.10.12 (Bounding first order terms, part 1)**.** *Consider a graph G and two sets $X, Y \subseteq V(G)$ with $X \cap Y = \emptyset$. Let $H = G/Y$ with y the identification of Y in G. Let $\Delta = \Delta^G(X,Y)$. Then*

$$\sum_{f \in G[X] \cup \partial_G X : 1/4 \leq \texttt{lev}_G(f) \leq 3/4} |\texttt{levcng}_{G \to H}(f)| \leq 32\Delta$$

Lemma 4.10.13 (Bounding first order terms, part 2)**.** *Consider a graph G and a set of edges $D \subseteq E(G)$. Then*

$$\sum_{e \in E(G) \setminus D : 1/4 \leq \texttt{lev}_G(e) \leq 3/4} |\texttt{levcng}_{G \to G \setminus D}(e)| \leq 4|D|$$

Lemma 4.10.14 (Bounding the second order term)**.** *Consider a graph G and two disjoint sets of vertices $X, Y \subseteq V(G)$. For any $s \in X$,*

$$\sum_{e \in E_G(X) \cup \partial_G X} \alpha^G_{s,Y}(e) = \sum_{e \in E_G(X) \cup \partial_G X} \max_{t \in Y} \frac{(b_{st}^T L_G^+ b_e)^2}{(b_{st}^T L_G^+ b_{st}) r_e} \leq 24 \xi^2_{buckets} \Delta^G(X,Y)$$

where $\xi_{buckets} = \log(m\alpha)$.

Appendix B.5.1 is dedicated towards proving these bounds. Notice that in Line 20, the algorithm chooses edges to condition on randomly. This allows us to exploit the above bounds in order to eliminate the `levcng` and α dependencies. This completes the outline of the proof of the bound on ρ_E for the normalized degree and illustrates how we prepare to apply Proposition 4.10.5 in general.

We now state all of the stability results that a stability oracle implies in the algorithm `Fix`. These are shown in Appendix B.5.2. These propositions analyze of Lines 20-22 by conditioning on a uniformly random sample one edge at a time.

Definition 4.10.15 (Stability propositions setup)**.** *Let $Z_0 = Z \leftarrow$ `Oracle`(I, S, S', D, A, B, W) and $I_0 = I$, where `Oracle` is (ρ, K)-stable. Assume that the input to `Oracle` satisfies the conditions described in Definition 4.10.7.*

Obtain Z_k and I_k for $k > 0$ by choosing a uniformly random edge $f_{k-1} \in Z_{k-1}$, letting $Z_k \leftarrow Z_{k-1} \backslash \{f_{k-1}\}$, and letting $I_k \leftarrow I_{k-1}[[f_{k-1}]]$. Let $\Delta_k := \Delta^{I_k \backslash D}(S, S') + \Delta^{I_k \backslash D}(S', S) + |D|$.

Notice that the set Z' defined in the algorithm `Fix` could analogously be defined by letting $Z' \leftarrow Z \setminus Z_K$. We now show that the following stability properties hold:

Proposition 4.10.16 (Stability with respect to Δ)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), 0)$-stable subset of W for the electrical functions*

$$\delta_{S,S'}(H \setminus D)$$

and

$$\delta_{S',S}(H \setminus D)$$

of H with high probability.

Proposition 4.10.17 (Stability with respect to sums of deferred potentials)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), r_{min}/n^4)$-stable subset of W for the electrical function*

$$\left(\sum_{\{u,v\} \in A} b_{ss'}^T L^+_{(H \backslash D)/(S,S')} (b_{su} + b_{sv}) \right) + \left(\sum_{\{u,v\} \in B} b_{ss'}^T L^+_{(H \backslash D)/(S,S')} (b_{us'} + b_{vs'}) \right)$$

of H with high probability.

Proposition 4.10.18 (Stability with respect to the main objective)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), 0)$-stable subset of W for the electrical function*

$$b_{ss'}^T L^+_{(H \backslash D)/(S,S')} b_{ss'}$$

of H with high probability.

Now, we use Proposition 4.10.5 to show the following result:

Proposition 4.10.19. *Immediately before Line 25 of the algorithm `Fix`, the graph I and the sets A and B have the following properties with high probability:*

- *(Main objective) Let I_0 be the graph supplied as input to `Fix`. Then $b_{ss'}^T L^+_{I \backslash D} b_{ss'} \geq (1 - \epsilon) b_{ss'}^T L^+_{I_0 \backslash D} b_{ss'}$*

- *(Normalized potentials of deferred edges are not too high on average)*

$$\frac{1}{b_{ss'}^T L_{I\setminus D}^+ b_{ss'}} \left(\left(\sum_{\{u,v\}\in A} b_{ss'}^T L_{I\setminus D}^+ (b_{su} + b_{sv}) \right) + \left(\sum_{\{u,v\}\in B} b_{ss'}^T L_{I\setminus D}^+ (b_{us'} + b_{vs'}) \right) \right)$$
$$\leq \tilde{O}(\rho^3 \epsilon^{-2} (\Delta^{I_0\setminus D}(S, S') + \Delta^{I_0\setminus D}(S', S) + |D|))$$

The first condition of the above proposition states that the value that `Fix` needs to preserve ($b_{ss'}^T L_{I\setminus D}^+ b_{ss'} = \frac{1}{c^{I\setminus D}(S,S')}$) is in fact similar to its value at the beginning of the algorithm. This is not enough, however, because the deferred edges have not been conditioned on before Line 25. The second condition of Proposition 4.10.19 ensures that contracting or deleting all but $\tilde{O}((\Delta^{I\setminus D}(S, S') + \Delta^{I\setminus D}(S', S) + |D|)\epsilon^{-1})$ of the deferred edges does not decrease the main objective by more than a factor of $1 - \epsilon$. Setting the remaining edges to be F' shows the desired result.

We now show Proposition 4.10.19:

Proof of Proposition 4.10.19. Number of "If" block visits. Start by bounding the number of times the "If" block in the algorithm `Fix` is entered. Each time the "If" statement is entered, Line 15 removes W from F. By construction, $|W| \geq |F|/(2i_{\max}) = |F|/(2\log n)$. Therefore, this removal can only take place $(\log |F|)(2\log n) \leq 2\log^2 n$ times, so the "If" statement only executes $2\log^2 n$ times over the course of `Fix`.

Verifying stable oracle input conditions and size of W. After Line 18, W only contains edges for which $|\texttt{lev}_{I\setminus D}(e) - \texttt{lev}_{I/(S,S')}(e)| \leq 1/32 + 2(1/64) \leq 1/16$. Therefore, W satisfies the "Bounded leverage score difference" condition. W satisfies one of the "Narrow potential neighborhood" conditions by definition of the X_is and Y_is.

Now, we lower bound the size of W after Line 18. Let U_S and $U_{S'}$ be the vertices with $(I \setminus D)/(S, S')$-normalized potentials greater than p and less than $1 - p$ respectively, with $s \leftarrow 0$ and $s' \leftarrow 1$. By Proposition 4.10.11 with $\gamma \leftarrow 1 - p$,

$$\Delta^{I\setminus D}(U_S, S') \leq \Delta^{I\setminus D}(S, S')/p$$

and

$$\Delta^{(I\setminus D)/S'}(U_{S'}, S) \leq \Delta^{I\setminus D}(U_{S'}, S) \leq \Delta^{I\setminus D}(S', S)/p$$

Every edge in W has an endpoint in $U_S \cap U_{S'}$ by the bucketing definition. Applying Lemma 4.10.12 twice and the first inequality of Remark 10 shows that

$$\sum_{e\in W} \texttt{lev}_{I\setminus D}(e) - \texttt{lev}_{(I\setminus D)/(S,S')}(e) \leq 32(\Delta^{I\setminus D}(S, S') + \Delta^{I\setminus D}(S', S))/p$$

By Lemma 4.10.13 and Remark 10,

$$\sum_{e\in W} \mathtt{lev}_{(I\setminus D)/(S,S')}(e) - \mathtt{lev}_{I/(S,S')}(e) \le |D|$$

Therefore, by Rayleigh monotonicity,

$$\sum_{e\in W} |\mathtt{lev}_{I\setminus D}(e) - \mathtt{lev}_{I/(S,S')}(e)| \le 32(\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|)/p$$

By the "If" condition,

$$|W| \ge 4096(\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|)/p$$

before Line 18. In particular,

$$\sum_{e\in W} |\mathtt{lev}_{I\setminus D}(e) - \mathtt{lev}_{I/(S,S')}(e)| \le |W|/128$$

which means that Line 18 can only remove $(|W|/128)/(1/32) \le |W|/4$ edges from W. Therefore, Line 18 only decreases the size of W by a factor of 3/4. In particular, $|W| \ge \tilde{\Omega}(\rho^3\epsilon^{-2}(\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|)/p)$ after Line 18.

Concentration in "Else"-only intervals. Now, define the functions

$$x_0(H) := b_{ss'}^T L_{(H\setminus D)/(S,S')} b_{ss'} = 1/c^{H\setminus D}(S,S')$$

$$x_1(H) := \delta_{S,S'}(H \setminus D)$$

$$x_2(H) := \delta_{S',S}(H \setminus D)$$

and

$$y_0(H) := \left(\sum_{\{u,v\}\in A} b_{ss'}^T L_{H\setminus D}^+ (b_{su} + b_{sv})\right) + \left(\sum_{\{u,v\}\in B} b_{ss'}^T L_{H\setminus D}^+ (b_{us'} + b_{vs'})\right)$$

All of these functions are electrical functions for the graph H, as they are preserved under splitting edges in series and in parallel.

We now check that Proposition 4.10.5 can be applied between "If" block executions to show that x_0, x_1, x_2, and y_0 do not change by more than a $(1 + \epsilon/(2\log^2 n))$-factor over each "Else"-only interval. Line 20 is equivalent picking uniformly random edges from Z' without replacement K times. Let $\Delta := \Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|$. By Propositions 4.10.16,4.10.17, and 4.10.18, each of the selected edges is sampled from a set that is a $(\tilde{O}(\rho), \tilde{O}(\rho(\Delta/p)), 0)$-stable subset of W for the x_is and $(\tilde{O}(\rho), \tilde{O}(\rho(\Delta/p)), r_{min}/n^4)$-stable for y_0. Therefore, we may apply Proposition 4.10.5 with

$$\begin{aligned}\rho_E(\{x_a(H)\}_a) &\leftarrow \tilde{O}(\rho(\Delta^{H\setminus D}(S,S') + \Delta^{H\setminus D}(S',S) + |D|)/p)) \\ &= \tilde{O}(\rho(\delta_{S,S'}(H\setminus D) + \delta_{S',S}(H\setminus D))/(pb_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_{ss'})) \\ &= \tilde{O}(\rho(x_1(H) + x_2(H))/(px_0(H)))\end{aligned}$$

$$\rho_L \leftarrow \tilde{O}(\rho)$$

$$\delta \leftarrow r_{min}/n^4$$

and

$$\sigma \leftarrow \tilde{\Omega}(\rho^3/\epsilon^2)$$

because ρ_E is a 1-multlipschitz function in its inputs. In order to apply Proposition 4.10.5, we need to bound τ_E. By the "If" statement and the "Size of W" bound earlier in this proof, $\tau_E \leq \epsilon/(100\log^2 n)$ and $\sqrt{\gamma\ell_1}\rho_L^{3/2}/\sqrt{\sigma} \leq \epsilon/(100\log^2 n)$. Therefore, Proposition 4.10.5 implies that each x_i function changes by at most a factor of $(1+\epsilon/(8\log^2 n))$ during each "Else"-only interval. Furthermore, y_0 changes by at most an $(1+\epsilon/(8\log^2 n))$ factor during each interval, along with an additive change of at most $\tilde{O}(\rho r_{min}|F|/n^4) \leq r_{min}/n^3$. This is the desired change in each "Else"-interval.

Main objective. Each "Else" interval causes $x_0(H) = b_{ss'}^T L^+_{H\setminus D} b_{ss'}$ to change by a factor of at most $(1+\epsilon/(8\log^2 n))$. By "Number of If block visits," there are at most $2\log^2 n$ of these intervals. Therefore, $x_0(I) \geq (1-\epsilon/(8\log^2 n))^{2\log^2 n} x_0(I_0) \geq (1-\epsilon)x_0(I_0)$. In particular, $b_{ss'}^T L^+_{I\setminus D} b_{ss'} \geq (1-\epsilon) b_{ss'}^T L^+_{I_0\setminus D} b_{ss'}$, as desired.

Normalized potentials are not too high on average. Each "Else"interval causes the quantity $\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S)$ to increase by a factor of at most $(1+\epsilon/(8\log^2 n))$ by Proposition 4.10.5 applied to all of the x_is. Therefore, the total increase over the course of the entire algorithm is at most a factor of $(1+\epsilon) \leq 2$, since there are at most $2\log^2 n$ "Else" intervals. Therefore, I in the bound of the "If" statement can be replaced with I_0 with at most a factor of 2 increase in the value.

Each "If" statement adds at most $\tilde{O}(\rho^3\epsilon^{-2}(\Delta^{I_0\setminus D}(S,S') + \Delta^{I_0\setminus D}(S',S) + |D|)2^i)$ edges to A or B. Each of these edges contributes at most $2^{-i-2}(b_{ss'}^T L^+_{I\setminus D} b_{ss'}) \leq 2^{-i-1}(b_{ss'}^T L^+_{I_0\setminus D} b_{ss'})$ (by Proposition 4.10.5) to the sum, so each "If" block increases y_0 by at most

$$\begin{aligned}&\tilde{O}(\rho^3\epsilon^{-2}(\Delta^{I_0\setminus D}(S,S') + \Delta^{I_0\setminus D}(S',S) + |D|)2^i)2^{-i-1}(b_{ss'}^T L^+_{I_0\setminus D} b_{ss'}) \\ &\leq \tilde{O}(\rho^3\epsilon^{-2}(\Delta^{I_0\setminus D}(S,S') + \Delta^{I_0\setminus D}(S',S) + |D|))(b_{ss'}^T L^+_{I_0\setminus D} b_{ss'})\end{aligned}$$

additively. Each "Else" statement increases the value of y_0 by at most a factor of $(1 + \epsilon/(8\log^2 n))$ along with an additive increase of at most r_{min}/n^3. Therefore, the value of y_0 immediately before Line 25 is at most $\tilde{O}(\rho^3\epsilon^{-2}(\Delta^{I_0\setminus D}(S,S')+\Delta^{I_0\setminus D}(S',S)+|D|))(b_{ss'}^T L_{I_0\setminus D}^+ b_{ss'})$, as desired. Dividing both sides by $(b_{ss'}^T L_{I_0\setminus D}^+ b_{ss'})$ and using the concentration of x_0 gives the desired result. □

We now use this proposition, along with the following result:

Proposition 4.10.20. *Consider two disjoint sets of vertices X and Y in a graph G. Let $G' = G/(X,Y)$, with x and y the identifications of X and Y respectively. Let A and B be sets of edges for which both endpoints have normalized $L_{G'}^+ b_{xy}$ potential at most γ and at least $1-\gamma$ respectively for some $\gamma \in (0, 1/2)$. Arbitrarily contract and delete edges in A and B in G to obtain the graph H. Then*

$$c^H(X,Y) \leq \frac{1}{(1-\gamma)^2} c^G(X,Y)$$

to show Lemma 4.10.8 using the intuition given earlier:

Proof of Lemma 4.10.8. Correctness and runtime of conditioning. The conditioning process given in `Fix` can be written in the form described in Proposition 4.9.9. Therefore, by Proposition 4.9.9, its output is equivalent in distribution to sampling from $I[F]$. Furthermore, Line 21 can be implemented using the sample J in constant time per edge by Proposition 4.9.8.

Conductance. By the "Main objective" condition, the $s-s'$ effective resistance in $I\setminus D$ is at least $(1-\epsilon)$ times its original value right before Line 25. Contracting or deleting $(A\cup B)\setminus F'$ in I only decreases the resistance by a factor of at most $(1-\epsilon/2)^2 \geq (1-\epsilon)$ by Proposition 4.10.20. By Rayleigh monotonicity, deleting F' only increases the $s-s'$ resistance. The total decrease due to all of these changes is at most a factor of $(1-\epsilon)^2 \geq 1 - O(\epsilon)$. Taking reciprocals and using the fact that $c^{J\setminus D\setminus F'}(S,S') = 1/(b_{ss'}^T L_{J\setminus D\setminus F'}^+ b_{ss'})$ yields the desired result.

Size. By Markov's Inequality and the "Normalized potentials of deferred edges are not too high on average" guarantee of Proposition 4.10.19, only $\tilde{O}(\rho^3\epsilon^{-3}(\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|))$ edges in $A \cup B$ can have any endpoint with normalized potential in the interval $[\epsilon/2, 1-\epsilon/2]$. Therefore, $|F'| \leq \tilde{O}(\rho^3\epsilon^{-3}(\Delta^{I\setminus D}(S,S') + \Delta^{I\setminus D}(S',S) + |D|))$, as desired.

Runtime. Each "While" loop iteration takes $\tilde{O}(m + \mathcal{T}(\texttt{Oracle}))$ time, so it suffices to bound the number of "While" loop iterations. Each "If" block "While" loop iteration reduces the size of $|F|$ by a factor of $1 - 1/(i_{\max}) \leq 1 - 1/(2\log n)$, so only $O(\log^2 n)$ such iterations can occur. Each "Else" block "While" loop iteration decreases the size of F in expectation by at least $K(|F|)/4$ by Proposition 4.9.6. By Chernoff bounds, for $K(|F|) \geq \text{polylog}(n)$, F decreases in size by at least $K(|F|)/8$ with high probability. Therefore, each "Else" iteration reduces the size of $|F|$ by a factor of $(1 - \frac{K(|F|)}{|F|})$ with high probability. Therefore, only

$\tilde{O}(|F|/K(|F|))$ "Else" iterations can occur. All "While" loop iterations either call "If" or "Else," so we have finished the runtime analysis. □

4.10.4 Warmup: A (polylog(n), 1)-stable oracle

We now give a stable oracle `SlowOracle` that suffices for proving Lemma 4.10.1. `SlowOracle` is similar in concept to picking the minimum energy edge in W. However, it needs to do so for multiple functions simultaneously and for more complicated functions than effective resistances. To cope with these complexities, we exploit Theorem 4.9.12 in place of the fact that the sum of the energies on edges is the effective resistance.

Algorithm 15: `SlowOracle`(I, S, S', D, A, B, W), never executed

1 Return all edges in W that satisfy all of the inequalities in the "$S - S'$-normalized degree change stability," "Deferred endpoint potential change stability," and "Main objective change stability" guarantees of `Oracle` with $\rho = 400 \log(\beta n)$

Proposition 4.10.21. *There is a* $(400 \log(\beta n), 1)$*-stable oracle* `SlowOracle`*.*

Proof. **Everything besides size of Z.** Let $Z \leftarrow$ `SlowOracle`(I, S, S', D, A, B, W). By construction, all edges in Z (in particular f_0) satisfy the "$S - S'$-normalized degree change stability," "Deferred endpoint potential change stability," and "Main objective change stability" guarantees of `Oracle`. The "Midpoint potential stability" and "Leverage score stability" guarantees follow from the fact that $K = 1$.

Size of Z. In each of the "Conductance term stability" quantities, the edge e has s' as an endpoint in the graph $(I_0 \setminus D)/(S, S')$. Furthermore, in the "Deferred endpoint potential change stability" quantity, $b_{su} + b_{sv}$ has one source that is the same as $b_{ss'}$'s source. Similarly, $b_{us'} + b_{vs'}$ and $b_{ss'}$ have the same sink. Therefore, Theorem 4.9.12 applies with $\tau = \beta^{12} n^{12}$ and shows that

$$\begin{aligned}
&\sum_{f\in W}\sum_{w\in S'}(b_{sw}^T L^+_{(I_0\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_0} w} \frac{|b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_f||b_f^T L^+_{(I_0\setminus D)/(S,S')} b_e|}{r_f r_e} \\
&= \sum_{w\in S'}(b_{sw}^T L^+_{(I_0\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_0} w}\sum_{f\in W} \frac{|b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_f||b_f^T L^+_{(I_0\setminus D)/(S,S')} b_e|}{r_f r_e} \\
&\le \sum_{w\in S'}(b_{sw}^T L^+_{(I_0\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_0} w} \left((12\log(n\beta))\frac{b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_e}{r_e} + \frac{n^2 r_{max}}{r_e \beta^{12} n^{12}}\right) \\
&\le \sum_{w\in S'}(b_{sw}^T L^+_{(I_0\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_0} w} \left((24\log(n\beta))\frac{b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_e}{r_e}\right) \\
&= 24\log(n\beta)\delta_{S,S'}(I\setminus D)
\end{aligned}$$

where the second-to-last inequality follows from the fact that effective resistances in any graph are within a factor of $n^2\beta$ of one another. Similarly, all five of the following inequalities hold:

$$\sum_{f\in W}\sum_{w\in S}(b_{s'w}^T L^+_{(I_0\setminus D)/S} b_{s'w}) \sum_{e\in\partial_{I_0} w} \frac{|b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_f||b_f^T L^+_{(I_0\setminus D)/(S,S')} b_e|}{r_f r_e} \le 24\log(n\beta)\delta_{S',S}(I\setminus D)$$

$$\sum_{f\in W}\sum_{w\in S'} \frac{(b_{sw}^T L^+_{(I_0\setminus D)/S} b_f)^2}{r_f} \sum_{e\in\partial_{I_0} w} \frac{b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_e}{r_e} \le \delta_{S,S'}(I\setminus D)$$

$$\sum_{f\in W}\sum_{w\in S} \frac{(b_{s'w}^T L^+_{(I_0\setminus D)/S'} b_f)^2}{r_f} \sum_{e\in\partial_{I_0} w} \frac{b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_e}{r_e} \le \delta_{S',S}(I\setminus D)$$

$$\begin{aligned}
&\sum_{f\in W}\left(\sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_f||b_f^T L^+_{(I_0\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_f}\right) \\
&+ \sum_{f\in W}\left(\sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_0\setminus D)/(S,S')} b_f||b_f^T L^+_{(I_0\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_f}\right) \\
&\le 12\log(\beta n) \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv}) \\
&+ 12\log(\beta n) \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'}) + \frac{r_{min}}{n^6}
\end{aligned}$$

$$\sum_{f \in W} \frac{(b_{ss'}^T L^+_{(I_0 \setminus D)/(S,S')} b_f)^2}{r_f} \leq b_{ss'}^T L^+_{(I \setminus D)/(S,S')} b_{ss'}$$

By Markov's Inequality, only $|W|/16$ edges in W can violate any one of the six conditions tested in `SlowOracle`. Therefore, $|Z| \geq |W| - 6|W|/16 \geq |W|/2$, as desired.

□

Proof of Lemma 4.10.1. Follows directly from Lemma 4.10.8, with `SlowOracle` substituted in for `Oracle` by Proposition 4.10.21. □

4.11 Efficient construction for the conductance concentration inequality

To accelerate `Fix`, we need to construct an almost-linear time $(m^{o(1)}, |W|m^{-o(1)})$-stable oracle. To do this, we need to do the following:

- Compute a large subset of W consisting of edges that respect all of the conditions of stable oracles in almost-linear time.
- Show that this large subset of W continues to satisfy the stable conditions even after conditioning on a significant fraction of W.

The first objective boils down to computing approximations to moments of Laplacian inner products. This can be done using techniques from streaming algorithms; for example [6, 44]. Specifically, we use Theorem 4.9.13. The second objective boils down to showing that a set of stable edges remains stable for many iterations. To show this, we use Theorem 5.4.1.

4.11.1 Exploiting localization

Concentration preliminaries

In this subsection, it is helpful to have a few concentration inequalities that allow us to control the ℓ_∞ norm of certain vectors. We apply Theorem 4.9.11 to obtain two concentration inequalities that will be applied directly:

Proposition 4.11.1. *Let $\{M^{(k)}\}_k \in \mathbb{R}^{n\times n}$ be a sequence of symmetric, nonnegative random matrices, $\{Z^{(k)}\}_k \in \{0,1\}^n$, and $S^{(k)} \subseteq [n]$ with the following properties:*

- *For all $i \in [n]$, $\sum_{j=1,j\neq i}^n M_{ij}^{(0)} \leq \sigma$ where $\sigma \leq \sigma_0$. Furthermore, $S^{(0)} = \emptyset$.*
- *The random variables $\{Z^{(k)}\}_k$ are defined by making $Z^{(k+1)}$ the indicator of a uniformly random choice $w^{(k+1)} \in [n] \setminus S^{(k)}$. Let $S^{(k+1)} := S^{(k)} \cup \{w^{(k+1)}\}$.*
- *For all $i, j \in [n]$ and k, $M_{ij}^{(k+1)} \leq M_{ij}^{(k)} + \gamma \sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)} M_{lj}^{(k)}$.*

With probability at least $1 - 1/n^8$,

$$\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \leq \sigma_1$$

for all $i \notin S^{(k)}$ and all $k \leq n/2$.

We also need a bound on how M affects a random vector v:

Proposition 4.11.2. *Let $\{M^{(k)}\}_k \in \mathbb{R}^{n\times n}$ be a sequence of symmetric, nonnegative random matrices, $\{v^{(k)}\}_k \in \mathbb{R}^n$ be a sequence of nonnegative random vectors, $\{Z^{(k)}\}_k \in \{0,1\}^n$, and $\{S^{(k)}\}_k \subseteq [n]$ with the following properties:*

- *For all $i \in [n]$ and all k, $\sum_{j\neq i, j\notin S^{(k)}} M^{(k)}_{ij} \leq \sigma_1$.*
- *The random variables $\{Z^{(k)}\}_k$ are defined by making $Z^{(k+1)}$ the indicator of a uniformly random choice $w^{(k+1)} \in [n] \setminus S^{(k)}$. Let $S^{(k+1)} := S^{(k)} \cup \{w^{(k+1)}\}$.*
- *For all $i \in [n]$, $v^{(0)}_i \leq \tau$.*
- *For all $i \in [n]$ and k, $v^{(k+1)}_i \leq v^{(k)}_i + \gamma \sum_{l=1}^n M^{(k)}_{il} Z^{(k+1)}_l (v^{(k)}_l + v^{(k)}_i)$.*

With probability at least $1 - 1/n^8$,

$$v^{(k)}_i \leq \tau_1$$

for all $i \notin S^{(k)}$ and all $k \leq n/2$.

We prove both of these propositions in Appendix B.6.1.

Flexible functions

We now discuss how to exploit the concentration inequalities from the previous subsubsection to make stable sets of edges remain stable for an almost-linear number of iterations. The bounds below are motivated by applications of Sherman-Morrison.

Definition 4.11.3 (Flexible families of functions)**.** *Let G' be a graph and consider any minor H of G'. Let $X \subseteq E(G')$ and consider a family of electrical functions $\{g_e\}_{e\in X}$ on minors of G' along with a function ϕ that maps minors H of G' to graphs with a subset of the edges in H. This family is called* flexible *if for any graph H and any edges $e, f \in X \cap E(H)$, all of the following hold:*

- *(ϕ commutes with modifications) For any minor H of G' and any edge $f \in E(H)$, $\phi(H/f) = \phi(H)/f$ and $\phi(H\backslash f) = \phi(H)\backslash f$.*
- *(Contractions) $|g_e(H/f) - g_e(H)| \leq \frac{|b_e^T L^+_{\phi(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} \frac{3}{(\mathtt{lev}_{\phi(H)}(f))^2}(g_f(H) + g_e(H))$*
- *(Deletions) $|g_e(H\backslash f) - g_e(H)| \leq \frac{|b_e^T L^+_{\phi(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} \frac{3}{(1-\mathtt{lev}_{\phi(H)}(f))^2}(g_f(H) + g_e(H))$*

Crudely controlling flexible functions

We now make use of Theorem 5.4.1 to find a set of edges for which flexible functions do not increase much if one splits and conditions for a linear number of iterations.

In the `Fix` algorithm, edges were split in one of two ways before conditioning on them. This ensured that they had a leverage score that was bounded away from both 0 and 1. In order to decide which way to split an edge, the algorithm must approximately know its effective resistance. Naively, this requires recomputing approximate effective resistances during each iteration. One can avoid recomputation for a linear number of iterations, however, by showing the following:

Proposition 4.11.4. *Given any set $X \subseteq E(G')$ in some graph G' along with a graph G'' on a subset of the edges of G' with the following property:*

- *Each edge $e \in X$ has $\texttt{lev}_{G''}(e) \in [1/4, 3/4]$*

there is a set $Y \leftarrow \texttt{VeryStable}_{G'}(G'', X)$ with $Y \subseteq X$ and the following additional properties:

- *(Size) $|Y| \geq |X|/\text{polylog}(n)$ with probability at least $\sigma/(16O(\log^2 n))$.*
- *(Leverage score bound) Pick a subset $Y_0 \subseteq Y$ and sample a random sequence of edges $f_0, f_1, \ldots, f_{|Y_0|/2}$ without replacement. For any integer $0 \leq i \leq |Y_0|/2$, let G''_i denote the graph obtained by arbitrarily deleting/contracting the edges $f_0, f_1, \ldots, f_{i-1}$ in G''. Then, with probability at least $1 - 1/n^6$,*

$$|\texttt{lev}_{G''_i}(e) - \texttt{lev}_{G''}(e)| \leq \frac{1}{8}$$

  for all $e \in Y_0 \setminus \{f_0, f_1, \ldots, f_{i-1}\}$.

- *(Flexible function bound) For any flexible family of functions $(\{g_e\}_{e \in X}, \phi)$ with $G'' = \phi(G')$,*

$$\max_{e \in Y_0 \setminus \{f_0, f_1, \ldots, f_{i-1}\}} g_e(G'_i) \leq 2 \max_{e \in Y_0} g_e(G')$$

  with probability at least $1 - 1/n^6$.

- *(Runtime) The algorithm takes $\tilde{O}(m + \mathcal{T}(\texttt{ColumnApxPreprocessing}) + |X|\mathcal{T}(\texttt{ColumnApx}))$ time.*

To apply this proposition, one just needs to run it $O((\log^3 n)/\sigma)$ times to obtain the desired set with high probability. We encourage the reader to ignore the approximation aspect of the v_es this is only included for efficiency purposes later on:

Algorithm 16: $\texttt{VeryStable}_{G'}(G'', X)$

Data: An ambient graph G', a graph G'' that stable functions are "defined" in (is the image of ϕ), and a set of edges $X \in E(G')$ for possible conditioning
Result: The set $Y \subseteq X$

$Z \leftarrow$ subset of edges $e \in X$, with each edge independently added to Z with probability $\sigma/(8O(\log^2 n))$
$\texttt{ColumnApxPreprocessing}(G'', Z)$
$Y \leftarrow \emptyset$
foreach $e \in Z$ **do**
  `// multiplicative 2-approximation to the quantity` $\sum_{f \in Z, f \neq e} \frac{|b_e^T L_{G''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$
  $v_e \leftarrow \texttt{ColumnApx}(e)$
  Add e to Y if $v_e \leq \sigma$
end
return Y

Proof. **Size.** By Markov's Inequality and Theorem 5.4.1 applied to the vector $w = 1_X$, the subset $X_0 \subseteq X$ consisting of edges $e \in X$ with $\sum_{f \in X} \frac{|b_e^T L_{G''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 2O(\log^2 n)$ has size at least $|X|/2$. For any edge $e \in X_0$,

$$\begin{aligned}
\mathbf{E}_Z \left[\sum_{f \in Z, f \neq e} \frac{|b_e^T L_{G''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \mid e \in Z \right] &= \mathbf{E}_Z \left[\sum_{f \in Z, f \neq e} \frac{|b_e^T L_{G''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \right] \\
&= \frac{\sigma}{8O(\log^2 n)} \sum_{f \in Z, f \neq e} \frac{|b_e^T L_{G''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \\
&\leq \frac{\sigma}{4}
\end{aligned}$$

where the first equality follows from the fact that edges in X are added to X_0 independently. Therefore, for $e \in X_0$, $\mathbf{E}[v_e | e \in Z] \leq \sigma/2$ and

$$\begin{aligned}\Pr_Z[e \in Y] &= \Pr_Z[e \in Y, e \in Z] \\ &= (1 - \Pr_Z[e \notin Y \mid e \in Z]) \Pr_Z[e \in Z] \\ &= \left(1 - \Pr_Z[v_e > \sigma \mid e \in Z]\right) \Pr_Z[e \in Z] \\ &\geq \frac{1}{2} \Pr_Z[e \in Z] \\ &= \frac{\sigma}{16O(\log^2 n)}\end{aligned}$$

where the inequality follows from Markov. Since $|Y \cap X_0| \leq |X_0|$,

$$\Pr_Z\left[|Y \cap X_0| > \frac{\sigma|X_0|}{(32O(\log^2 n))}\right] |X_0| + \frac{\sigma|X_0|}{(32O(\log^2 n))} \geq \mathbf{E}_Z[|Y \cap X_0|] \geq \frac{\sigma|X_0|}{(16O(\log^2 n))}$$

and

$$\Pr_Z\left[|Y| > \frac{\sigma|X|}{(64O(\log^2 n))}\right] \geq \frac{\sigma}{(32O(\log^2 n))}$$

thus completing the size bound.

Leverage score bound and continued electrical flow sum bound. We now show inductively that for all $k \leq |Y_0|/2$ and $e \in Y_0 \setminus \{f_0, f_1, \ldots, f_{k-1}\}$, both

$$\sum_{f \in Y_0 \setminus \{e, f_0, f_1, \ldots, f_{k-1}\}} \frac{|b_e^T L^+_{G''_k} b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 4\sigma \tag{4.1}$$

and

$$|\mathtt{lev}_{G''_k}(e) - \mathtt{lev}_{G''}| \leq \frac{1}{8} \tag{4.2}$$

where G''_k is a graph obtained by contracting or deleting the edges $f_0, f_1, \ldots, f_{k-1}$ in G''.

We start by checking the base case. The base case for (4.1) follows immediately from the approximation lower bound on v_e. The base case for (4.2) follows from the input condition to Proposition 4.11.4.

Now, we continue on to the inductive step. By the input condition,

$$\frac{1}{4} \leq \mathtt{lev}_{G''}(e) \leq \frac{3}{4}$$

for all $e \in Y_0$. Let $M_{ef}^{(k)} \leftarrow \frac{|b_e^T L_{G_k''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$, $\gamma \leftarrow 8$, and $\sigma \leftarrow 2\sigma$ (last one uses the lower bound for v_e). Notice that

$$M_{ef}^{(k+1)} \leq M_{ef}^{(k)} + \sum_{g \in Y_0} \frac{1}{\min(\mathtt{lev}_{G_k''}(g), \mathtt{nonlev}_{G_k''}(g))} Z_g^{(k+1)} M_{eg}^{(k)} M_{gf}^{(k)}$$

by Sherman-Morrison and the triangle inequality. $Z^{(k+1)}$ is the indicator for the edge f_{k+1}. By the inductive assumption and the triangle inequality, $\min(\mathtt{lev}_{G_k''}(g), \mathtt{nonlev}_{G_k''}(g)) \geq 1/8 = 1/\gamma$. Therefore, Proposition 4.11.1 applies. It shows that for all $k \leq |Y_0|/2$ and all $e \neq f_0, \ldots, f_{k-1}$,

$$\sum_{f \in Y_0 \setminus \{e, f_0, \ldots, f_{k-1}\}} \frac{|b_e^T L_{G_k''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq \sigma_1$$

with probability at least $1 - 1/n^8$. This is (4.1). Now, we bound how much e's effective resistance can change using martingale concentration. The above inequality implies that

$$\left| \mathbf{E}_{G_{k+1}''} \left[\frac{b_e^T L_{G_{k+1}''}^+ b_e}{r_e} \mid G_k'' \right] - \frac{b_e^T L_{G_k''}^+ b_e}{r_e} \right| \leq 8 \mathbf{E}_{G_{k+1}''} \left[\frac{(b_e^T L_{G_k''}^+ b_{f_k})^2}{r_e r_{f_k}} \right] \leq \frac{16}{|Y_0|} \sigma_1^2$$

$$\begin{aligned} \mathbf{Var}_{G_{k+1}''} \left[\frac{b_e^T L_{G_{k+1}''}^+ b_e}{r_e} \mid G_k'' \right] &= \mathbf{Var}_{G_{k+1}''} \left[\frac{b_e^T L_{G_{k+1}''}^+ b_e}{r_e} - \frac{b_e^T L_{G_k''}^+ b_e}{r_e} \mid G_k' \right] \\ &\leq 64 \mathbf{E}_{G_{k+1}''} \left[\frac{(b_e^T L_{G_k''}^+ b_{f_k})^4}{r_e^2 r_{f_k}^2} \mid G_k'' \right] \\ &\leq \frac{128 \sigma_1^4}{|Y_0|} \end{aligned}$$

$$\left| \frac{b_e^T L_{G_{k+1}''}^+ b_e}{r_e} - \frac{b_e^T L_{G_k''}^+ b_e}{r_e} \right| \leq 8\sigma_1^2$$

if $e \neq f_k$. By Theorem 4.9.11, $\Pr[|\mathtt{lev}_{G_k''}(e) - \mathtt{lev}_{G''}(e)| > 160(\log n)\sigma_1^2] \leq 1/n^{10}$. Since $\mathtt{lev}_{G''}(e) \in [1/4, 3/4]$ and $160(\log n)\sigma_1^2 < 1/8$, $|\mathtt{lev}_{G_k''}(e) - \mathtt{lev}_{G''}(e)| \leq 1/8$ for all k and all $e \neq f_0, f_1, \ldots, f_{k-1}$ with probability at least $1 - n^2/n^{10} = 1 - 1/n^8$. This verifies the inductive hypothesis and proves (4.2).

Flexible function bound. Set $M_{ef}^{(k)} \leftarrow \frac{|b_e^T L_{G_k''}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$, $v_e^{(k)} \leftarrow g_e(G_k')$, $\gamma \leftarrow 300$, $\sigma_1 \leftarrow \sigma_1$ (using (4.1)), and $\tau \leftarrow \max_{e \in Y_0} g_e(G')$. Notice that

$$M_{ef}^{(k)} = \frac{|b_e^T L_{\phi(G_k')}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$$

as well by the "ϕ commutes with modifications" property of ϕ. By definition of flexibility and (4.2),

$$\begin{aligned} v_e^{(k+1)} &\leq v_e^{(k)} + \sum_{f \in Y_0} \frac{3}{\min(\texttt{lev}_{G_k''}(f), 1 - \texttt{lev}_{G_k''}(f))} M_{ef}^{(k)} Z_f^{(k)} (v_f^{(k)} + v_e^{(k)}) \\ &\leq v_e^{(k)} + \sum_{f \in Y_0} \gamma M_{ef}^{(k)} Z_f^{(k)} (v_f^{(k)} + v_e^{(k)}) \end{aligned}$$

In particular, Proposition 4.11.2 applies and shows that

$$v_e^{(k)} \leq \tau_1$$

for all $e \in Y_0 \setminus \{f_0, f_1, \ldots, f_{k-1}\}$, as desired.

Runtime. `ColumnApx` is called at most $|X|$ times and `ColumnApxPreprocessing` is called once. □

4.11.2 A $(m^{o(1)}, |W|m^{-o(1)})$-stable oracle that runs in almost-linear time (`FastOracle`) given fast approximations to certain quantities

Now, we implement `FastOracle` modulo some subroutines that efficiently return approximations to certain quantities. We start by intuitively discussing how to make each of the quantities that `FastOracle` needs to control not change for multiple iterations.

The "Bounded leverage score difference" condition ensures that when an edge is split, it can be split in the same direction for all graphs whose edge resistances are between those in $I \setminus D$ and $I/(S, S')$. Specifically, in all graphs between $I \setminus D$ and $I/(S, S')$, splitting an edge e in one particular direction ensures that its leverage score is bounded away from 0 and 1.

Leverage score stability intuition

This bound follows immediately from returning a subset of $\texttt{VeryStable}(I \setminus D, \texttt{VeryStable}(I/(S, S'), W))$. The "Leverage score bound" of Proposition 4.11.4 yields the desired result.

Midpoint potential stability intuition

By Theorem 4.9.12, for any $\{u, v\} \in W$,

$$\sum_{f \in W} \frac{|b_{ss'}^T L_{(I \setminus D)/(S,S')}^+ b_f | |b_f^T L_{(I \setminus D)/(S,S')}^+ (b_{su} + b_{sv})/2|}{r_f} \leq b_{ss'}^T L_{(I \setminus D)/(S,S')}^+ (b_{su} + b_{sv})/2 \leq O(\log(n/p))p$$

In particular, using the birthday paradox (as in `VeryStable`), one can find a 1/polylog(n/p) fraction $W' \subseteq W$ for which

$$\frac{|b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_f || b_f^T L^+_{(I\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{r_f} \leq \frac{p}{\log n}$$

for all pairs of distinct edges $e = \{u, v\}, f \in W'$. Passing W' through `VeryStable` makes it so that the following flexible functions of H do not change by more than a factor of 2 over the course of many edge contractions or deletions:

$$g_f^{(0)}(H) := \frac{|b_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_f|}{\sqrt{r_f}}$$

$$\phi^{(0)}(H) := (H \setminus D)/(S, S')$$

$$g_f^{(1),X,s}(H) := \sum_{\{u,v\}\in X\setminus f} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_f}}$$

$$g_f^{(1),X,s'}(H) := \sum_{\{u,v\}\in X\setminus f} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')}(b_{us'} + b_{vs'})/2|}{\sqrt{r_f}}$$

$$\phi^{(1)}(H) := (H \setminus D)/(S, S')$$

for some set of edges $X \subseteq W$.

We have to bucket edges by similar value of these functions in order to fully exploit this (since the bound at the end is in terms of the maximum). Therefore, we can find a relatively large subset of W for which random sampling will on average causes a $(1-\tilde{O}(1/|W|))$-factor decrease in the function $b_{ss'}^T L^+_{(H\setminus D)/(S,S')}(b_{su} + b_{sv})/2$ and in the worst case only decreases it by a $(1 - 1/(\log n))$ factor. Therefore, by Theorem 4.9.11, $b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su} + b_{sv})/2$ only decreases by a small constant factor over the course of conditioning on this relatively large subset of W.

Main objective change stability intuition

$g_f^{(0)}(H)$ is the square root of the quantity that needs to be bounded here. Therefore, we are done by the discussion of the previous subsubsection.

Deferred endpoint potential change stability intuition

Exploit bucketing and Theorem 4.9.12, as discussed in Subsubsection 4.11.2. In particular, we will control the function

$$g_f^{(2)}(H) := \sum_{\{u,v\}\in A} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')}(b_{su}+b_{sv})|}{\sqrt{r_f}} + \sum_{\{u,v\}\in B} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{\sqrt{r_f}}$$

$$\phi^{(2)}(H) := (H\setminus D)/(S,S')$$

$S-S'$ normalized degree change stability intuition

The only complication over the previous subsubsection is that the coefficients $b_{sw}^T L^+_{(I_i\setminus D)/S} b_{sw}$ and $\frac{b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_e}{r_e}$ can increase. We show using Theorem 4.9.12 that stopping after $|W|m^{-o(1)}$ samples does not result in an overly large increase in the coefficients. This is the only part of this section that requires $\rho = m^{o(1)}$.

Specifically, the following proposition will help us. Think of $v^{(k)} = b_{sw}^T L^+_{(I_k\setminus D)/S} b_{sw}$ and $v^{(k)} = b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_e$ in two different applications of Proposition 4.11.5. It is proven in Appendix B.6.1:

Proposition 4.11.5. *Consider a random sequence $\{v^{(k)}\}_{k\geq 0}$ generated as follows. Given $v^{(k)}$,*

- *Pick $\{u_i^{(k)}\}_{i=1}^{\ell_k}$ and $\{w_i^{(k)}\}_{i=1}^{\ell_k}$, with $\sum_{i=1}^{\ell_k} u_i^{(k)} w_i^{(k)} \leq \eta v^{(k)}$*
- *Let $Z^{(k+1)} \in \{0,1\}^{\ell_k}$ denote the indicator of a uniformly random choice over $[\ell_k]$*
- *Pick $v^{(k+1)} \leq v^{(k)} + \gamma \sum_{i=1}^{\ell_k} u_i^{(k)} Z_i^{(k+1)} w_i^{(k)}$*

Let $m_0 = \min_k \ell_k$ and $M_0 = \max_k \ell_k$. Then with probability at least $1-2\tau$,

$$v^{(k')} \leq (2\gamma\eta)^\rho v^{(0)}$$

for all $k' \leq m_0 \min(\frac{1}{(\log(M_0^2/\tau))\eta^2\gamma^2}, \frac{1}{200\eta\gamma^2\log(M_0^2/\tau)}(\tau/M_0^2)^{1/\rho})$

We use this proposition to show that controlling the following functions with constant coefficients suffices:

$$g_f^{(3),s}(H) = \sum_{w\in S'} b_{sw}^T L^+_{(I\setminus D)/S} b_{sw} \sum_{e\in\partial_H w} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')} b_e|}{\sqrt{r_f} r_e}$$

$$g_f^{(3),s'}(H) = \sum_{w\in S} b_{s'w}^T L^+_{(I\setminus D)/S'} b_{s'w} \sum_{e\in\partial_H w} \frac{|b_f^T L^+_{(H\setminus D)/(S,S')} b_e|}{\sqrt{r_f} r_e}$$

$$\phi^{(3)}(H) := (H\setminus D)/(S,S')$$

$$g_f^{(4),s}(H) = \sum_{w \in S'} \left(\frac{|b_{sw}^T L^+_{(H \setminus D)/S} b_f|}{\sqrt{r_f}} \right)^2 \sum_{e \in \partial_I w} \frac{b_{ss'}^T L^+_{(I \setminus D)/(S,S')} b_e}{r_e}$$

$$\phi^{(4),s}(H) := (H \setminus D)/S$$

$$g_f^{(4),s'}(H) = \sum_{w \in S} \left(\frac{|b_{s'w}^T L^+_{(H \setminus D)/S'} b_f|}{\sqrt{r_f}} \right)^2 \sum_{e \in \partial_I w} \frac{b_{ss'}^T L^+_{(I \setminus D)/(S,S')} b_e}{r_e}$$

$$\phi^{(4),s'}(H) := (H \setminus D)/S'$$

Tying the parts together

Now, we implement `FastOracle`. This oracle is similar to `SlowOracle` but restricts the set W up front using `VeryStable` in order to take care of the flexibility of the $g^{()}$ functions.

Algorithm 17: `FastOracle`(I, S, S', D, A, B, W) part 1 (everything but return statement)

Data: graph I, sets $S, S' \subseteq V(I)$ for identification, deleted edges $D \subseteq E(I)$, deferred edges $A, B \subseteq E(I)$, input edges $W \subseteq E(I)$
Result: a relatively large subset $Z \subseteq W$ for which objectives remain stable

$I' \leftarrow$ graph obtained by splitting each edge of W
$W' \leftarrow$ arbitrary copy of each edge in W in I'
`// leverage scores`
$W' \leftarrow \texttt{VeryStable}_{I'}(I' \setminus D, W')$
$W' \leftarrow \texttt{VeryStable}_{I'}(I'/(S, S'), W')$
`// controlling flexible functions`
$W' \leftarrow \texttt{VeryStable}_{I'}((I' \setminus D)/(S, S'), W')$
$W' \leftarrow \texttt{VeryStable}_{I'}((I' \setminus D)/S, W')$
$W' \leftarrow \texttt{VeryStable}_{I'}((I' \setminus D)/S', W')$
`ApxPreprocessing`(I', S, S', D, A, B, W')
foreach $e \in W'$ **do**
  `// ApxQuery returns multiplicative 2-approximations` $h_e^{()}$ `to each` $g_e^{()}$
  $h_e^{()}(I') \leftarrow \texttt{ApxQuery}(g_e^{()}(I'))$
end
`// bucketing for deferred, degree, and midpoint objectives`
foreach $i \in \{0, 1, \ldots, i_{max} := \log(n^8\alpha^4)\}$ **do**
  $W'_i \leftarrow$ edges $f \in W'$ for which

$$\frac{h_f^{(0)}(I')}{\sqrt{b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} b_{ss'}}} \in \left[2^{-i-1}, 2^{-i}\right]$$

  with no lower bound for $i = i_{max}$
end
$W' \leftarrow$ the W'_i with maximum size
`// midpoint objective downsampling`
$W' \leftarrow$ uniform random sample of $1/(1000 \log^2(n/p))$ fraction of W'

Algorithm 18: `FastOracle` part 2 (return statement)

`// final output`

$Z \leftarrow$ edges $e \in W'$ with all of the following properties:

- (Midpoint s) $h_e^{(0)}(I')h_e^{(1),W',s}(I') \leq p(b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'})/(100 \log n)$ if "s narrow potential neighborhood" input condition is satisfied
- (Midpoint s') $h_e^{(0)}(I')h_e^{(1),W',s'}(I') \leq p(b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'})/(100 \log n)$ if "s' narrow potential neighborhood" input condition is satisfied
- (Conductance $s - s'$) $h_e^{(0)}(I')h_e^{(3),s}(I') \leq \frac{100(\log(n\alpha))}{|W'|}\delta_{S,S'}(I' \setminus D)$
- (Conductance $s' - s$) $h_e^{(0)}(I')h_e^{(3),s'}(I') \leq \frac{100(\log(n\alpha))}{|W'|}\delta_{S',S}(I' \setminus D)$
- (Energy $s - s'$) $h_e^{(4),s}(I') \leq \frac{100}{|W'|}\delta_{S,S'}(I' \setminus D)$
- (Energy $s' - s$) $h_e^{(4),s'}(I') \leq \frac{100}{|W'|}\delta_{S',S}(I' \setminus D)$
- (Deferred) $h_e^{(0)}(I')h_e^{(2)}(I') \leq \frac{100(\log(\alpha n))}{|W'|}\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{su} + b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{us'} + b_{vs'})\right) + r_{min}/n^4$
- (Main) $(h_e^{(0)}(I'))^2 \leq \frac{100}{|W'|} b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}$

return Z

Now, we outline the analysis of this algorithm. Simple calculations involving Sherman-Morrison show the following proposition:

Proposition 4.11.6. *For any graph J with $S, S' \subseteq V(J)$ and $A, B, D, X \subseteq E(J)$, the families of functions*

- $\mathcal{F}_0 := (\{g_e^{(0)}(H)\}_{e\in X}, \phi^{(0)})$
- $\mathcal{F}_{1,s} := (\{g_e^{(1),X\cap E(H),s}(H)\}_{e\in X}, \phi^{(1)})$
- $\mathcal{F}_{1,s'} := (\{g_e^{(1),X\cap E(H),s'}(H)\}_{e\in X}, \phi^{(1)})$
- $\mathcal{F}_2 := (\{g_e^{(2)}(H)\}_{e\in X}, \phi^{(2)})$
- $\mathcal{F}_{3,s} := (\{g_e^{(3),s}(H)\}_{e\in X}, \phi^{(3)})$
- $\mathcal{F}_{3,s'} := (\{g_e^{(3),s'}(H)\}_{e\in X}, \phi^{(3)})$
- $\mathcal{F}_{4,s} := (\{g_e^{(4),s}(H)\}_{e\in X}, \phi^{(4),s})$

- $\mathcal{F}_{4,s'} := (\{g_e^{(4),s'}(H)\}_{e\in X}, \phi^{(4),s'})$

are flexible for the graph J.

We prove the above proposition in Appendix B.6.1. Proposition 4.11.4 therefore implies that the maximum values of the functions in each family do not change by more than a factor of 2 with high probability over the course of $|W'|/2$ edge contractions and deletions. The bucketing lower bound for $h_e^{(0)}(I')$ together with the upper bounds given in bullets implies an upper bound on the values of all of the $h_e^{()}$ functions, which in turn gives upper bounds on the $g_e^{()}$ functions. This essentially completes all bounds besides the "Midpoint potential stability" bounds. We prove these bounds using a simple application of Theorem 4.9.11.

Now, we just have to check the "Size of Z" guarantee. This proof of this guarantee is similar to the analysis of `SlowOracle`.

Proposition 4.11.7. *There is an $(m^{o(1)}, |W|m^{-o(1)})$-stable oracle `FastOracle` with runtime*

$$\tilde{O}(m + \mathcal{T}(\texttt{VeryStable}) + \mathcal{T}(\texttt{ApxPreprocessing}) + |W|\mathcal{T}(\texttt{ApxQuery}))$$

In the following proof, notice that some statements have been reordered from their appearance in the definition of stable oracles. The statements are roughly stated in order of difficulty.

Proof. Algorithm well-definedness. By Proposition 4.11.8, the h functions are indeed approximations to the g functions. Therefore, to check that this algorithm is well-defined, it suffices to check that the input condition of `VeryStable` described in Proposition 4.11.4 is satisfied. The "Bounded leverage score difference" input condition and Rayleigh monotonicity imply that all leverage scores $\texttt{lev}_H(e)$ in graphs H used in calls to `VeryStable` are within $1/16$ additively of $\texttt{lev}_I(e)$. Therefore, by Proposition 4.9.6, after splitting, all leverage scores are within $1/8$ additively of $\texttt{lev}_{I'}(e)$. Also by Proposition 4.9.6, $\texttt{lev}_{I'}(e) \in [1/4, 3/4]$. Therefore, $\texttt{lev}_{H'}(e) \in [1/8, 7/8]$, where H' is the graph with W split for each graphs H supplied as an argument to `VeryStable`. In particular, splitting using I satisfies all of the input conditions to the `VeryStable` calls.

Runtime. `ApxQuery` is called $\tilde{O}(|W|)$ times while `ApxPreprocessing` and `VeryStable` are only called a constant number of times. The rest of the algorithm takes $\tilde{O}(m)$ time, as desired.

Leverage score stability. Notice that `VeryStable` is applied with both $I' \setminus D$ and $I'/(S, S')$ as arguments. Therefore, the desired result follows directly from the "Leverage score bound" guarantee of Proposition 4.11.4 applied to $I' \setminus D$ and $I'/(S, S')$.

Main objective change stability. By the "Flexible function bound" on the family $\{g_e^{(0)}\}_{e\in Z}$,

$$\begin{aligned}(g_{f_i}^{(0)}(I_i'))^2 &\leq 4\max_{e\in Z}(g_e^{(0)}(I'))^2\\ &\leq 16\max_{e\in Z}(h_e^{(0)}(I'))^2\\ &\leq \frac{1600}{|Z|}(b_{ss'}^T L^+_{(I\setminus D)/(S,S')}b_{ss'})\end{aligned}$$

for any $i \leq |Z|/2$ with high probability, where I_i' is the graph I_i with edges in Z split. By definition of $g_{f_i}^{(0)}(I_i')$ and the fact that the g functions are electrical,

$$g_{f_i}^{(0)}(I_i') = \frac{b_{ss'}^T L^+_{(I_i'\setminus D)/(S,S')}b_{f_i}}{\sqrt{r_{f_i}}} = \frac{b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')}b_{f_i}}{\sqrt{r_{f_i}}}$$

so plugging this in completes the stability proof since $1600 < \rho$.

Main objective change. By "Main objective change stability," "Leverage score stability," and the fact that f_i is split before contraction or deletion, each contraction or deletion causes a maximum change of

$$|b_{ss'}^T L^+_{(I_{i+1}\setminus D)/(S,S')}b_{ss'} - b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')}b_{ss'}| \leq 8\frac{1600 b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')}b_{ss'}}{|Z|}$$

for all i. Therefore, since $K = |W|m^{-o(1)} \leq |W|/12800$, the main objective can only increase by a factor of 2 over the course of conditioning on the f_is.

Deferred endpoint potential change stability. Suppose that W_a' is the bucket in W' that is chosen in Line 17 in Part 1 of `FastOracle`. For edges $e \in Z$, $h_e^{(0)}(I') \geq 2^{-a-1}\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}b_{ss'}}$ by the bucketing lower bound. Therefore, by the "Deferred" condition on edges in Z,

$$\begin{aligned}g_e^{(2)}(I') \leq 2h_e^{(2)}(I') \leq \frac{2^{a+2}}{\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}b_{ss'}}}\frac{100(\log(\alpha n))}{|Z|}\\ \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right)\end{aligned}$$

By the bucketing upper bound,

$$g_e^{(0)}(I') \leq 2^{-a+1}\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}}$$

These bounds hold for all $e \in Z$. Therefore, by the "Flexible function bound" of Proposition 4.11.4 applied to both of the flexible families $\{g_e^{(0)}\}_{e\in Z}$ and $\{g_e^{(2)}\}_{e\in Z}$,

$$g_{f_i}^{(0)}(I_i') \leq \max_{e\in Z} 2g_e^{(0)}(I') \leq 2^{-a+2}\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}}$$

and

$$\begin{aligned} g_{f_i}^{(2)}(I_i') &\leq \max_{e\in Z} 2g_e^{(2)}(I') \\ &\leq \frac{2^{a+3}}{\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}}} \frac{100(\log(\alpha n))}{|Z|} \\ &\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I'\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) \end{aligned}$$

with high probability for all $i \leq |Z|/2$. Therefore, since the gs are electrical functions,

$$\begin{aligned} &\left(\sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_{f_i}} \right) \\ &+ \left(\sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_{f_i}} \right) \\ &= g_{f_i}^{(0)}(I_i) g_{f_i}^{(2)}(I_i) \\ &\leq \frac{3200(\log(\alpha n))}{|Z|} \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) \end{aligned}$$

as desired, since $K < |Z|/2$.

$S - S'$-normalized degree change stability. We omit the $S' - S$ guarantees, as their proofs are the same with s and S swapped for s' and S' respectively in all places.

$S - S'$-conductance term stability. We break this analysis up into two parts.

Bounding increases in effective resistances. We show that with high probability, for all $w \in S'$ and all $i \leq m^{-o(1)}|Z|$,

$$b_{sw}^T L^+_{(I_i \setminus D)/S} b_{sw} \leq m^{o(1)} b_{sw}^T L^+_{(I \setminus D)/S} b_{sw}$$

We use Proposition 4.11.5 with

- $\rho \leftarrow \sqrt{\log n}$
- $\tau \leftarrow 1/n^6$
- $\gamma \leftarrow 8$
- $u_e^{(i)} = w_e^{(i)} \leftarrow |b_{sw}^T L^+_{(I_i \setminus D)/S} b_e| / \sqrt{r_e}$
- $v^{(i)} \leftarrow b_{sw}^T L^+_{(I_i \setminus D)/S} b_{sw}$
- $\eta \leftarrow 1$ (which works because the sum of energies on edges is at most the overall energy)
- $\ell_i \leftarrow$ number of remaining edges in Z in I_i

By electrical functions, Sherman-Morrison, and the triangle inequality,

$$\begin{aligned}
v^{(i+1)} &= b_{sw}^T L^+_{(I'_{i+1} \setminus D)/S} b_{sw} \\
&\leq b_{sw}^T L^+_{(I'_i \setminus D)/S} b_{sw} + \frac{(b_{sw}^T L^+_{(I'_i \setminus D)/S} b_{f_i})^2 / r_{f_i}}{\min(\texttt{lev}_{(I'_i \setminus D)/S}(f_i), 1 - \texttt{lev}_{(I'_i \setminus D)/S}(f_i))} \\
&\leq b_{sw}^T L^+_{(I'_i \setminus D)/S} b_{sw} + 8 (b_{sw}^T L^+_{(I'_i \setminus D)/S} b_{f_i})^2 / r_{f_i} \\
&\leq v^{(i)} + \gamma \sum_e u_e^{(i)} Z_e^{(i+1)} w_e^{(i)}
\end{aligned}$$

Therefore, Proposition 4.11.5 applies, which means that

$$v^{(i)} = b_{sw}^T L^+_{(I_i \setminus D)/S} b_{sw} \leq 16^{\sqrt{\log n}} b_{sw}^T L^+_{(I \setminus D)/S} b_{sw}$$

for all $i \leq |Z| n^{-6/\sqrt{\log n}} = |Z| 2^{-6\sqrt{\log n}}$ with high probability (at least $1 - 1/n^6$). This is the desired result for this part since $K < |Z| 2^{-6\sqrt{\log n}}$.

Bounding increases in flows with constant coefficients. We now show that

$$\sum_{w \in S'} (b_{sw}^T L^+_{(I \setminus D)/S} b_{sw}) \sum_{e \in \partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i \setminus D)/(S,S')} b_{f_i}| |b_{f_i}^T L^+_{(I_i \setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \leq \tilde{O}\left(\frac{1}{|Z|}\right) \delta_{S,S'}(I \setminus D)$$

with high probability for all $i \leq |Z|/2$. Assume that W'_a is the bucket chosen on Line 17. Then, by the lower bound on $h_e^{(0)}(I')$,

$$g_e^{(3),s}(I') \le 2h_e^{(3),s}(I') \le \frac{2^{a+2}}{\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}}} \frac{100\log(\alpha n)}{|Z|}\delta_{S,S'}(I'\setminus D)$$

for any $e \in Z$. By the bucketing upper bound,

$$g_e^{(0)}(I') \le 2h_e^{(0)}(I') \le 2^{-a+1}\sqrt{b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}}$$

By the "Flexible function bound" of Proposition 4.11.4,

$$\begin{aligned} g_{f_i}^{(0)}(I_i')g_{f_i}^{(3),s}(I_i') &\le 4(\max_{e\in Z} g_e^{(0)}(I'))(\max_{e\in Z} g_e^{(3),s}(I')) \\ &\le \frac{3200}{\log(\alpha n)}|Z|\delta_{S,S'}(I'\setminus D) \end{aligned}$$

for all $i \le |Z|/2$ with high probability. Substituting in the definitions of $g_{f_i}^{(0)}(I_i')$ and $g_{f_i}^{(3),s}(I_i')$ yields the desired result for this part.

Combining the parts. By the first part,

$$\begin{aligned} &\sum_{w\in S'}(b_{sw}^T L^+_{(I_i\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i\setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \\ &\le m^{o(1)} \sum_{w\in S'}(b_{sw}^T L^+_{(I\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i\setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \end{aligned}$$

By the second part,

$$m^{o(1)} \sum_{w\in S'}(b_{sw}^T L^+_{(I\setminus D)/S} b_{sw}) \sum_{e\in\partial_{I_i} w} \frac{|b_{ss'}^T L^+_{(I_i\setminus D)/(S,S')} b_{f_i}||b_{f_i}^T L^+_{(I_i\setminus D)/(S,S')} b_e|}{r_{f_i} r_e} \le \frac{3200 m^{o(1)}}{|Z|}\delta_{S,S'}(I\setminus D)$$

The desired result follows from substitution since $3200m^{o(1)} < \rho$.

$S - S'$ energy term stability. As in the previous bound, we break the analysis up into two parts:

Bounding increases in flows to S' vertices. We show that with high probability, for all $w \in S'$ and all $i \le m^{-o(1)}|Z|$,

$$\sum_{e\in\partial_{I_i}w}\frac{b_{ss'}^TL^+_{(I_i\setminus D)/(S,S')}b_e}{r_e}\le m^{o(1)}\sum_{e\in\partial_Iw}\frac{b_{ss'}^TL^+_{(I\setminus D)/(S,S')}b_e}{r_e}$$

We use Proposition 4.11.5 with

- $\rho\leftarrow\sqrt{\log n}$
- $\tau\leftarrow 1/n^6$
- $\gamma\leftarrow 8$
- $u_f^{(i)}\leftarrow\frac{|b_{ss'}^TL^+_{(I_i\setminus D)/(S,S')}b_f|}{\sqrt{r_f}}$
- $w_f^{(i)}\leftarrow\sum_{e\in\partial_{I_i}w}\frac{|b_f^TL^+_{(I_i\setminus D)/(S,S')}b_e|}{\sqrt{r_f}r_e}$
- $v^{(i)}\leftarrow\sum_{e\in\partial_{I_i}w}\frac{b_{ss'}^TL^+_{(I_i\setminus D)/(S,S')}b_e}{r_e}$
- $\eta\leftarrow O(\log(n\alpha))$ (which works by Theorem 4.9.12)
- $\ell_i\leftarrow$ number of remaining edges in Z in I_i

By electrical functions, Sherman-Morrison, and the triangle inequality,

$$\begin{aligned}v^{(i+1)}&=\sum_{e\in\partial_{I'_{i+1}}w}\frac{b_{ss'}^TL^+_{(I'_{i+1}\setminus D)/(S,S')}b_e}{r_e}\\&\le\left(\sum_{e\in\partial_{I'_i}w}\frac{b_{ss'}^TL^+_{(I'_i\setminus D)/(S,S')}b_e}{r_e}\right)+\frac{1}{\min(\texttt{lev}_{(I'_i\setminus D)/(S,S')}(f_i),1-\texttt{lev}_{(I'_i\setminus D)/(S,S')}(f_i))}\\&\left(\sum_{e\in\partial_{I_i}w}\frac{|b_{ss'}^TL^+_{(I_i\setminus D)/(S,S')}b_{f_i}|}{\sqrt{r_{f_i}}}\frac{|b_{f_i}^TL^+_{(I_i\setminus D)/(S,S')}b_e|}{\sqrt{r_{f_i}}r_e}\right)\\&\le v^{(i)}+\gamma\sum_e u_e^{(i)}Z_e^{(i+1)}w_e^{(i)}\end{aligned}$$

Therefore, Proposition 4.11.5 applies, which means that,

$$v^{(i)}=\sum_{e\in\partial_{I_i}w}\frac{b_{ss'}^TL^+_{(I_i\setminus D)/(S,S')}b_e}{r_e}\le(16\log n)^{\sqrt{\log n}}\sum_{e\in\partial_Iw}\frac{b_{ss'}^TL^+_{(I\setminus D)/(S,S')}b_e}{r_e}$$

for all $i \leq |Z|n^{-6/\sqrt{\log n}} = |Z|2^{-6\sqrt{\log n}}$ with high probability (at least $1 - 1/n^6$). This is the desired result for this part since $K < |Z|2^{-6\sqrt{\log n}}$.

Bounding increases in energies with constant coefficients. By the "Flexible function bound" of Proposition 4.11.4 applied to $g^{(4),s}$, with high probability for all i,

$$\begin{aligned} g^{(4),s}_{f_i}(I'_i) &\leq 2 \max_{e \in Z} g^{(4),s}_e(I') \\ &\leq 4 \max_{e \in Z} h^{(4),s}_e(I') \\ &\leq \frac{400}{|Z|} \delta_{S,S'}(I' \setminus D) \end{aligned}$$

as desired.

Combining the parts. By the first part,

$$\begin{aligned} &\sum_{w \in S'} \frac{(b^T_{sw} L^+_{(I_i \setminus D)/S} b_{f_i})^2}{r_{f_i}} \sum_{e \in \partial_{I_i} w} \frac{b^T_{ss'} L^+_{(I_i \setminus D)/(S,S')} b_e}{r_e} \\ &\leq m^{o(1)} \sum_{w \in S'} \frac{(b^T_{sw} L^+_{(I_i \setminus D)/S} b_{f_i})^2}{r_{f_i}} \sum_{e \in \partial_{I_i} w} \frac{b^T_{ss'} L^+_{(I \setminus D)/(S,S')} b_e}{r_e} \end{aligned}$$

By the second part,

$$m^{o(1)} \sum_{w \in S'} \frac{(b^T_{sw} L^+_{(I_i \setminus D)/S} b_{f_i})^2}{r_{f_i}} \sum_{e \in \partial_{I_i} w} \frac{b^T_{ss'} L^+_{(I \setminus D)/(S,S')} b_e}{r_e} \leq \frac{400 m^{o(1)}}{|Z|} \delta_{S,S'}(I \setminus D)$$

The desired result follows from substitution since $400 m^{o(1)} < \rho$.

Midpoint potential stability. We assume that the "s narrow potential neighborhood" input condition is satisfied. The "s' narrow potential neighborhood" case is the same with s and s' swapped.

Suppose that bucket W'_a was chosen on Line 17. Then for all $e \in Z$,

$$g^{(0)}_e(I') \leq 2h^{(0)}_e(I') \leq 2^{-a+1} \sqrt{b^T_{ss'} L^+_{(I' \setminus D)/(S,S')} b_{ss'}}$$

and

$$\begin{aligned} g^{(1),Z,s}_e(I') &\leq 2h^{(1),W',s}_e(I') \\ &\leq \frac{2^{a+2}}{\sqrt{b^T_{ss'} L^+_{(I' \setminus D)/(S,S')} b_{ss'}}} \frac{p(b^T_{ss'} L^+_{(I \setminus D)/(S,S')} b_{ss'})}{100 \log n} + \frac{p}{200(\log n) h^{(0)}_e(I')} \end{aligned}$$

by the bucketing upper and lower bounds respectively. By the "Flexible function bound" of Proposition 4.11.4 and the "Midpoint s" bound, for all $i \leq |Z|/2$, with high probability

$$g_{f_i}^{(0)}(I_i')g_{f_i}^{(1),Z\setminus\{f_0,f_1,\ldots,f_{i-1}\},s}(I_i') \leq \frac{32}{100(\log n)}p(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+ b_{ss'})$$

For any $\{u,v\} \in Z \setminus \{f_0, f_1, \ldots, f_{i-1}\}$,

$$\begin{aligned}\frac{|b_{ss'}^T L_{(I_i'\setminus D)/(S,S')}^+ b_{f_i}||b_{f_i}^T L_{(I_i'\setminus D)/(S,S')}(b_{su}+b_{sv})/2|}{r_{f_i}} &\leq g_{f_i}^{(0)}(I_i')g_{f_i}^{(1),Z\setminus\{f_0,f_1,\ldots,f_{i-1}\},s}(I_i') \\ &\leq \frac{32}{100(\log n)}p(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+ b_{ss'})\end{aligned}$$

Therefore, Theorem 4.9.11 applies with expectation change $O((\log(n/p))(3p)(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+ b_{ss'}))/|Z|$, stepwise variance $(O((\log(n/p))(3p)(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+ b_{ss'})))^2/|Z|$ (both by Theorem 4.9.12 and an upper bound inductive hypothesis with base case that is the upper bound of "s narrow potential neighborhood"), and maximum change $\frac{32}{100(\log n)}p(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+ b_{ss'})$. Theorem 4.9.11 shows that

$$\Pr[|(b_{ss'}^T L_{(I_i\setminus D)/(S,S')}^+(b_{su_i}+b_{sv_i})/2)-(b_{ss'}^T L_{(I\setminus D)/(S,S')}^+(b_{su_i}+b_{sv_i})/2)| > p/8 b_{ss'}^T L_{(I_i\setminus D)/(S,S')}^+ b_{ss'}] \leq 1/n^6$$

Therefore, since $2p + p/8 < 3p < 1 - (p/2)$ (because $p \leq 1/4$ by the "Narrow potential neighborhood" condition), the inductive hypothesis is verified and the "s' midpoint potential stability" guarantee is satisfied. Furthermore, by the "s narrow potential neighborhood" lower bound, the normalized potential of the midpoint of f_i is at least $p - p/8 > p/2$, as desired.

Size of Z. We show that $|Z| \geq |W|/\text{polylog}(n)$ with constant probability. This can be boosted to high probability by running `FastOracle` $\Theta(\log n)$ times, since all other guarantees are satisfied with high probability.

By the "Size" guarantee of Proposition 4.11.4, $|W'| \geq |W|/(\text{polylog}(n))^5$ immediately after all applications of `VeryStable`. After Line 17,

$$|W'| \geq |W|/((\text{polylog}(n))^5 i_{max})$$

Let W'' be the version of W' immediately after Line 18. Then

$$|W''| \geq |W|/((\text{polylog}(n))^5 i_{max} 1000 \log^2(n/p))$$

Now, we just need to upper how many edges in W'' are not in Z. We bound this on a constraint-by-constraint basis.

Midpoint constraints with the "s narrow potential neighborhood" input condition.This part is very similar in spirit to the "Size" bound proof in Proposition 4.11.4. This part under the "s' narrow potential neighborhood" assumption is the same with s and s' swapped, so we focus on the s case.

Start by exploiting how W'' was sampled from W' on Line 18, which we denote by W'_{orig} for clarity. By Theorem 4.9.12, for any $e \in W'_{orig}$

$$\begin{aligned}
&\sum_{e \in W'_{orig}} g_e^{(0)}(I') g_e^{(1),W'_{orig},s}(I') \\
&\qquad = \sum_{e \in W'_{orig}} \sum_{f=\{u,v\} \in W'_{orig} \setminus \{e\}} \frac{|b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} b_e | |b_e^T L^+_{(I' \setminus D)/(S,S')} (b_{su} + b_{sv})/2|}{r_e} \\
&\qquad \leq \sum_{f=\{u,v\} \in W'_{orig}} O(\log(n/p)) b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} (b_{su} + b_{sv})/2
\end{aligned}$$

By the "s narrow potential neighborhood" condition,

$$\sum_{f=\{u,v\} \in W'_{orig}} O(\log(n/p)) b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} (b_{su} + b_{sv})/2 \leq O(\log(n/p))(2p b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} b_{ss'}) |W'_{orig}|$$

Therefore, there is a set $W'_{low} \subseteq W'_{orig}$ with $|W'_{low}| \geq |W'_{orig}|/2$ with the property that

$$g_e^{(0)}(I') g_e^{(1),W'_{orig},s}(I') \leq O(\log(n/p))(4p b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} b_{ss'})$$

for all $e \in W'_{low}$. We now upper bound the expected number of edges in W'_{low} that violate the "Midpoint s" condition. First, for any $e \in W'_{low}$

$$\begin{aligned}
\mathbf{E}_{W''}[g_e^{(0)}(I') g_e^{(1),W'',s}(I') | e \in W''] &= g_e^{(0)}(I') \mathbf{E}_{W''}[g_e^{(1),W'',s}(I')] \\
&= \frac{1}{1000 \log^2(n/p)} g_e^{(0)}(I') g_e^{(1),W'_{orig},s}(I') \\
&\leq \frac{p}{400(\log n)} (b_{ss'}^T L^+_{(I' \setminus D)/(S,S')} b_{ss'})
\end{aligned}$$

The first equality follows from the fact that the sum in g_e does not include e. The second equality follows from the definition of W'' on Line 18. The last line follows from the definition of W'_{low}. By Markov's Inequality, for any $e \in W'_{low}$,

$$\begin{aligned}
&\Pr_{W''}[e \text{ satisfies "Midpoint } s\text{" condition and } e \in W''] \\
&\Pr_{W''}[e \text{ satisfies "Midpoint } s\text{" condition} | e \in W''] \Pr_{W''}[e \in W''] \\
&\geq 1 - \Pr_{W''}[g_e^{(0)}(I')g_e^{(1),W'',s}(I') > \frac{p}{200(\log n)}(b_{ss'}^T L^+_{(I'\setminus D)/(S,S')} b_{ss'}) | e \in W''] \frac{1}{1000\log^2(n/p)} \\
&\geq \frac{1}{2000\log^2(n/p)}
\end{aligned}$$

Let Z_{mid} be the set of edges in W'_{orig} satisfying the "Midpoint s" condition. Then

$$\begin{aligned}
\mathbf{E}_{W''}[|Z_{mid}|] &\geq \sum_{e \in W'_{low}} \Pr_{W''}[e \in Z_{mid}] \\
&= \sum_{e \in W'_{low}} \Pr_{W''}[e \text{ satisfies "Midpoint } s\text{" condition and } e \in W''] \\
&\geq \frac{|W'_{low}|}{2000\log^2(n/p)} \\
&\geq \frac{|W'_{orig}|}{4000\log^2(n/p)}
\end{aligned}$$

Since $|Z_{mid}| \subseteq |W'_{orig}|$, $|Z_{mid}| \geq \frac{|W'_{orig}|}{8000\log^2(n/p)}$ with probability at least $\frac{1}{8000\log^2(n/p)}$, as desired.

Conductance $s - s'$ constraint. We upper bound the number of elements in Z_{mid} that do not satisfy this. By Theorem 4.9.12,

$$\sum_{e \in Z_{mid}} g_e^{(0)}(I')g_e^{(3),s}(I') \leq (\log(n\alpha))\delta_{S,S'}(I' \setminus D)$$

By the approximation lower bound, for all edges e with $h_e^{(0)}(I')h_e^{(3),s}(I') \geq \frac{100(\log(n\alpha))}{|Z_{mid}|}\delta_{S,S'}(I' \setminus D)$,

$$g_e^{(0)}(I')g_e^{(3),s}(I') \geq \frac{25(\log(n\alpha))}{|Z_{mid}|}\delta_{S,S'}(I' \setminus D)$$

By the previous inequality, only $|Z_{mid}|/25$ edges in Z_{mid} can satisfy the above inequality, as desired.

Conductance $s' - s$ constraint. Same as above, but with s and S swapped for s' and S' and vice versa.

Energy $s - s'$ constraint. Since the sum of energies on edges is at most the overall energy,

$$\sum_{e \in Z_{mid}} g_e^{(4),s}(I') \leq \delta_{S,S'}(I' \setminus D)$$

For all edges with $h_e^{(4),s}(I') \geq \frac{100}{|Z_{mid}|}\delta_{S,S'}(I' \setminus D)$,

$$g_e^{(4),s}(I') \geq \frac{50}{|Z_{mid}|}\delta_{S,S'}(I' \setminus D)$$

By the previous inequality, only $|Z_{mid}|/50$ edges in Z_{mid} can satisfy the above inequality, as desired.

Energy $s' - s$ constraint. Same as above, but with s and S swapped for s' and S' and vice versa.

Deferred constraint. By Theorem 4.9.12,

$$\begin{aligned}
&\sum_{e \in Z_{mid}} g_e^{(0)}(I') g_e^{(2)}(I') \\
&\leq \left((\log(n\alpha)) \left(\sum_{\{u,v\} \in A} b_{ss'}^T L^+_{(I' \setminus D)/(S,S')}(b_{su} + b_{sv}) + \sum_{\{u,v\} \in B} b_{ss'}^T L^+_{(I' \setminus D)/(S,S')}(b_{us'} + b_{vs'}) \right) \right. \\
&\left. + r_{min}/n^6 \right)
\end{aligned}$$

Therefore, only $|Z_{mid}|/25$ edges $e \in Z_{mid}$ can have

$$\begin{aligned}
&h_e^{(0)}(I') h_e^{(2)}(I') \\
&\geq \left(\frac{100(\log(\alpha n))}{|W'|} \left(\sum_{\{u,v\} \in A} b_{ss'}^T L^+_{(I' \setminus D)/(S,S')}(b_{su} + b_{sv}) + \sum_{\{u,v\} \in B} b_{ss'}^T L^+_{(I' \setminus D)/(S,S')}(b_{us'} + b_{vs'}) \right) \right. \\
&\left. + r_{min}/n^4 \right)
\end{aligned}$$

as desired.

Main constraint. Since the sum of energies on edges is at most the overall energy,

$$\sum_{e \in Z_{mid}} (g_e^{(0)}(I'))^2 \leq b_{ss'}^T L^+_{(I \setminus D)/(S,S')} b_{ss'}$$

For edges e with $(h_e^{(0)}(I'))^2 \geq \frac{100}{|Z_{mid}|} b_{ss'}^T L^+_{(I \setminus D)} b_{ss'}$,

$$(g_e^{(0)}(I'))^2 \geq \frac{25}{|Z_{mid}|} b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}$$

By the previous inequality, this can only occur for $|Z_{mid}|/25$ edges in Z_{mid}, as desired.

Combining the parts. Adding all of the guarantees from the previous parts about Z_{mid} shows that at most $\frac{6|Z_{mid}|}{25}$ edges in Z_{mid} are removed by those constraints. This leaves a set Z with $|Z| \geq 19|Z_{mid}|/25$. Therefore, with probability at least $1/\text{polylog}(n)$, $|Z| \geq |W|/\text{polylog}(n)$. Repeating `FastOracle` $\text{polylog}(n)$ times therefore finds a set Z with the desired size lower bound with high probability.

□

4.11.3 Efficient approximations to required quantities

Now, we just need to efficiently compute approximations $h_e^{()}$ to the functions $g_e^{()}$ used in `FastOracle`. We do this by observing that

- Each of the functions are weighted ℓ_1 or ℓ_2 norms of columns of a matrix M with entries $M_{ij} := v_i^T L_H^+ v_j$.
- ℓ_1 and ℓ_2 norms of columns can be approximated using $O(\log n)$ "linear queries" of the form $v^T L_H^+ v_j$ by Theorem 4.9.13. All linear queries for all v_js can be computed using one Laplacian solve with demand vector v (preprocessing) and a constant amount of work for v_j (query). One can extract the desired information from the queries by taking the median, which takes $\tilde{O}(1)$ time per query.

While this is good enough for some of the functions we need to compute, we also need to leave out elements on the diagonal in some cases. To do this, we use the above idea to efficiently compute row sums for entries in some off-diagonal rectangle in the matrix. We show that all off-diagonal row sums can be approximated using $O(\log n)$ off-diagonal rectangle queries.

Full row norm approximations

In this subsection, we describe how to obtain 2-approximations for the functions $g_e^{(0)}$, $g_e^{(2)}$, $g_e^{(3),s}$, and $g_e^{(4),s}$ for arbitrary query edges e. $g_e^{(0)}(H)$ for all e can be computed using one Laplacian solve for the demand $b_{ss'}$ on the graph $(H \setminus D)/(S, S')$. $g_e^{(2)}(H)$ and $g_e^{(3),s}(H)$ are each weighted ℓ_1 norms, while $g_e^{(4),s}(H)$ is a weighted ℓ_2 norm.

We now implement the first parts of `ApxPreprocessing` and `ApxQuery` that compute these functions:

Algorithm 19: `ApxPreprocessing`(H, S, S', D, A, B, X) part 1 (functions with diagonal terms)

Input: a graph H, $S, S' \subseteq V(H)$, $D, A, B, X \subseteq E(H)$
Result: an implicit data structure for use by `ApxQuery`

`// main terms`

$X^{(0)} \leftarrow L^{+}_{(H\setminus D)/(S,S')} b_{ss'}$

`// deferred terms`

$C^{(2)} \leftarrow$ `SketchMatrix`$(|A| + |B|, 1/n^6, 1, 1/2)$

$D^{(2)} \leftarrow$ the $n \times (|A| + |B|)$ matrix with columns $b_{su} + b_{sv}$ for $\{u, v\} \in A$ and $b_{us'} + b_{vs'}$ for $\{u, v\} \in B$

$X^{(2)} \leftarrow L^{+}_{(H\setminus D)/(S,S')} D^{(2)} (C^{(2)})^T$

`// conductance terms`

$C^{(3),s} \leftarrow$ `SketchMatrix`$(|\partial_H S'|, 1/n^6, 1, 1/2)$

$D^{(3),s} \leftarrow$ the $n \times |\partial_H S'|$ matrix with columns $b_{sw}^T L^{+}_{(H\setminus D)/S} b_{sw} \sum_{e \in \partial_H w} \frac{b_e}{r_e}$ for edges $e \in \partial_H w$ for some $w \in S'$, where the effective resistance value is 1.1-approximated using Johnson-Lindenstrauss

$X^{(3),s} \leftarrow L^{+}_{(H\setminus D)/(S,S')} D^{(3),s} (C^{(3),s})^T$

$C^{(3),s'} \leftarrow$ `SketchMatrix`$(|\partial_H S'|, 1/n^6, 1, 1/2)$

Compute $X^{(3),s'}$ in the same way as $X^{(3),s}$ with s,S and s',S' interchanged

`// energy terms`

$C^{(4),s} \leftarrow$ `SketchMatrix`$(|S'|, 1/n^6, 2, 1/2)$

$D^{(4),s} \leftarrow$ the $n \times |S'|$ matrix with columns $\left(\sqrt{\sum_{e \in \partial_H w} \frac{b_{ss'}^T L^{+}_{(H\setminus D)/(S,S')} b_e}{r_e}} \right) b_{sw}$ for $w \in S'$, where the flow values are computed using the vector $L^{+}_{(H\setminus D)/(S,S')} b_{ss'}$ with one Laplacian solve

$X^{(4),s} \leftarrow L^{+}_{(H\setminus D)/S} D^{(4),s} (C^{(4),s})^T$

$C^{(4),s'} \leftarrow$ `SketchMatrix`$(|S'|, 1/n^6, 2, 1/2)$

Compute $X^{(4),s'}$ in the same way as $X^{(4),s}$, with s,S and s',S' swapped

Algorithm 20: `ApxQuery`$(g_f^{()}(H))$ for non-$g^{(1)}$ functions

Input: a function $g_f^{()}(H)$
Output: a 2-approximation to the value of the function
1 $d \leftarrow$ number of columns of $C^{()}$
2 $p \leftarrow 1$ or 2, depending on whether $g^{(4)}$ is being used
3 **return** *RecoverNorm*$((X^{()})^T(b_f/\sqrt{r_f}), d, 1/n^6, p, 1/2)$ *or this value squared if* $p = 2$

Row with diagonal element left out

Now, we compute $g^{(1)}$ and implement `ColumnApx`. These quantities differ from the ones discussed in the previous subsubsection in that they leave out one "diagonal" element in each row. One could try to subtract out this element, but this destroys multiplicative approximation.

Instead, we use sketching to approximate sums of rows of random off-diagonal submatrices. Specifically, the algorithm does the following for `ColumnApxPreprocessing`:

- Do $\Theta(\log n)$ times:
  - Pick a random balanced partition (Z_0, Z_1) of Z
  - For each $e \in Z_0$, approximate $a_e \leftarrow \sum_{f \in Z_1} \frac{|b_e^T L_{G'}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ using sketching
- For each $e \in Z$, average the a_es together and scale up the average by a factor of 4 to obtain an estimate for $\sum_{f \neq e \in Z} \frac{|b_e^T L_{G'}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$

This algorithm takes $\tilde{O}(m)$ time thanks to the sketching step. To show correctness, think about the indicator variable Y_{ef} that is 1 if and only if $e \in X_0$ and $f \in X_1$. This event happens with probability 1/4 in each of the $\Theta(\log n)$ trials if $e \neq f$ and with probability 0 otherwise. Since the trials are independent, the weight in front of each off-diagonal term concentrates around 1/4. Scaling up the average by a factor of 4 yields the desired result.

Now, we formally implement this idea. We analyze these implementations in the proof

of Proposition 4.11.8.

Algorithm 21: `ColumnApxPreprocessing`(G', Z)

Input: a graph G' and $Z \subseteq E(G')$
Result: nothing; implicit data structure

$K \leftarrow 100 \log n$
$a_e \leftarrow 0$ for each $e \in Z$
foreach $k \leftarrow 1, 2, \ldots, K$ **do**
  $Z_0, Z_1 \leftarrow$ uniformly random partition of Z with $|Z_0| - |Z_1| \leq 1$
  $C \leftarrow$ `SketchMatrix`$(|Z_1|, 1/n^6, 1, 1/4)$
  $D \leftarrow n \times |Z_1|$ matrix of columns $b_f/\sqrt{r_f}$ for $f \in Z_1$
  $U \leftarrow L^+_{G'} D C^T$
  foreach $e \in Z_0$ **do**
    Increment a_e by `RecoverNorm`$(U^T(b_e/\sqrt{r_e}), |Z_1|, 1/n^6, 1, 1/4)$
  end
end

Algorithm 22: `ColumnApx`(e)

Input: an edge $e \in Z$
Result: a 2-approximation to the value of $\sum_{f \neq e \in Z} \frac{|b_e^T L^+_{G'} b_f|}{\sqrt{r_e}\sqrt{r_f}}$

return $4a_e/K$

Algorithm 23: `ApxPreprocessing`(H, S, S', D, A, B, X) part 2 (functions without diagonal terms)

Input: a graph H, $S, S' \subseteq V(H)$, $D, A, B, X \subseteq E(H)$
Result: an implicit data structure for use by `ApxQuery`
`// code for` s `only given for clarity; swap` s, S `for` s', S'
$K \leftarrow 100 \log n$
$a_e \leftarrow 0$ for each $e \in X$
foreach $k \leftarrow 1, 2, \ldots, K$ **do**
  $X_0, X_1 \leftarrow$ uniformly random partition of X with $|X_0| - |X_1| \leq 1$
  $C \leftarrow$ `SketchMatrix`$(|X_1|, 1/n^6, 1, 1/4)$
  $D \leftarrow n \times |X_1|$ matrix of columns $(b_{su} + b_{sv})/2$ for $\{u, v\} \in X_1$
  $U \leftarrow L^{+}_{(H \setminus D)/(S,S')} DC^T$
  foreach $e \in X_0$ **do**
    Increment a_e by `RecoverNorm`$(U^T(b_e/\sqrt{r_e}), |X_1|, 1/n^6, 1, 1/4)$
  end
end

Algorithm 24: `ApxQuery`$(g_e^{(1),X,s})$

Input: the function $g_e^{(1),X,s}$
Result: a 2-approximation to its value on H
`// code for` s `only given for clarity; swap` s, S `for` s', S'
return $4a_e/K$

Combining the parts

Proposition 4.11.8. *The following guarantees hold for* ***`ApxQuery`****,* ***`ColumnApx`****,* ***`ApxPreprocessing`****, and* ***`ColumnApxPreprocessing`****:*

- *(Correctness) Each call to* ***`ApxQuery`*** *and* ***`ColumnApx`*** *returns a 2-approximation to the correct value.*
- *(Query runtime) Each call to* ***`ApxQuery`*** *and* ***`ColumnApx`*** *takes $\tilde{O}(1)$ time.*
- *(Preprocessing runtime) Each call to* ***`ApxPreprocessing`*** *and* ***`ColumnApxPreprocessing`*** *takes $\tilde{O}(m)$ time.*

Proof. **Correctness for `ApxQuery` for all but $g^{(1)}$.** Follows directly from the "Approximation" guarantee of Theorem 4.9.13.

Correctness for `ApxQuery` for $g^{(1)}$ and `ColumnApx`. We focus on $g_e^{(1),X,s}$, since an intuitive overview for `ApxQuery` was given earlier and the proof is very similar in both cases. Let $Y_{ef}^{(k)}$ be the indicator variable of the event $\{e \in X_0 \text{ and } f \in X_1\}$. By the

"Approximation" guarantee of Theorem 4.9.13, at the end of the outer foreach loop in `ApxPreprocessing`,

$$a_e \in [3/4, 5/4] \left(\sum_{f=\{u,v\} \in X} \frac{|b_e^T L^+_{(H \setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \right)$$

for each $e \in X$. Since $Y_{ee}^{(k)} = 0$ for all k and $e \in X$,

$$\left(\sum_{f=\{u,v\} \in X} \frac{|b_e^T L^+_{(H \setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \right)$$
$$= \left(\sum_{f=\{u,v\} \in X \setminus e} \frac{|b_e^T L^+_{(H \setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \right)$$

Notice that $\mathbf{E}[Y_{ef}^{(k)}] = 1/4$ if $e \neq f$ and that for any fixed e, f, the collection of random variables $\{Y_{ef}^{(k)}\}_{k=1}^{K}$ is independent. Therefore, by Chernoff bounds (using the value of K) and a union bound over all pairs $e \neq f \in X$,

$$3K/16 \leq \sum_{k=1}^{K} Y_{ef}^{(k)} \leq 5K/16$$

Therefore,

$$a_e \in [9K/64, 25K/64] \left(\sum_{f=\{u,v\} \in X \setminus e} \frac{|b_e^T L^+_{(H \setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \right)$$

which means that $4a_e/K$ is a 2-approximation, as desired.

Query runtime. `ColumnApx` and `ApxQuery` for $g^{(1)}$ just return precomputed values, so they both take constant time. `ApxQuery` for other functions computes a matrix-vector product with a vector that is supported on only two entries. Therefore, this product only takes time proportional to the number of rows of the matrix, which is $\ell = O(\log n)$ by Theorem 4.9.13. `RecoverNorm` only take $\text{poly}(\ell) = \text{polylog}(n)$ time by the "Runtime" guarantee of Theorem 4.9.13. Therefore, all queries take $\text{polylog}(n)$ time, as desired.

Preprocessing runtime. Each X matrix computation takes near-linear time, as it involves a ℓ sparse matrix-vector products to compute DC^T and ℓ Laplacian solves to compute L^+DC. Each `SketchMatrix` call takes $\tilde{O}(m)$ time by Theorem 4.9.13. All other operations take $\tilde{O}(m)$ time, so the total precomputation runtime is $\tilde{O}(m)$, as desired.

□

4.11.4 Proof of Lemma 4.8.2 (an algorithm for the fixing lemma with almost-linear runtime)

Now, we combine all of the results of this section to prove Lemma 4.8.2:

Proof of Lemma 4.8.2. The result follows immediately from Lemma 4.10.8 and Proposition 4.11.7 with `FastOracle` substituted in for `Oracle` in the `Fix` algorithm. Call this algorithm `FastFix`. `FastOracle`'s runtime is bounded using Proposition 4.11.8. □

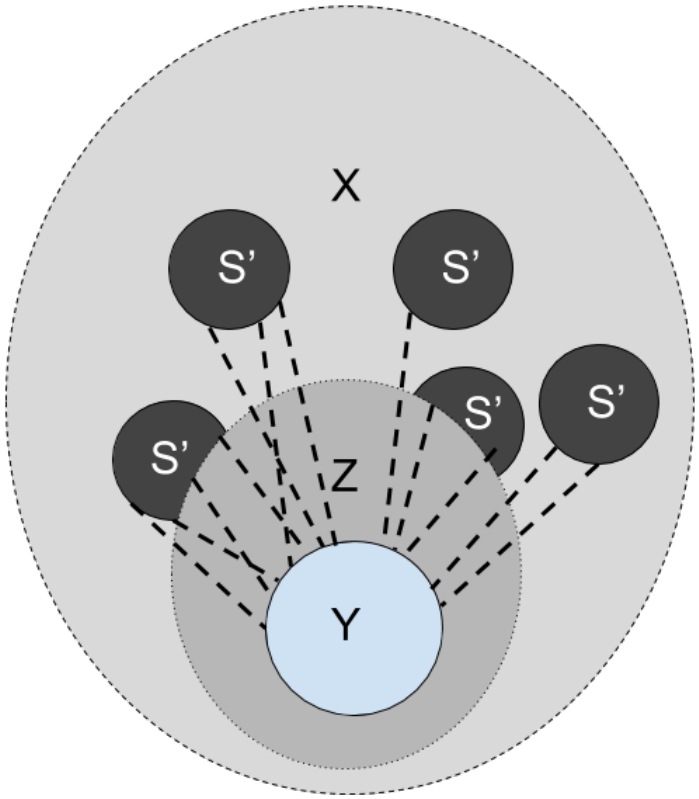

Figure 4.8.2: The first step in bounding conductivity. To bound the $Y - S'$ conductance in $\texttt{Schur}(H_{\mathcal{C}}, Y \cup S' \cup (V(H) \setminus X))$, it suffices to bound the $Y - (V(H) \setminus Z)$-conductance in $\texttt{Schur}(H_{\mathcal{C}}, Y \cup (V(H) \setminus Z))$, which is bounded because Z is 1/4th of the way from Y to ∂X. More formally, we apply Lemma 4.8.5 with $p = 1/4$.

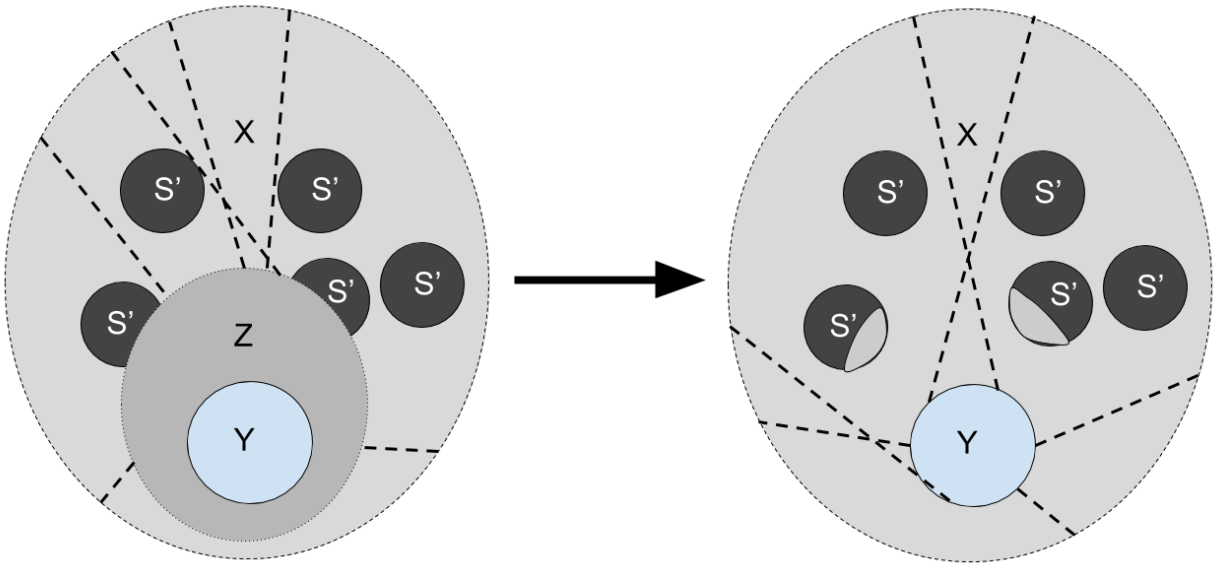

Figure 4.8.3: The second step in bounding conductivity. To bound the direct $Y - V(H) \setminus X$ conductance, it suffices to bound the direct $Z - V(H) \setminus X$ conductance. This is bounded using Lemma 4.8.5 on $X \setminus Z$ with $p = 3/4$.

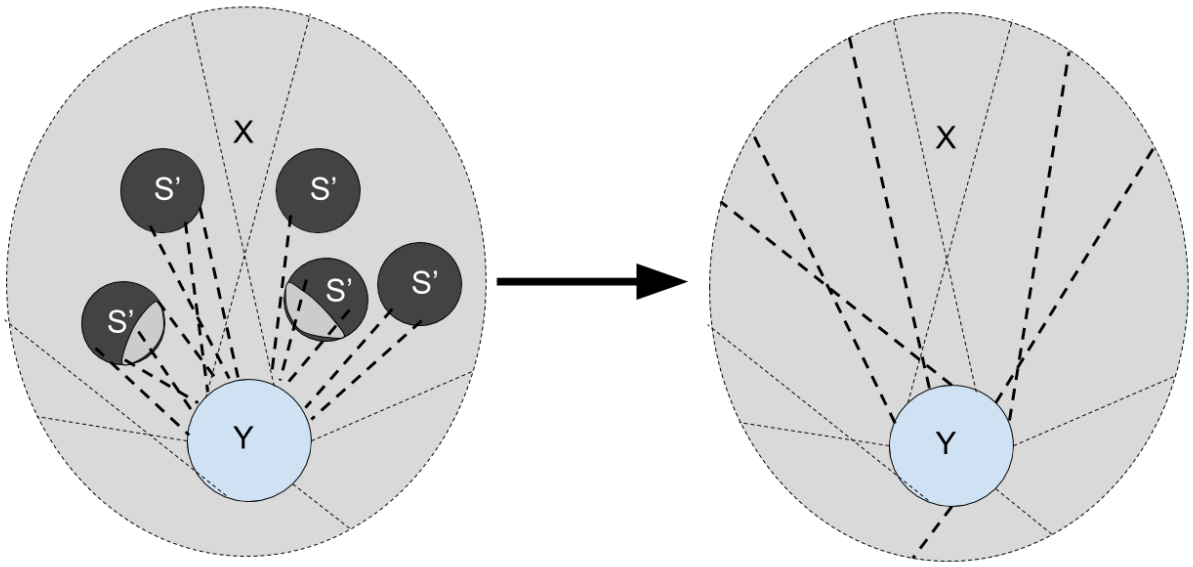

Figure 4.8.4: The third step in bounding conductivity. The light dashed edges are the direct $Y - V(H) \setminus X$ edges, whose total conductance was bounded in the second step. The dark dashed edges obtained by eliminating S' have smaller total conductance than the direct dark dashed edges to S' by Lemma 4.8.3. The total conductance of these edges in $\texttt{Schur}(H'_{\mathcal{C}}, Y \cup S' \cup V(H) \setminus X)$ is bounded by the first part and Lemma 4.8.2. The end goal was to bound $Y - (V(H) \setminus X)$ conductance in the Schur complement $\texttt{Schur}(H'_{\mathcal{C}}, Y \cup (V(H) \setminus X))$ obtained from the fixed graph $H'_{\mathcal{C}}$, so we are done.

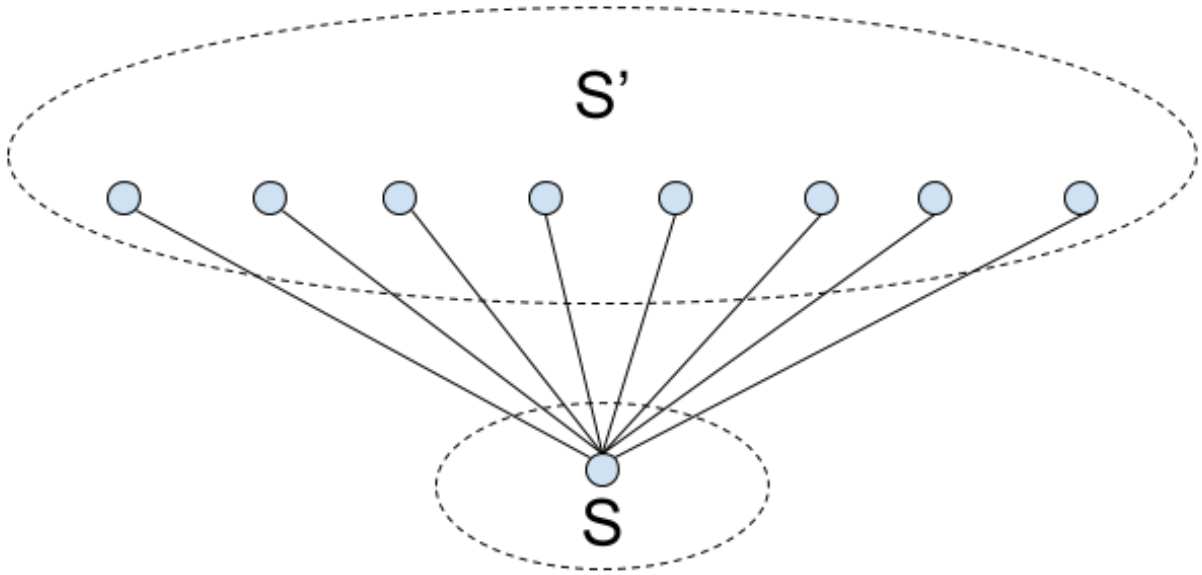

Figure 4.10.1: Why Lemma 4.10.1 has a dependence on $\Delta^I(S, S')$. In this diagram, let F be the set of all edges and $D = \emptyset$. Conditioning on a random spanning tree in I contracts all of the edges in F. In order to make the $S - S'$ conductance finite, all of the edges in F must be deleted. There are $\Delta^I(S, S')$ of these edges, no matter what the conductances of these edges are.

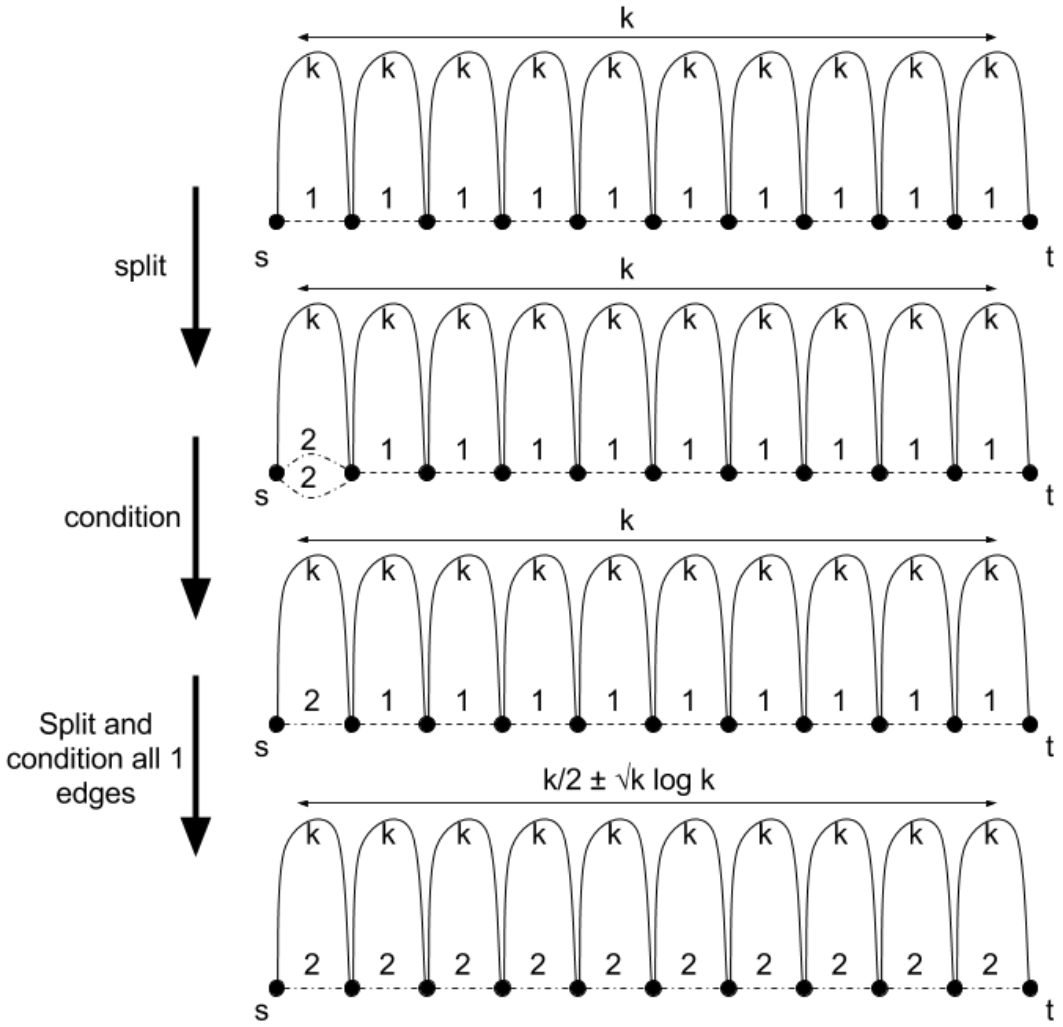

Figure 4.10.2: The `SpecialFix` algorithm for $|F|/2$ iterations. In the above example, each of the dashed edges is in F. During each iteration, one edge is chosen, split, and conditioned on. During the first conditioning round in this example, the chosen edge is deleted. In the last diagram, all contracted edges yield self-loops with resistance k, which are irrelevant for the $s-t$ resistance. With high probability, roughly $k/2$ edges are left in F after conditioning on each edge once. Furthermore, the $s-t$ resistance is extremely close to its original value by martingale concentration.

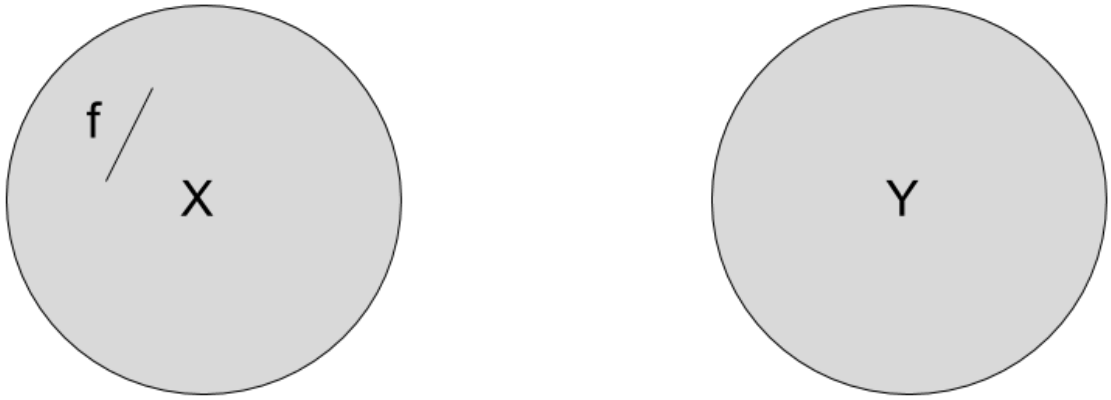

Figure 4.10.3: Depiction of Lemma 4.10.12. When Y is identified, the expected number of random spanning tree edges decreases.

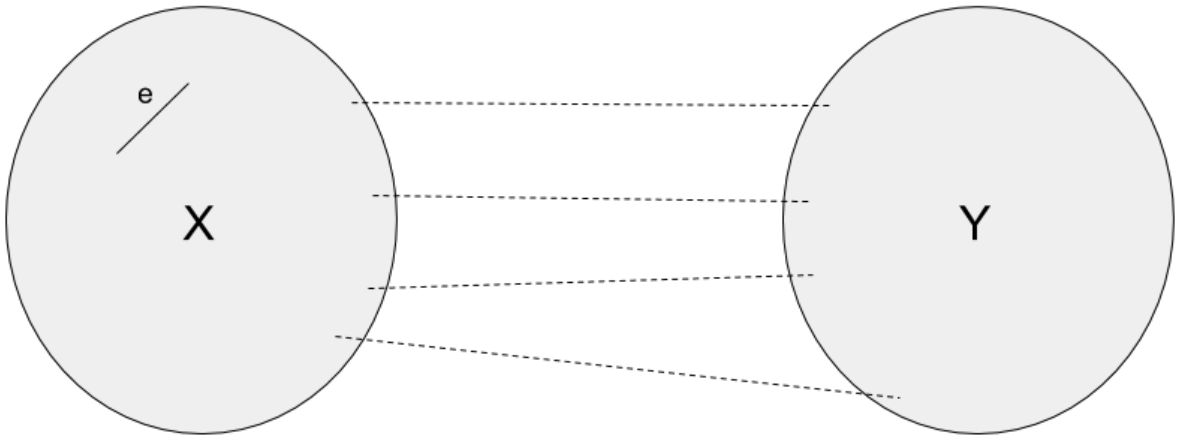

Figure 4.10.4: Depiction of Lemma 4.10.14. When $\Delta^G(X, Y)$ is low, the vertices in Y are similar to each other from the perspective of $s - Y$ demand vectors.

Chapter 5

Spectral Subspace Sparsification

5.1 Introduction

Graph sparsification has had a number of applications throughout algorithms and theoretical computer science. In this work, we loosen the requirements of spectral sparsification and show that this loosening enables us to obtain sparsifiers with fewer edges. Specifically, instead of requiring that the Laplacian pseudoinverse quadratic form is approximated for every vector, we just require that the sparsifier approximates the Laplacian pseudoinverse quadratic form on a subspace:

Definition 5.1.1 (Spectral subspace sparsifiers)**.** *Consider a weighted graph G, a vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$ that is orthogonal to $\mathbf{1}^{V(G)}$, and $\epsilon \in (0,1)$. For a minor H of G with contraction map $\phi : V(G) \to V(H)$, let $P \in \mathbb{R}^{V(H)\times V(G)}$ be a matrix with $P_{uv} = \mathbb{1}[u = \phi(v)]$ for all $u \in V(H), v \in V(G)$. A reweighted minor H of G is a called an* $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier *if for all vectors $x \in \mathcal{S}$,*

$$(1-\epsilon)x^T L_G^+ x \leq x_H^T L_H^+ x_H \leq (1+\epsilon)x^T L_G^+ x$$

where $x_H := Px$.

[52] also considers a form of specific form of subspace sparsification related to controlling the k smallest eigenvalues of a spectral sparsifier for $\mathcal{S} = \mathbb{R}^{V(G)}$. When $\mathcal{S}$ is the dimension $|S| - 1$ subspace of $\mathbb{R}^{|S|} \times \mathbf{0}^{n-|S|}$ that is orthogonal to $\mathbf{1}^{V(G)}$, a (S, ϵ)-spectral subspace sparsifier is a sparsifier for the Schur complement of G restricted to the set of vertices S. Schur complement sparsifiers are implicitly constructed in [58] and [59] by an approximate Gaussian elimination procedure and have been used throughout spectral graph theory. For example, they are used in algorithms for random spanning tree generation [33, 32], approximate maximum flow [77], and effective resistance computation [37, 38, 33].

Unlike the existing construction of Schur complement sparsifiers [33], our algorithm (a) produces a sparsifier with vertices outside of S and (b) produces a sparsifier that is a minor of the input graph. While (a) is a disadvantage to our approach, it is not a problem in

applications, in which the number of edges in the sparsifier is the most relevant feature for performance, as illustrated by our almost-optimal algorithm for ϵ-approximate effective resistance computation. (b) is an additional benefit to our construction and connects to the well-studied class of Steiner point removal problems [22, 34].

In the Approximate Terminal Distance Preservation problem [22], one is given a graph G and a set of k vertices S. One is asked find a reweighted minor H of G with size $\text{poly}(k)$ for which

$$d_G(u,v) \leq d_H(u,v) \leq \alpha d_G(u,v)$$

for all $u, v \in S$ and some small *distortion* $\alpha > 1$. The fact that H is a minor of G is particularly useful in the context of planar graphs. One can equivalently phrase this problem as a problem of finding a minor H in which the ℓ_1-norm of the ℓ_1-minimizing flow between any two vertices $s, t \in S$ is within an α-factor of the ℓ_1 norm of the ℓ_1-minimizing $s-t$ flow in G. The analogous problem for ℓ_∞ norms is the problem of constructing a *flow sparsifier* (with non-$s-t$ demands as well). Despite much work on flow sparsifiers [79, 65, 20, 74, 34, 25, 10, 83], it is still not known whether $\alpha = (1+\epsilon)$-flow sparsifiers with size $\text{poly}(k, 1/\epsilon)$ exist, even when the sparsifier is not a minor of the original graph.

5.1.1 Our Results

Our main result is the following:

Theorem 5.1.2. *Consider a weighted graph G, a d-dimensional vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$, and $\epsilon \in (0,1)$. Then an $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier for G with $O\left(\frac{d \log d}{\epsilon^2}\right)$ edges exists.*

When $\mathcal{S}$ is the maximal subspace of $\mathbb{R}^S \times \mathbf{0}^{V(G)\setminus S}$ orthogonal to $\mathbf{1}^{V(G)}$ for some set of vertices $S \subseteq V(G)$, $(\mathcal{S}, \epsilon)$-spectral subspace sparsifiers satisfy the same approximation guarantee as Schur complement sparsifiers. The approximation guarantee of a spectral subspace sparsifier H of G is equivalent to saying that for any demand vector $d \in \mathcal{S}$, the energy of the ℓ_2-minimizing flow for d in H is within a $(1+\epsilon)$ factor of the energy for the ℓ_2-minimizing flow for d in G. This yields an near-optimal (up to a $\log d$ factor) answer to the $(1+\epsilon)$-approximate Steiner vertex removal problem for the ℓ_2 norm. The ℓ_2 version is substantially different from the ℓ_1 problem, in which there do not exist $o(k^2)$-size minors that 2-approximate all terminal distances [22].

Unlike Schur complement sparsifiers, $(\mathbb{R}^S, \epsilon)$-spectral subspace sparsifiers may contain "Steiner nodes;" i.e. vertices outside of S. This is generally not relevant in applications, as we illustrate in Section 5.6. Allowing Steiner nodes allows us to obtain sparsifiers with fewer edges, which in turn allows us to obtain faster constructions. Specifically, we show the following result:

Theorem 5.1.3. *Consider a weighted graph G, a set of vertices $S \subseteq V(G)$, and $\epsilon \in (0,1)$. Let $\mathcal{T}_{rst}(G)$ denote the time it takes to generate a random spanning tree from a distribution with total variation distance at most $1/m^{10}$ from the uniform distribution. Then*

$(\mathbb{R}^S \times \mathbf{0}^{V(G)\setminus S}, \epsilon)$-spectral subspace sparsifier for G with $\min(m, O\left(|S|\frac{polylog(n)}{\epsilon^2}\right))$ edges can be constructed in $\mathcal{T}_{rst}(G) + O(mpolylog(n)) \leq m^{1+o(1)}$ time.

This sparsifier has as many edges as the Schur complement sparsifier given in [33] but improves on their $\tilde{O}(m + n/\epsilon^2)$ runtime. An important ingredient in the above construction is an algorithm for multiplicatively approximating changes in effective resistances due to certain modifications of G. This algorithm is called with $\delta = \Theta(1)$ in this paper:

Lemma 5.1.4. *Consider a weighted graph G, a set of vertices $S \subseteq V(G)$, and $\delta_0, \delta_1 \in (0,1)$. There is an $O(mpolylog(n)\log(m/\delta_1)/\delta_0^2)$-time algorithm* `DiffApx`$(G, S, \delta_0, \delta_1)$ *that outputs numbers ν_e for all $e \in E(G)$ with the guarantee that*

$$(1-\delta_0)\nu_e - \delta_1 \leq \frac{b_e^T L_G^+ b_e}{r_e} - \frac{b_e^T L_{G/S}^+ b_e}{r_e} \leq (1+\delta_0)\nu_e + \delta_1$$

Finally, we replace the use of Theorem 6.1 in [33] with our Theorem 5.1.3 in their improvement to Johnson-Lindenstrauss to obtain a faster algorithm:

Corollary 5.1.5. *Consider a weighted graph G, a set of pairs of vertices $P \subseteq V(G) \times V(G)$, and an $\epsilon \in (0,1)$. There is an $m^{1+o(1)} + \tilde{O}(|P|/\epsilon^2)$-time algorithm* `ResApx`$(G, P, \epsilon)$ *that outputs $(1+\epsilon)$-multiplicative approximations to the quantities*

$$b_{uv}^T L_G^+ b_{uv}$$

for all pairs $(u,v) \in P$.

This directly improves upon the algorithm in [33], which takes $O((m + (n + |P|)/\epsilon^2)\text{polylog}(n))$-time.

5.1.2 Technical Overview

To construct Schur complement sparsifiers, [33] eliminates vertices one-by-one and sparsifies the cliques resulting from those eliminations. This approach is fundamentally limited in that each clique sparsification takes $\Omega(1/\epsilon^2)$ time in general. Furthermore, in the $n+1$ vertex star graph with n vertices $v_1, v_2, \ldots, v_n$ connected to a single vertex v_{n+1}, a $(1+\epsilon)$-approximate Schur complement sparsifier without Steiner vertices for the set $\{v_1, v_2, \ldots, v_n\}$ must contain $\Omega(n/\epsilon^2)$ edges. As a result, it seems difficult to obtain Schur complement sparsifiers in time less than $\tilde{O}(m + n/\epsilon^2)$ time using vertex elimination.

Instead, we eliminate edges from a graph by contracting or deleting them. Edge elimination has the attractive feature that, unlike vertex elimination, it always reduces the number of edges. Start by letting $H := G$. To eliminate an edge e from the current graph H, sample $X_e \sim \text{Ber}(p_e)$ for some probability p_e depending on e, contract e if $X_e = 1$, and delete e if $X_e = 0$.

To analyze the sparsifier produced by this procedure, we set up a matrix-valued martingale and reduce the problem to bounding the maximum and minimum eigenvalues of a random matrix with expectation equal to the identity matrix. The right value for p_e for preserving this matrix in expectation turns out to be the probability that a uniformly random spanning tree of H contains the edge e. To bound the variance of the martingale, one can use the Sherman-Morrison rank one update formula to bound the change in L_H^+ due to contracting or deleting the edge e. When doing this, one sees that the maximum change in eigenvalue is at most a constant times

$$\max_{x \in \mathcal{S}} \frac{(x^T L_H^+ b_e)^2}{r_e \min(\texttt{lev}_H(e), 1 - \texttt{lev}_H(e))(x^T L_G^+ x)}$$

where $\texttt{lev}_H(e)$ is the probability that e is in a uniformly random spanning tree of H. This quantity is naturally viewed as the quotient of two quantities:

(a) The maximum fractional energy contribution of e to any demand vector in $\mathcal{S}$'s electrical flow.

(b) The minimum of the probabilities that e is in or is not in a uniformly random spanning tree of H.

We now make the edge elimination algorithm more specific to bound these two quantities. Quantity (a) is small on average over all edges in e (see Proposition 5.3.9), so choosing the lowest-energy edge yields a good bound on the maximum change. To get a good enough bound on the stepwise martingale variance, it suffices to sample an edge uniformly at random from the half of edges with lowest energy. Quantity (b) is often not bounded away from 0, but can be made so by modifying the sampling procedure. Instead of contracting or deleting the edge e, start by *splitting* it into two parallel edges with double the resistance or two series edges with half the resistance, depending on whether or not $\texttt{lev}_H(e) \leq 1/2$. Then, pick one of the halves e_0, contract it with probability p_{e_0}, or delete it otherwise. This produces a graph in which the edge e is either contracted, deleted, or reweighted. This procedure suffices for proving our main existence result (Theorem 5.1.2). This technique is similar to the technique used to prove Lemma 4.1.4.

While the above algorithm does take polynomial time, it does not take almost-linear time. We can accelerate it by batching edge eliminations together using what we call *steady oracles*. The contraction/deletion/reweight decisions for edges in H during each batch can be made by sampling just one $1/m^{10}$-approximate uniformly random spanning tree, which takes $m^{1+o(1)}$ time. The main remaining difficulty is finding a large set of edges for which quantity (a) does not change much over the course of many edge contractions/deletions. To show the existence of such a set, we exploit electrical flow localization (Chapter 3). To find this set, we use matrix sketching and a new primitive for approximating the change in leverage score due to the identification of some set of vertices S (Lemma 5.1.4), which may be of independent interest. The primitive for approximating the change works by writing

the change in an Euclidean norm, reducing the dimension by Johnson-Lindenstrauss Lemma, and then computing the embedding by Fast Laplacian Solvers in near-linear time.

We conclude by briefly discussing why localization is relevant for showing that quantity (a) does not change over the course of many iterations. The square root of the energy contribution of an edge e to x's electrical flow after deleting an edge f is

$$\begin{aligned}\left|\frac{x^T L_{H\setminus f}^+ b_e}{\sqrt{r_e}}\right| &= \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}} + \frac{(x^T L_H^+ b_f)(b_f^T L_H^+ b_e)}{(r_f - b_f^T L_H^+ b_f)\sqrt{r_e}}\right| \\ &= \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}} + \frac{1}{1-\mathtt{lev}_H(f)}\frac{x^T L_H^+ b_f}{\sqrt{r_f}}\frac{b_f^T L_H^+ b_e}{\sqrt{r_f}\sqrt{r_e}}\right| \\ &\leq \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}}\right| + \frac{1}{1-\mathtt{lev}_H(f)}\left|\frac{x^T L_H^+ b_f}{\sqrt{r_f}}\right|\left|\frac{b_f^T L_H^+ b_e}{\sqrt{r_f}\sqrt{r_e}}\right|\end{aligned}$$

by Sherman-Morrison. In particular, the new energy on e is at most the old energy plus some multiple of the energy on the deleted edge f. By Theorem 3.3.1 applied with edge weights 1, the average value of this multiplier over all edges e and f is $\tilde{O}(\frac{1}{|E(H)|})$, which means that the algorithm can do $\tilde{\Theta}(|E(H)|)$ edge deletions/contractions without seeing the maximum energy on edges e change by more than a factor of 2.

5.2 Preliminaries

5.2.1 Graphs and Laplacians

For a graph G and a subset of vertices S, let G/S denote the graph obtained by *identifying* S to a single vertex s. Specifically, for any edge $e = \{u, v\}$ in G, replace each endpoint $u, v \in S$ with s and do not change any endpoint not in S. Then, remove all self-loops to obtain G/S.

Let $G = (V(G), E(G))$ be a weighted undirected graph with n vertices, m edges, and edge weights $\{w_e\}_{e\in E(G)}$. The Laplacian of G is an $n \times n$ matrix given by:

$$(L_G)_{u,v} := \begin{cases} -w_{(u,v)} & \text{if } u \neq v \text{ and } (u,v) \in E(G), \\ \sum_{(u,w)\in E(G)} w_{(u,w)} & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases}$$

We define edge resistances $\{r_e\}_{e\in E(G)}$ by $r_e = 1/w_e$ for all $e \in E(G)$.

If we orient every edge $e \in E(G)$ arbitrarily, we can define the signed edge-vertex incidence matrix B_G by

$$(B_G)_{e,u} := \begin{cases} 1 & \text{if } u \text{ is } e\text{'s head,} \\ -1 & \text{if } u \text{ is } e\text{'s tail,} \\ 0 & \text{otherwise.} \end{cases}$$

Then we can write L_G as $L_G = B_G^T W_G B_G$, where W_G is a diagonal matrix with $(W_G)_{e,e} = w_e$.

For vertex sets $S, T \subseteq V$, $(L_G)_{S,T}$ denotes the submatrix of L_G with row indices in S and column indices in T.

L_G is always positive semidefinite, and only has one zero eigenvalue if G is connected. For a connected graph G, let $0 = \lambda_1(L_G) < \lambda_2(L_G) \leq \ldots \leq \lambda_n(L_G)$ be the eigenvalues of L_G. Let $u_1, u_2, \ldots, u_n$ be the corresponding set of orthonormal eigenvectors. Then, we can diagonalize L_G and write

$$L_G = \sum_{i=2}^{n} \lambda_i(L_G) u_i u_i^T.$$

The pseudoinverse of L_G is then given by

$$L_G^+ = \sum_{i=2}^{n} \frac{1}{\lambda_i(L_G)} u_i u_i^T.$$

In the rest of the paper, we will write $\lambda_{\min}(\cdot)$ to denote the smallest eigenvalue and $\lambda_{\max}(\cdot)$ to denote the largest eigenvalue. We will also write $\sigma_{\max}(\cdot)$ to denote the largest singular value, which is given by

$$\sigma_{\max}(A) = \sqrt{\lambda_{\max}(A^T A)}$$

for any matrix A.

We will also need to use Schur complements which are defined as follows:

Definition 5.2.1 (Schur Complements)**.** *The Schur complement of a graph G onto a subset of vertices $S \subset V(G)$, denoted by $SC(G, S)$ or $SC(L_G, S)$, is defined as*

$$SC(L_G, S) = (L_G)_{S,S} - (L_G)_{S,T} (L_G)_{T,T}^{-1} (L_G)_{T,S},$$

where $T := V(G) \setminus S$.

The fact below relates Schur complements to the inverse of graph Laplacian:

Fact 5.2.2 (see, e.g., Fact 5.4 in [33])**.** *For any graph G and $S \subset V(G)$,*

$$\left(I - \frac{1}{|S|} J\right) \left(L_G^+\right)_{S,S} \left(I - \frac{1}{|S|} J\right) = \left(SC(L_G, S)\right)^+,$$

where I denotes the identity matrix, and J denotes the matrix whose entries are all 1.

5.2.2 Leverage scores and rank one updates

For a graph G and an edge $e \in E(G)$, let $b_e \in \mathbb{R}^{V(G)}$ denote the signed indicator vector of the edge e; that is the vector with -1 on one endpoint, 1 on the other, and 0 everywhere else. Define the *leverage score* of e to be the quantity

$$\texttt{lev}_G(e) := \frac{b_e^T L_G^+ b_e}{r_e}$$

Let $d_1, d_2 \in \mathbb{R}^n$ be two vectors with $d_1, d_2 \perp \mathbf{1}^{V(G)}$. Then the following results hold by the Sherman-Morrison rank 1 update formula:

Proposition 5.2.3. *For a graph G and an edge f, let $G\backslash f$ denote the graph with f deleted. Then*

$$d_1^T L_{G\backslash f}^+ d_2 = d_1^T L_G^+ d_2 + \frac{(d_1^T L_G^+ b_f)(b_f^T L_G^+ d_2)}{r_f - b_f^T L_G^+ b_f}$$

Proposition 5.2.4. *For a graph G and an edge f, let G/f denote the graph with f contracted. Then*

$$d_1^T L_{G/f}^+ d_2 = d_1^T L_G^+ d_2 - \frac{(d_1^T L_G^+ b_f)(b_f^T L_G^+ d_2)}{b_f^T L_G^+ b_f}$$

5.2.3 Random spanning trees

We use the following result on uniform random spanning tree generation:

Theorem 5.2.5 (Restatement of Theorem 4.1.2)**.** *Given a weighted graph G with m edges, a random spanning tree T of G can be sampled from a distribution with total variation distance at most $1/m^{10}$ from the uniform distribution in time $m^{1+o(1)}$.*

Let $T \sim G$ denote the uniform distribution over spanning trees of G. We also use the following classic result:

Theorem 5.2.6 ([51])**.** *For any edge $e \in E(G)$, $\Pr_{T\sim G}[e \in T] = \texttt{lev}_G(e)$.*

For an edge $e \in E(G)$, let $G[e]$ denote a random graph obtained by contracting e with probability $\texttt{lev}_G(e)$ and deleting e otherwise.

5.2.4 Some useful bounds and tools

We now describe some useful bounds/tools we will need in our algorithms. In all the following bounds, we define the quantities $w_{\max}$ and $w_{\min}$ as follows:

$$w_{\max} := \max\left\{1, \max_{e\in E(G)} 1/r_e\right\},$$
$$w_{\min} := \min\left\{1, \min_{e\in E(G)} 1/r_e\right\}.$$

The following lemma bounds the range of eigenvalues for Laplacians and SDDM matrices:

Lemma 5.2.7. *For any Laplacian L_G and $S \subset V(G)$,*

$$\lambda_2(L_G) \geq w_{\min}/n^2, \tag{5.1}$$
$$\lambda_{\min}\left((L_G)_{S,S}\right) \geq w_{\min}/n^2, \tag{5.2}$$
$$\lambda_{\max}\left((L_G)_{S,S}\right) \leq \lambda_{\max}(L_G) \leq nw_{\max}. \tag{5.3}$$

Proof. Defered to Appendix C.1. □

The lemma below gives upper bounds on the largest eigenvalues/singular values for some useful matrices:

Lemma 5.2.8. *The following upper bounds on the largest singular values/eigenvalues hold:*

$$\sigma_{\max}(W_G^{1/2}B_G) \leq (nw_{\max})^{1/2}, \tag{5.4}$$
$$\lambda_{\max}(SC(L_G, S)) \leq nw_{\max}, \tag{5.5}$$
$$\sigma_{\max}((L_G)_{S,T}) = \sigma_{\max}((L_G)_{T,S}) \leq nw_{\max}, \tag{5.6}$$

where $T := V(G) \setminus S$.

Proof. Defered to Appendix C.2. □

We will need to invoke Fast Laplacian Solvers to apply the inverse of a Laplacian of an SDDM matrix. The following lemma characterizes the performance of Fast Laplacian Solvers:

Lemma 5.2.9 (Fast Laplacian Solver [91, 26])**.** *There is an algorithm $\tilde{x} = \texttt{LaplSolve}(M, b, \epsilon)$ which takes a matrix $M_{n\times n}$ either a Laplacian or an SDDM matrix with m nonzero entries, a vector $b \in \mathbb{R}^n$, and an error parameter $\epsilon > 0$, and returns a vector $\tilde{x} \in \mathbb{R}^n$ such that*

$$\|x - \tilde{x}\|_M \leq \epsilon \|x\|_M$$

holds with high probability, where $\|x\|_M := \sqrt{x^T M x}$, $x := M^{-1}b$, and M^{-1} denotes the pseudoinverse of M when M is a Laplacian. The algorithm runs in time $O(m\text{polylog}(n)\log(1/\epsilon))$.

The following lemmas show how to bound the errors of Fast Laplacian Solvers in terms of ℓ_2 norms, which follows directly from the bounds on Laplacian eigenvalues in Lemma 5.2.7:

Lemma 5.2.10. *For any Laplacian L_G, vectors $x, \tilde{x} \in \mathbb{R}^n$ both orthogal to $\mathbf{1}$, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_{L_G} \leq \epsilon \|x\|_{L_G},$$

the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\|.$$

Proof. Defered to Appendix C.3. □

Lemma 5.2.11. *For any Laplacian L_G, $S \subset V$, vectors $x, \tilde{x} \in \mathbb{R}^{|S|}$, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_M \leq \epsilon \|x\|_M ,$$

where $M := (L_G)_{S,S}$, the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\| .$$

Proof. Defered to Appendix C.3. □

When computing the changes in effective resistances due to the identification of a given vertex set (i.e. merging vertices in that set and deleting any self loops formed), we will need to use Johnson-Lindenstrauss lemma to reduce dimensions:

Lemma 5.2.12 (Johnson-Lindenstrauss Lemma [45, 1])**.** *Let $v_1, v_2, \ldots, v_n \in \mathbb{R}^d$ be fixed vectors and $0 < \epsilon < 1$ be a real number. Let k be a positive integer such that $k \geq 24 \log n/\epsilon^2$ and $Q_{k\times d}$ be a random ± 1 matrix. With high probability, the following statement holds for any $1 \leq i, j \leq n$:*

$$(1-\epsilon) \|v_i - v_j\|^2 \leq \|Qv_i - Qv_j\|^2 \leq (1+\epsilon) \|v_i - v_j\|^2 .$$

5.3 Existence of sparsifiers

In this section, we reduce the construction of spectral subspace sparsifiers to an oracle that outputs edges that have low energy with respect to every demand vector in the chosen subspace $\mathcal{S}$. We prove it by splitting and conditioning on edges being present in a uniformly random spanning tree one-by-one until $\tilde{O}(d/\epsilon^2)$ edges are left. This construction is a high-dimensional generalization of the construction given in Section 4.10.1. We use the following matrix concentration inequality:

Theorem 5.3.1 (Matrix Freedman Inequality applied to symmetric matrices [93])**.** *Consider a matrix martingale $(Y_k)_{k\geq 0}$ whose values are symmetric matrices with dimension s, and let $(X_k)_{k\geq 1}$ be the difference sequence $X_k := Y_k - Y_{k-1}$. Assume that the difference sequence is uniformly bounded in the sense that*

$$\lambda_{\max}(X_k) \leq R$$

almost surely for $k \geq 1$. Define the predictable quadratic variation process of the martingale:

$$W_k := \sum_{j=1}^{k} \boldsymbol{E}[X_j^2 | Y_{j-1}]$$

Then, for all $t \geq 0$ *and* $\sigma^2 > 0$,

$$\Pr[\exists k \geq 0 : \lambda_{\max}(Y_k - Y_0) \geq t \text{ and } \lambda_{\max}(W_k) \leq \sigma^2] \leq s \exp\left(\frac{-t^2/2}{\sigma^2 + Rt/3}\right)$$

Now, we give an algorithm $\texttt{SubspaceSparsifier}(G, \mathcal{S}, \epsilon)$ that proves Theorem 5.1.2. The algorithm simply splits and conditions on the edge that minimizes the martingale difference repeatedly until there are too few edges left. For efficiency purposes, $\texttt{SubspaceSparsifier}(G, \mathcal{S}, \epsilon)$ receives martingale-difference-minimizing edges from a steady oracle $\mathcal{O}$ with the additional guarantee that differences remain small after many edge updates. This oracle is similar to the stable oracles given in Section 4.10.

Definition 5.3.2 (Steady oracles)**.** *A* $(\rho, K(z))$-steady oracle *is a function* $Z \leftarrow \mathcal{O}(I, \mathcal{S})$ *that takes in a graph* I *and a subspace* $\mathcal{S} \subseteq \mathbb{R}^{V(I)}$ *that satisfy the following condition:*

- *(Leverage scores) For all* $e \in E(I)$, $\texttt{lev}_I(e) \in [3/16, 13/16]$.

and outputs a set $Z \subseteq E(I)$*. Let* $I_0 = I$ *and for each* $i > 0$*, obtain* I_i *by picking a uniformly random edge* $f_{i-1} \in Z$*, arbitrarily letting* $I_i \leftarrow I_{i-1} \setminus f_{i-1}$ *or* $I_i \leftarrow I_{i-1}/f_{i-1}$*, and letting* $Z \leftarrow Z \setminus \{f_{i-1}\}$*.* $\mathcal{O}$ *satisfies the following guarantees with high probability for all* $i < K(|E(I)|)$*:*

- *(Size of* Z*)* $|Z| \geq |E(I)|/\rho$
- *(Leverage score stability)* $\texttt{lev}_{I_i}(f_i) \in [1/8, 7/8]$
- *(Martingale change stability)* $\max_{x \in \mathcal{S}} \frac{(x_{I_i}^T L_{I_i}^+ b_{f_i})^2}{r_{f_i}(x^T L_I^+ x)} \leq \frac{\rho dim(\mathcal{S})}{|E(I)|}$

We now state the main result of this section:

Lemma 5.3.3. *Consider a weighted graph* G*, a* d*-dimensional vector space* $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$*, and* $\epsilon \in (0, 1)$*. There is an algorithm* $\texttt{SubspaceSparsifier}(G, \mathcal{S}, \epsilon)$ *that, given access to a* $(\rho, K(z))$*-steady-oracle* $\mathcal{O}$*, computes a* $(\mathcal{S}, \epsilon)$*-spectral subspace sparsifier for* G *with*

$$O\left(\frac{\rho^2 d \log d}{\epsilon^2}\right)$$

edges in time

$$O\left((\log n)(\max_{z\leq|E(G)|} z/K(z))(\mathcal{T}_{rst}+\mathcal{T}_{\mathcal{O}}+m)\right)$$
$$\leq O\left((\log n)(\max_{z\leq|E(G)|} z/K(z))(m^{1+o(1)}+\mathcal{T}_{\mathcal{O}})\right)$$

where $\mathcal{T}_{rst}$ is the time required to generate a spanning tree of G from a distribution with total variation distance $\leq n^{-10}$ from uniform and $\mathcal{T}_{\mathcal{O}}$ is the runtime of the oracle.

The algorithm will use two simple subroutines that modify the graph by splitting edges. `Split` replaces each edge with approximate leverage score less than $1/2$ with a two-edge path and each edge with approximate leverage score greater than $1/2$ with two parallel edges. `Unsplit` reverses this split for all pairs that remain in the graph. We prove the following two results about this subroutines in the appendix:

Proposition 5.3.4. *There is a linear-time algorithm $(I,\mathcal{P})\leftarrow$ `Split`(H) that, given a graph H, produces a graph I with $V(H)\subseteq V(I)$ and a set of pairs of edges $\mathcal{P}$ with the following additional guarantees:*

- *(Electrical equivalence) For all $x\in\mathbb{R}^{V(I)}$ that are supported on $V(H)$, $x^TL_I^+x = x_H^TL_H^+x_H$.*
- *(Bounded leverage scores) For all $e\in E(I)$, $\texttt{lev}_I(e)\in[3/16,13/16]$*
- *($\mathcal{P}$ description) Every edge in I is in exactly one pair in $\mathcal{P}$. Furthermore, there is a bijection between pairs $(e_0,e_1)\in\mathcal{P}$ and edges $e\in E(H)$ for which either (a) e_0,e_1 and e have the same endpoint pair or (b) $e_0=\{u,w\}$, $e_1=\{w,v\}$, and $e=\{u,w\}$ for some degree 2 vertex w.*

Proposition 5.3.5. *There is a linear-time algorithm $H\leftarrow$ `Unsplit`$(I,\mathcal{P})$ that, given a graph I and a set of pairs $\mathcal{P}$ of edges in I, produces a minor H with $V(H)\subseteq V(I)$ and the following additional guarantees:*

- *(Electrical equivalence) For all $x\in\mathbb{R}^{V(I)}$ that are supported on $V(H)$, $x^TL_I^+x = x_H^TL_H^+x_H$.*
- *(Edges of H) There is a surjective map $\phi: E(I)\to E(H)$ from non-self-loop,non-leaf edges of I such that for any pair $(e_0,e_1)\in\mathcal{P}$, $\phi(e_0)=\phi(e_1)$. Furthermore, for each $e\in E(H)$, either (a) $\phi^{-1}(e)=e$, (b) $\phi^{-1}(e)=\{e_0,e_1\}$, with $(e_0,e_1)\in\mathcal{P}$ and e_0,e_1 having the same endpoints as e or (c) $\phi^{-1}(e)=\{e_0,e_1\}$, with $(e_0,e_1)\in\mathcal{P}$ and $e_0=\{u,w\},e_1=\{w,v\}$, and $e=\{u,v\}$ for a degree 2 vertex w.*

Algorithm 25: `SubspaceSparsifier`$(G, \mathcal{S}, \epsilon)$

Input: A weighted graph G, a vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$, $\epsilon \in (0,1)$, and (implicitly) a $(\rho, K(z))$-steady oracle $\mathcal{O}$

Output: A $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier for G

$H \leftarrow G$
while $|E(H)| \geq 10000\rho^2(dim(\mathcal{S})\log(dim(\mathcal{S})))/\epsilon^2$ **do**
  $(I, \mathcal{P}) \leftarrow$ `Split`(H)
  $Z \leftarrow \mathcal{O}(I, \mathcal{S})$
  $Z' \leftarrow$ a uniformly random subset of Z with size $K(|E(I)|)$
  $T \leftarrow$ a spanning tree of I drawn from a distribution with TV distance $\leq \kappa_0 := 1/n^{10}$ from uniform
  $I' \leftarrow$ the graph with all edges in $E(T) \cap Z'$ contracted and all edges in $Z' \setminus E(T)$ deleted
  $H \leftarrow$ `Unsplit`$(I', \mathcal{P})$
end
return H

We analyze the approximation guarantees of H by setting up two families of matrix-valued martingales. In all of the proof besides the final "Proof of Lemma 5.3.3," we sample T from the uniform distribution rather than from a distribution with total variation distance κ_0 from uniform. We bound the error incurred from doing this in the final "Proof of Lemma 5.3.3."

We start by defining the first family, which just consists of one martingale. Let $H_0 := G$ and let H_k be the graph H between iterations k and $k+1$ of the while loop of `SubspaceSparsifier`. Let $d = \dim(\mathcal{S})$. Since $\mathcal{S}$ is orthogonal to $\mathbf{1}^{V(G)}$, $\dim((L_G^+)^{1/2}\mathcal{S}) = \dim(\mathcal{S}) = d$, which means that $\mathcal{S}$ has a basis $\{y_i\}_{i=1}^d$ for which $y_i^T L_G^+ y_j = 0$ for all $i \neq j \in [d]$ and $y_i^T L_G^+ y_i = 1$ for all $i \in [d]$. Let Y_k be the $|V(H_k)| \times d$ matrix with ith column $(y_i)_{H_k}$ and let $Y := Y_0$. Let $M_k := Y_k^T L_{H_k}^+ Y_k - Y^T L_G^+ Y$. Since the y_is form a basis of $\mathcal{S}$, there is a vector a_x for which $x = Ya_x$ for any $x \in \mathcal{S}$. Furthermore, $x_{H_k} = Y_k a_x$ for any $k \geq 0$. In particular,

$$\left| \frac{x_{H_k}^T L_{H_k}^+ x_{H_k}}{x^T L_G^+ x} - 1 \right| = \left| \frac{a_x^T M_k a_x}{||a_x||_2^2} \right|$$

so it suffices to show that $\lambda_{\max}(M_k) \leq \epsilon$ for all $k \leq k_{\text{final}}$, where k_{final} is the number of while loop iterations.

In order to bound the change between M_k and M_{k+1}, we introduce a second family of martingales consisting of one martingale for each while loop iteration. Let $I_{k,0} := I$ during the kth iteration of the while loop in `SubspaceSparsifier`. Generate Z' in Z during iteration k of the while loop by sampling a sequence of edges $f_{k,0}, f_{k,1}, \ldots, f_{k,K(|E(I)|)-1}$ without replacement from Z. Let $I_{k,t} = I_{k,t-1}[f_{k,t-1}]$ for all $t > 0$. For a vector $v \in \mathbb{R}^{V(G)}$, let $v_{I_{k,0}} \in \mathbb{R}^{V(I_{k,0})}$ be the vector with $v_{I_{k,0}}(p) = v_{H_k}(p)$ for $p \in V(H_k)$ and $v_{I_{k,0}}(p) = 0$ for $p \in V(I_{k,0}) \setminus V(H_k)$. For $t > 0$ and $v \in \mathbb{R}^{V(G)}$, let $v_{I_{k,t}} := (v_{I_{k,0}})_{I_{k,t}}$. Let $Y_{k,t}$ be the $|V(I_{k,t})| \times d$ matrix with

ith column $(y_i)_{I_{k,t}}$. Let $N_{k,t} := Y_{k,t}^T L_{I_{k,t}}^+ Y_{k,t} - Y^T L_G^+ Y$. For any $x \in \mathcal{S}$, $t \geq 0$, and $k \geq 0$, $x_{I_{k,t}} = Y_{k,t} a_x$. In particular,

$$\left| \frac{x_{I_{k,t}}^T L_{I_{k,t}}^+ x_{I_{k,t}}}{x^T L_G^+ x} - 1 \right| = \left| \frac{a_x^T N_{k,t} a_x}{||a_x||_2^2} \right|$$

Next, we write an equivalent formulation for the steady oracle "Martingale change stability" guarantee that is easier to analyze:

Proposition 5.3.6.

$$\max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x^T L_G^+ x)} = \frac{b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f}{r_f}$$

Proof. Notice that

$$\begin{aligned}
\max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x^T L_G^+ x)} &= \max_{x \in \mathcal{S}} \frac{(a_x^T Y_{k,t}^T L_{I_{k,t}}^+ b_f)(b_f^T L_{I_{k,t}}^+ Y_{k,t} a_x)}{r_f ||a_x||_2^2} \\
&= \max_{a \in \mathbb{R}^d} \frac{a^T Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t} a}{r_f ||a||_2^2} \\
&= \lambda_{\max} \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f} \right) \\
&= \frac{b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f}{r_f}
\end{aligned}$$

as desired. □

Now, we analyze the inner family of matrices $N_{k,t}$. Let $Z_{k,t}$ denote the set Z during iteration k of the while loop after sampling t edges without replacement.

Proposition 5.3.7. *$Y_t := N_{k,t}$ for fixed $k \geq 0$ and varying $t \geq 0$ is a matrix martingale. Furthermore, if*

$$\frac{x_{I_{k,s}}^T L_{I_{k,s}}^+ x_{I_{k,s}}}{x^T L_G^+ x} \leq 10$$

for all $x \in \mathcal{S}$, $k \geq 0$, and $s \leq t$ for some $t \geq 0$, $\lambda_{\max}(X_{t+1}) \leq \frac{90d}{|E(I_{k,t})|}$ and $\lambda_{\max}(\boldsymbol{E}[X_{t+1}^2 | Y_t]) \leq \frac{25600\rho^2 d}{|E(I_{k,0})|^2}$, where X_{t+1} is defined based on the Y_ss as described in Theorem 5.3.1.

Proof. We compute the conditional expectation of $X_{t+1} = Y_{t+1} - Y_t$ given Y_t using Sherman-Morrison:

$$\begin{aligned}\mathbf{E}[X_{t+1}|Y_t] &= \mathbf{E}[N_{k,t+1} - N_{k,t}|N_{k,t}] \\ &= \frac{1}{|Z_{k,t}|}\sum_{f\in Z_{k,t}} -\frac{b_f^T L_{I_{k,t}}^+ b_f}{r_f}\left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{b_f^T L_{I_{k,t}}^+ b_f}\right) \\ &+ \frac{1}{|Z_{k,t}|}\sum_{f\in Z_{k,t}}\left(1-\frac{b_f^T L_{I_{k,t}}^+ b_f}{r_f}\right)\left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f - b_f^T L_{I_{k,t}}^+ b_f}\right) \\ &= 0\end{aligned}$$

Therefore, $(Y_t)_{t\geq 0}$ is a martingale. Since $I_{k,0}$ is the output of `Split`, all edges in $I_{k,0}$ have leverage score between 3/16 and 13/16 by Proposition 5.3.4. In particular, the input condition to $\mathcal{O}$ is satisfied. Furthermore,

$$\begin{aligned}\lambda_{\max}(X_{t+1}) &\leq \lambda_{\max}(N_{k,t+1} - N_{k,t}) \\ &\leq \frac{1}{|Z_{k,t}|}\sum_{f\in Z_{k,t}} \lambda_{\max}\left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{\min(b_f^T L_{I_{k,t}}^+ b_f, r_f - b_f^T L_{I_{k,t}}^+ b_f)}\right) \\ &\leq 8\max_{f\in Z_{k,t}} \lambda_{\max}\left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f}\right) \\ &= 8\max_{f\in Z_{k,t}}\max_{x\in\mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f(x^T L_G^+ x)} \\ &\leq 80\max_{f\in Z_{k,t}}\max_{x\in\mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f(x_{I_{k,0}}^T L_{I_{k,0}}^+ x_{I_{k,0}})} \\ &\leq \frac{90\rho d}{|E(I_{k,0})|}\end{aligned}$$

where the third inequality follows from "Leverage score stability," the equality follows from Proposition 5.3.6, the fourth inequality follows from the input condition, and the last inequality follows from "Martingale change stability." Also,

$$
\begin{aligned}
&\lambda_{\max}(\mathbf{E}[X_{t+1}^2|Y_t]) \\
&= \lambda_{\max}\left(\mathbf{E}[(N_{k,t+1}-N_{k,t})^2|Y_t]\right) \\
&\le \frac{256}{|Z_{k,t}|}\lambda_{\max}\left(\sum_{f\in Z_{k,t}}\frac{1}{r_f^2}Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}\right) \\
&\le \frac{256d}{|Z_{k,t}|}\left(\max_{f\in Z_{k,t}}\frac{1}{r_f}b_f^T L_{I_{k,t}}^+ Y_{k,t}Y_{k,t}^T L_{I_{k,t}}^+ b_f\right)\lambda_{\max}\left(\sum_{f\in Z_{k,t}}\frac{1}{r_f}Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}\right) \\
&\le \frac{2560\rho d}{|Z_{k,t}||E(I_{k,0})|}\lambda_{\max}\left(Y_{k,t}^T L_{I_{k,t}}^+ Y_{k,t}\right) \\
&= \frac{2560\rho d}{|Z_{k,t}||E(I_{k,0})|}\max_{x\in\mathcal{S}}\frac{x_{I_{k,t}}^T L_{I_{k,t}}^+ x_{I_{k,t}}}{x^T L_G^+ x} \\
&\le \frac{25600\rho^2 d}{|E(I_{k,0})|^2}
\end{aligned}
$$

where the second inequality follows from Sherman-Morrison and "Leverage score stability," the fourth follows from "Martingale change stability," and the last follows from "Size of Z" and the input condition. □

Now, consider the sequence of matrices $((N_{k,t})_{t=0}^{K(|E(I_{k,t})|)})_{k\ge 0}$ obtained by concatenating the $(N_{k,t})_t$ martingales for each k. We now analyze this sequence:

Proposition 5.3.8. *The sequence of matrices $(Y_{kt})_{k,t}$ ordered lexicographically by (k,t) pairs defined by $Y_{kt} := N_{k,t}$ is a matrix martingale. Furthermore, if for any $k \ge 0, t \ge 0$, any pairs (l,s) lexicographically smaller than (k,t), and any $x \in \mathcal{S}$,*

$$\frac{x_{I_{l,s}}^T L_{I_{l,s}}^+ x_{I_{l,s}}}{x^T L_G^+ x} \le 10$$

then

$$\lambda_{\max}(X_{k't'}) \le \frac{90d}{|E(I_{k',t'})|}$$

, $\lambda_{\max}(\boldsymbol{E}[X_{k't'}^2|Y_{kt}]) \le \frac{25600\rho^2 d}{|E(I_{k',t'})|^2}$, and

$$\lambda_{\max}(W_{kt}) \le \sum_{(l,s)\le(k,t)}\frac{25600\rho^2 d}{|E(I_{l,s})|^2}$$

where (k',t') is the lexicographic successor to (k,t) and $X_{k't'} = Y_{k't'} - Y_{kt}$ as described in Theorem 5.3.1.

Proof. Consider a pair (k', t') with lexicographic predecessor (k, t). If $k = k'$, then $t' = t + 1$, which means that

$$\mathbf{E}[Y_{k't'}|Y_{kt}] = \mathbf{E}[N_{k,t+1}|N_{k,t}] = N_{k,t} = Y_{kt}$$

, $\lambda_{\max}(X_{k't'}) \leq \frac{90d}{|E(I_{k',t'})|}$, and $\lambda_{\max}(\mathbf{E}[X_{k't'}^2|Y_{kt}]) \leq \frac{25600\rho^2 d}{|E(I_{k',t'})|^2}$ by Proposition 5.3.7. If $k = k' - 1$, then $t' = 0$. As a result, $Y_{kt} = N_{k,t} = M_k$ by the "Electrical equivalence" guarantee of Proposition 5.3.5 and $M_k = N_{k',t'} = Y_{k't'}$ by the "Electrical equivalence" guarantee of Proposition 5.3.4. In particular, $X_{k't'} = 0$ and satisfies the desired eigenvalue bounds. The bound on $\lambda_{\max}(W_{kt})$ follows directly from the stepwise bound $\lambda_{\max}(\mathbf{E}[X_{k't'}^2|Y_{kt}])$ and the definition of W_{kt} as the predictable quadratic variation. □

Now, we are ready to prove Lemma 5.3.3.

Proof of Lemma 5.3.3. **H a minor of G.** It suffices to show that for every $k \geq 0$, H_{k+1} is a minor of H_k. I' is a minor of I, as I is only modified by deletion and contraction. Now, we show that the unweighted version of H_{k+1} can be obtained from H_k by contracting each edge $e \in E(H_k)$ with an I'-self-loop in its pair $(e_0, e_1) \in P$ and deleting each edge $e \in E(H_k)$ with an I'-leaf edge in its pair. Let H'_{k+1} be the result of this procedure.

We show that $H'_{k+1} = H_{k+1}$ without weights. We start by showing that $V(H_{k+1}) = V(H'_{k+1})$. Each vertex $v \in V(H_{k+1})$ corresponds to a set of vertices in $V(H_k)$ that were identified, as the "Edges of H" requirement ensures that H_{k+1} contains no vertices that were added to H_k by `Split`. Since T is a tree, each vertex $v \in V(H_{k+1})$ corresponds to a subtree of identified vertices in H_k. Since Z only contains one edge for each pair in $\mathcal{P}$, the self-loop edges in I' match the edges contracted to form the subtree for v, which means that $V(H_{k+1}) = V(H'_{k+1})$. $E(H_{k+1}) \subseteq E(H'_{k+1})$ because for every $e \in E(H_{k+1})$, $\phi^{-1}(e)$ does not contain an I' self-loop or leaf by the "Edges of H" and "$\mathcal{P}$ description" guarantees. $E(H'_{k+1}) \subseteq E(H_{k+1})$ because each $e \in E(H'_{k+1})$ does not map to a self-loop or leaf in I', which means that $\phi^{-1}(e)$ exists by surjectivity of ϕ. Therefore, $H'_{k+1} = H_{k+1}$. Since H'_{k+1} is a minor of H_k, H_{k+1} is also a minor of H_k, as desired.

Number of edges. This follows immediately from the while loop termination condition.

Approximation bound. Let (k_τ, t_τ) be the final martingale index pair that the while loop encounters before termination. We start by obtaining a high-probability bound on $W_{k_\tau t_\tau}$ given that T is drawn from the exact uniform distribution on spanning trees of I. By Proposition 5.3.8,

$$W_{k_\tau t_\tau} \leq \sum_{(k,t) \leq (k_\tau, t_\tau)} \frac{25600\rho^2 d}{|E(I_{k,t})|^2}$$

The process of generating $I_{k,t+1}$ from $I_{k,t}$ does not increase the number of edges and decreases the number of edges by 1 with probability at least $1/8$, by "Leverage score stability." Therefore, by Azuma's Inequality, $|E(I_{k,t})| \leq 2|E(G)| - c_{k,t}/8 + 10\sqrt{\log d}\sqrt{c_{k,t}}$ with probability at least $1 - 1/d^5$, where $c_{k,t}$ is the number of pairs that are lexicographically less than (k, t). Therefore, as long as $|E(G)| > 20 \log d$, which is true when $d > 10000000 = \Theta(1)$,

$$|E(I_{k,t})| \leq 2|E(G)| - c_{k,t}/16$$

with probability at least $1 - 1/d^3$ for all pairs (k, t). This means that

$$c_{k_\tau, t_\tau} \leq 32000000|E(G)|$$

and that

$$W_{k_\tau, t_\tau} \leq \frac{32000000000\rho^2 d}{|E(I_{k_\tau, t_\tau})|} \leq \epsilon^2/(10 \log d)$$

with probability at least $1 - 1/d^3$.

Now, we apply the Matrix Freedman Inequality (Theorem 5.3.1). Apply it to the martingale $(Y_{kt})_{k,t}$ to bound $\lambda_{\max}(Y_{k_\tau t_\tau} - Y_{00})$. By Proposition 5.3.8 and the termination condition for the while loop, we may set $R \leftarrow \epsilon/(10 \log d) \geq \frac{90d}{|E(I_{k_\tau, t_\tau})|}$. By Theorem 5.3.1,

$$\begin{aligned}\Pr\left[\lambda_{\max}(Y_{k_\tau t_\tau}) \geq \epsilon \text{ and } \lambda_{\max}(W_{k_\tau t_\tau}) \leq \epsilon^2/(10 \log d)\right] &\leq d \exp\left(\frac{-\epsilon^2/2}{\epsilon^2/(10 \log d) + \epsilon^2/(30 \log d)}\right) \\ &\leq 1/d^2\end{aligned}$$

Therefore,

$$\Pr_{T \text{ uniform}}[\lambda_{\max}(Y_{k_\tau t_\tau}) \geq \epsilon] \leq 1/d^2 + 1/d^3 \leq 2/d^2$$

Now, switch uniform spanning tree sampling to κ_0-approximate random spanning tree sampling. The total number of iterations is at most m, so the total TV distance of the joint distribution sampled throughout all iterations is at most $m\kappa_0$. Therefore,

$$\Pr_{T\ \kappa_0\text{-uniform}}[\lambda_{\max}(Y_{k_\tau t_\tau})] \leq 2/d^2 + m\kappa_0 \leq 3/d^2$$

In particular, with probability at least $1 - 3/d^2$,

$$\left|\frac{x_{H_{k_\tau}} L^+_{H_{k_\tau}} x_{H_{k_\tau}}}{x^T L_G^+ x} - 1\right| \leq \lambda_{\max}(M_{k_\tau}) = \lambda_{\max}(Y_{k_\tau t_\tau}) \leq \epsilon$$

for all $x \in \mathcal{S}$, as desired.

Runtime. By Azuma's Inequality,

$$|E(H_k)| \leq |E(H_{k-1})| - K(|E(H_{k-1})|)/32 \leq (1 - \min_{z \geq 0} K(z)/32z)|E(H_{k-1})|$$

for all $k \leq k_\tau$ with probability at least $1 - 1/d^2$. Therefore,

$$|E(H_k)| \leq (1 - \min_{z \geq 0} K(z)/(32z))^k |E(G)|$$

which means that the termination condition is satisfied with high probability after

$$O((\log n) \max_{z \leq |E(G)|} z/K(z))$$

iterations with high probability. Each iteration samples one spanning tree, calls the oracle once, and does a linear amount of additional work, yielding the desired runtime. □

5.3.1 Slow oracle and proof of existence

In this section, we prove Theorem 5.1.2 by exhibiting a $(2,1)$-steady oracle `MultiDimSlowOracle`$(I, \mathcal{S})$. The oracle just returns all edges in the bottom half by maximum energy fraction:

Algorithm 26: `MultiDimSlowOracle`, never executed

Input: A graph I and a subspace $\mathcal{S} \subseteq \mathbb{R}^{V(I)}$
Output: A set Z of edges satisfying the steady oracle definition
1 **return** *all edges* $e \in E(I)$ *with* $\max_{x \in \mathcal{S}} \frac{(x^T L_I^+ b_e)^2}{r_e (x^T L_I^+ x)} \leq \frac{2 dim(\mathcal{S})}{|E(I)|}$

To lower bound the number of edges added to Z, we use the following result and Markov's Inequality:

Proposition 5.3.9.

$$\sum_{f \in E(I)} \max_{x \in \mathcal{S}} \frac{(x^T L_I^+ b_f)^2}{r_f (x^T L_I^+ x)} = dim(\mathcal{S})$$

Proof. Let Y_I be a $V(I) \times \dim(\mathcal{S})$-matrix consisting of a basis $(y_i)_{i=1}^d$ for $\mathcal{S}$ with $y_i^T L_I^+ y_j = 0$ for all $i \neq j \in [\dim(\mathcal{S})]$ and $y_i^T L_I^+ y_i = 1$ for all $i \in [\dim(\mathcal{S})]$. By Proposition 5.3.6,

$$\begin{aligned}
\sum_{f \in E(I)} \max_{x \in \mathcal{S}} \frac{(x^T L_I^+ b_f)^2}{r_f (x^T L_I^+ x)} &= \sum_{f \in E(I)} \frac{b_f^T L_I^+ Y_I Y_I^T L_I^+ b_f}{r_f} \\
&= \sum_{f \in E(I)} \text{trace}\left(\frac{b_f^T L_I^+ Y Y^T L_I^+ b_f}{r_f} \right) \\
&= \sum_{f \in E(I)} \text{trace}\left(\frac{L_I^+ Y_I Y_I^T L_I^+ b_f b_f^T}{r_f} \right) \\
&= \text{trace}(L_I^+ Y_I Y_I^T) \\
&= \sum_{i=1}^d y_i^T L_I^+ y_i \\
&= \dim(\mathcal{S})
\end{aligned}$$

as desired. □

Now, we prove Theorem 5.1.2:

Proof of Theorem 5.1.2. By Lemma 5.3.3, it suffices to show that `MultiDimSlowOracle` is a $(2,1)$-steady oracle.

Size of Z. By Markov's Inequality and Proposition 5.3.9, $|Z| \geq |E(I)|/2$.

Leverage score stability. We are only interested in $i = 0$, for which the "Leverage score" input condition immediately implies the "Leverage score stability" guarantee.

Martingale change stability. We are only interested in $i = 0$. The return statement specifies the "Martingale change stability" guarantee for $\rho = 2$. □

5.4 Fast oracle

In this section, we give a $(O(\log^3 n), \Omega(z/\log^3 n))$-steady oracle `MultiDimFastOracle` that proves Theorem 5.1.3 when plugged into `SubspaceSparsifier`. To do this, we use localization (Chapter 3) to find a set of edges whose leverage scores and martingale changes do not change much over time. We use sketching and Lemma 5.1.4 to find these edges efficiently. This section can be described using the *flexible function* framework given in Chapter 4, but we give a self-contained treatment here.

5.4.1 Efficient identification of low-change edges

`MultiDimFastOracle` needs to find a large collection of edges whose electrical energies do not change over the course of many iterations. This collection exists by the following result:

Theorem 5.4.1 (Restatement of Theorem 3.3.1)**.** *Let I be a graph. Then for any vector $w \in \mathbb{R}^{E(I)}$,*

$$\sum_{e,f \in E(I)} w_e w_f \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq c_{local}(\log^2 n)||w||_2^2$$

for some constant c_{local}.

Plugging in $w \leftarrow \mathbf{1}^{E(I)}$ shows that at least half of the edges $e \in E(I)$,

$$\sum_{f \in E(I)} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 2c_{\text{local}} \log^2 n$$

We decrease this bound by subsampling the edges in I to obtain Z. To identify the edges with low sum, we use matrix sketching:

Theorem 5.4.2 (Theorem 3 of [44] stated for ℓ_1)**.** *An efficiently computable, polylog(d)-space linear sketch exists for ℓ_1 norms. That is, given a $d \in \mathbb{Z}_{\geq 1}$, $\delta \in (0,1)$, and $\epsilon \in (0,1)$, there is a matrix $C = \texttt{SketchMatrix}(d, \delta, \epsilon) \in \mathbb{R}^{l \times d}$ and an algorithm $\texttt{RecoverNorm}(s, d, \delta, \epsilon)$ with the following properties:*

- *(Approximation) For any vector $v \in \mathbb{R}^d$, with probability at least $1 - \delta$ over the randomness of `SketchMatrix`, the value $r = \texttt{RecoverNorm}(Cv, d, \delta, \epsilon)$ is as follows:*

$$(1 - \epsilon)||v||_1 \leq r \leq (1 + \epsilon)||v||_1$$

- $l = c/\epsilon^2 \log(1/\delta)$
- *(Runtime)* `SketchMatrix` *and* `RecoverNorm` *take* $O(ld)$ *and* $\text{poly}(l)$ *time respectively.*

Approximation of column norms

Consider a graph I and a set $W \subseteq E(I)$. We can obtain multiplicative approximations the quantities $\sum_{f \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ for all $e \in W$ in near-linear time using Theorem 5.4.2. However, we actually need to multiplicatively approximate the quantities $\sum_{f \in W, f \neq e} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$. In particular, we need to estimate the ℓ_1 norm of the rows of the matrix M with $M_{ef} := \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ with the diagonal left out. To do this, we tile the matrix as described in the proof of Proposition 4.11.8 (self-contained treatment below):

- Do $\Theta(\log n)$ times:
  - Pick a random balanced partition (W_0, W_1) of W
  - For each $e \in W_0$, approximate $a_e \leftarrow \sum_{f \in Z_1} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ using sketching
- For each $e \in W$, average the a_es together and scale up the average by a factor of 4 to obtain an estimate for $\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$

The expected contribution of each off-diagonal entry is 1, while no diagonal entry can contribute. After $\Theta(\log n)$ trials, the averages concentrate by Chernoff and a union bound. Now, we formally implement this idea:

Proposition 5.4.3. *There is a near-linear time algorithm* $(a_e)_{e \in W} \leftarrow$ `ColumnApx`(I, W) *that takes a graph* I *and a set of edges* $W \subseteq E(I)$ *and returns estimates* a_e *for which*

$$a_e/2 \leq \sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 3a_e/2$$

for all $e \in W$.

Algorithm 27: `ColumnApx`(I, W)

Input: a graph I and $W \subseteq E(I)$

Output: approximations to the values $\{\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\}_{e \in W}$

```
K ← 100 log n
κ_e ← 0 for each e ∈ W
foreach k ← 1, 2, ..., K do
    W_0, W_1 ← partition of W with e ∈ W_0 or e ∈ W_1 i.i.d. with probability 1/2
    C ← SketchMatrix(|W_1|, 1/n^6, 1/4)
    D ← V(I) × |W_1| matrix of columns b_f/√r_f for f ∈ W_1
    U ← L_I^+ D C^T
    foreach e ∈ W_0 do
        Increase κ_e by RecoverNorm(U^T(b_e/√r_e), |W_1|, 1/n^6, 1/4)
    end
end
return (4κ_e/K)_{e∈W}
```

Proof of Proposition 5.4.3. **Approximation.** Let $Y_{ef}^{(k)}$ be the indicator variable of the event $\{e \in W_0 \text{ and } f \in W_1 \text{ in iteration } k\}$. By the "Approximation" guarantee of Theorem 5.4.2, at the end of the foreach loop in `ColumnApx`,

$$\kappa_e \in [3/4, 5/4] \left(\sum_{f \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \right)$$

for each $e \in W$. Since $Y_{ee}^{(k)} = 0$ for all k and $e \in W$,

$$\begin{aligned} &\sum_{f \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \\ &= \sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \left(\sum_{k=1}^{K} Y_{ef}^{(k)} \right) \end{aligned}$$

Notice that for $e \neq f$, $\{Y_{ef}^{(k)}\}_k$ is a family of independent Bernoullis with mean $1/4$. Therefore, by Chernoff bounds and our choice of K, $K(1/4)(7/8) \leq \sum_{k=1}^{K} Y_{ef}^{(k)} \leq K(1/4)(9/8)$ for all $e \neq f$ with probability at least $1 - 1/n^5$. As a result,

$$\kappa_e \in \frac{K}{4}[1/2, 3/2] \left(\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \right)$$

with high probability, as desired.

Runtime. Lines 5 and 9 contribute at most $\tilde{O}(|E(I)|)$ to the runtime of `ColumnApx` by the "Runtime" guarantee of Theorem 5.4.2. Line 7 only takes $\tilde{O}(|E(I)|)$ time to compute U because C^T only has $O(\log n)$ columns. All other lines take linear time, so `ColumnApx` takes near-linear time. □

Construction of concentrated edges

Now, we subsample localized sets:

Proposition 5.4.4. *Given a graph I and $\gamma \in (0,1)$, there is a set of edges $W \subseteq E(I)$ with two properties:*

- *(Size)* $|W| \geq (\gamma/4)|E(I)|$
- *(Value) For all* $e \in W$, $\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq \psi$ *for all* $e \in W$, *where* $\psi := 100 c_{local} \gamma (\log^2 n)$

Furthermore, there is an $\tilde{O}(|E(I)|/\gamma)$-expected time algorithm `Subsample`(I, γ) that produces W.

Algorithm 28: `Subsample`(I, γ)

```
1 while W does not satisfy Proposition 5.4.4 do
2 |   W_0 ← random subset of E(I), with each edge of e ∈ E(I) included i.i.d. with
  |     probability 2γ
3 |   (a_e)_{e∈W_0} ← ColumnApx(I, W_0)
4 |   W ← set of edges e ∈ W_0 with a_e ≤ ψ/2
5 end
6 return W
```

Proof. We show that each iteration of the while loop terminates with probability at least $1/\text{polylog}(n)$. As a result, only $\text{polylog}(n)$ iterations are required to find the desired set. We do this by setting up an intermediate family of subsets of $E(I)$ to obtain W.

Size. Let $X_1 \subseteq E(I)$ be the set of edges e with $\sum_{f \in E(I)} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 2c_{\text{local}} \log^2 n$. By Theorem 5.4.1, $|X_1| \geq |E(I)|/2$.

Let $W_1 := X_1 \cap W_0$. W_1 can alternatively be sampled by sampling W_1 from X_1, including each element of X_1 in W_1 i.i.d. with probability 2γ. Furthermore,

$$\begin{aligned}\mathbf{E}_{W_1}\left[\sum_{f\neq e\in W_0}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\Big|e\in W_1\right] &= \mathbf{E}_{W_1}\left[\sum_{f\neq e\in W_0}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\right]\\ &= 2\gamma\sum_{f\neq e\in E(I)}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\\ &\leq 4\gamma c_{\text{local}}(\log^2 n)\end{aligned}$$

By the approximation upper bound for a_e and Markov's Inequality,

$$\begin{aligned}\Pr_{W_1}[e\notin W|e\in W_1] &\leq \Pr_{W_1}[a_e>\psi/2|e\in W_1]\\ &\leq \Pr_{W_1}\left[\sum_{f\neq e\in E(I)}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}>\psi/4|e\in W_1\right]\\ &\leq \frac{4\gamma c_{\text{local}}(\log^2 n)}{\psi/4}\\ &\leq 1/2\end{aligned}$$

for every $e\in X_1$. Therefore,

$$\mathbf{E}[|W|]>(1/2)\mathbf{E}[|W_1|]=\gamma|X_1|\geq\gamma|E(I)|/2$$

Since $0\leq|W|\leq|E(I)|$, $|W|\geq\gamma|E(I)|/4$ with probability at least $\gamma/4$, as desired.

Value. By the upper bound on a_e due to Proposition 5.4.3, all edges $e\in W$ have the property that

$$\sum_{f\neq e\in W}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\leq\sum_{f\neq e\in W_0}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\leq\psi$$

as desired.

Runtime. Each iteration of the while loop succeeds with probability at least $\gamma/4$, as discussed in the "Size" analysis. Each iteration takes $\tilde{O}(|E(I)|)$ time by the runtime guarantee for `ColumnApx`. Therefore, the expected overall runtime is $\tilde{O}(|E(I)|/\gamma)$.

□

5.4.2 `MultiDimFastOracle`

We now implement the $(\Theta(\log^3 n),\Theta(z/\log^3 n))$-steady oracle `MultiDimFastOracle`. It starts by finding a set W guaranteed by Proposition 5.4.4 with $\gamma=\Theta(1/\log^3 n)$. It then

further restricts W down to the set of edges satisfying "Martingale change stability" for I_0 and returns that set. The "Value" guarantee of Proposition 5.4.4 ensures that these edges continue to satisfy the "Martingale change stability" guarantee even after conditioning on edges in Z.

Algorithm 29: `MultiDimFastOracle`$(I, \mathcal{S})$

Input: a graph I with leverage scores in $[3/16, 13/16]$ and a subspace $\mathcal{S} \subseteq V(I)$ with $\mathcal{S} := \mathbb{R}^S \times \mathbf{0}^{V(I)\setminus S}$ for some $S \subseteq V(I)$

Output: a set $Z \subseteq E(I)$ satisfying the steady oracle guarantees

1 $W \leftarrow$ `Subsample`(I, γ), where $\gamma := 1/(100000000 c_{\text{local}}(\log^3 n)))$

2 $\{\nu_e\}_{e\in E(I)} \leftarrow$ `DiffApx`$(I, S, 1/4, 1/m^5)$

3 **return** *all* $e \in W$ *for which* $\nu_e \leq \frac{4|S|}{|W|}$

To ensure that `DiffApx` is applicable, note the following equivalence to what its approximating and the quantity in the "Martingale change stability" guarantee:

Proposition 5.4.5.

$$\max_{x\in\mathbb{R}^S} \frac{(x^T L_H^+ b_f)^2}{r_f(x^T L_H^+ x)} = \frac{b_f^T L_H^+ b_f}{r_f} - \frac{b_f^T L_{H/S}^+ b_f}{r_f}$$

Proof. Define an $n \times (|S|-1)$ matrix C with signed indicator vectors of edges in a star graph on C. For every $x \in \mathbb{R}^S$ with $x^T\mathbf{1} = 0$, $x = Cc_x$ for some unique $c_x \in \mathbb{R}^{|S|-1}$. Therefore,

$$\begin{aligned}
\max_{x\in\mathbb{R}^S} \frac{(x^T L_H^+ b_f)^2}{r_f(x^T L_H^+ x)} &= \max_{c\in\mathbb{R}^{|S|-1}} \frac{c^T C^T L_H^+ b_f b_f^T L_H^+ C c}{r_f(c^T C^T L_H^+ C c)} \\
&= \frac{1}{r_f}\lambda_{\max}((C^T L_H^+ C)^{-1/2} C^T L_H^+ b_f b_f^T L_H^+ C (C^T L_H^+ C)^{-1/2}) \\
&= \frac{1}{r_f}(b_f^T L_H^+ C (C^T L_H^+ C)^{-1} C^T L_H^+ b_f) \\
&= \frac{b_f^T L_H^+ b_f}{r_f} - \frac{b_f^T L_{H/S}^+ b_f}{r_f}
\end{aligned}$$

where the last equality follows from the Woodbury formula. □

To analyze `MultiDimFastOracle`, we start by showing that any set of localized edges remains localized under random edge modifications:

Proposition 5.4.6. *Consider a graph I and a set of edges $Z \subseteq E(I)$ that satisfy the following two initial conditions:*

- *(Initial leverage scores)* $\texttt{lev}_I(e) \in [3/16, 13/16]$ *for all* $e \in Z$.

- *(Initial localization)* $\sum_{f\in Z}\frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \le \tau$ *for all* $e \in Z$*, where* $\tau = \frac{1}{10000\log n}$.

Sample a sequence of minors $\{I_k\}_{k\ge 0}$ *of* I *and sets* $Z_k \subseteq E(I_k)$ *by letting* $I_0 := I$ *and for each* $k \ge 0$*, sampling a uniformly random edge* $e_k \in Z_k$*, letting* $I_{k+1} \leftarrow I_k\backslash e_k$ *or* $I_{k+1} \leftarrow I_k/e_k$ *arbitrarily, and letting* $Z_{k+1} \leftarrow Z_k\backslash e_k$*. Then with probability at least* $1 - 1/n^2$*, the following occurs for all* i*:*

- *(All leaverage scores)* $\texttt{lev}_{I_k}(e) \in [1/8, 7/8]$ *for all* $e \in Z_k$.
- *(All localization)* $\sum_{f\in Z_k, f\ne e}\frac{|b_e^T L_{I_k}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \le \tau'$ *for all* $e \in Z_k$*, where* $\tau' = 2\tau$.

To prove this result, we cite the following submartingale inequality:

Theorem 5.4.7 (Theorem 27 of [24] with $a_i = 0$ for all i)**.** *Let* $(Y_i)_{i\ge 0}$ *be a submartingale with difference sequence* $X_i := Y_i - \boldsymbol{E}[Y_i|Y_{i-1}]$ *and* $W_i := \sum_{j=1}^{i}\boldsymbol{E}[X_j^2|Y_{j-1}]$*. Suppose that both of the following conditions hold for all* $i \ge 0$*:*

- $W_i \le \sigma^2$
- $X_i \le M$

Then

$$\Pr[Y_i - Y_0 \ge \lambda] \le \exp\left(-\frac{\lambda^2}{2(\sigma^2 + M\lambda/3)}\right)$$

Proof of Proposition 5.4.6. We prove this result by induction on k. For $k = 0$, "Initial leverage scores" and "Initial localization" imply "All leverage scores" and "All localization" respectively. For $k > 0$, we use submartingale concentration to show the inductive step. For any edge $e \in Z_k$, define two random variables $U_e^{(k)} := \texttt{lev}_{I_k}(e)$ and $V_e^{(k)} := \sum_{f\in Z_k, f\ne e}\frac{|b_e^T L_{I_k}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$. Let

$$\widehat{U}_e^{(k)} := U_e^{(k)} - \sum_{l=0}^{k-1}\mathbf{E}[U_e^{(l+1)} - U_e^{(l)}|U_e^{(l)}]$$

and

$$\widehat{V}_e^{(k)} := V_e^{(k)} - \sum_{l=0}^{k-1}\mathbf{E}\left[V_e^{(l+1)} - V_e^{(l)} + \frac{|b_e^T L_{I_l}^+ b_{e_l}|}{\sqrt{r_e}\sqrt{r_{e_l}}}\Big|V_e^{(l)}\right]$$

$(\widehat{U}_e^{(k)})_{k\ge 0}$ is a martingale and $(\widehat{V}_e^{(k)})_{k\ge 0}$ is a submartingale for all $e \in Z_k$. Let

$$\widehat{XU}_e^{(k)} := \widehat{U}_e^{(k)} - \mathbf{E}[\widehat{U}_e^{(k)}|\widehat{U}_e^{(k-1)}] = U_e^{(k)} - U_e^{(k-1)} - \mathbf{E}[U_e^{(k)} - U_e^{(k-1)}|U_e^{(k-1)}]$$

$$\widehat{XV}_e^{(k)} := \widehat{V}_e^{(k)} - \mathbf{E}[\widehat{V}_e^{(k)}|\widehat{V}_e^{(k-1)}] = V_e^{(k)} - V_e^{(k-1)} - \mathbf{E}[V_e^{(k)} - V_e^{(k-1)}|V_e^{(k-1)}]$$

$$\widehat{WU}_e^{(k)} := \sum_{j=1}^{k} \mathbf{E}[(\widehat{XU}_e^{(j)})^2 | \widehat{U}_e^{(j-1)}]$$

and

$$\widehat{WV}_e^{(k)} := \sum_{j=1}^{k} \mathbf{E}[(\widehat{XV}_e^{(j)})^2 | \widehat{V}_e^{(j-1)}]$$

By Sherman-Morrison and the inductive assumption applied to the edges $e, e_{k-1} \in Z_{k-1}$,

$$\begin{aligned}
|\widehat{XU}_e^{(k)}| &\leq |U_e^{(k)} - U_e^{(k-1)}| + \mathbf{E}[|U_e^{(k)} - U_e^{(k-1)}||U_e^{(k-1)}] \\
&\leq 2\frac{(b_e^T L_{I_{k-1}}^+ b_{e_{k-1}})^2}{r_e \min(b_{e_{k-1}}^T L_{I_{k-1}}^+ b_{e_{k-1}}, r_{e_{k-1}} - b_{e_{k-1}}^T L_{I_{k-1}}^+ b_{e_{k-1}})} \\
&\leq 16(\tau')^2
\end{aligned}$$

$$\begin{aligned}
\widehat{XV}_e^{(k)} &= \left(V_e^{(k)} - \left(V_e^{(k-1)} - \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}|}{\sqrt{r_e}\sqrt{r_{e_{k-1}}}}\right)\right) - \mathbf{E}\left[V_e^{(k)} - \left(V_e^{(k-1)} - \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}|}{\sqrt{r_e}\sqrt{r_{e_{k-1}}}}\right)\right] \\
&\leq 2\sum_{g \in Z_k, g \neq e} \left| \frac{|b_e^T L_{I_k}^+ b_g|}{\sqrt{r_e}\sqrt{r_g}} - \frac{|b_e^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_e}\sqrt{r_g}} \right| \\
&\leq 2\sum_{g \in Z_k, g \neq e} \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}||b_{e_{k-1}}^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_e} r_{e_{k-1}} \min(1 - \mathtt{lev}_{I_{k-1}}(e_{k-1}), \mathtt{lev}_{I_{k-1}}(e_{k-1}))\sqrt{r_g}} \\
&\leq 16\frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}|}{\sqrt{r_e}\sqrt{r_{e_{k-1}}}} \sum_{g \in Z_{k-1}, g \neq e} \frac{|b_{e_{k-1}}^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_g}\sqrt{r_{e_{k-1}}}} \\
&\leq 16(\tau')^2
\end{aligned}$$

$$
\begin{aligned}
&\mathbf{E}[(\widehat{XU}_e^{(k)})^2|\widehat{U}_e^{(k-1)}, e \neq e_{k-1}, \ldots, e \neq e_0] \\
&\leq 4\mathbf{E}[(U_e^{(k)} - U_e^{(k-1)})^2|U_e^{(k-1)}, e \neq e_{k-1}, \ldots, e \neq e_0] \\
&\leq 4\mathbf{E}_{e_{k-1}}\left[\frac{(b_e^T L_{I_{k-1}}^+ b_{e_{k-1}})^4}{r_e^2 r_{e_{k-1}}^2 \min(1-\mathtt{lev}_{I_{k-1}}(e_{k-1}), \mathtt{lev}_{I_{k-1}}(e_{k-1}))^2}\middle| e \neq e_{k-1}\right] \\
&\leq 256\mathbf{E}_{e_{k-1}}\left[\frac{(b_e^T L_{I_{k-1}}^+ b_{e_{k-1}})^4}{r_e^2 r_{e_{k-1}}^2}\middle| e \neq e_{k-1}\right] \\
&\leq \frac{256}{|Z_{k-1}|-1}\left(\sum_{f\in Z_{k-1}, f\neq e}\frac{|b_e^T L_{I_{k-1}}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}\right)^4 \\
&\leq \frac{256(\tau')^4}{|Z_{k-1}|-1}
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{E}[(\widehat{XV}_e^{(k)})^2|\widehat{V}_e^{(k-1)}, e \neq e_{k-1}, \ldots, e \neq e_0] \\
&\leq 4\mathbf{E}\left[\left(V_e^{(k)} - \left(V_e^{(k-1)} - \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}|}{\sqrt{r_e}\sqrt{r_{e_{k-1}}}}\right)\right)^2\middle| V_e^{(k-1)}, e \neq e_{k-1}, \ldots, e \neq e_0\right] \\
&\leq 4\mathbf{E}_{e_{k-1}}\left[\left(\sum_{g\in Z_k, g\neq e}\frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}||b_{e_{k-1}}^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_e} r_{e_{k-1}} \min(1-\mathtt{lev}_{I_{k-1}}(e_{k-1}), \mathtt{lev}_{I_{k-1}}(e_{k-1}))\sqrt{r_g}}\right)^2\middle| e \neq e_{k-1}\right] \\
&\leq 256\mathbf{E}_{e_{k-1}}\left[\left(\frac{(b_e^T L_{I_{k-1}}^+ b_{e_{k-1}})^2}{r_e r_{e_{k-1}}}\right)\left(\sum_{g\in Z_k, g\neq e}\frac{|b_{e_{k-1}}^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_{e_{k-1}}}\sqrt{r_g}}\right)^2\middle| e \neq e_{k-1}\right] \\
&\leq \frac{256}{|Z_{k-1}|-1}\max_{f\in Z_{k-1}}\left(\sum_{g\in Z_{k-1}, g\neq f}\frac{|b_f^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_f}\sqrt{r_g}}\right)^4 \\
&\leq \frac{256(\tau')^4}{|Z_{k-1}|-1}
\end{aligned}
$$

Therefore, for all $k \leq |Z|/2$, $|\widehat{WU}_e^{(k)}| \leq 256(\tau')^4$ and $|\widehat{WV}_e^{(k)}| \leq 256(\tau')^4$ given the inductive hypothesis. By Theorem 5.4.7,

$$
\Pr[|\widehat{U}_k - \widehat{U}_0| > 2000(\log n)(\tau')^2] \leq \exp\left(-\frac{(2000(\log n)(\tau')^2)^2}{512(\tau')^4 + 512(\tau')^2(2000(\log n)(\tau')^2/3)}\right) \leq \frac{1}{n^5}
$$

and

$$\Pr[\widehat{V}_k - \widehat{V}_0 > 2000(\log n)(\tau')^2] \leq \exp\left(-\frac{(2000(\log n)(\tau')^2)^2}{512(\tau')^4 + 512(\tau')^2(2000(\log n)(\tau')^2/3)}\right) \leq \frac{1}{n^5}$$

Now, we bound $U_k - \widehat{U}_k$ and $V_k - \widehat{V}_k$. By Sherman-Morrison and the inductive assumption for Z_{k-1},

$$\begin{aligned}
&\mathbf{E}[|U_e^{(k)} - U_e^{(k-1)}||U_e^{(k-1)}, e \neq e_{k-1}] \\
&\leq \mathbf{E}_{e_{k-1}}\left[\frac{(b_e^T L_{I_{k-1}}^+ b_{e_{k-1}})^2}{r_e \min(1 - \mathtt{lev}_{I_{k-1}}(e_{k-1}), \mathtt{lev}_{I_{k-1}}(e_{k-1}))r_{e_{k-1}}}\middle| e \neq e_{k-1}\right] \\
&\leq \frac{8(\tau')^2}{|Z_{k-1}| - 1}
\end{aligned}$$

and

$$\begin{aligned}
&\mathbf{E}\left[V_e^{(k)} - V_e^{(k-1)} + \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}|}{\sqrt{r_e}\sqrt{r_{e_{k-1}}}}\middle| V_e^{(k-1)}, e \neq e_{k-1}\right] \\
&\leq \mathbf{E}_{e_{k-1}}\left[\sum_{g \in Z_k, g \neq e} \frac{|b_e^T L_{I_{k-1}}^+ b_{e_{k-1}}||b_{e_{k-1}}^T L_{I_{k-1}}^+ b_g|}{\sqrt{r_e} r_{e_{k-1}} \min(1 - \mathtt{lev}_{I_{k-1}}(e_{k-1}), \mathtt{lev}_{I_{k-1}}(e_{k-1}))\sqrt{r_g}}\middle| e \neq e_{k-1}\right] \\
&\leq \frac{8(\tau')^2}{|Z_{k-1}| - 1}
\end{aligned}$$

so for $k \leq |Z|/2$, $|U_k - \widehat{U}_k| \leq 8(\tau')^2$ and $V_k - \widehat{V}_k \leq 8(\tau')^2$. In particular, with probability at least $1 - 2/n^5$,

$$\begin{aligned}
|\mathtt{lev}_{I_k}(e) - \mathtt{lev}_{I_0}(e)| &= |U_e^{(k)} - U_e^{(0)}| \\
&\leq |U_e^{(k)} - \widehat{U}_e^{(k)}| + |\widehat{U}_e^{(k)} - \widehat{U}_e^{(0)}| + |\widehat{U}_e^{(0)} - U_e^{(0)}| \\
&\leq 8(\tau')^2 + 2000(\log n)(\tau')^2 + 0 \\
&\leq 1/16
\end{aligned}$$

Therefore, $\mathtt{lev}_{I_k}(e) \in [1/8, 7/8]$ with probability at least $1 - 2/n^5$ for all $e \in Z_k$. Furthermore,

$$\begin{aligned}\sum_{g\in Z_k, g\neq e}\frac{|b_e^T L_{I_k}^+ b_g|}{\sqrt{r_e}\sqrt{r_g}} &= V_e^{(k)}\\ &= (V_e^{(k)} - \widehat{V}_e^{(k)}) + (\widehat{V}_e^{(k)} - \widehat{V}_e^{(0)}) + (\widehat{V}_e^{(0)} - V_e^{(0)}) + V_e^{(0)}\\ &\leq 8(\tau')^2 + 2000(\log n)(\tau')^2 + 0 + \tau\\ &\leq 2\tau = \tau'\end{aligned}$$

This completes the inductive step and the proof of the proposition.

□

Now, we prove Theorem 5.1.3. By Lemma 5.3.3, it suffices to show that `MultiDimFastOracle` is a $(O(\log^3 n), \Omega(z/\log^3 n)))$-steady oracle with runtime $\tilde{O}(|E(I)|)$.

Proof of Theorem 5.1.3. **Size of Z.** By Proposition 5.3.9, Proposition 5.4.5, and the approximation upper bound for ν_e,

$$\sum_{e\in E(I)}\nu_e \leq (5/4)|S| + 1/m^4 \leq (3/2)|S|$$

Therefore, by Markov's Inequality, $|Z| \geq 5|W|/8$. By the "Size" guarantee of Proposition 5.4.4, $|W| \geq \Omega(1/(\log n)^3)|E(I)|$, so $|Z| \geq \Omega(1/(\log n)^3)|E(I)|$, as desired.

Leverage score stability. We start by checking that the input conditions for Proposition 5.4.6 are satisfied for Z. The "Initial leverage scores" condition is satisfied thanks to the "Leverage scores" input guarantee for steady oracles. The "Initial localization" condition is satisfied because of the "Value" output guarantee for `Subsample` applied to W. Therefore, Proposition 5.4.6 applies. The "All leverage scores" guarantee of Proposition 5.4.6 is precisely the "Leverage score stability" guarantee of steady oracles, as desired.

Martingale change stability. Let $(y_i)_{i=1}^d$ be a basis of $\mathcal{S}$ for which $y_i L_I^+ y_j = 0$ for $i \neq j$ and $y_i L_I^+ y_i$ for all $i \in [\dim(\mathcal{S})]$. Let Y_t be a $V(I_t) \times \dim(\mathcal{S})$ matrix with columns $(y_i)_{I_t}$. By Proposition 5.3.6 applied with $G \leftarrow I$,

$$\max_{x\in\mathcal{S}}\frac{(x_{I_t}^T L_{I_t}^+ b_f)^2}{r_f(x^T L_I^+ x)} = \frac{b_f^T L_{I_t}^+ Y_t Y_t^T L_{I_t}^+ b_f}{r_f}$$

for all $f \in E(I_t)$. We now bound this quantity over the course of deletions and contractions of the edges e_i by setting up a martingale. For all $t \geq 0$ and $f \in E(I)$, let

$$A_f^{(t)} := \frac{b_f^T L_{I_t}^+ Y_t Y_t^T L_{I_t}^+ b_f}{r_f}$$

and

$$\widehat{A}_f^{(t)} := A_f^{(t)} - \sum_{s=0}^{t-1} [A_f^{(s+1)} - A_f^{(s)} | A_f^{(s)}]$$

For each $f \in E(I)$, $(A_f^{(t)})_{t \geq 0}$ is a martingale. Let

$$\widehat{XA}_f^{(t)} := \widehat{A}_f^{(t)} - \widehat{A}_f^{(t-1)} = A_f^{(t)} - A_f^{(t-1)} - \mathbf{E}[A_f^{(t)} - A_f^{(t-1)} | A_f^{(t-1)}]$$

and

$$\widehat{WA}_f^{(t)} := \sum_{s=1}^{t} \mathbf{E}[(\widehat{XA}_f^{(s)})^2 | A_f^{(s-1)}]$$

We now inductively show that for all $f \in Z_t$ (which includes f_t),

$$A_f^{(t)} \leq \frac{\xi' |S|}{|E(I)|}$$

where $\xi := 8\frac{E(I)}{|W|} \leq O(\log^3 n)$ and $\xi' := 2\xi$. Initially,

$$A_f^{(0)} \leq \frac{8|S|}{|W|} = \frac{\xi |S|}{|W|}$$

for all $f \in Z$ by the approximation lower bound for ν_e, completing the base case. For $t > 0$, we bound $A_f^{(t)}$ for $f \in Z_t$ by using martingale concentration. We start by bounding differences using the "All leverage scores" guarantee of Proposition 5.4.6, Sherman-Morrison, Cauchy-Schwarz, and the inductive assumption:

$$
\begin{aligned}
|A_f^{(t)} - A_f^{(t-1)}| &= \left| \frac{b_f^T L_{I_t}^+ Y_t Y_t^T L_{I_t}^+ b_f}{r_f} - \frac{b_f^T L_{I_{t-1}}^+ Y_{t-1} Y_{t-1}^T L_{I_{t-1}}^+ b_f}{r_f} \right| \\
&\leq 2 \left| \frac{b_f^T L_{I_{t-1}}^+ b_{f_{t-1}} b_{f_{t-1}}^T L_{I_{t-1}}^+ Y_{t-1} Y_{t-1}^T L_{I_{t-1}}^+ b_f}{r_{f_{t-1}} \min(1 - \mathtt{lev}_{I_{t-1}}(f_{t-1}), \mathtt{lev}_{I_{t-1}}(f_{t-1})) r_f} \right| \\
&+ \left| \frac{b_f^T L_{I_{t-1}}^+ b_{f_{t-1}} b_{f_{t-1}}^T L_{I_{t-1}}^+ Y_{t-1} Y_{t-1}^T L_{I_{t-1}}^+ b_{f_{t-1}} b_{f_{t-1}}^T L_{I_{t-1}}^+ b_f}{r_{f_{t-1}}^2 (\min(1 - \mathtt{lev}_{I_{t-1}}(f_{t-1}), \mathtt{lev}_{I_{t-1}}(f_{t-1})))^2 r_f} \right| \\
&\leq 16 \frac{|b_f^T L_{I_{t-1}}^+ b_{f_{t-1}}| |b_{f_{t-1}}^T L_{I_{t-1}}^+ Y_{t-1} Y_{t-1}^T L_{I_{t-1}}^+ b_f|}{r_{f_{t-1}} r_f} \\
&+ 64 \frac{(b_f^T L_{I_{t-1}}^+ b_{f_{t-1}})^2 b_{f_{t-1}}^T L_{I_{t-1}}^+ Y_{t-1} Y_{t-1}^T L_{I_{t-1}}^+ b_{f_{t-1}}}{r_{f_{t-1}}^2 r_f} \\
&\leq 16 \frac{|b_f^T L_{I_{t-1}}^+ b_{f_{t-1}}|}{\sqrt{r_{f_{t-1}}} \sqrt{r_f}} \sqrt{A_{f_{t-1}}^{(t-1)}} \sqrt{A_f^{(t-1)}} \\
&+ 64 \left(\frac{|b_f^T L_{I_{t-1}}^+ b_{f_{t-1}}|}{\sqrt{r_f} \sqrt{r_{f_{t-1}}}} \right)^2 A_{f_{t-1}}^{(t-1)} \\
&\leq 80 \frac{|b_f^T L_{I_{t-1}}^+ b_{f_{t-1}}|}{\sqrt{r_{f_{t-1}}} \sqrt{r_f}} \frac{\xi' |S|}{|E(I)|}
\end{aligned}
$$

By the "All localization" guarantee of Proposition 5.4.6,

$$
\begin{aligned}
|\widehat{XA}_f^{(t)}| &\leq |A_f^{(t)} - A_f^{(t-1)}| + \mathbf{E}[|A_f^{(t)} - A_f^{(t-1)}| | A_f^{t-1}] \\
&\leq 160 \tau' \frac{\xi' |S|}{|E(I)|}
\end{aligned}
$$

and

$$
\begin{aligned}
\mathbf{E}[(\widehat{XA}_f^{(t)})^2 | \widehat{A}_f^{(t-1)}] &\leq 4 \mathbf{E}_{f_{t-1}}[(A_f^{(t)} - A_f^{(t-1)})^2 | A_f^{(t-1)}, f \neq f_{t-1}] \\
&\frac{6400 (\tau')^2}{|Z_{t-1}| - 1} \left(\frac{\xi' |S|}{|E(I)|} \right)^2
\end{aligned}
$$

Since $K(|Z|) \leq |Z|/2$, $|\widehat{WA}_f^{(t)}| \leq 6400(\tau')^2 \left(\frac{\xi'|S|}{|E(I)|}\right)^2$. Therefore, by Theorem 5.4.7 applied to the submartingales $(\widehat{A}_f^{(t)})_{t\geq 0}$ and $(-\widehat{A}_f^{(t)})_{t\geq 0}$,

$$\Pr\left[|\widehat{A}_f^{(t)} - \widehat{A}_f^{(0)}| > \frac{\xi'|S|}{5|E(I)|}\right] \leq \exp\left(-\frac{((\xi'|S|)/(5|E(I)|))^2}{(6400(\tau')^2 + 160(\tau'))((\xi'|S|)/(|E(I)|))^2}\right) \leq 1/n^5$$

Since $\widehat{A}_f^{(0)} = A_f^{(0)}$, we just need to bound $|A_f^{(t)} - \widehat{A}_f^{(t)}|$. We do this by bounding expectations of differences:

$$\begin{aligned}\mathbf{E}[|A_f^{(t)} - A_f^{(t-1)}||A_f^{(t-1)}, f \neq f_{t-1}] &\leq 80\mathbf{E}\left[\frac{|b_f^T L_{I_{t-1}}^+ b_{f_{t-1}}|}{\sqrt{r_{f_{t-1}}}\sqrt{r_f}}\frac{\xi'|S|}{|E(I)|}\Big| f \neq f_{t-1}\right] \\ &\leq \frac{80\tau'}{|Z_{t-1}| - 1}\frac{\xi'|S|}{|E(I)|}\end{aligned}$$

Therefore, $|A_f^{(t)} - \widehat{A}_f^{(t)}| \leq \sum_{s=0}^{t-1} \mathbf{E}[|A_f^{(s+1)} - A_f^{(s)}||A_f^{(s)}, f \neq f_s] \leq \frac{\xi'|S|}{5|E(I)|}$. This means that

$$A_f^{(t)} \leq |A_f^{(t)} - \widehat{A}_f^{(t)}| + |\widehat{A}_f^{(t)} - \widehat{A}_f^{(0)}| + A_f^{(0)} \leq \frac{2\xi'|S|}{5|E(I)|} + \frac{\xi|S|}{|E(I)|} \leq \frac{\xi'|S|}{|E(I)|}$$

with probability at least $1 - 1/n^5$, which completes the inductive step.

Therefore, by a union bound and the fact that $f_t \in Z_t$ for all $t \geq 0$,

$$\max_{x\in\mathcal{S}} \frac{(x_{I_t} L_{I_t}^+ b_{f_t})^2}{r_{f_t}(x^T L_I^+ x)} = A_{f_t}^{(t)} \leq \frac{\xi'|S|}{|E(I)|} \leq \frac{O(\log^3 n)|S|}{|E(I)|}$$

completing the "Martingale change stability" proof.

□

5.5 Efficient approximation of differences

In this section, we show how to approximate changes in effective resistances due to the identification of a given vertex set S, and thus prove Lemma 5.1.4. Namely, given a vertex set $S \subset V$, we need to approximate the following quantity for all edges $e \in E(G)$:

$$(b_e^T L_G^+ b_e) - (b_e^T L_{G/S}^+ b_e).$$

By a proof similar to that of Proposition 5.4.5, this quantity equals

$$\max_{x\perp 1, x\in\mathbb{R}^S} \frac{(x^T L_G^+ b_e)^2}{x^T L_G^+ x}, \tag{5.7}$$

where 1 denotes the all-one vector.

Lemma 5.5.1. *The decrease in the effective resistance of an edge $e \in E(G)$ due to the identification of a vertex set $S \subset V$ equals*

$$(b_e^T L_G^+ b_e) - (b_e^T L_{G/S}^+ b_e) = \max_{x \perp 1, x \in \mathbb{R}^S} \frac{(x^T L_G^+ b_e)^2}{x^T L_G^+ x}.$$

Proof. Let C be the $n \times (|S| - 1)$ matrix with signed indicator vectors of edges in a star graph supported on S. Then we have

$$\begin{aligned}
&(b_e^T L_G^+ b_e) - (b_e^T L_{G/S}^+ b_e) \\
=&b_e^T L_H^+ C (C^T L_H^+ C)^{-1} C^T L_H^+ b_e \qquad \text{by Woodbury} \\
=&\lambda_{\max}((C^T L_H^+ C)^{-1/2} C^T L_H^+ b_e b_e^T L_H^+ C (C^T L_H^+ C)^{-1/2}) \\
=&\max_{c \in \mathbb{R}^{|S|-1}} \frac{c^T C^T L_H^+ b_e b_e^T L_H^+ C c}{c^T C^T L_H^+ C c} \\
=&\max_{x \perp 1, x \in \mathbb{R}^S} \frac{(x^T L_G^+ b_e)^2}{x^T L_H^+ x},
\end{aligned}$$

where the last equality follows from that the columns of C form a basis of the subspace of $\mathbb{R}^S$ orthogonal to the all-ones vector. □

Let $k := |S|$, and suppose without loss of generality that S contains the first k vertices in G. We construct a basis (plus an extra vector) of the subspace of $\mathbb{R}^S$ orthogonal to the all-ones vector by letting

$$C_{n \times k} := \begin{pmatrix} I_{k \times k} - \frac{1}{k} J_{k \times k} \\ 0_{(n-k) \times k} \end{pmatrix}, \tag{5.8}$$

where I denotes the identity matrix, and J denotes the matrix whose entries are all 1. Let $P_{n \times k} := \begin{pmatrix} I_{k \times k} & 0 \end{pmatrix}^T$ be the projection matrix taking the first k coordinates, and let

$\Pi_{k\times k} := I_{k\times k} - \frac{1}{k}J_{k\times k}$. Now we can write (5.7) as

$$\begin{aligned}
&\max_{x\perp 1, x\in\mathbb{R}^S} \frac{(x^T L_G^+ b_e)^2}{x^T L_G^+ x}\\
=&\max_{c\in\mathbb{R}^k} \frac{c^T C^T L_G^+ b_e b_e^T L_G^+ C c}{c^T C^T L_G^+ C c}\\
=&\max_{c\in\mathbb{R}^k} \frac{\left(c^T\Pi_{k\times k}\right)\ C^T L_G^+ b_e b_e^T L_G^+ C\ \left(\Pi_{k\times k}c\right)}{\left(c^T\Pi_{k\times k}\right)\ C^T L_G^+ C\ \left(\Pi_{k\times k}c\right)} \qquad \text{by } C\Pi_{k\times k} = C\\
=&\max_{c\in\mathbb{R}^k} \frac{\left(c^T(C^T L_G^+ C)^{+/2}\right)\ C^T L_G^+ b_e b_e^T L_G^+ C\ \left((C^T L_G^+ C)^{+/2}c\right)}{\left(c^T(C^T L_G^+ C)^{+/2}\right)\ C^T L_G^+ C\ \left((C^T L_G^+ C)^{+/2}c\right)}\\
&\qquad \text{since } (C^T L_G^+ C)^{+/2} \text{ and } \Pi_{k\times k} \text{ have the same column space}\\
=&\lambda_{\max}((C^T L_G^+ C)^{+/2} C^T L_G^+ b_e b_e^T L_G^+ C (C^T L_G^+ C)^{+/2})\\
=&b_e^T L_G^+ C (C^T L_G^+ C)^+ C^T L_G^+ b_e\\
=&b_e^T L_G^+ C (\Pi_{k\times k} P^T L_G^+ P \Pi_{k\times k})^+ C^T L_G^+ b_e \qquad \text{by } P\Pi_{k\times k} = C\\
=&b_e^T L_G^+ C\ SC(L_G, S)\ C^T L_G^+ b_e \qquad \text{by Fact 5.2.2.}
\end{aligned} \tag{5.9}$$

To approximate (5.9), we further write it as

$$\begin{aligned}
&b_e^T L_G^+ C\ SC(L_G, S)\ C^T L_G^+ b_e\\
=&b_e^T L_G^+ C\ SC(L_G, S)(SC(L_G, S))^+ SC(L_G, S)\ C^T L_G^+ b_e\\
=&b_e^T L_G^+ C\ SC(L_G, S)\ C^T L_G^+ C\ SC(L_G, S)\ C^T L_G^+ b_e\\
=&b_e^T L_G^+ C\ SC(L_G, S)\ C^T L_G^+ (B_G^T W_G B_G) L_G^+ C\ SC(L_G, S)\ C^T L_G^+ b_e,
\end{aligned}$$

where the last equality follows from $L_G^+ = L_G^+ L_G L_G^+$ and $L_G = B_G^T W_G B_G$.

We now write the change in the effective resistance of an edge e in a square of an Euclidean norm as

$$b_e^T L_G^+ b_e - b_e^T L_{G/S}^+ b_e = \left\| W_G^{1/2} B_G L_G^+ C (SC(L_G, S)) C^T L_G^+ b_e \right\|^2.$$

We then use Johnson-Lindenstrauss Lemma to reduce dimensions. Let $Q_{k\times m}$ be a random ± 1 matrix where $k \geq 24\log n/\epsilon^2$. By Lemma 5.2.12, the following statement holds for all e with high probability:

$$\left\| W_G^{1/2} B_G L_G^+ C (SC(L_G, S)) C^T L_G^+ b_e \right\|^2 \approx_{1+\epsilon} \left\| Q W_G^{1/2} B_G L_G^+ C (SC(L_G, S)) C^T L_G^+ b_e \right\|^2. \tag{5.10}$$

To compute the matrix on the rhs, we note that C is easy to apply by applying I and J, and L_G^+ can be applied to high accuracy by Fast Laplacian Solvers. Thus, we only need to apply the Schur complement $SC(L_G, S)$ to high accuracy fast. We recall Definition 5.2.1 of Schur complements:

$$SC(L_G, S) := (L_G)_{S,S} - (L_G)_{S,T}(L_G)_{T,T}^{-1}(L_G)_{T,S},$$

where $T := V \setminus S$. Since $(L_G)_{T,T}$ is a principle submatrix of L_G, it is an SDDM matrix and hence its inverse can be applied also by Fast Laplacian Solvers to high accuracy.

5.5.1 The subroutine and proof of Lemma 5.1.4

We give the algorithm for approximating changes in effective resistances due to the identification of S as follows:

Algorithm 30: `DiffApx`$(G, S, \delta_0, \delta_1)$

Input: A weighted graph G, a set of vertices $S \subseteq V(G)$, and $\delta_0, \delta_1 \in (0, 1)$
Output: Estimates $\{\nu_e\}_{e \in E(G)}$ to differences in effective resistances in G and G/S

1 Let $Q_{k \times m}$ be a random ± 1 matrix where $k \geq 24 \log n / \delta_0^2$.

2 Compute each row of $Y_{k \times n} := QW_G^{1/2} B_G L_G^+ C(SC(L_G, S))C^T L_G^+$ by applying L_G^+ and $L_{V \setminus S, V \setminus S}^{-1}$ to accuracy

$$\epsilon = \frac{\delta_1}{48\sqrt{k} \cdot n^{8.5} \cdot w_{\max}^{2.5} w_{\min}^{-3}}.$$

3 $\nu_e \leftarrow \|Yb_e\|^2$ for all $e \in E(G)$

4 **return** $\{\nu_e\}_{e \in E(G)}$

To prove the approximation ratio for `DiffApx`, we first track the errors for applying Schur complement in the following lemma:

Lemma 5.5.2. *For any Laplacian L_G, $S \subset V(G)$, vector $b \in \mathbb{R}^n$, and $\epsilon > 0$, the following statement holds:*

$$\|x - \tilde{x}\| \leq \epsilon n^{3.5} w_{\max}^{2.5} w_{\min}^{-0.5} \|b\|,$$

where

$$\begin{aligned} x &:= \left((L_G)_{S,S} - (L_G)_{S,T}(L_G)_{T,T}^{-1}(L_G)_{T,S}\right) b, \\ \tilde{x} &:= (L_G)_{S,S} b - (L_G)_{S,T} \tilde{x}_1, \\ \tilde{x}_1 &:= \texttt{LaplSolve}((L_G)_{T,T}, (L_G)_{T,S} b, \epsilon). \end{aligned}$$

Using Lemma 5.5.2, we track the errors for computing the embedding in (5.10) as follows:

Lemma 5.5.3. *For any Laplacian L_G, $S \subset V(G)$, vector $q \in \mathbb{R}^n$ with entries ± 1, and*

$$0 < \epsilon < 1/\left(4n^6 \cdot w_{\max}^{2.5} w_{\min}^{-1.5}\right), \tag{5.11}$$

and a matrix $C_{n \times k}$ defined by

$$\begin{aligned} C_{S,1:k} &= I_{k \times k} - \frac{1}{k} J_{k \times k}, \\ C_{V \setminus S,1:k} &= 0, \end{aligned}$$

the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon \cdot 8n^8 \cdot \left(\frac{w_{\max}}{w_{\min}}\right)^{2.5},$$

where

$$\begin{aligned}
x &:= \left(q^T W_G^{1/2} B_G L_G^+ C(SC(L_G, S))C^T L_G^+\right)^T, \\
\tilde{x} &:= \texttt{LaplSolve}(L_G, C\tilde{x}_1, \epsilon), \\
\tilde{x}_1 &:= (L_G)_{S,S}(C^T\tilde{x}_2) - (L_G)_{S,T}\ \texttt{LaplSolve}((L_G)_{T,T}, (L_G)_{T,S}(C^T\tilde{x}_2), \epsilon), \\
\tilde{x}_2 &:= \texttt{LaplSolve}(L_G, B_G^T W_G^{1/2} q, \epsilon).
\end{aligned}$$

Before proving the above two lemmas, we show how they imply Lemma 5.1.4.

Proof of Lemma 5.1.4. The running time follows directly from the running time of `LaplSolve`. Let $X_{k\times n} := QW_G^{1/2}B_G L_G^+ C(SC(L_G, S))C^T L_G^+$. The multiplicative approximation follows from Johnson-Lindenstrauss Lemma. To prove the additive approximation, we write the difference between $\|Xb_e\|^2$ and $\|Yb_e\|^2$ as

$$\left|\|Xb_e\|^2 - \|Yb_e\|^2\right| = |\|Xb_e\| - \|Yb_e\|| \cdot (\|Xb_e\| + \|Yb_e\|).$$

Let u, v be the endpoints of e. We upper bound $|\|Xb_e\| - \|Yb_e\||$ by

$$\begin{aligned}
|\|Xb_e\| - \|Yb_e\|| &\leq \|(X-Y)b_e\| = \|(X-Y)(e_u - e_v)\| && \text{by triangle ineq.} \\
&\leq \|(X-Y)e_u\| + \|(X-Y)e_v\| && \text{by triangle ineq.} \\
&\leq \sqrt{2}\left(\|(X-Y)e_u\|^2 + \|(X-Y)e_v\|^2\right)^{1/2} && \text{by Cauchy-Schwarz} \\
&\leq \sqrt{2}\,\|X-Y\|_F = \sqrt{2}\left(\sum_{i=1}^{k}\left\|(X-Y)^T e_i\right\|^2\right)^{1/2} \\
&\leq \sqrt{2k}\cdot\epsilon\cdot 8n^8\cdot\left(\frac{w_{\max}}{w_{\min}}\right)^{2.5} && \text{by Lemma 5.5.3} \\
&\leq \frac{\delta_1}{3\sqrt{2}n^{1/2}w_{\min}^{-1/2}}
\end{aligned}$$

and upper bound $\|Xb_e\| + \|Yb_e\|$ by

$$\begin{aligned}
\|Xb_e\| + \|Yb_e\| &\leq 2\,\|Xb_e\| + |\|Xb_e\| - \|Yb_e\|| \\
&\leq 2\left((1+\delta_0)\left(b_e^T L_G^+ b_e - b_e^T L_{G/S}^+ b_e\right)\right)^{1/2} + |\|Xb_e\| - \|Yb_e\|| && \text{by Lemma 5.2.12} \\
&\leq 2\left((1+\delta_0)n/w_{\min}\right)^{1/2} + |\|Xb_e\| - \|Yb_e\|| && \text{upper bounding } b_e^T L_G^+ b_e \\
&\leq 3\sqrt{2}n^{1/2}w_{\min}^{-1/2} && \text{by } \delta_0 < 1
\end{aligned}$$

Combining these two upper bounds gives

$$\left|\|Xb_e\|^2 - \|Yb_e\|^2\right| \leq \delta_1,$$

which proves the additive error. □

5.5.2 Analysis of additional errors

We now prove Lemma 5.5.2 and 5.5.3.

Proof of Lemma 5.5.2. We upper bound $\|x - \tilde{x}\|$ by

$$\begin{aligned}
\|x - \tilde{x}\| =& \left\|(L_G)_{S,T}\left((L_G)_{T,T}^{-1}(L_G)_{T,S}b - \tilde{x}_1\right)\right\| \\
\leq& n w_{\max}\left\|(L_G)_{T,T}^{-1}(L_G)_{T,S}b - \tilde{x}_1\right\| \qquad \text{by (5.6)} \\
\leq& \epsilon n^{2.5} w_{\max}^{1.5} w_{\min}^{-0.5}\left\|(L_G)_{T,S}b\right\| \qquad \text{by Lemma 5.2.11} \\
\leq& \epsilon n^{3.5} w_{\max}^{2.5} w_{\min}^{-0.5}\|b\| \qquad \text{by (5.6).}
\end{aligned}$$

□

Proof of Lemma 5.5.3. We first bound the norm of vector $L_G^+ B_G^T W_G^{1/2} q$ by

$$\begin{aligned}
\left\|L_G^+ B_G^T W_G^{1/2} q\right\| \leq& \frac{n^2}{w_{\min}}\|q\| \qquad \text{by } \sigma_{\max}(L_G^+ B_G^T W_G^{1/2}) = \lambda_{\max}(L_G^+) \text{ and (5.1)} \\
=& \frac{n^{2.5}}{w_{\min}} \qquad \text{since } q\text{'s entries are } \pm 1, \qquad (5.12)
\end{aligned}$$

and upper bound the norm of vector $SC(L_G, S)C^T L_G^+ B_G^T W_G^{1/2} q$ by

$$\begin{aligned}
\left\|SC(L_G, S)C^T L_G^+ B_G^T W_G^{1/2} q\right\| \leq& n w_{\max}\left\|L_G^+ B_G^T W_G^{1/2} q\right\| \qquad \text{by (5.5)} \\
\leq& n^{3.5}\frac{w_{\max}}{w_{\min}}. \qquad (5.13)
\end{aligned}$$

The error of $\tilde{x}_2$ follows by

$$\begin{aligned}
\left\|L_G^+ B_G^T W_G^{1/2} q - \tilde{x}_2\right\| \leq& \epsilon n^{1.5}\left(\frac{w_{\max}}{w_{\min}}\right)^{1/2}\left\|L_G^+ B_G^T W_G^{1/2} q\right\| \qquad \text{by Lemma 5.2.10} \\
\leq& \epsilon n^4 w_{\max}^{1/2} w_{\min}^{-1.5} \qquad \text{by (5.12).} \qquad (5.14)
\end{aligned}$$

The norm of $\tilde{x}_2$ can be upper bounded by

$$\begin{aligned}
\|\tilde{x}_2\| \leq& \left\|L_G^+ B_G^T W_G^{1/2} q\right\| + \left\|L_G^+ B_G^T W_G^{1/2} q - \tilde{x}_2\right\| \qquad \text{by triangle inequality} \\
\leq& \frac{2n^{2.5}}{w_{\min}} \qquad \text{by (5.12) and (5.11).} \qquad (5.15)
\end{aligned}$$

The error of $\tilde{x}_1$ follows by

$$\begin{aligned}
&\left\|SC(L_G,S)C^TL_G^+B_G^TW_G^{1/2}q-\tilde{x}_1\right\| \\
\leq&\left\|SC(L_G,S)C^T\left(L_G^+B_G^TW_G^{1/2}q-\tilde{x}_2\right)\right\|+\left\|SC(L_G,S)C^T\tilde{x}_2-\tilde{x}_1\right\| \quad \text{by triangle ineq.} \\
\leq&\epsilon n^5 w_{\max}^{1.5}w_{\min}^{-1.5}+\left\|SC(L_G,S)C^T\tilde{x}_2-\tilde{x}_1\right\| \quad \text{by (5.14), (5.5) and } \sigma_{\max}(C)=1 \\
\leq&\epsilon n^5 w_{\max}^{1.5}w_{\min}^{-1.5}+\epsilon\cdot\left\|C^T\tilde{x}_2\right\|\cdot n^{3.5}\cdot w_{\max}^{2.5}w_{\min}^{-0.5} \quad \text{by Lemma 5.5.2} \\
\leq&\epsilon n^5 w_{\max}^{1.5}w_{\min}^{-1.5}+\epsilon\cdot 2n^6\cdot w_{\max}^{2.5}w_{\min}^{-1.5} \quad \text{by (5.15)} \\
\leq&\epsilon\cdot 4n^6\cdot w_{\max}^{2.5}w_{\min}^{-1.5}. \qquad (5.16)
\end{aligned}$$

The norm of $\tilde{x}_1$ can be upper bounded by

$$\begin{aligned}
&\|\tilde{x}_1\| \\
\leq&\left\|SC(L_G,S)C^TL_G^+B_G^TW_G^{1/2}q\right\|+\left\|SC(L_G,S)C^TL_G^+B_G^TW_G^{1/2}q-\tilde{x}_1\right\| \quad \text{by triangle ineq.} \\
\leq&2n^{3.5}\frac{w_{\max}}{w_{\min}} \quad \text{by (5.13) and (5.11).} \qquad (5.17)
\end{aligned}$$

Finally, the error of $\tilde{x}$ follows by

$$\begin{aligned}
&\|x-\tilde{x}\| \\
\leq&\left\|L_G^+C\left(SC(L_G,S)C^TL_G^+B_G^TW_G^1/2q-\tilde{x}_1\right)\right\|+\left\|L_G^+C\tilde{x}_1-\tilde{x}\right\| \quad \text{by triangle ineq.} \\
\leq&\epsilon\cdot 4n^8\cdot w_{\max}^{2.5}w_{\min}^{-2.5}+\left\|L_G^+C\tilde{x}_1-\tilde{x}\right\| \quad \text{by (5.16) and (5.1)} \\
\leq&\epsilon\cdot 4n^8\cdot w_{\max}^{2.5}w_{\min}^{-2.5}+\epsilon n^{1.5}w_{\max}^{0.5}w_{\min}^{-0.5}\left\|C\tilde{x}_1\right\| \quad \text{by Lemma 5.2.10} \\
\leq&\epsilon\cdot 4n^8\cdot w_{\max}^{2.5}w_{\min}^{-2.5}+2\epsilon n^5 w_{\max}^{1.5}w_{\min}^{-1.5} \quad \text{by (5.17)} \\
\leq&\epsilon\cdot 8n^8\cdot w_{\max}^{2.5}w_{\min}^{-2.5} \qquad (5.18)
\end{aligned}$$

□

5.6 Better effective resistance approximation

In this section, we use divide-and-conquer based on Theorem 5.1.3 to ϵ-approximate effective resistances for a set of pairs of vertices $P \subseteq V(G)\times V(G)$ in time $O(m^{1+o(1)}+(|P|/\epsilon^2)\text{polylog}(n))$. The reduction we use is the same as in [33]. We give the algorithm

ResApx as follows:

Algorithm 31: ResApx(G, P, ϵ), never executed

Input: A weighted graph G, a set of pairs of vertices P, and an $\epsilon \in (0, 1)$
Output: Estimates $\{\tilde{r}_{u,v}\}_{(u,v)\in P}$ to effective resistances between vertex pairs in P

if $|P| = 1$ **then**
  Compute the Schur complement H of G onto P with error ϵ
  return $\left\{\tilde{r}_{u,v} := b_{u,v}^T L_H^+ b_{u,v}\right\}$ for the only $(u, v) \in P$
end
Let $\epsilon_1 := \frac{1}{2} \cdot \epsilon \cdot (1/\log|P|)$ and $\epsilon_2 := \epsilon \cdot (1 - 1/\log|P|)$.
Divide P into subsets $P^{(1)}$ and $P^{(2)}$ with equal sizes.
Let $V^{(1)}$ and $V^{(2)}$ be the respective set of vertices in $P^{(1)}$ and $P^{(2)}$.
Compute the Schur complement $H^{(1)}$ of G onto $V^{(1)}$ with error ϵ_1
Compute the Schur complement $H^{(2)}$ of G onto $V^{(2)}$ with error ϵ_2
$\tilde{r} \leftarrow$ ResApx$(H^{(1)}, P^{(1)}, \epsilon_2) \cup$ ResApx$(H^{(2)}, P^{(2)}, \epsilon_2)$
return $\tilde{r}$

Proof of Corollary 5.1.5. The approximation guarantees follows from

$$\begin{aligned}\tilde{r}_{u,v} \geq& \left(1 - \frac{1}{2} \cdot \epsilon/\log|P|\right)^{\log|P|-1} \cdot \left(b_{u,v}^T L_G^+ b_{u,v}\right)\\ \geq& (1-\epsilon) b_{u,v}^T L_G^+ b_{u,v}\end{aligned}$$

and

$$\begin{aligned}\tilde{r}_{u,v} \leq& \left(1 + \frac{1}{2} \cdot \epsilon/\log|P|\right)^{\log|P|-1} \cdot \left(b_{u,v}^T L_G^+ b_{u,v}\right)\\ \leq& (1+\epsilon) b_{u,v}^T L_G^+ b_{u,v}.\end{aligned}$$

We then prove the running time. Let $T(p, \epsilon)$ denote the running time of ResApx(G, P, ϵ) when $|P| = p$ and $|E(G)| = O((p/\epsilon^2)\text{polylog}(n))$. Clearly, the total running time of ResApx(G, P, ϵ) for any G with m edges is at most

$$2 \cdot T(|P|/2, \epsilon \cdot (1 - 1/\log|P|)) + O\left(m^{1+o(1)} + (|P|/\epsilon^2)\text{polylog}(n)\right), \tag{5.19}$$

since the first step of ResApx will divide the graph into two Schur complements with $O((|P|/\epsilon^2)\text{polylog}(n))$ edges each. Furthermore, we can write $T(p, \epsilon)$ in a recurrence form as

$$T(p, \epsilon) = 2 \cdot T(p/2, \epsilon \cdot (1 - 1/\log p)) + O\left(p^{1+o(1)} + (p/\epsilon^2)\text{polylog}(n)\right),$$

which gives

$$T(p, \epsilon) = O\left(p^{1+o(1)} + (p/\epsilon^2)\text{polylog}(n)\right).$$

Combining this with (5.19) gives the overall running time

$$O\left(m^{1+o(1)} + (|P|/\epsilon^2)\text{polylog}(n)\right).$$

□

Bibliography

[1] Dimitris Achlioptas. "Database-friendly random projections". In: *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*. 2001. DOI: 10.1145/375551.375608. URL: http://doi.acm.org/10.1145/375551.375608.

[2] David J. Aldous. "The Random Walk Construction of Uniform Spanning Trees and Uniform Labelled Trees". In: *SIAM J. Discret. Math.* 3.4 (Nov. 1990), pp. 450–465. ISSN: 0895-4801. DOI: 10.1137/0403039. URL: http://dx.doi.org/10.1137/0403039.

[3] Vedat Levi Alev et al. In: *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*. 2018, 41:1–41:16. DOI: 10.4230/LIPIcs.ITCS.2018.41. URL: https://doi.org/10.4230/LIPIcs.ITCS.2018.41.

[4] N. Alon and V. D. Milman. *Lambda 1, Isoperimetric Inequalities for Graphs, and Superconcentrators*. 1985.

[5] Noga Alon. "Eigenvalues and expanders". In: *Combinatorica* 6.2 (1986), pp. 83–96.

[6] Noga Alon, Yossi Matias, and Mario Szegedy. "The Space Complexity of Approximating the Frequency Moments". In: *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: ACM, 1996, pp. 20–29. ISBN: 0-89791-785-5. DOI: 10.1145/237814.237823. URL: http://doi.acm.org/10.1145/237814.237823.

[7] Ingo Althöfer et al. "On sparse spanners of weighted graphs". In: *Discrete & Computational Geometry* 9.1 (Jan. 1, 1993), pp. 81–100. ISSN: 1432-0444. DOI: 10.1007/BF02189308. URL: https://doi.org/10.1007/BF02189308.

[8] Nima Anari, Shayan Oveis Gharan, and Alireza Rezaei. "Monte Carlo Markov Chain Algorithms for Sampling Strongly Rayleigh Distributions and Determinantal Point Processes". In: *29th Annual Conference on Learning Theory*. Ed. by Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir. Vol. 49. Proceedings of Machine Learning Research. Columbia University, New York, New York, USA: PMLR, 23–26 Jun 2016, pp. 103–115. URL: http://proceedings.mlr.press/v49/anari16.html.

[9] Nima Anari et al. "Log-concave polynomials II: high-dimensional walks and an FPRAS for counting bases of a matroid". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*. 2019, pp. 1–12. DOI: 10.1145/3313276.3316385. URL: https://doi.org/10.1145/3313276.3316385.

[10] Alexandr Andoni, Anupam Gupta, and Robert Krauthgamer. "Towards $(1 + \epsilon)$ - Approximate Flow Sparsifiers". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*. 2014, pp. 279–293. DOI: 10.1137/1.9781611973402.20. URL: https://doi.org/10.1137/1.9781611973402.20.

[11] Alexandr Andoni and Piotr Indyk. "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions". In: *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*. 2006, pp. 459–468. DOI: 10.1109/FOCS.2006.49. URL: https://doi.org/10.1109/FOCS.2006.49.

[12] Arash Asadpour et al. "An O(Log N/ Log Log N)-approximation Algorithm for the Asymmetric Traveling Salesman Problem". In: *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '10. Austin, Texas: Society for Industrial and Applied Mathematics, 2010, pp. 379–389. ISBN: 978-0-898716-98-6. URL: http://dl.acm.org/citation.cfm?id=1873601.1873633.

[13] Baruch Awerbuch and David Peleg. "Sparse Partitions (Extended Abstract)". In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*. 1990, pp. 503–513. DOI: 10.1109/FSCS.1990.89571. URL: https://doi.org/10.1109/FSCS.1990.89571.

[14] Baruch Awerbuch et al. "Near-Linear Time Construction of Sparse Neighborhood Covers". In: *SIAM J. Comput.* 28.1 (Feb. 1999), pp. 263–277. ISSN: 0097-5397. DOI: 10.1137/S0097539794271898. URL: http://dx.doi.org/10.1137/S0097539794271898.

[15] Matthew Baker and Farbod Shokrieh. "Chip-firing games, potential theory on graphs, and spanning trees". In: *Journal of Combinatorial Theory, Series A* 120.1 (2013), pp. 164–182. ISSN: 0097-3165. DOI: https://doi.org/10.1016/j.jcta.2012.07.011. URL: http://www.sciencedirect.com/science/article/pii/S0097316512001355.

[16] Afonso S. Bandeira, Amit Singer, and Daniel A. Spielman. "A Cheeger Inequality for the Graph Connection Laplacian". In: *SIAM J. Matrix Analysis Applications* 34.4 (2013), pp. 1611–1630.

[17] Béla Bollobás. *Modern graph theory*. Vol. 184. Springer Science & Business Media, 2013.

[18] A. Broder. "Generating Random Spanning Trees". In: *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*. SFCS '89. Washington, DC, USA: IEEE Computer Society, 1989, pp. 442–447. ISBN: 0-8186-1982-1. DOI: 10.1109/SFCS.1989.63516. URL: https://doi.org/10.1109/SFCS.1989.63516.

[19] Ashok K. Chandra et al. "The Electrical Resistance of a Graph Captures its Commute and Cover Times". In: *Computational Complexity* 6.4 (1997), pp. 312–340. DOI: 10.1007/BF01270385. URL: https://doi.org/10.1007/BF01270385.

[20] Moses Charikar et al. "Vertex Sparsifiers and Abstract Rounding Algorithms". In: *CoRR* abs/1006.4536 (2010). arXiv: 1006.4536. URL: http://arxiv.org/abs/1006.4536.

[21] Jeff Cheeger. "A lower bound for the smallest eigenvalue of the Laplacian". English (US). In: *Proceedings of the Princeton conference in honor of Professor S. Bochner*. 1969, pp. 195–199.

[22] Yun Kuen Cheung, Gramoz Goranci, and Monika Henzinger. "Graph Minors for Preserving Terminal Distances Approximately - Lower and Upper Bounds". In: *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*. 2016, 131:1–131:14. DOI: 10.4230/LIPIcs.ICALP.2016.131. URL: https://doi.org/10.4230/LIPIcs.ICALP.2016.131.

[23] Paul Christiano et al. "Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs". In: *Proceedings of the forty-third annual ACM symposium on Theory of computing*. ACM. 2011, pp. 273–282.

[24] Fan Chung and Linyuan Lu. "Concentration inequalities and martingale inequalities: a survey". In: *Internet Math.* 3.1 (2006), pp. 79–127. URL: http://projecteuclid.org/euclid.im/1175266369.

[25] Julia Chuzhoy. "On vertex sparsifiers with Steiner nodes". In: *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*. 2012, pp. 673–688. DOI: 10.1145/2213977.2214039. URL: http://doi.acm.org/10.1145/2213977.2214039.

[26] Michael B. Cohen et al. "Solving SDD Linear Systems in Nearly Mlog1/2N Time". In: *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*. STOC '14. New York, New York: ACM, 2014, pp. 343–352. ISBN: 978-1-4503-2710-7. DOI: 10.1145/2591796.2591833. URL: http://doi.acm.org/10.1145/2591796.2591833.

[27] Charles J. Colbourn, Robert P. J. Day, and Louis D. Nel. "Unranking and Ranking Spanning Trees of a Graph". In: *J. Algorithms* 10.2 (1989), pp. 271–286.

[28] Charles J. Colbourn, Wendy J. Myrvold, and Eugene Neufeld. "Two Algorithms for Unranking Arborescences". In: *Journal of Algorithms* 20.2 (Mar. 1996), pp. 268–281. ISSN: 0196-6774. DOI: 10.1006/jagm.1996.0014.

[29] Mary Cryan, Heng Guo, and Giorgos Mousa. "Modified log-Sobolev inequalities for strongly log-concave distributions". In: *CoRR* abs/1903.06081 (2019). arXiv: 1903.06081. URL: http://arxiv.org/abs/1903.06081.

[30] Samuel I. Daitch and Daniel A. Spielman. "Faster approximate lossy generalized flow via interior point algorithms". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. 2008, pp. 451–460. DOI: 10.1145/1374376.1374441. URL: https://doi.org/10.1145/1374376.1374441.

[31] H. Davenport and A. Schinzel. "A Combinatorial Problem Connected with Differential Equations". In: *American Journal of Mathematics* 87.3 (1965), pp. 684–694. ISSN: 00029327, 10806377. URL: http://www.jstor.org/stable/2373068.

[32] David Durfee et al. "Determinant-Preserving Sparsification of SDDM Matrices with Applications to Counting and Sampling Spanning Trees". In: *CoRR* abs/1705.00985 (2017). arXiv: 1705.00985. URL: http://arxiv.org/abs/1705.00985.

[33] David Durfee et al. "Sampling random spanning trees faster than matrix multiplication". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM. 2017, pp. 730–742.

[34] Matthias Englert et al. "Vertex Sparsifiers: New Results from Old Techniques". In: *SIAM J. Comput.* 43.4 (2014), pp. 1239–1262. DOI: 10.1137/130908440. URL: https://doi.org/10.1137/130908440.

[35] Wai Shing Fung and Nicholas J. A. Harvey. "Graph Sparsification by Edge-Connectivity and Random Spanning Trees". In: *CoRR* abs/1005.0265 (2010).

[36] Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. "A Randomized Rounding Approach to the Traveling Salesman Problem". In: *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*. FOCS '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 550–559. ISBN: 978-0-7695-4571-4. DOI: 10.1109/FOCS.2011.80. URL: http://dx.doi.org/10.1109/FOCS.2011.80.

[37] Gramoz Goranci, Monika Henzinger, and Pan Peng. "Dynamic Effective Resistances and Approximate Schur Complement on Separable Graphs". In: *CoRR* abs/1802.09111 (2018). arXiv: 1802.09111. URL: http://arxiv.org/abs/1802.09111.

[38] Gramoz Goranci, Monika Henzinger, and Pan Peng. "Improved Guarantees for Vertex Sparsification in Planar Graphs". In: *25th Annual European Symposium on Algorithms (ESA 2017)*. Ed. by Kirk Pruhs and Christian Sohler. Vol. 87. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 44:1–44:14. ISBN: 978-3-95977-049-1. DOI: 10.4230/LIPIcs.ESA.2017.44. URL: http://drops.dagstuhl.de/opus/volltexte/2017/7833.

[39] Navin Goyal, Luis Rademacher, and Santosh Vempala. "Expanders via Random Spanning Trees". In: *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '09. New York, New York: Society for Industrial and Applied Mathematics, 2009, pp. 576–585. URL: http://dl.acm.org/citation.cfm?id=1496770.1496834.

[40] A. Guenoche. "Random spanning tree". In: *Journal of Algorithms* 4 (1983), pp. 214–220.

[41] Sariel Har-Peled, Piotr Indyk, and Anastasios Sidiropoulos. "Euclidean spanners in high dimensions". In: *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*. 2013, pp. 804–809. DOI: 10.1137/1.9781611973105.57. URL: https://doi.org/10.1137/1.9781611973105.57.

[42] Prahladh Harsha et al. "Minimizing average latency in oblivious routing". In: *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2008, pp. 200–207.

[43] Nicholas J. A. Harvey and Keyulu Xu. "Generating Random Spanning Trees via Fast Matrix Multiplication". In: *LATIN 2016: Theoretical Informatics: 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*. Ed. by Evangelos Kranakis, Gonzalo Navarro, and Edgar Chávez. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 522–535. ISBN: 978-3-662-49529-2. DOI: 10.1007/978-3-662-49529-2_39. URL: https://doi.org/10.1007/978-3-662-49529-2_39.

[44] Piotr Indyk. "Stable distributions, pseudorandom generators, embeddings, and data stream computation". In: *J. ACM* 53.3 (2006), pp. 307–323. DOI: 10.1145/1147954.1147955. URL: http://doi.acm.org/10.1145/1147954.1147955.

[45] William B Johnson and Joram Lindenstrauss. "Extensions of Lipschitz mappings into a Hilbert space". In: *Contemporary Mathematics* 26.189-206 (1984), p. 1.

[46] Jonathan A Kelner and Aleksander Madry. "Faster generation of random spanning trees". In: *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*. IEEE. 2009, pp. 13–21.

[47] Jonathan A Kelner, Gary L Miller, and Richard Peng. "Faster approximate multicommodity flow using quadratically coupled flows". In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM. 2012, pp. 1–18.

[48] Jonathan A. Kelner et al. "A Simple, Combinatorial Algorithm for Solving SDD Systems in Nearly-linear Time". In: *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*. STOC '13. Palo Alto, California, USA: ACM, 2013, pp. 911–920. ISBN: 978-1-4503-2029-0. DOI: 10.1145/2488608.2488724. URL: http://doi.acm.org/10.1145/2488608.2488724.

[49] Jonathan A. Kelner et al. "Metric uniformization and spectral bounds for graphs". In: *CoRR* abs/1008.3594 (2010).

[50] Jonathan Kelner and Petar Maymounkov. "Electric routing and concurrent flow cutting". In: *Theoretical Computer Science* 412.32 (2011), pp. 4123–4135.

[51] G. Kirchhoff. "Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird". In: *Annalen der Physik* 148 (1847), pp. 497–508. DOI: 10.1002/andp.18471481202.

[52] Alexandra Kolla et al. "Subgraph sparsification and nearly optimal ultrasparsifiers". In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. 2010, pp. 57–66. DOI: 10.1145/1806689.1806699. URL: http://doi.acm.org/10.1145/1806689.1806699.

[53] Ioannis Koutis, Alex Levin, and Richard Peng. "Faster Spectral Sparsification and Numerical Algorithms for SDD Matrices". In: *ACM Trans. Algorithms* 12.2 (Dec. 2015), 17:1–17:16. ISSN: 1549-6325. DOI: 10.1145/2743021. URL: http://doi.acm.org/10.1145/2743021.

[54] Ioannis Koutis, Gary L. Miller, and Richard Peng. "Approaching Optimality for Solving SDD Linear Systems". In: *SIAM Journal on Computing* 43.1 (2014), pp. 337–354. DOI: 10.1137/110845914. eprint: https://doi.org/10.1137/110845914. URL: https://doi.org/10.1137/110845914.

[55] V. G. Kulkarni. "Generating random combinatorial objects". In: *Journal of Algorithms* 11.2 (1990), pp. 185–207. ISSN: 0196-6774. DOI: 10.1016/0196-6774(90)90002-V.

[56] Tsz Chiu Kwok et al. "Improved Cheeger's Inequality: Analysis of Spectral Partitioning Algorithms Through Higher Order Spectral Gap". In: *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*. STOC '13. Palo Alto, California, USA: ACM, 2013, pp. 11–20. ISBN: 978-1-4503-2029-0. DOI: 10.1145/2488608.2488611. URL: http://doi.acm.org/10.1145/2488608.2488611.

[57] Rasmus Kyng. "Approximate Gaussian Elimination". PhD thesis. Yale University, 2017.

[58] Rasmus Kyng and Sushant Sachdeva. "Approximate Gaussian Elimination for Laplacians: Fast, Sparse, and Simple". In: *CoRR* abs/1605.02353 (2016). URL: http://arxiv.org/abs/1605.02353.

[59] Rasmus Kyng et al. "Sparsified Cholesky and multigrid solvers for connection laplacians". In: *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. 2016, pp. 842–850. DOI: 10.1145/2897518.2897640. URL: http://doi.acm.org/10.1145/2897518.2897640.

[60] Gregory Lawler and Hariharan Narayanan. "Mixing times and l p bounds for Oblivious routing". In: *Proceedings of the Meeting on Analytic Algorithmics and Combinatorics*. Society for Industrial and Applied Mathematics. 2009, pp. 66–74.

[61] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. "Multi-way Spectral Partitioning and Higher-order Cheeger Inequalities". In: *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*. STOC '12. New York, New York, USA: ACM, 2012, pp. 1117–1130. ISBN: 978-1-4503-1245-5. DOI: 10.1145/2213977.2214078. URL: http://doi.acm.org/10.1145/2213977.2214078.

[62] Yin Tat Lee, Richard Peng, and Daniel A. Spielman. "Sparsified Cholesky Solvers for SDD linear systems". In: *CoRR* abs/1506.08204 (2015). URL: http://arxiv.org/abs/1506.08204.

[63] Yin Tat Lee, Satish Rao, and Nikhil Srivastava. "A new approach to computing maximum flows using electrical flows". In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM. 2013, pp. 755–764.

[64] Yin Tat Lee and Aaron Sidford. "Path Finding Methods for Linear Programming: Solving Linear Programs in $\tilde{O}$(vrank) Iterations and Faster Algorithms for Maximum Flow". In: *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*. 2014, pp. 424–433. DOI: 10.1109/FOCS.2014.52. URL: https://doi.org/10.1109/FOCS.2014.52.

[65] Frank Thomson Leighton and Ankur Moitra. "Extensions and limits to vertex sparsification". In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. 2010, pp. 47–56. DOI: 10.1145/1806689.1806698. URL: http://doi.acm.org/10.1145/1806689.1806698.

[66] Tom Leighton and Satish Rao. "Multicommodity Max-flow Min-cut Theorems and Their Use in Designing Approximation Algorithms". In: *J. ACM* 46.6 (Nov. 1999), pp. 787–832. ISSN: 0004-5411. DOI: 10.1145/331524.331526. URL: http://doi.acm.org/10.1145/331524.331526.

[67] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006. URL: https://pages.uoregon.edu/dlevin/MARKOV/markovmixing.pdf.

[68] Huan Li and Aaron Schild. "Spectral Subspace Sparsification". In: *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*. 2018, pp. 385–396. DOI: 10.1109/FOCS.2018.00044. URL: https://doi.org/10.1109/FOCS.2018.00044.

[69] Anand Louis et al. "Many Sparse Cuts via Higher Eigenvalues". In: *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*. STOC '12. New York, New York, USA: ACM, 2012, pp. 1131–1140. ISBN: 978-1-4503-1245-5. DOI: 10.1145/2213977.2214079. URL: http://doi.acm.org/10.1145/2213977.2214079.

[70] Ulrike Luxburg. "A Tutorial on Spectral Clustering". In: *Statistics and Computing* 17.4 (Dec. 2007), pp. 395–416. ISSN: 0960-3174. DOI: 10.1007/s11222-007-9033-z. URL: http://dx.doi.org/10.1007/s11222-007-9033-z.

[71] Russell Lyons and Yuval Peres. *Probability on Trees and Networks*. Vol. 42. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, New York, 2016, pp. xv+699. ISBN: 978-1-107-16015-6. DOI: 10.1017/9781316672815. URL: http://dx.doi.org/10.1017/9781316672815.

[72] Aleksander Madry. "Navigating central path with electrical flows: From flows to matchings, and back". In: *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*. IEEE. 2013, pp. 253–262.

[73] Aleksander Madry, Damian Straszak, and Jakub Tarnawski. "Fast Generation of Random Spanning Trees and the Effective Resistance Metric". In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*. 2015, pp. 2019–2036. DOI: 10.1137/1.9781611973730.134. URL: https://doi.org/10.1137/1.9781611973730.134.

[74] Konstantin Makarychev and Yury Makarychev. "Metric Extension Operators, Vertex Sparsifiers and Lipschitz Extendability". In: *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. 2010, pp. 255–264. DOI: 10.1109/FOCS.2010.31. URL: https://doi.org/10.1109/FOCS.2010.31.

[75] Peter Matthews. "Covering Problems for Brownian Motion on Spheres". In: 16 (Jan. 1988).

[76] Marina Meila and Jianbo Shi. "Learning Segmentation by Random Walks". In: *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*. 2000, pp. 873–879. URL: http://papers.nips.cc/paper/1830-learning-segmentation-by-random-walks.

[77] Gary L. Miller and Richard Peng. "Approximate Maximum Flow on Separable Undirected Graphs". In: *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '13. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2013, pp. 1151–1170. ISBN: 978-1-611972-51-1. URL: http://dl.acm.org/citation.cfm?id=2627817.2627900.

[78] Gary L. Miller, Noel J. Walkington, and Alex L. Wang. "Hardy-Muckenhoupt Bounds for Laplacian Eigenvalues". In: *CoRR* abs/1812.02841 (2018). arXiv: 1812.02841. URL: http://arxiv.org/abs/1812.02841.

[79] Ankur Moitra. "Approximation Algorithms for Multicommodity-Type Problems with Guarantees Independent of the Graph Size". In: *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*. 2009, pp. 3–12. DOI: 10.1109/FOCS.2009.28. URL: https://doi.org/10.1109/FOCS.2009.28.

[80] Richard Peng and Daniel A. Spielman. "An Efficient Parallel Solver for SDD Linear Systems". In: *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*. STOC '14. New York, New York: ACM, 2014, pp. 333–342. ISBN: 978-1-4503-2710-7. DOI: 10.1145/2591796.2591832. URL: http://doi.acm.org/10.1145/2591796.2591832.

[81] Jim Propp. Personal communication with Aleksander Madry. 2010.

[82] Harald Racke. "Minimizing congestion in general networks". In: *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*. IEEE. 2002, pp. 43–52.

[83] Harald Räcke, Chintan Shah, and Hanjo Täubig. "Computing Cut-Based Hierarchical Decompositions in Almost Linear Time". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*. 2014, pp. 227–238. DOI: 10.1137/1.9781611973402.17. URL: https://doi.org/10.1137/1.9781611973402.17.

[84] Aaron Schild. "A Schur Complement Cheeger Inequality". In: *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*. 2019, 65:1–65:15. DOI: 10.4230/LIPIcs.ITCS.2019.65. URL: https://doi.org/10.4230/LIPIcs.ITCS.2019.65.

[85] Aaron Schild. "An almost-linear time algorithm for uniform random spanning tree generation". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*. 2018, pp. 214–227. DOI: 10.1145/3188745.3188852. URL: https://doi.org/10.1145/3188745.3188852.

[86] Aaron Schild, Satish Rao, and Nikhil Srivastava. "Localization of Electrical Flows". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*. 2018, pp. 1577–1584. DOI: 10.1137/1.9781611975031.103. URL: https://doi.org/10.1137/1.9781611975031.103.

[87] Jack Sherman and Winifried J. Morrison. "Adjustment of an inverse matrix corresponding to a change in one element of a given matrix." English. In: *Ann. Math. Stat.* 21 (1950), pp. 124–127. ISSN: 0003-4851. DOI: 10.1214/aoms/1177729893.

[88] Jianbo Shi and Jitendra Malik. "Normalized Cuts and Image Segmentation". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22.8 (Aug. 2000), pp. 888–905. ISSN: 0162-8828. DOI: 10.1109/34.868688. URL: https://doi.org/10.1109/34.868688.

[89] Alistair Sinclair. "Improved bounds for mixing rates of Markov chains and multicommodity flow". In: *Combinatorics, Probability and Computing* 1 (1992), pp. 351–370.

[90] Daniel A. Spielman and Nikhil Srivastava. "Graph sparsification by effective resistances". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. 2008, pp. 563–568. DOI: 10.1145/1374376.1374456. URL: http://doi.acm.org/10.1145/1374376.1374456.

[91] Daniel A. Spielman and Shang-Hua Teng. "Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems". In: *SIAM J. Matrix Analysis Applications* 35.3 (2014), pp. 835–885. DOI: 10.1137/090771430. URL: https://doi.org/10.1137/090771430.

[92] John Steenbergen, Caroline Klivans, and Sayan Mukherjee. "A Cheeger-type inequality on simplicial complexes". In: *Advances in Applied Mathematics* 56 (2014), pp. 56–77. ISSN: 0196-8858. DOI: https://doi.org/10.1016/j.aam.2014.01.002. URL: http://www.sciencedirect.com/science/article/pii/S0196885814000116.

[93] Luca Trevisan. *Lecture 4 from cs359g: Graph partitioning and expanders, Stanford University*. http://theory.stanford.edu/~trevisan/cs359g/lecture04.pdf. Jan. 2011.

[94] Joel A. Tropp. "User-Friendly Tail Bounds for Sums of Random Matrices". In: *Foundations of Computational Mathematics* 12.4 (Aug. 1, 2012), pp. 389–434. ISSN: 1615-3383. DOI: 10.1007/s10208-011-9099-z. URL: https://doi.org/10.1007/s10208-011-9099-z.

[95] Nisheeth K. Vishnoi. "Lx = b". In: *Foundations and Trends in Theoretical Computer Science* 8.1-2 (2013), pp. 1–141. DOI: 10.1561/0400000054. URL: https://doi.org/10.1561/0400000054.

[96] David Bruce Wilson. "Generating Random Spanning Trees More Quickly Than the Cover Time". In: *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: ACM, 1996, pp. 296–303. ISBN: 0-89791-785-5. DOI: 10.1145/237814.237880. URL: http://doi.acm.org/10.1145/237814.237880.

Appendix A

Schur Complement Cheeger Appendix

A.1 Proof of Theorem 2.1.3

Proof of Theorem 2.1.3. For any two sets of vertices S_1, S_2 in a graph G,

$$\texttt{Reff}_G(S_1, S_2) \min(\text{vol}_G(S_1), \text{vol}_G(S_2)) = \frac{1}{\sigma^G_{S_1,S_2}}$$

Therefore, the desired result follows from Lemmas 2.4.1 and 2.3.1.

□

A.2 Proof of Proposition 2.4.4

Proof of Proposition 2.4.4. Without loss of generality, suppose that $a \leq b$. We break the analysis up into cases:

Case 1: $a \leq 0$. In this case, $\kappa_q(a) = q/2$ for all $q \geq 0$, so

$$\begin{aligned}\int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^b \frac{(q/2 - q)^2}{q} \\ &+ \int_b^{2b} \frac{(q/2 - b)^2}{q} dq \\ &+ \int_{2b}^\infty \frac{(q/2 - q/2)^2}{q} dq \\ &= \frac{b^2}{8} + \int_b^{2b} (q/4 - b + b^2/q) dq \\ &= \frac{b^2}{2} - b^2 + b^2(\ln 2) \\ &\leq 10(a-b)^2\end{aligned}$$

as desired.

Case 2: $a > 0$ **and** $b \leq 2a$. In this case,

$$\begin{aligned}\int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^a \frac{(q - q)^2}{q} dq \\ &+ \int_a^b \frac{(a - q)^2}{q} dq \\ &+ \int_b^{2a} \frac{(a - b)^2}{q} dq \\ &+ \int_{2a}^{2b} \frac{(q/2 - b)^2}{q} dq \\ &+ \int_{2b}^\infty \frac{(q/2 - q/2)^2}{q} dq \\ &\leq \int_a^{2b} \frac{(a - b)^2}{q} dq \\ &= (a-b)^2 \ln(2b/a) \\ &\leq (a-b)^2 \ln 4 \leq 10(a-b)^2\end{aligned}$$

as desired.

Case 3: $a > 0$ **and** $b > 2a$. In this case,

$$\begin{aligned}
\int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^a \frac{(q-q)^2}{q} dq \\
&+ \int_a^{2a} \frac{(a-q)^2}{q} dq \\
&+ \int_{2a}^b \frac{(q/2-q)^2}{q} dq \\
&+ \int_b^{2b} \frac{(q/2-b)^2}{q} dq \\
&+ \int_{2b}^\infty \frac{(q/2-q/2)^2}{q} dq \\
&\leq \int_a^{2b} \frac{(q/2-q)^2}{q} dq \\
&\leq b^2/2 \\
&\leq 2(a-b)^2 \leq 10(a-b)^2
\end{aligned}$$

as desired.

□

Appendix B

Random Spanning Tree Appendix

B.1 Facts about electrical potentials

B.1.1 Bounding potentials using effective resistances

Lemma B.1.1. *Consider a graph I with three vertices $u, v, w \in V(I)$. Then*

$$\Pr_v[t_u > t_w] \leq \frac{\mathit{Reff}_I(u,v)}{\mathit{Reff}_I(u,w)}$$

Proof. This probability can be written in terms of normalized potentials:

$$\Pr_v[t_u > t_w] = \frac{b_{uw}^T L_I^+ b_{uv}}{b_{uw}^T L_I^+ b_{uw}}$$

Since electrical potentials are maximized on the support of the demand vector,

$$\frac{b_{uw}^T L_I^+ b_{uv}}{b_{uw}^T L_I^+ b_{uw}} \leq \frac{b_{uv}^T L_I^+ b_{uv}}{b_{uw}^T L_I^+ b_{uw}} = \frac{\mathtt{Reff}_I(u,v)}{\mathtt{Reff}_I(u,w)}$$

as desired. □

Lemma 4.8.6. *Consider a graph I with two clusters C_1 and C_2 with two properties:*

- *The I-effective resistance diameters of C_1 and C_2 are both at most R.*
- *The minimum effective resistance between a vertex in C_1 and a vertex in C_2 is at least γR for $\gamma > 4$.*

Let J be the graph with C_1 and C_2 identified to s and t respectively. Then $\mathit{Reff}_J(s,t) \geq (\gamma - 4)R$.

To prove this, we take advantage of the dual formulation of electrical flow given in [48]:

Remark 11 (Equation (2) of [48]). *The solution x for the optimization problem $Lx = b$ is also the vector that optimizes*

$$\max_{p \in \mathbb{R}^n} 2b^T p - p^T L p$$

Proof of Lemma 4.8.6. Pick arbitrary $s \in C_1, t \in C_2$ in I and use the electrical potentials $p = L_I^+ b_{st}$ to construct a good solution q to the dual formulation of electrical flows in J. Construct J from I by adding 0-resistance edges from s and t to all C_1 and C_2 vertices respectively. Notice that $p_s = \min_v p_v$ and $p_t = \max_v p_v$. For all v, let $q_v = \max(p_s + R, \min(p_t - R, p_v))$.

q is feasible, so by the above remark,

$$\texttt{Reff}_J(s,t) \geq 2b_{st}^T q - q^T L_J q$$

Notice that $2b_{st}^T q = 2(p_t - p_s - 2R)$, so we just need to upper bound $q^T L_J q$. Notice that for any $x \in C_1$, $p_x - p_s = b_{st}^T L_I^+ b_{sx} \leq b_{sx}^T L_I^+ b_{sx} \leq R$. Similarly, for any $y \in C_2$, $p_t - p_y \leq R$. Therefore, all vertices $x \in C_1$ and $y \in C_2$ have $q_x = p_s + R$ and $q_y = p_t - R$ respectively. This means that all 0-resistance edges in J have potential drop 0 across them, which means that $q^T L_J q$ is defined.

No other edges were added to create J and potential drops are only smaller using q instead of p, so $q^T L_J q \leq p^T L_I p = p_t - p_s$. Therefore,

$$\texttt{Reff}_J(s,t) \geq p_t - p_s - 4R \geq (\gamma - 4)R$$

as desired. □

B.1.2 Schur complement facts

Lemma 4.8.3. *Consider any three disjoint sets of vertices $S_0, S_1, S_2 \subseteq V(I)$ and $S_0' \subseteq S_0$. Let $J = \texttt{Schur}(I, S_0 \cup S_1 \cup S_2)$ and $J' = \texttt{Schur}(I, S_0' \cup S_1 \cup S_2)$. Then*

$$c^{J'}(E_{J'}(S_0', S_1)) \leq c^J(E_J(S_0, S_1))$$

Proof. It suffices to show this when $S_0 = S_0' \cup \{v\}$ for some vertex $v \in V(I)$. By Remark 4, $J' = \texttt{Schur}(J, V(J) \setminus \{v\})$. By the formula in Definition 5.2.1, each edge $\{u, v\}$ for $u \in S_1$ is mapped to a set of edges $\{u, w\}$ with $c_{uw} = c_{uv}^J c_{vw}^J / c_v^J$, where c_v^J is the total conductance of edges incident with v in J. In particular, summing over all w and using the fact that

$$\sum_{w \in S_0'} c_{vw}^J \leq c_v^J$$

shows that the sum of the conductances of the edges created when v is eliminated is less than the original uv conductance. This is the desired result. □

Lemma 4.8.4. *Consider any two disjoint sets $S_0, S_1 \subseteq V(I)$ with $S_0' \subseteq S_0$. Then $c^I(S_0', S_1) \leq c^I(S_0, S_1)$.*

Proof. By Proposition 4.9.1, $1/c^I(S_0', S_1) = b_{s_0's_1}^T L_{I/(S_0',S_1)}^+ b_{s_0's_1}$ and $1/c^I(S_0, S_1) = b_{s_0s_1}^T L_{I/(S_0,S_1)}^+ b_{s_0s_1}$. $I/(S_0, S_1)$ can be obtained from $I/(S_0', S_1)$ by identifying vertices. This only reduces the quadratic form by Rayleigh monotonicity, as desired. □

Lemma 4.8.5. *For any cluster S_C in a graph I and any $p \in (0, 1)$,*

$$c^I(C, V(I) \setminus S_{\{C,V(I)\setminus S_C\}}(p, C)) \leq \frac{c^I(C, V(I) \setminus S_C)}{p}$$

Proof. Let $J = I/(C, V(I) \setminus S_C)$, with C and $V(I) \setminus S_C$ identified to s and t respectively. Let J' be the graph obtained by subdividing every edge that crosses the p normalized st-potential threshold in J, identifying the vertices created through subdivision, and making J' the induced subgraph on vertices with normalized st-potential at most p. The st electrical flow on J restricted to J' is also an electrical flow in J' with energy exactly $p(b_{st}^T L_J^+ b_{st})$. $S_{C,V(I)\setminus S_C}(p, C)$ contains all edges of J', so by Proposition 4.9.1 and Rayleigh monotonicity,

$$c^I(C, V(I) \setminus S_{C,V(I)\setminus S_C}(p, C)) \leq \frac{1}{p b_{st}^T L_J^+ b_{st}}$$

By 4.9.1 again,

$$b_{st}^T L_J^+ b_{st} = \frac{1}{c^I(C, V(I) \setminus S_C)}$$

Substitution yields the desired result. □

B.1.3 Carving clusters from shortcutters does not increase conductance much

Lemma B.1.2. *Consider a graph I and three disjoint sets $S_0, S_1, S_2 \subseteq V(I)$ with $S_2' \subseteq S_2$. Let $J = \texttt{Schur}(I, S_0 \cup S_1 \cup S_2)$ and $J' = \texttt{Schur}(I, S_0 \cup S_1 \cup S_2')$. Then*

$$c^J(E_J(S_0, S_1)) \leq c^{J'}(E_{J'}(S_0, S_1))$$

Proof. It suffices to show this result when $S_2 = S_2' \cup \{v\}$ for some vertex $v \in V(I)$, since one can eliminate the vertices of S_2 one at a time to get to S_2'. In this case, $J' = \texttt{Schur}(J, V(J) \setminus \{v\})$ by Remark 4. By Definition 5.2.1,

$$L_{J'} = L_J[V(J) \setminus \{v\}, V(J) \setminus \{v\}] - L_J[V(J) \setminus \{v\}, \{v\}] 1/c_v^J L_J[\{v\}, V(J) \setminus \{v\}]$$

where c_v^J is the total conductance of edges incident with v in the graph J. $L_J[V(J) \setminus \{v\}, \{v\}]$ only consists of nonpositive numbers, so conductances in J' are only larger than they are in J. In particular, summing over all edges in $E_J(S_0, S_1)$ shows that

$$c^J(E_J(S_0, S_1)) \leq c^{J'}(E_{J'}(S_0, S_1))$$

as desired. □

Lemma 4.7.3. *Consider a graph H and two clusters C and S_C, with $C \subseteq S_C$. Let C' be disjoint from C. Additionally, suppose that*

- *The effective resistance diameters of C and C' in H are both at most R.*
- *The effective resistance distance between any pair of points in C and C' in H is at least $\beta_1 R$.*
- $c^H(S_C) \leq \frac{\tau}{R}$.

Then $c^H(S_C \setminus C') \leq \frac{\tau + 1/(\beta_1 - 4)}{R}$.

Proof. Let $H_0 = \texttt{Schur}(H, C \cup C' \cup (V(H) \setminus S_C))$. By Lemma 4.8.6 and Proposition 4.9.1,

$$c^H(C, C') \leq \frac{1}{(\beta_1 - 4)R}$$

By Lemma B.1.2 with $I \leftarrow H$, $S_0 \leftarrow C$, $S_1 \leftarrow C'$, $S_2' \leftarrow \emptyset$, and $S_2 \leftarrow V(H) \setminus S_C$,

$$c^{H_0}(E_{H_0}(C, C')) \leq c^H(C, C')$$

By Lemma B.1.2 with $I \leftarrow H$, $S_0 \leftarrow C$, $S_1 \leftarrow V(H) \setminus S_C$, $S_2' \leftarrow \emptyset$ and $S_2 \leftarrow C'$

$$c^{H_0}(E_{H_0}(C, V(H) \setminus S_C)) \leq c^H(S_C)$$

Therefore,

$$\begin{aligned} c^H(S_C \setminus C') &= c^H(C, C' \cup (V(H) \setminus S_C)) \\ &= c^{H_0}(E_{H_0}(C, C')) + c^{H_0}(E_{H_0}(C, V(H) \setminus S_C)) \\ &\leq \frac{1}{(\beta_1 - 4)R} + c^H(S_C) \end{aligned}$$

as desired. □

B.2 Deferred proofs for Section 4.5

B.2.1 Efficient construction of covering communities (`CoveringCommunity` implementation)

We now prove Lemma 4.5.3. Our construction is similar to the construction of sparse covers given in [14] and is made efficient for low-dimensional ℓ_2^2 metrics using locality-sensitive hashing ([11]). Our use of locality-sensitive hashing is inspired by [41].

The algorithm maintains a set of uncovered vertices S and builds families of clusters one at a time. We build each family by initializing a set $S' \leftarrow S$ and repeatedly building clusters one-by-one. Each cluster is built by picking a vertex in S' and building a effective resistance ball around it. We repeatedly consider growing the effective resistance radius of the ball by a factor of γ if it could contain a much larger number of edges (an m^{1/z_0} factor). This neighborhood can be generated efficiently (in time comparable to its size) using locality-sensitive hashing. After generating this neighborhood, one can grow it slightly in a way that decreases its boundary size using ball-growing (for example [66]).

Before giving the `CoveringCommunity` algorithm, we start by giving the ball-growing algorithm `BallGrow`. Ideally, `BallGrow` would do standard ball-growing done by sorting vertices $x \in X$ with respect to the values $||D(x_0) - D(x)||_2^2$. Unfortunately, $||D(x) - D(y)||_2^2$ is not necessarily a metric on X, despite the fact that it approximates the metric $\texttt{Reff}(x, y)$. Luckily, though, we only need to preserve distances to some vertex x_0 along with distances between endpoints of edges. This can be done by modifying D slightly, as described in the first line of `BallGrow`:

Algorithm 32: $\texttt{BallGrow}_D(x_0, X, I, R_1, R_2)$

1 $J \leftarrow$ the graph with $V(J) = V(I)$ and $E(J) = E_I(X) \cup$ edges from x_0 to each edge in X
2 $d_J \leftarrow$ the shortest path metric of J with $\{x, y\}$ edge weight $||D(x) - D(y)||_2^2$
3 $x_0, x_1, \ldots, x_k \leftarrow$ vertices in X in increasing order of $d_J(x_0, x_i)$ value
4 $i^* \leftarrow$ the value for which $c^I(\partial\{x_0, x_1, \ldots, x_{i^*}\})$ is minimized subject to the constraint that $d_J(x_0, x_i) \in (R_1, R_2]$ or $d_J(x_0, x_{i+1}) \in (R_1, R_2]$
5 **return** $\{x_0, x_1, \ldots, x_{i^*}\}$

Proposition B.2.1. *$\texttt{BallGrow}_D(x_0, X, I, R_1, R_2)$, given the Johnson-Lindenstrauss embedding D of the effective resistance metric of I with $\epsilon = 1/2$, returns a cluster C with the following properties:*

- *(Subset of input) $C \subseteq X$.*
- *(Large enough) C contains the subset of X with effective resistance distance $2R_1/3$ of x_0.*
- *(Not too large) The I-effective resistance diameter of C is at most $4R_2$.*

- *(Boundary size)* $c^I(\partial C) \leq \frac{2|E_I(X)|}{R_2 - R_1} + c^I(\partial C \cap \partial X)$.

Furthermore, it does so in $\tilde{O}(|E_I(X) \cup \partial X|)$ *time.*

Proof. **Subset of input.** The x_is are all in X and C only consists of x_is.

Large enough. By the lower bound constraint on i^*, C contains all vertices $x \in X$ with $d_J(x_0, x) \leq R_1$. Since the edge $\{x_0, x\} \in E(J)$, $d_J(x_0, x) \leq ||D(x_0) - D(x)||_2^2 \leq (3/2)\texttt{Reff}_I(x_0, x)$ by the upper bound of Theorem 4.3.3. For any vertex x with effective resistance distance at most $2R_1/3$ from x_0, $d_J(x_0, x) \leq R_1$. Therefore, any such x is in C, as desired.

Not too large. Consider any vertex $x \in C$. By the upper bound constraint on i^*, $d_J(x_0, x) \leq R_2$. By the triangle inequality for the effective resistance metric, to get a diameter bound of $4R_2$ on C, it suffices to show that for any $x \in X$, $\texttt{Reff}_I(x_0, x) \leq 2d_J(x_0, x)$.

Consider any path $\{y_0 = x_0, y_1, y_2, \ldots, y_k = x\}$ in J. The length of this path is $\sum_{i=0}^{k-1} ||D(y_i) - D(y_{i+1})||_2^2$ by definition of J. By the lower bound of Theorem 4.3.3,

$$\sum_{i=0}^{k-1} \texttt{Reff}_I(y_i, y_{i+1})/2 \leq \sum_{i=0}^{k-1} ||D(y_i) - D(y_{i+1})||_2^2$$

By the triangle inequality for the effective resistance metric,

$$\texttt{Reff}_I(x_0, x)/2 = \texttt{Reff}_I(y_0, y_k)/2 \leq \sum_{i=0}^{k-1} \texttt{Reff}_I(y_i, y_{i+1})/2$$

so all paths from x_0 to x in J have length at least $\texttt{Reff}_I(x_0, x)/2$. Therefore, $\texttt{Reff}_I(x_0, x)/2 \leq d_J(x_0, x)$, as desired.

Boundary size. It suffices to bound the conductance of edges in ∂C that have both endpoints in X. Consider the quantity

$$Q = \sum_{e=\{x,y\} \in E_I(X) : d_J(x_0,x) \leq d_J(x_0,y)} c_e^I (d_J(x_0, y) - d_J(x_0, x))$$

with x closer to x_0 than y in d_J-distance. We start by showing that there is a d_J distance threshold cut with conductance at most $\frac{Q}{R_2 - R_1}$. For any number $a \in \mathbb{R}$, let $\text{clamp}(a) = \max(\min(a, R_2), R_1)$. Notice that

$$
\begin{aligned}
&(R_2 - R_1) \min_{i: d_J(x_0,x_i) \in (R_1,R_2] \text{ or } d_J(x_0,x_{i+1}) \in (R_1,R_2]} \sum_{e \in E_I(X) \cap \partial\{x_0,x_1,\ldots,x_i\}} c_e^I \\
&\leq \sum_{i=0}^{k-1} (\text{clamp}(d_J(x_0,x_{i+1})) - \text{clamp}(d_J(x_0,x_i))) \sum_{e \in E_I(X) \cap \partial\{x_0,x_1,\ldots,x_i\}} c_e^I \\
&= \sum_{e=\{x,y\} \in E_I(X): d_J(x_0,x) \leq d_J(x_0,y)} c_e^I (\text{clamp}(d_J(x_0,y)) - \text{clamp}(d_J(x_0,x))) \\
&\leq Q
\end{aligned}
$$

Dividing both sides by $(R_2 - R_1)$ shows that the minimizing cut (for defining i^*) has total conductance at most $Q/(R_2 - R_1)$. Now, we upper bound Q. By the upper bound of Theorem 4.3.3 and the triangle inequality for d_J,

$$
\begin{aligned}
Q &\leq \sum_{e=\{x,y\} \in E_I(X)} c_e^I d_J(x,y) \\
&\leq \sum_{e=\{x,y\} \in E_I(X)} c_e^I ||D(x) - D(y)||_2^2 \\
&\leq \sum_{e \in E_I(X)} c_e^I (3/2) \texttt{Reff}_I(e) \\
&\leq 2|E_I(X)|
\end{aligned}
$$

This yields the desired conductance bound.

Runtime. Only $|X| - 1$ edges are added to I to make J. It only takes $\tilde{O}(|E_I(X) \cup \partial X|)$ time to compute all of the distances to x_0, which are the only ones that are queried. Finding i^* only takes one linear sweep over the x_is, which takes $\tilde{O}(|E_I(X) \cup \partial X|)$ time. □

Now, we implement `CoveringCommunity`:

Algorithm 33: $\texttt{CoveringCommunity}_D(X, I, R)$

```
  // the community that we output
1 F ← ∅
  // the set of uncovered vertices
2 S ← X
3 while S ≠ ∅ do
      // set to form F_is from
4     S' ← S
5     for i = 0, ..., 2z_0 do
6         F_i ← ∅
7         H_i ← (Rγ^i, Rγ^{i+1}, 1/n^{1/γ}, 1/n)-sensitive hash family for the ℓ_2^2 metric
          ||D(x) − D(y)||_2^2
8         H'_i ← a sample of (log n)n^{1/γ} hash functions from H_i
9         bucket all vertices by hash values for each function in H'_i
10    end
11    while S' ≠ ∅ do
12        C ← { arbitrary vertex v_0 in S'}
13        C' ← X
14        C'' ← X
15        i ← 0
16        while |E_I(C') ∪ ∂C'| > m^{1/z_0}|E_I(C) ∪ ∂C| do
              // update C' to be the set of nearby vertices to C
17            C' ← the set of vertices v ∈ X with h(v) = h(v_0) for some h ∈ H'_{i+1}
18            C'' ← subset of C' within D-distance (2R/3)γ^{i+1} of v_0 found using a scan
              of C'
19            C ← BallGrow_D(v_0, C'', I, (3/2)Rγ^i, 2Rγ^i)
20            i ← i + 1
21        end
22        F_i ← F_i ∪ {C}
23        S' ← S' \ C'
24        S ← S \ C
25    end
26    add all F_is to C
27 end
28 return C
```

Lemma 4.5.3. *The algorithm* $\texttt{CoveringCommunity}_D(X, I, R)$*, when given a cluster* X*, a graph* I*, a radius* R*, and a Johnson-Lindenstrauss embedding* D *of the vertices of* $V(I)$*, returns an* $\mu_{rad}R$*-community* $\mathcal{D}$ *with the following properties:*

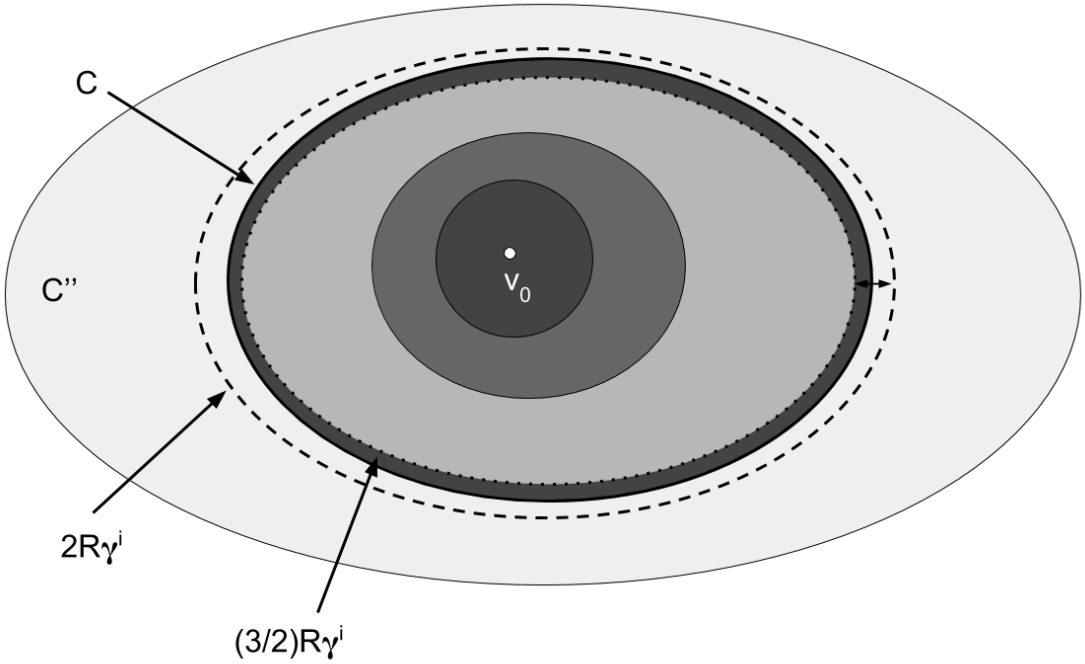

Figure B.2.1: How `CoveringCommunity` constructs one core. When clustering stops at some C'', a ball C with small boundary between the $(3/2)R\gamma^i$ and $2R$ radii balls centered around C_0 is found using `BallGrow`. C'' is removed from the set S' of candidates that are well-separated from v_0 and C is removed from the set of vertices that need to be covered.

- *(Input constraint) $\mathcal{D}$ is X-constrained.*
- *(Covering) Each vertex in X is in some cluster of some family in $\mathcal{D}$.*
- *(Boundedness) Each family $\mathcal{F} \in \mathcal{D}$ satisfies*

$$\sum_{C \in \mathcal{F}} c^I(\partial_I C) \leq \frac{\kappa_0 |E_I(X) \cup \partial_I X|}{R} + c^I(\partial_I X)$$

- *(Well-separatedness) $\mathcal{D}$ is γ_{ds}-well-separated.*
- *(Number of families) $\mathcal{D}$ has at most μ_{app} families.*

Furthermore, `CoveringCommunity` takes almost-linear time in $|E(X) \cup \partial X|$.

Proof of Lemma 4.5.3. **$\mu_{\mathbf{rad}}R$-community.** By the "Not too large" condition of Proposition B.2.1, each cluster added to some family has diameter at most $8R\gamma^{i_{max}} \leq \mu_{\text{rad}}R$, where i_{max}

is the maximum value of i over the course of the algorithm. To show that $\mathcal{C}$ is an $\mu_{\text{rad}}R$-community, we therefore just have to show that

$$i_{max} \leq 2z_0$$

, where $z_0 = (1/10)\log_\gamma \mu_{\text{rad}}$.

To do this, we show that the innermost while loop executes at most $2z_0$ times. Let C_i, C_i', and C_i'' be the values of C, C', and C'' set for the value of i in the innermost while loop. By the second (high distance) condition of locality-sensitive families, C_i' only consists of vertices in X with D-distance at most $R\gamma^{i+2}$ from v_0 with high probability. By the first (low distance) condition of locality-sensitive families, C_i'' contains all of the vertices in X with D-distance at most $(2R/3)\gamma^{i+1}$ from v_0. Therefore, by the "Large enough" condition of Proposition B.2.1, C_i consists of all vertices in X with D-distance at most $R\gamma^i$ from v_0. Therefore, $C_i' \subseteq C_{i+2}$ for all i. By the inner while loop condition, $|E(C_{i+2}) \cup \partial C_{i+2}| \geq m^{1/z_0}|E(C_i) \cup \partial C_i|$ for all i for which iteration $i+2$ occurs. But $|E(C_{i+2}) \cup \partial C_{i+2}| \leq n$ for all i. This means that $i_{max} \leq 2z_0$, as desired.

Input constraint. Each cluster consists of vertices in S, which is initialized to X and only decreases in size. Therefore, $\mathcal{C}$ is X-constrained at the end of the algorithm.

Covering. Vertices are only removed from S in Line 24. When they are removed from S, they are added to some cluster in $\mathcal{F}$ in Line 22. The algorithm only terminates when S is empty. Therefore, if `CoveringCommunity` terminates, it outputs a covering community.

Boundedness. Let C be a cluster and consider the C' and C'' used to generate it. By the "Boundary size" condition of Proposition B.2.1,

$$c^I(\partial C) \leq \frac{4|E_I(C'')|}{R\gamma^i} + c^I(\partial C \cap \partial C'')$$

Now, we bound the conductance of the set $\partial C \cap \partial C''$. By the first (upper bound) condition of locality-sensitive families, C'' contains all vertices in X within $(4R/9)\gamma^{i+1} \geq 16R\gamma^i$ I-effective resistance distance of v_0. By the "Not too large" condition of Proposition B.2.1, C contains vertices with I-effective resistance distance $8R\gamma^i$ of v_0. Therefore, by the triangle inequality for effective resistance, each edge in $\partial C \cap \partial C''$ with both endpoints in X has conductance at most $1/(8R\gamma^i)$. This means that

$$c^I(\partial C \cap \partial C'') \leq \frac{|\partial C''|}{8R\gamma^i} + c^I(\partial C \cap \partial X)$$

The C''s for clusters C in the same family are disjoint by Line 23. Therefore, the total boundary size of all clusters C in one family $\mathcal{F}$ is at most

$$\sum_{C \in \mathcal{F}} \left(\frac{4|E_I(C'')|}{R} + \frac{4|\partial C''|}{R} + c^I(\partial X \cap \partial C) \right) = c^I(\partial X) + \frac{4|E_I(X) \cup \partial X|}{R}$$

since $\gamma > 1$. This is the desired result.

Well-separatedness. By the "Not too large" condition of Proposition B.2.1 and Theorem 4.3.3, C' contains all vertices with I-effective resistance distance at most $(2R/3)\gamma^{i+1}$ from v_0. The corresponding cluster C has I-effective resistance diameter at most $8R\gamma^i$. Therefore, if C is added to a family $\mathcal{F}_{i+1}$, it is $\gamma/12$-well-separated from any cluster added to $\mathcal{F}_{i+1}$ in the future.

Since $\mathcal{F}_{i+1}$ only consists of clusters that were added to $\mathcal{F}_{i+1}$ with separation at least $(2R/3)\gamma^{i+1}$ from the remaining vertices in S', each cluster in $\mathcal{F}_{i+1}$ is also $\gamma/12$-well-separated from any cluster added to $\mathcal{F}_{i+1}$ in the past. Therefore, each F_i is a well-separated family.

Number of families. When a cluster C is added to a family $\mathcal{F}_i$, the number of edges incident with S' decreases by at most $|E_I(C') \cup \partial C'| \leq m^{1/z_0}|E_I(C) \cup \partial C|$. The number of edges incident with S decreases by at least $|E_I(C) \cup \partial C|$. Therefore, when S' is empty, S must have had at least m^{1-1/z_0} incident edges removed. Therefore, the outer while loop can only execute m^{1/z_0} times. Each outer while loop iteration adds $2z_0$ families to $\mathcal{C}$, so the total number of families is at most $2z_0 m^{1/z_0}$, as desired.

Runtime. The innermost while loop runs for $2z_0$ iterations. Line 17 takes time proportional to the number of elements in the bin $h(v_0)$ for the $n^{1/\gamma} = m^{o(1)}$ hash functions in $\mathcal{H}'_{i+1}$. The runtime of the innermost while loop is therefore at most $O(n^{1/\gamma} z_0 |E_I(C') \cup \partial C'|)$ for the final C' that it outputs by the "Runtime" condition of Proposition B.2.1. This runtime can be charged to the removal of C' (and its incident edges) from S'. Therefore, each middle while loop takes $O(n^{1/\gamma} z_0 |E_I(X) \cup \partial X|)$ time. As described in the "Number of families" condition, the outermost while loop only executes $2z_0 m^{1/z_0}$ times. Therefore, the total runtime is $O(m^{1/z_0} n^{1/\gamma} z_0^2 |E_I(X) \cup \partial X|) \leq |E_I(X) \cup \partial X| m^{o(1)}$, as desired. □

B.2.2 Efficient construction of Voronoi diagrams (`Voronoi` implementation)

We now prove Lemma 4.5.6. Ideally, the algorithm would just compute the set of vertices from which a random walk has a probability of 7/8 of hitting one cluster in $\mathcal{F}$ before any other. While this satisfies the correctness constraints, doing this would take too long. Computing the set with probability 7/8 of hitting one cluster before another would take a Laplacian solve on Z for each cluster. This is prohibitive, as we have no bound on the number of clusters in $\mathcal{F}$. This algorithm does not take advantage of slack on the lower bound for S_C.

Instead, we arbitrarily split the clusters into two groups (X_i, Y_i) in $\log n$ different ways, generate $S_{\{X_i,Y_i\}}(1/(8\log n), X_i)$ and $S_{\{X_i,Y_i\}}(1/(8\log n), Y_i)$, and refine the resulting clusters. This only requires $O(\log n)$ Laplacian solves on Z. The lower bound on S_C follows immediately from implication. The upper bound on S_C follows from thinking in terms of random walks. The event in which a random walk hitting C before any other cluster in $\mathcal{F}$ is equivalent to the conjunction of the $\log n$ events in which the random walk hits C's half in each partition before the other half. The probability that any of these events can bound

bounded satisfactorially with a union bound.

Algorithm 34: `Voronoi`$(I, \mathcal{F})$

```
C_1, C_2, ..., C_k ← arbitrary ordering of clusters in F
S_{C_1}, ..., S_{C_k} ← V(I)
for i = 0, 1, ..., log k do
    X_i ← the union of clusters C_j ∈ F with ith index digit 0 in binary
    Y_i ← {C_1, ..., C_k} \ X_i
    for j = 1, ..., k do
        Z_ij ← the member of {X_i, Y_i} that contains C_j;
        S_{C_j} ← S_{C_j} ∩ S_{{X_i,Y_i}}(1/(8 log k), Z_ij)
    end
end
return {S_{C_1}, ..., S_{C_k}}
```

Lemma 4.5.6. *The algorithm* `Voronoi`$(I, \mathcal{F})$ *takes a family* $\mathcal{F}$ *in the graph* I *and outputs a clan* $\mathcal{C}$ *in near-linear time in* $|E(Z) \cup \partial Z|$ *with the property that for each* $C \in \mathcal{F}$*, there is a shortcutter* $S_C \in \mathcal{C}$ *with the property that* $S_{\mathcal{F}}(1/(8 \log n), C) \subseteq S_C \subseteq S_{\mathcal{F}}(1/8, C)$*, where* $Z = V(I) \setminus (\cup_{C \in \mathcal{F}} C)$*.*

Proof of Lemma 4.5.6. **Lower bound.** We show that $S_{\mathcal{F}}(1/(8 \log k), C) \subseteq S_C$ for each $C \in \mathcal{F}$. Consider a vertex $x \in S_{\mathcal{F}}(1/(8 \log k), C)$. This vertex has the property that $\Pr_x[t_C > t_{\mathcal{F} \setminus C}] \leq (1/8 \log k)$. Let $C_j := C$. Since $C \subseteq Z_{ij}$, $x \in S_{\{X_i, Y_i\}}(1/(8 \log k), Z_{ij})$ for all i. Therefore, it is in the intersection of them as well. Since S_C is this intersection, $x \in S_C$, as desired.

Upper bound. We show that $S_C \subseteq S_{\mathcal{F}}(1/8, C)$. Consider a vertex $x \in S_C$. Let $C_j := C$. Since some $\{X_i, Y_i\}$ partition separates C from each other cluster in $\mathcal{F}$,

$$\Pr_x[t_C > t_{\mathcal{F} \setminus C}] = \Pr_x[\exists i \; t_{Z_{ij}} > t_{\mathcal{F} \setminus Z_{ij}}]$$

By a union bound,

$$\Pr_x[\exists i \; t_{Z_{ij}} > t_{\mathcal{F} \setminus Z_{ij}}] \leq \sum_{i=1}^{\log k} \Pr_x[t_{Z_{ij}} > t_{\mathcal{F} \setminus Z_{ij}}] \leq \frac{\log k}{8 \log k} \leq 1/8$$

Therefore, $x \in S_{\mathcal{F}}(1/8, C)$, as desired.

Runtime. Each $S_{\{X_i, Y_i\}}(1/(8 \log k), Z_{ij})$ can be computed in near-linear time in the number of edges incident with Z by Remark 7. Since there are only $\log k$ is and only 2 possible Z_{ij}s for each i, the algorithm takes near-linear time. □

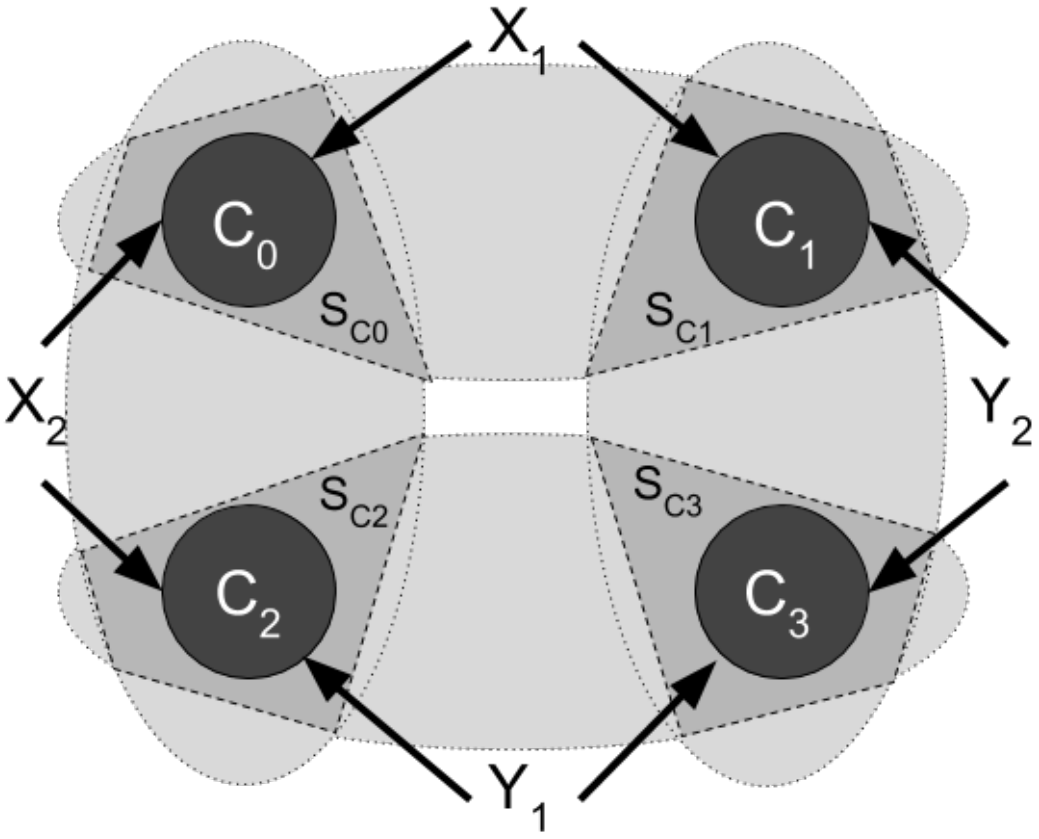

Figure B.2.2: How `Voronoi` constructs shortcutters. The horizontal and vertical dotted boundary sets are the $S_{\{X_1,Y_1\}}(1/(8\log k), Z_{i1})$ and $S_{\{X_2,Y_2\}}(1/(8\log k), Z_{i2})$ respectively.

B.2.3 A generalization of $\sum_{e\in E(G)} \texttt{lev}_G(e) = n-1$ to edges between well-separated clusters in the effective resistance metric (Lemma 4.5.4)

In this section, we prove Lemma 4.5.4:

Lemma 4.5.4. *Consider a $\gamma_{ds} = 2^{(\log n)^{2/3}}$ well-separated R-family of clusters $\mathcal{F}$ in a graph G. Let $H := \texttt{Schur}(G, \cup_{C\in\mathcal{F}} C)$. Then*

$$\sum_{e\in E(C,C'), C\neq C'\in\mathcal{F}} \frac{\texttt{Reff}_H(e)}{r_e^H} \leq \mu_{app}|\mathcal{F}|$$

We will use the Aldous-Broder algorithm for sampling random spanning trees (see Theorem 4.2.1) to reduce proving this result to a combinatorial problem. This problem is very closely related to Davenport-Schinzel strings from computational geometry:

Definition B.2.2 (Davenport-Schinzel strings [31])**.** *Consider a string $S = s_1 s_2 \dots s_y$ from an alphabet of size n. S is called an* (n, s)-Davenport-Schinzel *string if the following two properties hold:*

- $s_i \neq s_{i+1}$ *for all i*
- *There is no subsequence of S of the form $ABA \dots BA$ of length $s+2$ for two distinct letters A and B. We call such a subsequence an* alternating subsequence *with length $s+2$.*

Davenport-Schinzel strings are useful because they cannot be long. Here is a bound that is good for nonconstant s, but is not optimal for constant s:

Theorem B.2.3 (Theorem 3 and Inequality (34) of [31])**.** (n, s)*-Davenport-Schinzel strings cannot be longer than* $nC_0(n, s) := n(3s)! \exp(10\sqrt{s \log s \log n})$.

We now outline the proof of Lemma 4.5.4. Recall that the leverage score of an edge is the probability that the edge is in a random spanning tree (see Theorem 4.3.8). Therefore, it suffices to bound the expected number of edges in a random spanning tree between clusters of $\mathcal{F}$ in H.

We bound this by reasoning about Aldous-Broder. Start by considering the case in which each cluster in $\mathcal{F}$ is a single vertex. Then H is a graph with k vertices. All spanning trees of this graph have at most k edges, so the total leverage score of edges between clusters is at most k.

When $\mathcal{F}$ consists of (possibly large) well-separated clusters, we exploit Lemma B.1.1 to show that a random walk that goes between any two clusters more than $\log_{\gamma_{\text{ds}}} n$ times covers both clusters with high probability. This ensures that the sequence of destination clusters $(s_i)_i$ for edges added to the tree by Aldous-Broder satisfies the second condition of Davenport-Schnizel sequences for $s = \log_{\gamma_{\text{ds}}} n = (\log n)^{1/3}$ with high probability. If two clusters did alternate more than s times in this sequence, both would be covered, which means that Aldous-Broder would not add any more edges with a desination vertex in either cluster.

The sequence $(s_i)_i$ does not necessarily satisfy the first definition of Davenport-Schinzel because the random walk could go from an uncovered cluster C to a covered one C' and back. This motivates all of the pseudocode in the if statement on Line 15 of `InterclusterEdges`. If C is visited more than ζ times in a row, the random walk must visit many clusters that are very "tied" to C in the sense that they have a high probability of returning to C before hitting any other unvisited cluster. These ties are represented in a forest T on the clusters of $\mathcal{F}$. Each time a sequence of ζ Cs occurs, one can add another edge to T. T can only

have k edges because it is a forest. These ideas show that $(s_i)_i$ is only ζ times longer than an $(k, \log_{\gamma_{\text{ds}}} n)$-Davenport-Schinzel after removing ζk additional letters that are charged to edges of T. Theorem B.2.3 applies and finishes the proof.

Now, we discuss this intuition more formally. Let $k = |\mathcal{F}|$. It suffices to show that a random spanning tree in H contains $O(kC_0(k, \log_{\gamma_{\text{ds}}} n))$ edges between the clusters of $\mathcal{F}$ with high probability. Generate a random spanning tree using Aldous-Broder and organize the

intercluster edges using the following algorithm:

Algorithm 35: `InterclusterEdges`$(H, \mathcal{F})$, never executed

Data: a family of disjoint clusters $\mathcal{F}$ and a graph H on the vertices in $\cup_{C\in\mathcal{F}} C$
Result: A sequence of visited clusters $(s_i)_{i=0}^{\ell}$ and a rooted forest T with $V(T) = \mathcal{F}$ with all edges pointing towards from the root

$i \leftarrow 0$
$\zeta \leftarrow 100 \log^2(n\alpha)$
$\beta \leftarrow 1/(2 \log n)$
`// start Aldous-Broder`
$u_0 \leftarrow$ arbitrary vertex of H
$c \leftarrow 1$
$T \leftarrow (\mathcal{F}, \emptyset)$
while *H is not covered* **do**
  $u_c \leftarrow$ random neighbor of u_{c-1} in H
  $u \leftarrow u_{c-1}, v \leftarrow u_c$
  $C_u \leftarrow$ cluster in $\mathcal{F}$ containing u
  $C_v \leftarrow$ cluster in $\mathcal{F}$ containing v
  if *$C_u \neq C_v$ and C_v is not covered* **then**
    $s_i \leftarrow C_v$
    $i \leftarrow i + 1$
    if *$s_j = C_v$ for all $j \in [i - \zeta, i - 1]$, and $i \geq \zeta$* **then**
      `// in a multiset, members can be counted multiple times`
      $\mathcal{E} \leftarrow$ the multiset of clusters $A \in \mathcal{F}$ for which (1) A is visited immediately after one of the last ζ occurrences of C_v, (2) The root cluster B_A of A's tree in T is visited before the next visit to C_v and (3) $\Pr_x[$ hit C_v before any other uncovered cluster$] \geq 1 - \beta$ for any $x \in B_A$
      while *$\mathcal{E}$ contains a cluster C whose tree in T contains more than half of the clusters in the new tree formed by attaching $\mathcal{E}$'s trees to C_v* **do**
        Remove all clusters in C's arborescence that are also in $\mathcal{E}$ from $\mathcal{E}$
      end
      for *clusters $A \in \mathcal{E}$* **do**
        add an edge from B_A to C_v in T if one does not exist
      end
      remove the last ζ occurrences of C_v from $(s_j)_j$
      $i \leftarrow i - \zeta$
    end
  end
  $c \leftarrow c + 1$
end
return $(s_i)_i, T$

Let $(s'_i)_i$ be the sequence $(s_i)_i$ obtained by removing lines 14 to 25 from `InterclusterEdges` and removing the statement "and C_v is not covered" from the outer if statement; that is s'_i is the list of all cluster visits. We break up the analysis of this algorithm into a few subsections.

Invariants of the forest T

We start by discussing some invariants that hold for the directed graph T over the entire course of the algorithm:

Invariant B.2.4. *T is a forest of directed arborescences with all edges pointing towards the root.*

Proof. Initially, T just consists of isolated vertices, which trivially satisfy the invariant. Line 21 is the only line of `InterclusterEdges` that adds edges to T. When Line 21 adds an edge, it adds an edge from a root of one arborescence to the root of another arborescence. This preserves the invariant, so we are done. □

Invariant B.2.5. *Each arborescence with size κ in T has maximum leaf-to-root path length $\log \kappa$.*

Proof. Initially, T consists of isolated vertices, each of which has diameter $\log 1 = 0$. Line 19 ensures that the arborescence produced by the for loop containing Line 21 contains at least twice as many vertices as any of the constituent child arborescences. Furthermore, the length of the path to the root from each leaf increases by at most 1. Therefore, the new maximum path length is at most $1 + \log \kappa \leq \log(2\kappa) \leq \log \kappa'$, where κ is the maximum size of a child arborescence and κ' is the size of the combined arborescence created by Line 21. Therefore, Line 21 maintains the invariant. □

Invariant B.2.6. *For each edge $(C_0, C_1) \in E(T)$ and any $x \in C_0$,*

$$\Pr_x[\textit{hits } C_1 \textit{ before any uncovered cluster}] \geq 1 - \beta$$

Proof. Initially, there are no edges, so the invariant is trivially satisfied. When an edge (C_0, C_1) is added to T, property (3) on Line 16 ensures that the invariant is satisfied. As `InterclusterEdges` progresses, clusters that are covered remain covered, so

$\Pr_x[\text{hits } C_1 \text{ before any uncovered cluster}]$ only increases. In particular, the invariant remains satisfied. □

These invariants imply the following proposition, which will effectively allow us to replace occurences of a cluster $A \in \mathcal{F}$ in the sequence $\{s'_j\}_j$ with the root B_A of A's arborescence in T:

Proposition B.2.7. *For each cluster $A \in \mathcal{F}$ in a component of T with root $B_A \in \mathcal{F}$,*

$$\Pr_x[\textit{hits } B_A \textit{ before any uncovered cluster}] \geq 1/2$$

for any $x \in A$.

Proof. By Invariant B.2.5, there is a path from A to B_A in T with length at most $\log \kappa \leq \log n$. Let $C_0 = A, C_1, \ldots, C_\ell = B_A$ with $\ell \leq \log n$ be the path in A's arborescence from A to B_A. Notice that

$$\begin{aligned}
&\Pr_x[\text{hits } B_A \text{ before any uncovered cluster}] \\
&\geq \Pr_x[\wedge_{p=0}^{\ell} \text{hits } C_p \text{ after } C_{p-1}, C_{p-2}, \ldots, C_1, C_0 \text{ and before any uncovered cluster}] \\
&= \prod_{p=0}^{\ell-1} \Pr_x[\text{hits } C_{p+1} \text{ after } C_p, C_{p-1}, \ldots, C_0 \text{ and before any uncovered cluster} \\
&|\text{hits } C_p \text{ after } C_{p-1} \text{ after } \ldots \text{ after } C_0 \text{ and before any uncovered cluster}] \\
&= \prod_{p=0}^{\ell-1} \Pr_x[\text{hits } C_{p+1} \text{ after } C_p \text{ and before any uncovered cluster} \\
&|\text{hits } C_p \text{ after } C_{p-1} \text{ after } \ldots \text{ after } C_0 \text{ and before any uncovered cluster}]
\end{aligned}$$

By the Markov property,

$$\begin{aligned}
&\prod_{p=0}^{\ell-1} \Pr_x[\text{hits } C_{p+1} \text{ after } C_p \text{ and before any uncovered cluster} \\
&|\text{hits } C_p \text{ after } C_{p-1} \text{ after } \ldots \text{ after } C_0 \text{ and before any uncovered cluster}] \\
&= \prod_{p=0}^{\ell-1} \Pr_x[\text{hits } C_{p+1} \text{ after } C_p \text{ and before any uncovered cluster} \\
&|\text{hits } C_p \text{ before any uncovered cluster}]
\end{aligned}$$

Furthermore, by Invariant B.2.6,

$$\begin{aligned}
&\Pr_x[\text{hits } C_{p+1} \text{ after } C_p \text{ and before any uncovered cluster}|\text{hits } C_p \text{ before any uncovered cluster}] \\
&\geq \max_{y \in C_p} \Pr_y[\text{hits } C_{p+1} \text{ before any uncovered cluster}] \\
&\geq 1 - \beta
\end{aligned}$$

Combining these inequalities shows that

$$\begin{aligned}\Pr_x[\text{hits } B_A \text{ before any uncovered cluster}] &\geq (1-\beta)^\ell \\ &\geq (1-\ell\beta) \\ &\geq (1-(\log n)\beta) \\ &\geq 1/2\end{aligned}$$

as desired. □

Number of possible alternations before coverage

Next, we show that any two clusters can alternate a small number of times before both being covered. This is useful for bounding both the length of $(s_i)_i$ at the end of the algorithm and the number of edges added to T.

Specifically, we show the following:

Proposition B.2.8. *Consider two clusters C and C' in some graph G. Suppose that the C and C' have effective resistance diameter γR and that $\min_{u\in C, v\in C'} \texttt{Reff}_G(u,v) \geq \gamma R$. Then, for any $x \in C$ or C' and any $z > 0$,*

$$\Pr_x[C \text{ or } C' \text{ is not covered after alternating between them more than } T \text{ times}] \leq n\left(\frac{1}{\gamma-4}\right)^\tau$$

Proof. Consider two vertices $y_0, y_1 \in C$. By Lemmas B.1.1 and 4.8.6

$$\Pr_{y_0}[\text{hit } C' \text{ before visiting } y_1]\frac{R}{(\gamma-4)R} = \frac{1}{\gamma-4}$$

In particular, by the Markov property,

$$\begin{aligned}&\Pr_x[y \text{ not visited before alternating between } C \text{ and } C' \ T \text{ times}] \\ &\leq \prod_{t=0}^{T-1} \max_{y'\in C} \Pr_{y'}[y \text{ not visited before alternating between } C \text{ and } C' \text{ once}] \\ &\leq \left(\frac{1}{\gamma-4}\right)^\tau\end{aligned}$$

If C or C' has not been covered, then there exists a vertex in C or C' that has not yet been visited. Therefore, by a union bound, the probability that there exists an unvisited vertex in C or C' after T alternations is at most $(1/(\gamma-4))^\tau$, as desired. □

For all subsequent sections, set $\tau = 5(\log(\alpha m))/(\log(\gamma - 4))$. This ensure that any pair of clusters is covered with probability at least $1 - n(1/(m\alpha)^5) \geq 1 - (m\alpha)^4$ after the random walk alternates between both of them T times. In particular, the following property holds:

Definition B.2.9 (Fresh sequences)**.** *Consider the two sequences $(s_i)_i$ and the supersequence $(s'_j)_j$ defined using the algorithm* `InterclusterEdges`*. This pair of sequences is called* fresh *if any cluster $C \in \mathcal{F}$'s appearances in $(s_i)_i$ do not alternate with some cluster C' more than τ times in the supersequence $(s'_j)_j$.*

We exploit Proposition B.2.8 to show the following:

Proposition B.2.10. *The pair of sequences $(s_i)_i$ and $(s'_j)_j$ is fresh with probability at least $1 - 1/(m\alpha)^2$.*

Proof. $(s_i)_i$ is a sequence of uncovered clusters. Therefore, by Proposition B.2.8, the appearances of a cluster C in $(s_i)_i$ cannot alternate with the appearances of some other cluster C' in $(s'_j)_j$ more than τ times with probability at least $1 - 1/(m\alpha)^4$. Union bounding over all pairs of clusters C and C' yields the desired result. □

Hitting probabilities are similar within a cluster

We now show the following fact, which will allow us to turn a maximum probability lower bound into a bound on the minimum. In the following proposition, think of $\mathcal{F}'$ as being the set of uncovered clusters besides C_v:

Proposition B.2.11. *Let $C, C' \in \mathcal{F}$ and $\mathcal{F}' \subseteq F$, with $C, C' \notin \mathcal{F}'$. Then*

$$(1-1/(\gamma-4)) \max_{y \in C} \Pr_y[\textit{hit } C' \textit{ before any cluster in } \mathcal{F}'] \leq \min_{y \in C} \Pr_y[\textit{hit } C' \textit{ before any cluster in } \mathcal{F}']$$

Proof. Let $y_0 \in C$ be the maximizer of $\Pr_y[\text{hit } C' \text{ before any cluster in } \mathcal{F}']$. By Lemmas B.1.1 and 4.8.6, $\Pr_y[\text{hit } C' \text{ before } y_0] \leq \frac{R}{(\gamma-4)R} = \frac{1}{\gamma-4}$ for any $y \in C$. Furthermore,

$$\begin{aligned}
\Pr_y[\text{hit } C' \text{ before any cluster in } \mathcal{F}'] &\geq \Pr_y[(\text{hit } C' \text{ before any cluster in } \mathcal{F}') \wedge (\text{hit } y_0 \text{ before } C')] \\
&= \Pr_y[\text{hit } C' \text{ before any cluster in } \mathcal{F}' | \text{hit } y_0 \text{ before } C'] \\
&\quad \Pr_y[\text{hit } y_0 \text{ before } C'] \\
&= \Pr_{y_0}[\text{hit } C' \text{ before any cluster in } \mathcal{F}'] \Pr_y[\text{hit } y_0 \text{ before } C'] \\
&\geq \Pr_{y_0}[\text{hit } C' \text{ before any cluster in } \mathcal{F}'](1 - 1/(\gamma - 4))
\end{aligned}$$

as desired. □

Line 21 always adds an edge

Here, we assume that the pair of sequences $(s_i)_i, (s'_j)_j$ is fresh in light of Proposition B.2.10. Subject to this assumption, we examine the effect of Lines 16 and 19 on $\mathcal{E}$. Specifically, we show that passing $\mathcal{E}$ through these lines does not make it empty in a sequence of four propositions.

Proposition B.2.12. *Let $\mathcal{E}_1$ be the multiset of clusters $A \in \mathcal{F}$ for which Property (1) on Line 16 is satisfied. Then $|\mathcal{E}_1| \geq \zeta - 1$.*

Proof. The number of clusters visited immediately after each of the last ζ visits to C_v is at least $\zeta - 1$, since all prior visits to C_v were followed immediately by a visit to another cluster. Therefore, $|\mathcal{E}_1| \geq \zeta - 1$, as desired. □

We use the following supermartingale concentration inequality to bound the drop from $\mathcal{E}_1$ to $\mathcal{E}_2$:

Theorem B.2.13 (Theorem 29 in [24] with $\phi_i = 0$ for all i and $M = 0$)**.** *Suppose a supermartingale X associated with filter $\boldsymbol{F}$ satisfies, for all $1 \leq i \leq n$,*

$$\boldsymbol{Var}(X_i|\mathcal{F}_{i-1}) \leq \sigma_i^2$$

and

$$\boldsymbol{E}[X_i|\mathcal{F}_{i-1}] \leq a_i$$

Then

$$\Pr[X_n \leq X_0 - \lambda] \leq e^{\frac{-\lambda^2}{2\sum_{i=1}^n (\sigma_i^2 + a_i^2)}}$$

Proposition B.2.14. *Let $\mathcal{E}_2$ be the submultiset of $\mathcal{E}_1$ for which Property (2) on Line 16 is satisfied. Suppose that $|\mathcal{E}_1| > 256 \log(m\alpha)$. Then $|\mathcal{E}_2| \geq |\mathcal{E}_1|/4$ with probability at least $1 - 1/(m\alpha)^3$.*

Proof. Let $A_1, A_2, \ldots, A_{|\mathcal{E}_1|}$ be the (possibly nondistinct) clusters in $\mathcal{E}_1$ listed in visitation order. For each $i \in \{1, 2, \ldots, |E_1|\}$, let X_i be the indicator variable of the event

$$\{B_{A_i} \text{ is visited between } A_i \text{ and the next visit to } C_v\}$$

and let $Z_i = (\sum_{j \leq i} X_i) - i/2$. By Proposition B.2.7 applied to $A \leftarrow A_i$, $\mathbf{E}[X_i] \geq 1/2$ for all i. This means that $\{Z_i\}$ is a supermartingale with stepwise variance 1 and change in mean at most 1. By Theorem B.2.13,

$$\Pr[Z_i = Z_i - Z_0 \leq -i/4] \leq \exp(-(i/4)^2/(4i)) = \exp(-i/64)$$

Since $|\mathcal{E}_1| > 256 \log(m\alpha)$, at least $|\mathcal{E}_1|/4$ A_is visit B_{A_i} before returning to C_v with probability at least $1 - 1/(m\alpha)^4$. Union bounding over all C_vs gives the desired result that $|\mathcal{E}_2| \geq |\mathcal{E}_1|/4$ with high probability. □

Proposition B.2.15. *Let $\mathcal{E}_3$ be the submultiset of $\mathcal{E}_2$ for which Property (3) on Line 16 is satisfied. Then $|\mathcal{E}_3| \geq |\mathcal{E}_2| - 10(\log^2(m\alpha))$ with probability at least $1 - 1/(m\alpha)^4$.*

Proof. Suppose that

$$\Pr[\text{ all roots for clusters in } \mathcal{E}_2 \text{ hit } C_v \text{ before another uncovered cluster }] \geq 1/(m\alpha)^3$$

Let x_i be a random variable denoting the first vertex that the random walk visits in B_{A_i}. Then

$$\begin{aligned}&\Pr[\text{ all roots for clusters in } \mathcal{E}_2 \text{ hit } C_v \text{ before another uncovered cluster }]\\ &= \Pr[\wedge_{i:A_i\in\mathcal{E}_2}\text{random walk starting at } x_i \text{ hits } C_v \text{ before another uncovered cluster}]\end{aligned}$$

By the Markov property applied after writing the above wedge as a product of conditional probabilities,

$$\begin{aligned}&\prod_{i:A_i\in\mathcal{E}_2} \max_{y\in B_{A_i}} \Pr_y[\text{hits } C_v \text{ before another uncovered cluster}]\\ &\geq \Pr[\text{ all clusters in } \mathcal{E}_2 \text{ hit } C_v \text{ before another uncovered cluster }]\\ &\geq 1/(m\alpha)^4\end{aligned}$$

Therefore, for all but $10\log^2(m\alpha)$ of the A_is,

$$\max_{y\in B_{A_i}} \Pr_y[\text{hits } C_v \text{ before another uncovered cluster}] \geq 1 - 1/(4\log n)$$

For each of these A_is, apply Proposition B.2.11 with $C \leftarrow B_{A_i}$, $C' \leftarrow C_v$, and $\mathcal{F}' \leftarrow \{\text{uncovered clusters besides } C_v\}$ to show that

$$\min_{y\in B_{A_i}} \Pr_y[\text{hits } C_v \text{ before another uncovered cluster}] \geq (1 - 1/(4\log n))(1 - 1/(\gamma - 4)) \geq 1 - \beta$$

Therefore, each of these A_is is also in $\mathcal{E}_3$. As a result, $|\mathcal{E}_3| \geq |\mathcal{E}_2| - 10(\log^2(m\alpha))$ if

$$\Pr[\text{ all roots for clusters in } \mathcal{E}_2 \text{ hit } C_v \text{ before another uncovered cluster }] \geq 1/(m\alpha)^4$$

The other case happens with probability at most $1/(m\alpha)^4$ for each C_v. Union bounding over all clusters in $\mathcal{F}$ gives the desired result. □

Proposition B.2.16. *Let $\mathcal{E}_{final}$ be the submultiset of $\mathcal{E}_3$ that remains in $\mathcal{E}$ after Line 19. Suppose that the pair $(s_i)_i, (s'_j)_j$ is fresh. Then $|\mathcal{E}_{final}| \geq |\mathcal{E}_3| - \tau(\log n)$.*

Proof. We start by noticing that no arborescence in T can contain more than τ clusters (possibly repeated) in the multiset $\mathcal{E}_3$. Otherwise, by Property (2), C_v would alternate with the root of that arborescence more than τ times, which contradicts the freshness of the pair $(s_i)_i, (s'_j)_j$. Therefore, each iteration of the while loop on Line 19 can only remove τ elements from $\mathcal{E}_3$. Each iteration reduces the size of the arborescence formed by joining $\mathcal{E}$'s arborescences to C_v by at least a factor of 2. Therefore, the while loop can only execute for $\log n$ iterations. Therefore, $|\mathcal{E}_{final}| \geq |\mathcal{E}_3| - \tau(\log n)$, as desired. □

We now combine all of these propositions into one key observation:

Corollary B.2.17. $\mathcal{E}_{final} \neq \emptyset$.

Proof. Since $\zeta \geq 100 \log^2(m\alpha)$, the size reductions due to Propositions B.2.12, B.2.14, B.2.15, and B.2.16 ensure that $|\mathcal{E}_{final}| > 0$, as desired. □

As a result, `InterclusterEdges` adds an edge to T every time the if statement on Line 15 is triggered. Therefore, the ζ elements that are removed from $(s_i)_i$ can be charged to the addition of one edge to T. By Invariant B.2.4, T can only have $|\mathcal{F}| - 1$ edges, which means that only $(|\mathcal{F}| - 1)\zeta$ letters can be removed from $(s_i)_i$ over the course of the algorithm.

Tying the parts together

The upside to removing elements of $(s_i)_i$ is that the resulting sequence does not contain more than ζ consecutive occurrences of any cluster. Since $(s_i)_i, (s'_j)_j$ is a fresh pair, $(s_i)_i$ with all consecutive duplicates removed is a $(|\mathcal{F}|, \tau)$-Davenport-Schinzel string. Therefore, its length can be bounded using Theorem B.2.3. Since each edge in a random spanning tree can be charged to a specific visit in s_i or a visit removed from s_i, we are done.

Proof of Lemma 4.5.4. We start with the high probability regime in which all of the above propositions hold. By Theorem 4.3.8,

$$\sum_{e \in E(C,C'), C \neq C' \in \mathcal{F}} \frac{\texttt{Reff}_H(e)}{r_e^H} = \mathbf{E}[\text{random spanning tree edges in } H \text{ between clusters in } \mathcal{F}]$$

By Theorem 4.2.1, the number of spanning tree edges between clusters in $\mathcal{F}$ is at most the number of clusters appended to the sequence $(s_i)_i$. The number of clusters appended to $(s_i)_i$ is equal to the number removed over the course of the algorithm plus the number left at the end of the algorithm. We bound these two quantities separately:

Clusters removed from $(s_i)_i$. Only Line 23 removes elements from $(s_i)_i$. By Corollary B.2.17, every ζ deletions from $(s_i)_i$ result in the addition of at least one edge to T. By

Invariant B.2.4, only $|\mathcal{F}|-1$ edges can be added to T, so at most $\zeta|\mathcal{F}|$ elements of $(s_i)_i$ are removed over the course of the algorithm.

Clusters remaning in $(s_i)_i$ at the end. At the end of `InterclusterEdges`, no cluster appears more than ζ times consecutively in $(s_i)_i$ by the if condition on Line 15. Therefore, the sequence $(s_i'')_i$ obtained by removing consecutive duplicates of clusters in $(s_i)_i$ is at most ζ times shorter than $(s_i)_i$. Furthermore, if $(s_i)_i, (s_j')_j$ is a fresh pair (which by Proposition B.2.10 occurs with high probability), $(s_i'')_i$ is an $(|\mathcal{F}|,\tau)$-Davenport-Schinzel string. Therefore, by Theorem B.2.3, the length of $(s_i'')_i$ is at most $|\mathcal{F}|C_0(m,\tau) \le |\mathcal{F}|(\mu_{\text{app}}/(2\zeta))$. Therefore, the length of $(s_i)_i$ at the end of the algorithm is at most $\zeta|\mathcal{F}|m^{o(1)}$.

Combining the parts. Therefore, with probablity at least $1-1/(m\alpha)^2$, the number of random spanning tree edges in H between clusters in $\mathcal{F}$ is at most

$$\zeta|\mathcal{F}|(\mu_{\text{app}}/(2\zeta)) + \zeta|\mathcal{F}|$$

The maximum number of edges that can be added is at most n^2, as there are at most n vertices in H. Therefore, the expected number of edges is at most

$$\zeta|\mathcal{F}|(\mu_{\text{app}}/(2\zeta)) + \zeta|\mathcal{F}| + n^2/(m\alpha)^2 \le \mu_{\text{app}}|\mathcal{F}|$$

as desired. □

B.2.4 Proof of Proposition 4.5.7

Proposition 4.5.7. *Consider a γ-well-separated R-family $\mathcal{F}$ in $Y \subseteq X \subseteq V(H_{\mathcal{C}})$ and let $I := \textit{Schur}(H_{\mathcal{C}}, \cup_{C\in\mathcal{F}}C)$. Suppose that*

$$c^{\textit{Schur}(H_{\mathcal{C}}, Y\cup(V(H)\setminus X))}(E(Y, V(H)\setminus X)) \le \xi$$

For any $C \in \mathcal{F}$, let

$$\Delta_{\mathcal{F}}(C) := \sum_{e\in E_I(C,C'), C'\neq C\in\mathcal{F}} \frac{\textit{Reff}_I(e)}{r_e^I}$$

Let $\mathcal{F}' := \mathcal{F}\cup\{V(H)\setminus X\}$ and consider any clusters S_C with $S_{\mathcal{F}'}(p,C) \subseteq S_C$ for all $C \in \mathcal{F}$. Then

$$\sum_{C\in\mathcal{F}} c^{\mathcal{C}}(S_C) \le \left(\sum_{C\in\mathcal{F}} \frac{\Delta_{\mathcal{F}}(C)}{p(\gamma-4)R}\right) + \frac{\xi}{p}$$

Proof. Let $J := \texttt{Schur}(H_{\mathcal{C}}, (V(H_{\mathcal{C}})\setminus X)\cup(\cup_{C\in\mathcal{F}}C))$. By Lemma 4.8.3 with $I \leftarrow H_{\mathcal{C}}$, $S_0 \leftarrow Y$, $S_0' \leftarrow \cup_{C\in\mathcal{F}}C$, and $S_1 \leftarrow V(H_{\mathcal{C}})\setminus X$, and $S_2 \leftarrow \emptyset$,

$$\sum_{C\in\mathcal{F}} c^J(E_J(C, (V(H_{\mathcal{C}})\setminus X))) \le \xi$$

By Lemma B.1.2 with $I \leftarrow H_{\mathcal{C}}$, $S_0 \leftarrow C$, $S_1 \leftarrow C'$, $S_2 \leftarrow (V(H_{\mathcal{C}}) \setminus X) \cup (\cup_{C'' \in \mathcal{F}, C'' \neq C, C'} C'')$, and $S_2' \leftarrow S_2 \setminus (V(H_{\mathcal{C}}) \setminus X)$ and Lemma 4.8.6, for any $C \in \mathcal{F}$

$$\begin{aligned}\sum_{C' \in \mathcal{F} \setminus \{C\}} c^J(E_J(C, C')) &\leq \sum_{C' \in \mathcal{F} \setminus \{C\}} c^I(E_I(C, C')) \\ &= \sum_{e \in E_I(C, C')} c_e^I \\ &\leq \frac{1}{(\gamma - 4)R} \sum_{e \in E_I(C, C')} \texttt{Reff}_I(e) c_e^I \\ &= \frac{\Delta_{\mathcal{F}}(C)}{(\gamma - 4)R}\end{aligned}$$

By definition of S_C, $S_{\mathcal{F}'}(p, C) \subseteq S_C$ for all $C \in \mathcal{F}$. Therefore, by Lemma 4.8.5,

$$\begin{aligned}\sum_{C \in \mathcal{F}} c^{\mathcal{C}}(S_C) &\leq \sum_{C \in \mathcal{F}} c^{\mathcal{C}}(S_{\mathcal{F}'}(p, C)) \\ &\leq \sum_{C \in \mathcal{F}} \frac{c^J(E_J(C, \cup_{C' \in \mathcal{F}', C' \neq C} C'))}{p}\end{aligned}$$

Substitution shows that

$$\begin{aligned}\sum_{C \in \mathcal{F}} \frac{c^J(E_J(C, \cup_{C' \in \mathcal{F}', C' \neq C} C'))}{p} &= \sum_{C \in \mathcal{F}} \frac{c^J(E_J(C, \cup_{C' \in \mathcal{F}, C' \neq C} C'))}{p} + \sum_{C \in \mathcal{F}} \frac{c^J(E_J(C, V(H_{\mathcal{C}}) \setminus X))}{p} \\ &\leq \left(\sum_{C \in \mathcal{F}} \frac{\Delta_{\mathcal{F}}(C)}{p(\gamma - 4)R} \right) + \frac{\xi}{p}\end{aligned}$$

as desired. □

B.2.5 Proof of Proposition 4.5.8

Proposition 4.5.8. *Consider a family $\mathcal{F}$ in a graph H. Let $C \in \mathcal{F}$ be a cluster with H-effective resistance diameter R. Consider some S_C for which $C \subseteq S_C \subseteq S_{\mathcal{F}}(p, C)$ for any $p \in (0, 1/2)$. Consider a cluster C' that is tied to C. Then $C' \subseteq S_{\mathcal{F}}(p + 3/10, C)$.*

Proof. Consider a vertex $x \in C' \cap S_C$. This vertex exists by the "Intersection" condition of ties. Let J be the graph obtained by identifying C to a vertex s and all vertices in clusters of $\mathcal{F} \setminus \{C\}$ to a vertex t. Then $b_{st}^T L_J^+ b_{sx} \leq p b_{st}^T L_J^+ b_{st}$ since $S_C \subseteq S_{\mathcal{F}}(p, C)$.

Let y be any vertex in C'. Let $H_0 = \texttt{Schur}(H, (\cup_{C'' \in \mathcal{F}} C'') \cup \{x, y\})$ and let H_1 be the graph obtained by identifying C to s. By Lemma 4.8.6 and the "Well-Separatedness" condition of ties, $b_{sx}^T L_{H_1}^+ b_{sx} \geq \beta_0 R = 100R$ and $b_{sy}^T L_{H_1}^+ b_{sy} \geq 90R$. This means that $r_{sx}^{H_1} \geq 100R$ and that $r_{sy}^{H_1} \geq 90R$.

Let $J_0 = \texttt{Schur}(J, \{s, x, y, t\})$. We now reason about the $s - t$ electrical flow on the edges $\{s, x\}$ and $\{x, y\}$. Since J is obtained by identifying $\mathcal{F} \setminus \{C\}$ to t, the $\{s, x\}$ and $\{s, y\}$ conductances are uneffected in obtaining J_0 from H_1, so $r_{sx}^{J_0} = r_{sx}^{H_1} \geq 100R$ and $r_{sy}^{J_0} = r_{sy}^{H_1} \geq 90R$. Let $f \in \mathbb{R}^{E(J_0)}$ be the $s - t$ unit electrical flow vector for J_0. Since flow times resistance is the potential drop,

$$f_{sx} = \frac{b_{st}^T L_{J_0}^+ b_{sx}}{r_{sx}^{J_0}} \leq \frac{b_{st}^T L_{J_0}^+ b_{st}}{100R}$$

By Rayleigh monotonicity, $b_{xy}^T L_{J_0}^+ b_{xy} \leq 10R$. Therefore, either $r_{xy}^{J_0} \leq 30R$ or $r_{xt}^{J_0} + r_{ty}^{J_0} \leq 30R$ since $r_{sx}^{J_0}, r_{sy}^{J_0} \geq 90R$. We start by showing that the latter case does not happen. In the latter case, the potential drop from s to t is

$$\begin{aligned} b_{st}^T L_J^+ b_{st} &= b_{st}^T L_{J_0}^+ b_{st} \\ &\leq b_{st}^T L_{J_0}^+ b_{sx} + r_{xt}^{J_0} f_{xt} \\ &\leq p b_{st}^T L_J^+ b_{st} + r_{xt}^{J_0} f_{sx} \\ &\leq (1/2 + 3/10) b_{st}^T L_J^+ b_{st} \\ &< b_{st}^T L_J^+ b_{st} \end{aligned}$$

which is a contradiction. Therefore, $r_{xy}^{J_0} \leq 30R$. In this case, if y's potential is less than x's, we are done. Otherwise, by flow conservation,

$$f_{xy} \leq f_{sx} \leq \frac{b_{st}^T L_{J_0}^+ b_{st}}{100R}$$

and the $x - y$ potential drop is therefore at most

$$r_{xy}^{J_0} f_{xy} \leq (30R) \frac{b_{st}^T L_{J_0}^+ b_{st}}{100R} \leq (3/10) b_{st}^T L_{J_0}^+ b_{st}$$

In particular, y's normalized potential is at most 3/10 greater than x's, as desired. □

B.3 Deferred proofs for Section 4.6

B.3.1 Bound on the number of random walk steps before covering a neighborhood

We prove Lemma 4.2.3 in this section. We exploit the Matthews trick in this proof [75]:

Lemma 4.2.3 (Key result for bounding the number of shortcutter uses)**.** *Consider an arbitrary vertex u_0 in a graph I, an edge $\{u, v\} = f \in E(I)$, and an $R \geq 0$. Let $\mathcal{B}(u, R) \subseteq V(I)$ denote the set of vertices in I with I-effective resistance distance at most R from u. The expected number of times that the random walk starting at u_0 traverses f from $u \to v$ before all vertices in $\mathcal{B}(u, R)$ have been visited is at most $\tilde{O}(c_f R)$, where c_f is the conductance of the edge f.*

Proof of Lemma 4.2.3. Let $B := \mathcal{B}(u, R)$. For any two vertices $x, y \in B$, the random walk from x to y in I traverses f from u to v at most $\texttt{Reff}_I(x, y)c_f^I$ times in expectation by Theorem 4.3.2. By the triangle inequality, this is at most $2Rc_f^I$.

Pick a random bijection $\pi : \{1, 2, \ldots, |B| - 1\} \to B \setminus \{u\}$ of the vertices in $B \setminus \{u\}$. All traversals across f from $u \to v$ within distance R of S occur between the first visit to u and the last first visit to any vertex in B. Let τ_i be the random variable denoting the number of $u \to v$ f traversals before the last first visit to $\{\pi_1, \ldots, \pi_i\}$. Let τ_0 be the the first visit to u. Notice that for all $i > 0$,

$$\begin{aligned}
\mathbf{E}[\tau_i - \tau_{i-1}] &\leq \Pr_\pi[t_{\pi_i} < t_{\{\pi_{i-1}, \ldots, \pi_1\}}] \max_{x,y \in B} \mathbf{E}[\ f \text{ traversals from } x \to y\ ] \\
&\leq \Pr_\pi[t_{\pi_i} < t_{\{\pi_{i-1}, \ldots, \pi_1\}}] 2Rc_f^I \\
&\leq \frac{1}{i} 2Rc_f^I
\end{aligned}$$

Summing over all i shows that

$$\mathbf{E}[K] = \mathbf{E}[\tau_i] = \tau_0 + \sum_{i=1}^{|B|-1} \mathbf{E}[\tau_i - \tau_{i-1}] \leq 0 + O(\log n) Rc_f^I$$

as desired. □

B.4 Deferred proofs for Sections 4.7 and 4.8

B.4.1 Ball splitting after deletions

We now prove the following:

Lemma 4.7.2. *Consider a graph H and a set of clusters $\mathcal{D}$, each with effective resistance diameter at most R. Let F be a set of edges in H. Then there is a set of clusters $\mathcal{D}'$ with the following properties:*

- *(Covering) Each vertex in a cluster of $\mathcal{D}$ is in a cluster of $\mathcal{D}'$.*

- *(Diameter) The effective resistance diameter of each cluster in $\mathcal{D}'$ is at most $\mu_{app}R$ in the graph $H \setminus F$.*
- *(Number of clusters) $|\mathcal{D}'| \leq \mu_{app}(|\mathcal{D}| + |F|)$.*

Our proof uses Lemma 4.5.4 to reduce to the case where both (1) $\mathcal{D}$ just consists of one cluster C and (2) $F = (\partial C) \cup F'$, where $F' \subseteq E(C)$. We deal with this case using a sparsification technique that peels many low-stretch spanners off of the graph to show that the effective resistance between two particular vertices is low [53].

The case where $\mathcal{D}$ just contains one cluster and $F = (\partial C) \cup F'$

Before proving this result, we discuss low-stretch sparse spanners [7]:

Theorem B.4.1 ([7])**.** *Any unweighted graph G has a subgraph H with the following two properties:*

- *(Distortion) For any two vertices $u, v \in V(G)$, $d_H(u, v) \leq 2(\log n)d_G(u, v)$, where d_I is the shortest path metric on the vertices of the graph I.*
- *(Sparsity) H has at most $100n \log n$ edges.*

We use this as a tool to sparsify G in a way that ignores the deleted edges in the deleted set F.

Proposition B.4.2. *Consider a resistance-weighted graph G. Let C be a cluster with effective resistance diameter R in G. Let $F \subseteq E(G)$ and suppose that $k = |F|$. Furthermore, suppose that F is the union of ∂C and a set of edges F' with both endpoints in C.*

Let $H = G \setminus F$. Then C is the union of at most $O(k \log^6 n)$ clusters in H that have effective resistance diameter $O(R \log^6 n)$.

Proof of Proposition B.4.2. The following iterative algorithm, `PartitionBall`, partitions C

into the desired clusters.

Algorithm 36: `PartitionBall`(G, C, F), never executed

Data: A graph G, a cluster C, and a set of edges F to delete from G
Result: A partition $C_1 \cup \ldots \cup C_\tau = C$

$H \leftarrow G \backslash F$
// set of current centers
$K \leftarrow V(C)$
$i \leftarrow 0$
while $|K| \geq 31k \log n$ **do**
  $I_i \leftarrow \texttt{Schur}(G, V(F) \cup K)$
  $K' \leftarrow K \setminus V(F)$
  foreach j *from 0 to* $2 \log n$ **do**
    $B_j \leftarrow$ the set of edges $e \in E(I_i) \setminus F$ with $2^j \texttt{Reff}_{I_i}(e) \leq r_e^{I_i} \leq 2^{j+1} \texttt{Reff}_{I_i}(e)$
  end
  $j^* \leftarrow$ the index $j \in \{0, 1, \ldots, 2 \log n\}$ for which the number of vertices incident with leverage score at least $1/(2 \log n)$ in B_j is maximized
  $I_i' \leftarrow I_i[B_{j^*}]$
  foreach *of* $\lfloor 2^{j^*}/(64 \log^3 n) \rfloor$ *rounds* **do**
    $J \leftarrow$ a $2(\log n) - 1$ spanner of I_i' as per Theorem B.4.1
    $I_i' \leftarrow I_i' \setminus E(J)$
  end
  $K \leftarrow$ the minimum dominating set of the unweighted version of I_i'
  $i \leftarrow i + 1$
end
return Voronoi diagram of C with respect to K in the metric $\texttt{Reff}_{G \setminus F}$

Now, we analyze this algorithm. We start by showing that the while loop only executes $O(\log^2 n)$ times. Start by noticing that $|K'| \geq 30k \log n \geq |K|/2$ because $|V(F)| \leq 2k$ and $|K| \geq 31k \log n$.

For an edge $e \in I_i$, let $z_e = \texttt{lev}_{I_i}(e)$. For a set $S \subseteq E(I_i)$, let $z(S) = \sum_{e \in S} z_e$. The total z-weight of edges incident with each vertex in I_i is at least 1 since the total leverage score incident with a vertex in any graph is at least 1. Since there are only $2 \log n$ B_js, at least one contributes $1/(2 \log n)$ to the z-weight incident with each vertex. Therefore, the number of vertices in K' with more than $1/(2 \log n)$ incident z-weight in I_i' is at least $|K'|/(2 \log n) \geq (|K| + k)/(2 \log n)$ before the second ForEach loop. Therefore, by the bucketing resistance lower bound, $|E(I_i')| \geq (|K| + k) 2^{j^*}/(2 \log n)$ before the second ForEach loop.

The spanners created during the second ForEach loop delete at most $(2^{j^*}/(64 \log^3 n))(2(|K| + k) \log n) = (|K| + k) 2^{j*}/(32 \log^2 n)$ edges from I_i'. Each of the $(|K| + k)/(2 \log n)$ high z-weight vertices is incident with at least $2^{j^*}/(2 \log n)$ edges in B_{j^*}. Each edge is incident with two vertices. Notice that the number of vertices that can have all of their incident edges deleted is at most

$$\frac{2((|K|+k)2^{j*}/(32\log^2 n))}{2^{j*}/(2\log n)} = (|K|+k)/(8\log n)$$

so the number of vertices with an incident edge after the second ForEach loop is at least $(|K|+k)/(4\log n)$. The spanners are edge-disjoint, so together they contain at least $2^{j*}/(64\log^3 n)$ disjoint paths with at most $2\log n$ edges between the endpoints of each edge of I'_i. In particular, each path has resistance length at most $(2\log n)2^{j*+1}R$ by the bucketing resistance upper bound. Therefore, the effective resistance between the endpoints of any edge left over in $I'_i \backslash F$ is at most $256(\log^4 n)R$.

Since the while loop only executes when $|K| \geq 31k\log n$, the number of vertices in K' with an incident edge in I'_i after the second ForEach loop is at least $8k$, for a total of at least $4k$ disjoint pairs. Removing one side of each K pair certifies that there is a dominating set with size at most $|K| - (|K|+k)/(8\log n) + 2k \leq (1 - 1/(16\log n))|K|$. Therefore, the algorithm will only take $16\log^2 n$ iterations.

At the end of those iterations, for every vertex in C there is a connected sequence of $16\log^2 n$ pairs to some vertex in K with each pair having $G\backslash F$ effective resistance distance at most $256(\log^4 n)R$. By the triangle inequality, this means that every vertex in C is within $G\backslash F$ distance $O(R\log^6 n)$ of some vertex in K at the end of the algorithm. This is the desired result. □

Sparsification

We now prove a version of spectral sparsification that allows a large part of the graph to not be changed. We are sure that this is known, but provide a proof for completeness.

Proposition B.4.3 (Subset sparsification)**.** *Let G be a weighted graph and let $F \subseteq E(G)$. Let $\epsilon \in (0,1)$. Then, there exists a graph H with $V(H) = V(G)$ and the following additional properties:*

- *(Only F modified) The weights of edges in $G \setminus F$ are not modified.*
- *(Sparsity of F) $|E(H) \cap F| \leq (8(\log n)/\epsilon^2)(1 + \sum_{e\in F} \texttt{lev}_G(e))$*
- *(Spectral approximation) For any vector $d \in \mathbb{R}^n$,*

$$(1-\epsilon)d^T L_G^+ d \leq d^T L_H^+ d \leq (1+\epsilon)d^T L_G^+ d$$

The proof is a simple application of matrix Chernoff bounds, which we state here:

Theorem B.4.4 (Theorem 1.1 in [94])**.** *Consider a finite sequence $\{X_k\}_k$ of independent, random, self-adjoint matrices with dimension n. Assume that each random matrix satisfies*

$$X_k \succeq 0$$

and

$$\lambda_{\max}(X_k) \leq R$$

Define

$$\mu_{\min} := \lambda_{\min}(\sum_k \boldsymbol{E}[X_k])$$

and

$$\mu_{\max} := \lambda_{\max}(\sum_k \boldsymbol{E}[X_k])$$

Then

$$\Pr[\lambda_{\min}(\sum_k X_k) \leq (1-\delta)\mu_{\min}] \leq n\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu_{\min}/R}$$

and

$$\Pr[\lambda_{\max}(\sum_k X_k) \leq (1+\delta)\mu_{\max}] \leq n\left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu_{\max}/R}$$

Proof of Proposition B.4.3. Let $q = 20(\log n)/\epsilon^2$. For each edge $e \in E(G)$, define q matrices $X_e^{(k)}$ for $k \in \{1, 2, \ldots q\}$, with

- $X_e^{(k)} = \frac{c_e^G}{\texttt{lev}_G(e)q}(L_G^+)^{1/2}b_e b_e^T(L_G^+)^{1/2}$ with probability $\texttt{lev}_G(e)$ and 0 otherwise for $e \in F$
- $X_e^{(k)} = \frac{c_e^G}{q}(L_G^+)^{1/2}b_e b_e^T(L_G^+)^{1/2}$ deterministically for $e \notin F$.

Notice that

$$\sum_{e\in E(G)}\sum_{k=1}^{q}\mathbf{E}[X_e^{(k)}] = I$$

so $\mu_{min} = \mu_{max} = 1$. Apply Theorem B.4.4 with $\delta = \epsilon$ and $R = 1/q$ (since $\lambda_{\max}(X_e^{(k)}) = c_e^G(b_e^T L_G^+ b_e)/(q\texttt{lev}_G(e)) = 1/q$ for all $e \in F$) to reason about the random matrix $M_H = \sum_{e\in E(G)}\sum_{k=1}^{q} X_e^{(k)}$ and $L_H = L_G^{1/2} M_H L_G^{1/2}$. We now analyze the guarantees one by one.

Only F modified. The contribution of edge edge $e \notin F$ to L_H is $c_e^G b_e b_e^T$, which is the same as its weight in G.

Sparsity of F. By standard Chernoff bounds on each edge, the total number of edges from F in H is at most $8(\log n)/\epsilon^2$.

Spectral approximation. By Theorem B.4.4,

$$\begin{aligned}\Pr[(1+\epsilon)I \succeq M_H \succeq (1-\epsilon)I] &\geq 1 - n\left(\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}}\right)^q - n\left(\frac{e^{\epsilon}}{(1+\epsilon)^{1+\epsilon}}\right)^q \\ &\geq 1 - n\left(1-\epsilon^2/4\right)^q - n\left(1-\epsilon^2/4\right)^q \\ &\geq 1 - 1/n^4\end{aligned}$$

By standard facts about the Loewner ordering, $(1+\epsilon)L_G \succeq L_H \succeq (1-\epsilon)L_G$ and $(1+\epsilon)L_G^+ \succeq L_H^+ \succeq (1-\epsilon)L_G^+$, as desired. □

Tying the parts together

Now, we prove Lemma 4.7.2. To do this, we exploit the algorithm `CoveringCommunity` to split the clusters of $\mathcal{D}$ into a small number of well-separated families. All vertices outside of these clusters can be Schur complemented out, as they are irrelevant. By Lemma 4.5.4, the intercluster edges have small total leverage score. Proposition B.4.3 allows us to replace these intercluster edges with $m^{o(1)}|\mathcal{D}|$ reweighted edges that separate all of the clusters. Therefore, the clusters in each well-separated family can be handled independently using Proposition B.4.2.

Proof of Lemma 4.7.2. Start by applying Lemma 4.5.3 to produce a community $\mathcal{G} \leftarrow$ `CoveringCommunity`$(V(H), H, R)$. Form a new community $\mathcal{G}'$ by taking each cluster C in a family of $\mathcal{G}$ and replacing it with a cluster C' formed by adding all vertices in clusters of $\mathcal{D}$ that intersect C'. Furthermore, remove all clusters in $\mathcal{G}'$ that do not intersect a cluster in $\mathcal{D}$.

Each family in $\mathcal{G}'$ is $\gamma_{\text{ds}}/3$-well-separated after doing this by the "Well-separatedness" guarantee of `CoveringCommunity`, the "R-community" guarantee, and the fact that each cluster in $\mathcal{D}$ has effective resistance diameter at most R. Furthermore, by the "Covering" guarantee of $\mathcal{G}$, each cluster in $\mathcal{D}$ intersects some cluster in a family of $\mathcal{G}$. This means that each cluster in $\mathcal{D}$ is completely contained in some cluster in some family of $\mathcal{G}'$. Therefore, we can focus on each family of $\mathcal{G}'$ independently.

The above paragraph effectively reduced the problem to the case where $\mathcal{D}$ is a small number of well-separated families of clusters. Now, we exploit this. By Lemma 4.5.4, the total leverage score of the intercluster edges in a family is at most $O(C_0(\ell, \gamma_{\text{ds}})\ell) \leq m^{o(1)}\ell$, where ℓ is the number of clusters in all families in $\mathcal{G}'$. Since each family consists of clusters that contain at least one cluster in $\mathcal{D}$, $\ell \leq |\mathcal{G}||\mathcal{D}| \leq m^{o(1)}|\mathcal{D}|$. By Proposition B.4.3 applied to all intercluster edges, a graph H_i can be made for each family $\mathcal{G}'_i \in \mathcal{G}$ that spectrally approximates H and only has $m^{o(1)}|\mathcal{D}|$ intercluster edges. Add these intercluster edges to F to obtain F_i. Each of these edges is incident with at most two clusters in $\mathcal{G}_i$.

Apply Proposition B.4.2 to each cluster $C \in \mathcal{G}_i$ in the graph H_i with deleted edge set consisting of the edges of F_i incident with the cluster. This shows that C splits into

$\tilde{O}(|F_i \cap (C \cup \partial_{H_i} C)|)$ clusters with radius $\tilde{O}(R)$. Add these clusters to $\mathcal{D}'$. We now analyze the guarantees one-by-one:

Covering. Each vertex in a cluster in $\mathcal{D}$ is in some cluster of $\mathcal{G}$, which is in turn covered by the clusters produced using Proposition B.4.2, as desired.

Diameter. Since $\mathcal{G}$ is R-bounded, each cluster in $\mathcal{G}$ has H-effective resistance diameter at most $m^{o(1)}R$. By the "Spectral approximation" guarantee of Proposition B.4.3, each cluster in $\mathcal{G}$ has H_i-effective resistance diameter at most $(1+\epsilon)m^{o(1)}R = m^{o(1)}R$ as well. Proposition B.4.2 implies that after deleting the edges in F, all clusters added to $\mathcal{D}'$ have $H_i \setminus F = H \setminus F$-effective resistance diameter $O(\log n)m^{o(1)}R = m^{o(1)}$, as desired.

Number of clusters. Summing up $\tilde{O}(|F_i \cap (C \cup \partial_{H_i} C)|)$ over all clusters in $\mathcal{G}_i$ shows that applications of Proposition B.4.2 add $\tilde{O}(|F_i|) \leq m^{o(1)}(|\mathcal{D}| + |F|)$ clusters to $\mathcal{D}'$. Doing this for all $|\mathcal{G}'| \leq m^{o(1)}$ families in $\mathcal{G}$ yields the desired result. □

B.5 Deferred proofs for Section 4.10

B.5.1 Stable objective subresults

We also use the following folklore fact about leverage scores:

Remark 12. *In any graph G with n vertices,*

$$\sum_{e \in E(G)} \mathit{lev}_G(e) = n - 1$$

Preliminaries for first-order terms

Proposition B.5.1. *Consider a graph G, a vertex $x \in V(G)$, a set $Y \subseteq V(G)$ with $x \notin Y$, and $\gamma \in [0,1]$. Calculate electrical potentials with boundary conditions $p_x = 0$ and $p_w = 1$ for every $w \in Y$. Suppose that there is no edge $\{u,v\} \in E(G)$ with $p_u < \gamma$ and $p_v > \gamma$.*

Let U be the set of vertices u with $p_u \leq \gamma$. Make a new electrical flow with boundary conditions $q_u = 0$ for $u \in U$ and $q_w = 1$ for all $w \in Y$. Then for any $w \in Y$,

$$\sum_{w' \in N_G(w)} \frac{q_w - q_{w'}}{r_{ww'}} = \frac{1}{1-\gamma} \sum_{w' \in N_G(w)} \frac{p_w - p_{w'}}{r_{ww'}}$$

Proof. For any edge $e = (u,v)$ with $p_v \geq p_u$, let $f_e = \frac{p_v - p_u}{r_e}$,

$$g_e = \begin{cases} \frac{1}{1-\gamma} f_e & \text{if } p_u \geq \gamma \\ 0 & \text{if } p_v \leq \gamma \end{cases}$$

and

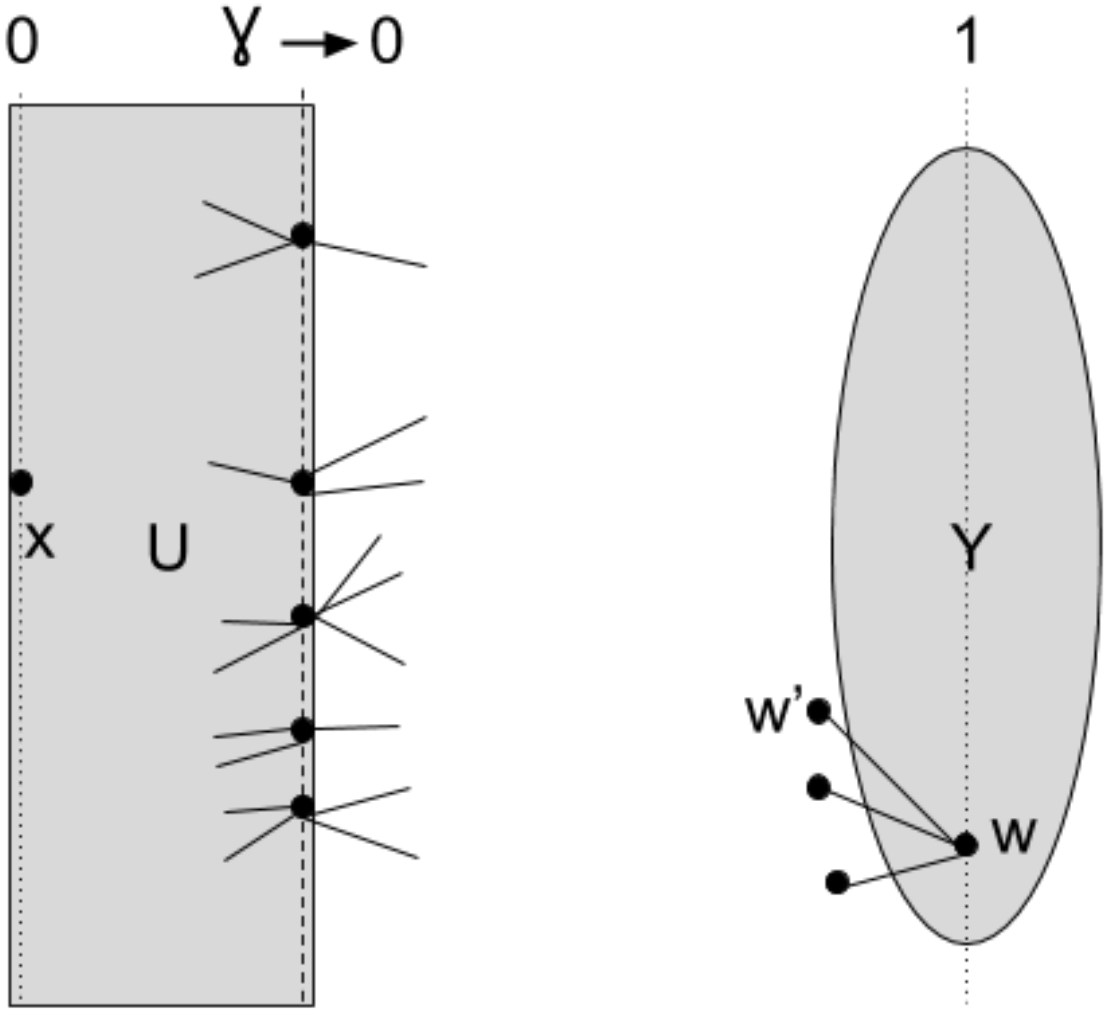

Figure B.5.1: Depiction of Proposition B.5.1. The proof follows from the observation that identifying the vertices of U to x does not change the $x-Y$ electrical flow (with y identified) on any edge outside of U.

$$q_w = \max(0, 1 - \frac{1 - p_w}{1 - \gamma})$$

The cases for g_e form a partition of the edges thanks to the condition that no edge can cross the γ potential threshold. One can check that g_e is an electrical flow with respect to the potentials q_w, $q_w = 1$ for any $w \in Y$, and $q_u = 0$ for all $u \in U$. Therefore, since no edge crosses the γ threshold,

$$\sum_{w' \in N_G(w)} \frac{q_w - q_{w'}}{r_{ww'}} = \frac{1}{1 - \gamma} \sum_{w' \in N_G(w)} \frac{p_w - p_{w'}}{r_{ww'}}$$

as desired. □

Proposition B.5.2. *Consider a vertex x and a set Y in a graph G. Let Z be a set of edges with both endpoints having electrical potential at most γ, with $p_x = 0$ and $p_y = 1$ in G/Y with identification y. Let G' be a graph obtained by deleting and contracting edges in Z in an arbitrary way. Let $H' = \texttt{Schur}(G', \{x\} \cup Y)$ and $H = \texttt{Schur}(G, \{x\} \cup Y)$. Then for any $w \in Y$,*

$$r_{xw}^{H'} \geq (1 - \gamma) r_{xw}^{H}$$

Proof. Let G_0 be a graph with every edge $\{u, v\}$ with $p_u < \gamma$ and $p_v > \gamma$ subdivided at potential γ. Let U be the set of vertices u with $p_u \leq \gamma$. Let $H_0 = \texttt{Schur}(G_0, Y \cup U)$ and $H_0' = \texttt{Schur}(G_0 \backslash Z, Y \cup U)$. Since $Z \subseteq E(G[U])$, $H_0' = H_0 \backslash Z$, so for any $w \in Y$,

$$\sum_{u \in U} \frac{1}{r_{wu}^{H_0'}} = \sum_{u \in U} \frac{1}{r_{wu}^{H_0}}$$

By Proposition B.5.1 applied to H_0, for any $w \in Y$,

$$\sum_{u \in U} \frac{1}{r_{wu}^{H_0}} = \frac{1}{1 - \gamma} \frac{1}{r_{wx}^{H}}$$

Obtain H' from H_0' by eliminating vertices of $U \backslash \{x\}$ one by one. Let $U_0 = U = \{u_0, u_1, \ldots, u_\ell, x\}$, $U_i = \{u_i, \ldots, u_\ell, x\}$, and H_i' be the graph obtained by eliminating $u_0, u_1, \ldots, u_{i-1}$. By the formula for elimination of one vertex,

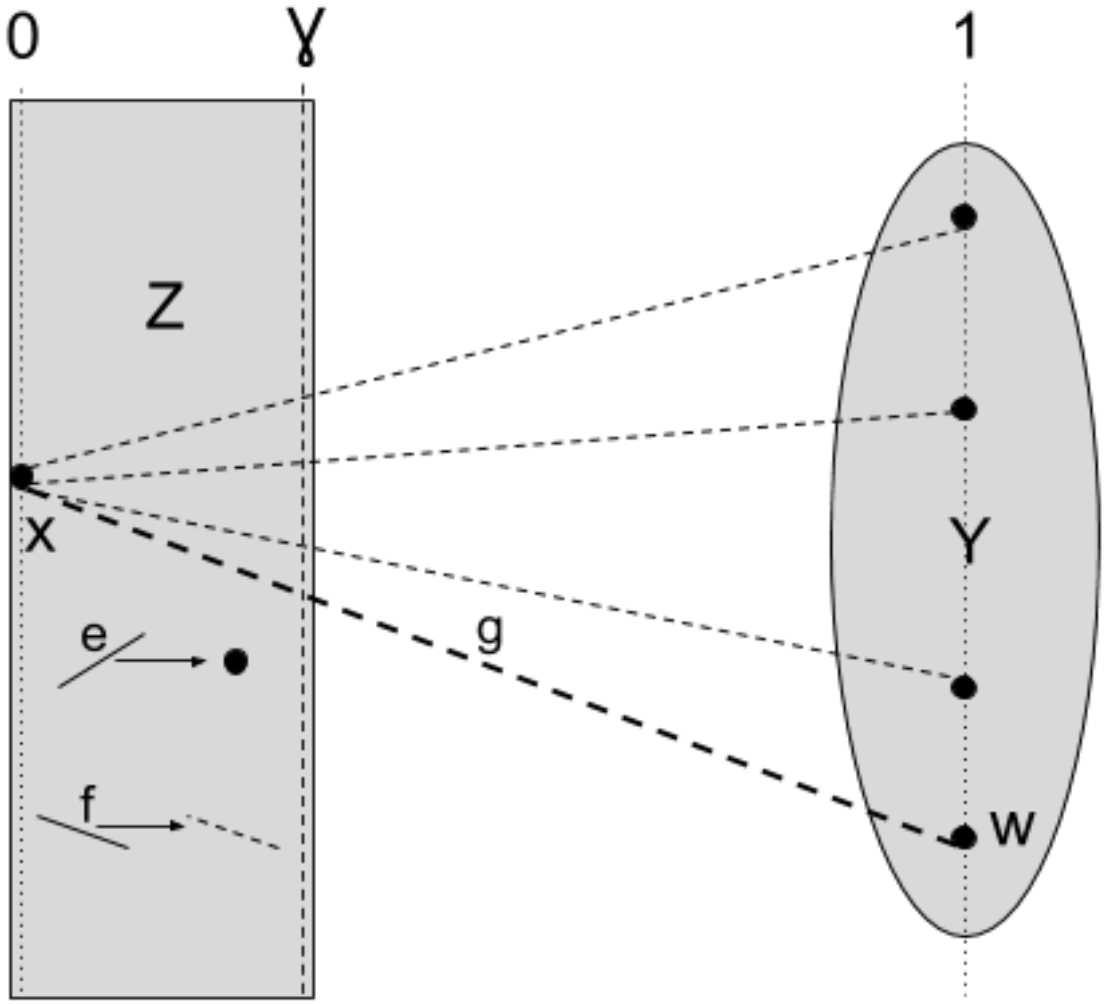

Figure B.5.2: Depiction of Proposition B.5.2. Contracting edges like e and deleting edges like f can only decrease resistances of edges like g by a small amount.

$$\begin{aligned}\sum_{u\in U_{i+1}} c_{wu}^{H'_{i+1}} &= \sum_{u\in U_{i+1}} \left(c_{wu}^{H'_i} + \frac{c_{wu_i}^{H'_i} c_{u_i u}^{H'_i}}{\sum_{v\in U_{i+1}\cup Y} c_{u_i v}^{H'_i}} \right) \\ &\leq c_{wu_i}^{H'_i} + \sum_{u\in U_{i+1}} c_{wu}^{H'_i} \\ &= \sum_{u\in U_i} c_{wu}^{H'_i}\end{aligned}$$

Chaining these inequalities shows that

$$c_{wx}^{H'} \leq \frac{1}{1-\gamma} c_{wx}^{H}$$

since $U_{\ell+1} = \{x\}$ and $H'_{\ell+1} = H'$. Taking reciprocals shows the desired result. □

Proposition B.5.3. *Consider two sets $X, Y \subseteq V(G)$ with $X \cap Y = \emptyset$. Let G_0 be a graph obtained by splitting each edge $e \in \partial_G X$ into a path of length 2 consisting of edges e_1 and e_2 with $r_{e_1} = r_{e_2} = r_e/2$, with e_1 having an endpoint in X. Add X and all endpoints of edges e_1 to a set X_0. Let x and x_0 denote the identifications of X and X_0 in G/X and G_0/X_0 respectively. Then*

$$\Delta^{G_0}(X_0, Y) \leq 2\Delta^G(X, Y)$$

Proof. Recall that

$$\Delta^{G_0}(X_0, Y) = \sum_{y_i\in Y} b_{x_0 y_i}^T L_{G_0/X_0}^+ b_{x_0 y_i} c_{x_0 y_i}^{H_0}$$

where $H_0 = \texttt{Schur}(G_0, \{x_0\}\cup Y)$. By Rayleigh monotonocity and $X \subseteq X_0$, $b_{x_0 y_i}^T L_{G_0/X_0}^+ b_{x_0 y_i} \leq b_{xy_i}^T L_{G/X}^+ b_{xy_i}$. By Proposition B.5.2 and the fact that all vertices in X_0 have normalized potential at most $1/2$ in G with $p_x = 0$ and $p_w = 1$ for all $w \in Y$,

$$c_{x_0 y_i}^{H_0} \leq 2 c_{xy_i}^{H}$$

where $H = \texttt{Schur}(G, \{x\} \cup Y)$. Therefore,

$$\Delta^{G_0}(X_0, Y) \leq 2 \sum_{y_i\in Y} (b_{xy_i}^T L_{G/X}^+ b_{xy_i}) c_{xy_i}^H = 2\Delta^G(X, Y)$$

as desired. □

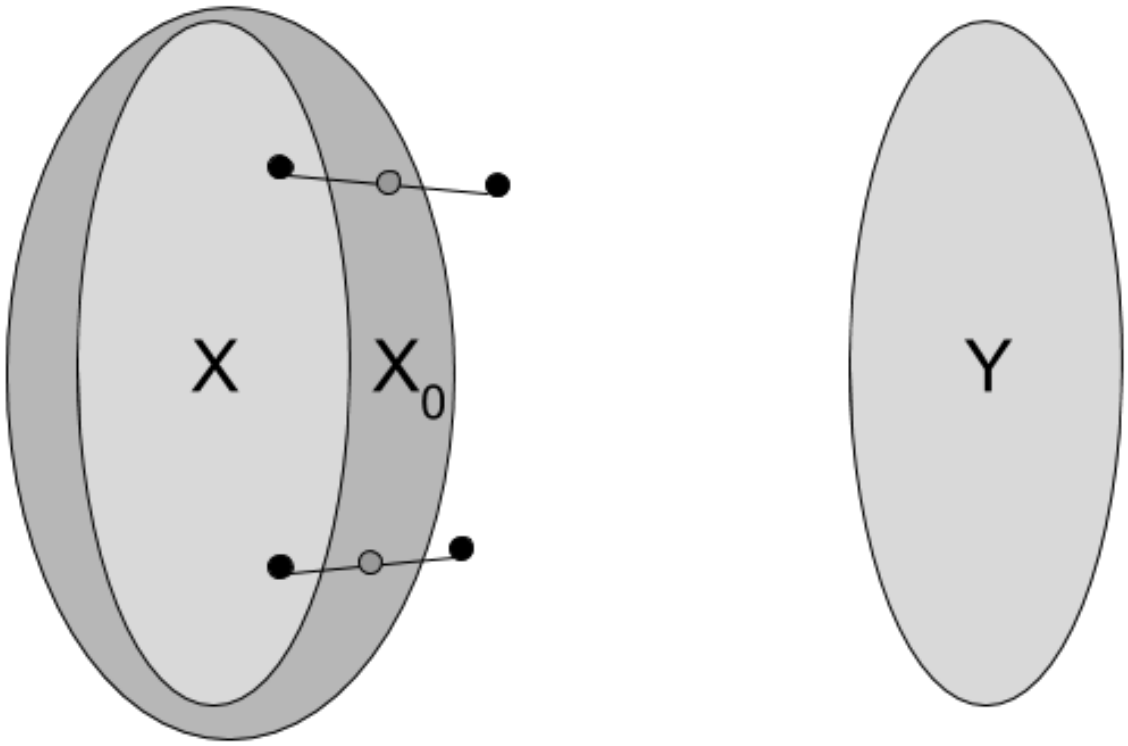

Figure B.5.3: Depiction of Proposition B.5.3.

Lemma 4.10.12 (Bounding first order terms, part 1)**.** *Consider a graph G and two sets $X, Y \subseteq V(G)$ with $X \cap Y = \emptyset$. Let $H = G/Y$ with y the identification of Y in G. Let $\Delta = \Delta^G(X, Y)$. Then*

$$\sum_{f \in G[X] \cup \partial_G X : 1/4 \leq \mathtt{lev}_G(f) \leq 3/4} |\mathtt{levcng}_{G \to H}(f)| \leq 32\Delta$$

Proof of Lemma 4.10.12. Split every edge in $\partial_G X$ into a path of two edges with half the resistance. Let X_0 be the set of edges with endpoints in X or one of the copies incident with X and let G_0 be this graph. We start by showing that

$$\sum_{f \in G_0[X_0]} (\mathtt{lev}_{G_0}(f) - \mathtt{lev}_{G_0/Y}(f)) \leq \Delta_0$$

where $\Delta_0 = \Delta^{G_0}(X_0, Y)$, $G_0' = G_0/X_0$, and x_0 is the identification of X_0. First, by Remark 12

$$\sum_{f \in H_0'[Y]} \texttt{lev}_{H_0'}(f) = |Y| - \Delta_0$$

where $H_0' = \texttt{Schur}(G_0', \{x_0\} \cup Y)$. By Rayleigh monotonicity,

$$\sum_{f \in H_0[Y]} \texttt{lev}_{H_0}(f) \geq |Y| - \Delta_0$$

where $H_0 = \texttt{Schur}(G_0, X_0 \cup Y)$. By Remark 12,

$$\sum_{f \in E(H_0) \setminus H_0[Y]} \texttt{lev}_{H_0}(f) \leq |X_0| + \Delta_0 - 1$$

But by Remark 12,

$$\sum_{f \in E(H_0) \setminus H_0[Y]} \texttt{lev}_{H_0/Y}(f) = |X_0|$$

so therefore

$$\sum_{f \in E(H_0) \setminus H_0[Y]} (\texttt{lev}_{H_0}(f) - \texttt{lev}_{H_0/Y}(f)) \leq \Delta_0 - 1$$

By Rayleigh monotonicity, the summand of the above sum is always nonnegative. Therefore, the above inequality also applies for any subset of $E(H_0) \setminus H_0[Y]$. In particular, it applies for $G_0[X_0]$, completing the proof that

$$\sum_{f \in G_0[X_0]} (\texttt{lev}_{G_0}(f) - \texttt{lev}_{G_0/Y}(f)) \leq \Delta_0$$

Consider an edge f that is one of the two copies of an edge e resulting from subdivision with $\texttt{lev}_G(e) \geq \frac{1}{4}$. Consider the three-vertex Schur complement of G with respect to $V(e) \cup V(f)$. Identifying Y only changes the Schur complement resistance of the edge with endpoints $V(e)$. Let f' be the remaining of the three edges in this Schur complement. $\texttt{lev}_{G_0}(f) = \texttt{lev}_{G_0}(f')$ and $\texttt{lev}_{G_0/Y}(f) = \texttt{lev}_{G_0/Y}(f')$ since the edge weights of f and f' are the same and they are part of the same path with length 2. Therefore, by Remark 12 applied to the three-vertex Schur complement,

$$\begin{aligned}\texttt{lev}_G(e) - \texttt{lev}_{G/Y}(e) &= (1 - (3 - 1 - 2\texttt{lev}_G(f))) - (1 - (3 - 1 - 2\texttt{lev}_{G/Y}(f))) \\ &= 2\texttt{lev}_G(f) - 2\texttt{lev}_{G/Y}(f)\end{aligned}$$

so

$$\sum_{f \in G[X] \cup \partial_G X} (\texttt{lev}_G(f) - \texttt{lev}_{G/Y}(f)) \leq 2\Delta_0$$

.

By Proposition B.5.3, $\Delta_0 \leq 2\Delta$. Combining this with Remark 10 proves the lemma. □

Lemma 4.10.13 (Bounding first order terms, part 2)**.** *Consider a graph G and a set of edges $D \subseteq E(G)$. Then*

$$\sum_{e \in E(G) \setminus D : 1/4 \leq \texttt{lev}_G(e) \leq 3/4} |\texttt{levcng}_{G \to G \setminus D}(e)| \leq 4|D|$$

Proof. By Remark 10, it suffices to show that

$$\sum_{e \in E(G) \setminus D} (\texttt{lev}_{G \setminus D}(e) - \texttt{lev}_G(e)) \leq |D|$$

Furthermore, we may assume that D consists of just one edge f, because deleting one edge at a time suffices for proving the above inequality for general D. By Sherman-Morrison,

$$\begin{aligned}
\sum_{e \in E(G) \setminus f} (\texttt{lev}_{G \setminus \{f\}}(e) - \texttt{lev}_G(e)) &= \sum_{e \in E(G) \setminus f} \frac{(b_e^T L_G^+ b_f)^2}{r_e(r_f - b_f^T L_G^+ b_f)} \\
&= \frac{1}{r_f - b_f^T L_G^+ b_f}(b_f^T L_G^+ b_f - \frac{(b_f^T L_G^+ b_f)^2}{r_f}) \\
&= \texttt{lev}_G(f) \\
&\leq 1
\end{aligned}$$

as desired. □

Preliminaries for second-order terms

Proposition B.5.4. *Consider a graph G with a set $Y \subseteq V(G)$, $X := V(G) \setminus Y$, $e = \{a, b\} \in E(X)$, $s \in X$, and $t \in Y$. Then*

$$|b_{st}^T L_G^+ b_e| \leq \sum_{f = \{p,q\} \in E(Y,X)} |b_{sq}^T L_G^+ b_e| \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f}$$

Proof. We start by showing that

$$b_{st}^T L_G^+ b_{su} = \sum_{f = \{p,q\} \in E(Y,X)} b_{sq}^T L_G^+ b_{su} \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f}$$

for any vertex $u \in X$. To show this, interpret the statement probabilistically. Notice that

$$\Pr_t[t_s > t_u] = \frac{b_{st}^T L_G^+ b_{su}}{b_{su}^T L_G^+ b_{su}}$$

This probability can be factored based on which edge is used to exit Y first:

$$\begin{aligned}
\Pr_t[t_s > t_u] &= \sum_{f=\{p,q\}\in E(Y,X)} \Pr_t[t_s > t_u, f \text{ used to exit } Y \text{ for the first time}] \\
&= \sum_{f=\{p,q\}\in E(Y,X)} \Pr_t[t_s > t_u | f \text{ used to exit } Y \text{ for the first time}] \\
&\Pr_t[f \text{ used to exit } Y \text{ for the first time}] \\
&= \sum_{f=\{p,q\}\in E(Y,X)} \Pr_q[t_s > t_u] \Pr_t[f \text{ used to exit } Y \text{ for the first time}] \\
&= \sum_{f=\{p,q\}\in E(Y,X)} \frac{b_{sq}^T L_G^+ b_{su}}{b_{su}^T L_G^+ b_{su}} \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f}
\end{aligned}$$

Multiplying both sides by $b_{su}^T L_G^+ b_{su}$ shows that

$$b_{st}^T L_G^+ b_{su} = \sum_{f=\{p,q\}\in E(Y,X)} b_{sq}^T L_G^+ b_{su} \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f}$$

For any two vertices $u, v \in X$, subtracting the resulting equations shows that

$$b_{st}^T L_G^+ b_{uv} = \sum_{f=\{p,q\}\in E(Y,X)} b_{sq}^T L_G^+ b_{uv} \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f}$$

Letting $b_{uv} := b_e$ and orienting the edge in the positive direction shows the desired inequality. □

Proposition B.5.5. *Consider a graph G with a set $Y \subseteq V(G)$, $X := V(G) \setminus Y$, $u \in X$, $s \in X$, and $t \in Y$. Let S_γ be the set of edges $f = \{p, q\} \in E(Y, X)$ with $|b_{sq}^T L_G^+ b_{su}| \geq \gamma |b_{st}^T L_G^+ b_{st}|$. Then*

$$\sum_{f\in S_\gamma} \frac{|b_{st}^T L_{G/X}^+ b_f|}{r_f} \leq \frac{3}{\gamma}$$

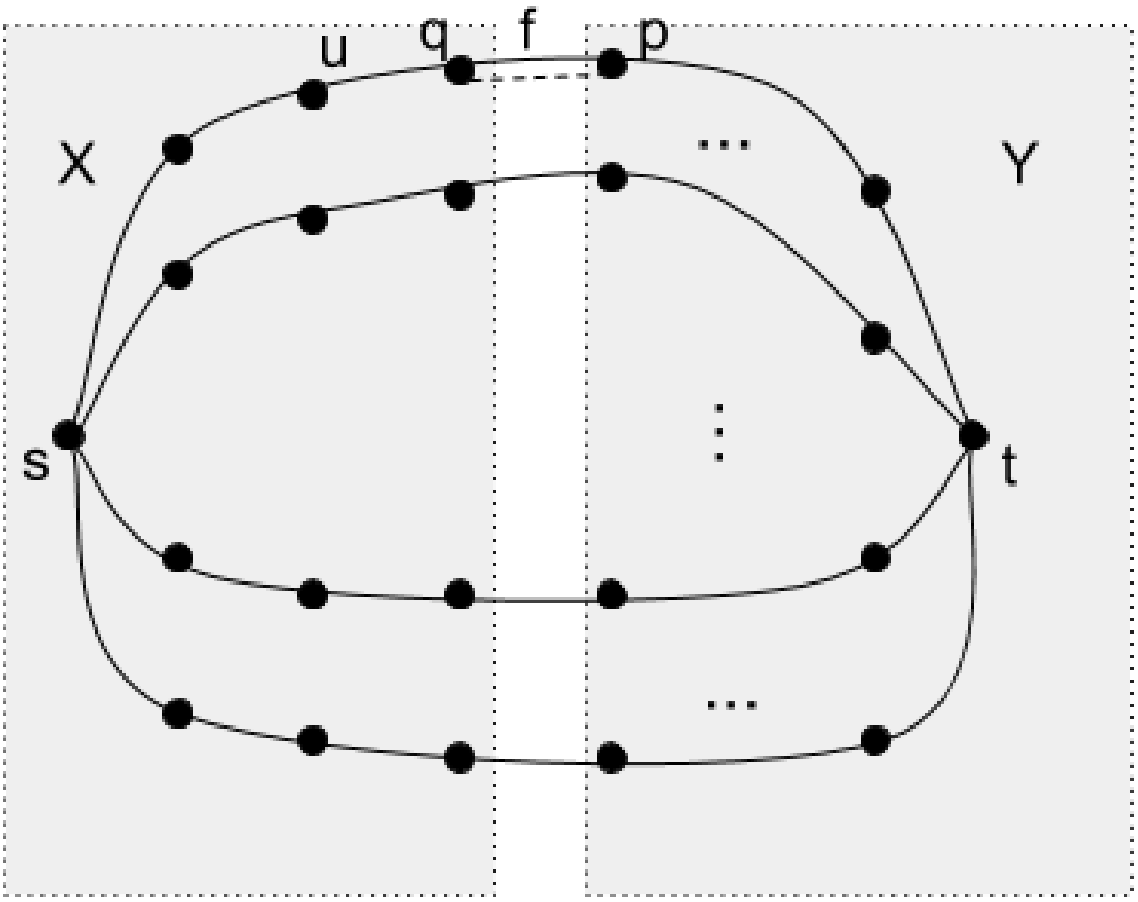

Figure B.5.4: Depiction of Proposition B.5.5. In this example, there are k parallel paths with length k from s to t and the $X - Y$ cut cuts the middle edge of each path. u is close to the cutedge f. f is the one $X - Y$ cutedge for which q has roughly k times the $s - u$ potential of t. The proposition implies that f has at most $O(1/k)$ flow.

Proof. Split each edge $e \in S_\gamma$ with resistance r_e into a path with two edges e_1 and e_2 with resistance $r_e/2$. Let the resulting graph be H. For an edge $\{p, q\} \in E(Y, X)$, suppose that it is split into $\{p, w\}$ and $\{w, q\}$. Notice that $|b_{st}^T L_{H/X}^+ b_{e_2}|/r_{e_2}^H = |b_{st}^T L_H^+ b_e|/r_e^G$, so it suffices to bound the st-flow on the edges e_2 for each edge $e \in S_\gamma$.

Let S'_γ be the set of edges e_2 for edges $e \in S_\gamma$.Notice that $|b_{su}^T L_H^+ b_{sz}| \geq \gamma b_{st}^T L_G^+ b_{st}/2 = \gamma |b_{st}^T L_H^+ b_{st}|/2$ for each $z \in V(S'_\gamma)$. Let $I = \texttt{Schur}(H, V(S'_\gamma) \cup X \cup \{t\})$. By the harmonic property of vertex potentials,

$$\begin{aligned}
\left(\sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I \right) b_{st}^T L_I^+ b_{st} &\geq \left(\sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I \right) b_{su}^T L_I^+ b_{st} \\
&= \sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I b_{su}^T L_I^+ b_{sx} \\
&\geq \sum_{x \in V(S'_\gamma) \setminus X} c_{xt}^I b_{su}^T L_I^+ b_{sx} \\
&\geq \frac{\gamma}{2} b_{st}^T L_I^+ b_{st} \sum_{x \in V(S'_\gamma) \setminus X} c_{xt}^I
\end{aligned}$$

so at least a $1 - 2/\gamma$ fraction of the conductance incident with t is also incident with X. By Rayleigh monotonicity, $1/\left(\sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I\right) \leq b_{st}^T L_{I/X}^+ b_{st}$. Therefore, the total flow in I/X incident with t that does not come directly from s is

$$\begin{aligned}
\sum_{x \in V(S'_\gamma) \setminus X} c_{xt}^{I/X} b_{st}^T L_{I/X}^+ b_{xt} &= 1 - c_{st}^{I/X} b_{st}^T L_{I/X}^+ b_{st} \\
&= 1 - \left(\sum_{x \in X} c_{xt}^I \right) b_{st}^T L_{I/X}^+ b_{st} \\
&\leq 1 - \frac{\sum_{x \in X} c_{xt}^I}{\sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I} \\
&= \frac{\sum_{x \in V(S'_\gamma) \setminus X} c_{xt}^I}{\sum_{x \in V(S'_\gamma) \cup X} c_{xt}^I} \\
&\leq \frac{2}{\gamma}
\end{aligned}$$

Notice that I contains the edges in S'_γ and that all $s-t$ flow on these edges in I/X enters t through some vertex besides s. Since all edges of S'_γ are incident with X, flow only travels across one edge in S'_γ. This means that

$$\sum_{f \in S'_\gamma} c_f^I |b_{st}^T L_{I/X}^+ b_f| \leq \frac{2}{\gamma}$$

The desired result follows from the fact that $|b_{st}^T L_{I/X}^+ b_f| = |b_{st}^T L_{G/X}^+ b_f|$ by definition of the Schur complement for all $f \in S'_\gamma$ and the fact that $c_f^I \geq c_f^G$. □

Proposition B.5.6. *Consider a graph I with two vertices $s, t \in V(I)$. Let F be the set of edges $e = \{a, b\} \in E(I)$ with $\max_{c \in \{a,b\}} b_{st}^T L_I^+ b_{sc} \leq \rho$ for some $\rho > 0$. Then*

$$\sum_{e \in F} (b_{st}^T L_I^+ b_e)^2 / r_e \leq \rho$$

Proof. Write energy as current times potential drop. Doing this shows that

$$\begin{aligned} \sum_{e \in F} (b_{st}^T L_I^+ b_e)^2 / r_e &\leq \int_0^\rho \sum_{e \in x \text{ threshold cut}} (|b_{st}^T L_I^+ b_e| / r_e) dx \\ &= \int_0^\rho dx \\ &= \rho \end{aligned}$$

as desired. □

Lemma 4.10.14 (Bounding the second order term)**.** *Consider a graph G and two disjoint sets of vertices $X, Y \subseteq V(G)$. For any $s \in X$,*

$$\sum_{e \in E_G(X) \cup \partial_G X} \alpha_{s,Y}^G(e) = \sum_{e \in E_G(X) \cup \partial_G X} \max_{t \in Y} \frac{(b_{st}^T L_G^+ b_e)^2}{(b_{st}^T L_G^+ b_{st}) r_e} \leq 24 \xi_{buckets}^2 \Delta^G(X, Y)$$

where $\xi_{buckets} = \log(m\alpha)$.

Proof of Lemma 4.10.14. We start by reducing to the case where all edges have both endpoints in X. Subdividing all edges in $\partial_G X$ to obtain a graph G_0. Let X_0 be the set of vertices in G_0 that are endpoints of edges in $E_G(X) \cup \partial_{G_0} X$. Let x_0 be the identification of X_0 in G_0/X_0. Notice that

$$\begin{aligned}\sum_{e\in E_G(X)\cup\partial_G X}\max_{t\in Y}\frac{(b_{st}^T L_G^+ b_e)^2}{(b_{st}^T L_G^+ b_{st})r_e^G} &= \sum_{e\in E_{G_0}(X)}\max_{t\in Y}\frac{(b_{st}^T L_{G_0}^+ b_e)^2}{(b_{st}^T L_{G_0}^+ b_{st})r_e^{G_0}} \\ &+ \sum_{e\in\partial_{G_0}X}\max_{t\in Y}\frac{4(b_{st}^T L_{G_0}^+ b_e)^2}{(b_{st}^T L_{G_0}^+ b_{st})2r_e^{G_0}} \\ &\le 2\sum_{e\in E_{G_0}(X_0)}\max_{t\in Y}\frac{(b_{st}^T L_{G_0}^+ b_e)^2}{(b_{st}^T L_{G_0}^+ b_{st})r_e^{G_0}}\end{aligned}$$

with all equalities and inequalities true termwise. By Proposition B.5.3,

$$\Delta^{G_0}(X_0, Y) \le 2\Delta^G(X, Y)$$

so to prove the lemma it suffices to show that

$$\sum_{e\in E_{G_0}(X_0)}\max_{t\in Y}\frac{(b_{st}^T L_{G_0}^+ b_e)^2}{(b_{st}^T L_{G_0}^+ b_{st})r_e^{G_0}} \le 6\xi_{\text{buckets}}^2\Delta^{G_0}(X_0, Y)$$

Let $H = \texttt{Schur}(G_0, X_0 \cup Y)$. The Schur complement only adds edges to $E(X_0)$, so

$$\sum_{e\in E_{G_0}(X_0)}\max_{t\in Y}\frac{(b_{st}^T L_{G_0}^+ b_e)^2}{(b_{st}^T L_{G_0}^+ b_{st})r_e^{G_0}} \le \sum_{e\in E_H(X_0)}\max_{t\in Y}\frac{(b_{st}^T L_H^+ b_e)^2}{(b_{st}^T L_H^+ b_{st})r_e^H}$$

For an edge $g \in E_H(X_0, Y)$, let x_g and y_g denote its endpoints in X_0 and Y respectively, $S_{g,i} \subseteq E_H(X_0)$ be the set of edges $e = \{a, b\}$ with $\max_{c\in\{a,b\}} |b_{sx_g}^T L_G^+ b_{sc}| \in [2^i, 2^{i+1}]$ for integers $i \in (\log(r_{min}/m), \log r_{max}]$ and $\max_{c\in\{a,b\}} |b_{sx_g}^T L_G^+ b_{sc}| \in [0, 2^{i+1}]$ for $i = \log(r_{min}/m)$. By Proposition B.5.4,

$$|b_{st}^T L_H^+ b_e| \le \sum_{g\in E_H(X_0,Y)} |b_{sx_g}^T L_H^+ b_e| \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H}$$

Applying Cauchy-Schwarz twice and Proposition B.5.5 shows that

$$\sum_{e\in E_H(X_0)} \max_{t\in Y} \frac{(b_{st}^T L_H^+ b_e)^2}{(b_{st}^T L_H^+ b_{st}) r_e^H}$$

$$\leq \sum_{e\in E_H(X_0)} \max_{t\in Y} \frac{1}{(b_{st}^T L_H^+ b_{st}) r_e^H} \left(\sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{g:e\in S_{g,i}} |b_{sx_g}^T L_H^+ b_e| \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right)^2$$

$$\leq \xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{e\in E_H(X_0)} \max_{t\in Y} \frac{1}{(b_{st}^T L_H^+ b_{st}) r_e^H} \left(\sum_{g:e\in S_{g,i}} |b_{sx_g}^T L_H^+ b_e| \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right)^2$$

$$\leq \xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{e\in E_H(X_0)} \Bigg(\max_{t\in Y} \frac{1}{(b_{st}^T L_H^+ b_{st}) r_e^H} \left(\sum_{g:e\in S_{g,i}} (b_{sx_g}^T L_H^+ b_e)^2 \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right) \left(\sum_{g:e\in S_{g,i}} \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right) \Bigg)$$

$$\leq \xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)+1}^{\log r_{max}} \sum_{e\in E_H(X_0)} \max_{t\in Y} \left(\sum_{g:e\in S_{g,i}} \frac{(b_{sx_g}^T L_H^+ b_e)^2}{(b_{st}^T L_H^+ b_{st}) r_e^H} \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right) \frac{3 b_{st}^T L_H^+ b_{st}}{2^i}$$

$$+ \xi_{\text{buckets}} \sum_{e\in E_H(X_0)} \max_{t\in Y} \left(\sum_{g:e\in S_{g,\log(r_{min}/m)}} \frac{(b_{sx_g}^T L_H^+ b_e)^2}{(b_{st}^T L_H^+ b_{st}) r_e^H} \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right) m$$

$$\leq 3\xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{g\in E_H(X_0,Y)} \sum_{e\in S_{g,i}} \frac{(b_{sx_g}^T L_H^+ b_e)^2}{2^i r_e^H} \left(\max_{t\in Y} \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right)$$

By Proposition B.5.6,

$$\sum_{e\in S_{g,i}} \frac{(b_{sx_g}^T L_H^+ b_e)^2}{r_e^H} \leq 2^{i+1}$$

so

$$\sum_{e\in E_H(X_0)} \max_{t\in Y} \frac{(b_{st}^T L_H^+ b_e)^2}{(b_{st}^T L_H^+ b_{st}) r_e^H} \leq 6\xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{g\in E_H(X_0,Y)} \left(\max_{t\in Y} \frac{|b_{st}^T L_{H/X_0}^+ b_g|}{r_g^H} \right)$$
$$\leq 6\xi_{\text{buckets}} \sum_{i=\log(r_{min}/m)}^{\log r_{max}} \sum_{g\in E_H(X_0,Y)} \frac{b_g^T L_{H/X_0}^+ b_g}{r_g^H}$$
$$= 6\xi_{\text{buckets}}^2 \Delta^{G_0}(X_0, Y)$$

as desired. □

Other preliminaries

Proposition 4.10.11. *Consider two disjoint sets of vertices X and Y in a graph G. Let $G' = G/(X \cup Y)$, with x and y the identifications of X and Y respectively. Let Z be the set of vertices v with electrical potential $p_v \leq \gamma$ for some $\gamma \in (0, 1)$ with boundary conditions $p_x = 0$ and $p_y = 1$. Then*

$$\Delta^G(Z, Y) \leq \frac{1}{1-\gamma} \Delta^G(X, Y)$$

where z is the identification of Z in G/Z.

Proof. Let $H_z = \texttt{Schur}(G/Z, \{z\} \cup Y)$. By definition,

$$\Delta^G(Z, Y) = \sum_{v\in Y} \frac{b_{zv}^T L_{G/Z}^+ b_{zv}}{r_{zv}^{H_z}}$$

By Rayleigh monotonicity and the fact that $X \subseteq Z$,

$$b_{zv}^T L_{G/Z}^+ b_{zv} \leq b_{xv}^T L_{G/X}^+ b_{xv}$$

By Proposition B.5.1 applied to $U = Z$ and G',

$$r_{zv}^{H_z} \geq \frac{1}{1-\gamma} r_{xv}^{H_x}$$

since the conductance in the Schur complement for a vertex $v \in Y$ is proportional to the incoming flow by Proposition 4.9.2. Substituting these inequalities and using the definition of $\Delta^G(X, Y)$ gives the desired result. □

Proposition 4.10.20. *Consider two disjoint sets of vertices X and Y in a graph G. Let $G' = G/(X, Y)$, with x and y the identifications of X and Y respectively. Let A and B be sets of edges for which both endpoints have normalized $L_{G'}^+ b_{xy}$ potential at most γ and at*

least $1-\gamma$ *respectively for some* $\gamma \in (0, 1/2)$. *Arbitrarily contract and delete edges in* A *and* B *in* G *to obtain the graph* H. *Then*

$$c^H(X, Y) \leq \frac{1}{(1-\gamma)^2} c^G(X, Y)$$

Proof. Apply Proposition B.5.2 twice, first with $Z \leftarrow A$ and second with $Z \leftarrow B$. Do this for all $w \in X \cup Y$. Each application increases the $X - Y$ conductance by at most a factor of $1/(1-\gamma)$, proving the desired result. □

B.5.2 Objectives that are stable thanks to stable oracles

Before proving these stability propositions, we prove one fact that is common to all of them: that splitting can be done in the same direction (parallel or series) for all of the graphs I_k:

Proposition B.5.7. *Consider some* $k \in \{0, 1, \ldots, K-1\}$ *and any* $e \notin \{f_0, f_1, \ldots, f_{k-1}\}$. *Let* $(I_0'', \{e^{(0)}, e^{(1)}\}) \leftarrow \texttt{Split}(I_0, e)$ *and obtain* $I_k'' \leftarrow I_{k-1}''[[f_{k-1}]]$. *Then,*

- *(Splitting later)* I_k'' *is equivalent in distribution to the graph obtained by splitting* e *in* I_k *in the same direction as* e *was split in* I_0 *to obtain* I_0''
- *(Leverage score bounds) With high probability,*

$$1/8 \leq \texttt{lev}_{I_k''/(S,S')}(e^{(0)}) \leq \texttt{lev}_{I_k''\setminus D}(e^{(0)}) \leq 7/8$$

Proof. **Splitting later.** Effective resistances are electrical functions, so order of splitting does not matter.

Leverage score bounds. By the "Bounded leverage score difference" condition in Definition 4.10.7, the "Leverage score stability" guarantees in Definition 4.10.7, and the triangle inequality,

$$|\texttt{lev}_{I_k\setminus D}(e) - \texttt{lev}_{I_0}(e)| \leq 3/16$$

and

$$|\texttt{lev}_{I_k/(S,S')}(e) - \texttt{lev}_{I_0}(e)| \leq 3/16$$

If $\texttt{lev}_{I_0}(e) \geq 1/2$, then e is split in parallel in I_0 to obtain I_0''. Furthermore, $\texttt{lev}_{I_k/(S,S')}(e) \geq 1/2 - 3/16 > 1/4$. This means that $1/2 \geq \texttt{lev}_{I_k''/(S,S')}(e^{(0)}) \geq (1/4)/2 = 1/8$ since $\texttt{lev}_{I_k''/(S,S')}(e^{(0)}) = \texttt{lev}_{I_k/(S,S')}(e)/2$. By Rayleigh monotonicity, $\texttt{lev}_{I_k''\setminus D}(e) \geq \texttt{lev}_{I_k''/(S,S')}(e) \geq 1/8$. As a result, $1/2 \geq \texttt{lev}_{I_k''\setminus D}(e^{(0)}) \geq 1/8$ as well.

If $\texttt{lev}_{I_0}(e) \leq 1/2$, then e is split in series in I_0 to obtain I_0''. Furthermore, $\texttt{lev}_{I_k\setminus D}(e) \leq 1/2 + 3/16 \leq 3/4$. This means that $1/2 \leq \texttt{lev}_{I_k''\setminus D}(e^{(0)}) \leq (1/2)(3/4) + 1/2 = 7/8$ since

$\texttt{lev}_{I_k''\setminus D}(e^{(0)}) \le (1/2)\texttt{lev}_{I_k\setminus D}(e) + 1/2$. By Rayleigh monotonicity, $\texttt{lev}_{I_k''/(S,S')}(e) \le 7/8$. As a result, $1/2 \le \texttt{lev}_{I_k''/(S,S')}(e^{(0)}) \le 7/8$ as well.

This completes the desired result in both cases. □

Now, we proceed with the proofs of the stability propositions:

Proposition 4.10.16 (Stability with respect to Δ)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), 0)$-stable subset of W for the electrical functions*

$$\delta_{S,S'}(H \setminus D)$$

and

$$\delta_{S',S}(H \setminus D)$$

of H with high probability.

Proof of Proposition 4.10.16. We focus on $\delta_{S,S'}(H \setminus D)$, as the argument for $\delta_{S',S}(H \setminus D)$ is the same, with S and S' swapped.

Well-definedness. Recall that $\delta_{S,S'}(H \setminus D)$ is an electrical function of H since $\delta_{S,S'}(H \setminus D)$ is preserved under Schur complementation of vertices besides the ones referenced in the summand of $\delta_{S,S'}(H \setminus D)$. Therefore, stability is well-defined.

Degree of midpoints. Let I_k'' be the graph obtained from I_k by splitting all edges of $W \setminus \{f_0, f_1, \ldots, f_{k-1}\}$ in series. By the "Midpoint potential stability" guarantee of `Oracle`, the midpoints of these edges are contained in the sets $U_S, U_{S'} \subseteq V(I_k'')$ defined to be the sets of vertices with $s = 0 - s' = 1$ normalized potential less than $1 - p/2$ and greater than $p/2$ respectively. By Proposition 4.10.11,

$$\Delta^{I_k''\setminus D}(U_S, S') \le 2\Delta^{I_k\setminus D}(S, S')/p$$

and

$$\Delta^{I_k''\setminus D}(U_{S'}, S) \le 2\Delta^{I_k\setminus D}(S', S)/p$$

Lipschitz contractions. For simplicity of notation in the following proof, let I_k denote the graph in which the edge f_k has been split and let f_k denote an arbitrary copy. By Sherman-Morrison,

$$
\begin{aligned}
&\delta_{S,S'}((I_k \setminus D)/f_k) - \delta_{S,S'}(I_k \setminus D) \\
&= \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} - \frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \\
&\left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} - \frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right) \\
&- \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&= - \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&- \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right) \\
&+ \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right)
\end{aligned}
$$

Therefore, by Proposition B.5.7, Rayleigh monotonicity, and the fact that the total energy is an upper bound on the energy of an edge,

$$
\begin{aligned}
&|\delta_{S,S'}((I_k \setminus D)/f_k) - \delta_{S,S'}(I_k \setminus D)| \\
&\leq 8 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k}} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&+ 8 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{|b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e|}{r_{f_k} r_e} \right) \\
&+ 64 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{|b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e|}{r_{f_k} r_e} \right)
\end{aligned}
$$

By the "$S - S'$ normalized degree change stability" guarantees of `Oracle`,

$$|\delta_{S,S'}((I_k \setminus D)/f_k) - \delta_{S,S'}(I_k \setminus D)| \leq \frac{8\rho}{|W|}\delta_{S,S'}(I \setminus D) + \frac{8\rho}{|W|}\delta_{S,S'}(I \setminus D) + \frac{64\rho}{|W|}\delta_{S,S'}(I \setminus D)$$
$$\leq \frac{80\rho}{|W|}\delta_{S,S'}(I \setminus D)$$

as desired.

Lipschitz deletions. This bound is very similar to the contraction case. By Sherman-Morrison,

$$\begin{aligned}
&\delta_{S,S'}((I_k \setminus D)\backslash f_k) - \delta_{S,S'}(I_k \setminus D) \\
&= \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} + \frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \\
&\left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} + \frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right) \\
&- \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&= \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&+ \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right) \\
&+ \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/S} b_{f_k}} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{(r_{f_k} - b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}) r_e} \right)
\end{aligned}$$

Therefore, by Proposition B.5.7, Rayleigh monotonicity, and the fact that the total energy is an upper bound on the energy of an edge,

$$
\begin{aligned}
&|\delta_{S,S'}((I_k \setminus D)\backslash f_k) - \delta_{S,S'}(I_k \setminus D)| \\
&\leq 8 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k}} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&+ 8 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{|b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e|}{r_{f_k} r_e} \right) \\
&+ 64 \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{|b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e|}{r_{f_k} r_e} \right)
\end{aligned}
$$

By the "$S - S'$ normalized degree change stability" guarantees of `Oracle`,

$$
\begin{aligned}
|\delta_{S,S'}((I_k \setminus D)\backslash f_k) - \delta_{S,S'}(I_k \setminus D)| &\leq \frac{8\rho}{|W|} \delta_{S,S'}(I \setminus D) + \frac{8\rho}{|W|} \delta_{S,S'}(I \setminus D) + \frac{64\rho}{|W|} \delta_{S,S'}(I \setminus D) \\
&\leq \frac{80\rho}{|W|} \delta_{S,S'}(I \setminus D)
\end{aligned}
$$

as desired.

Change in expectation. Write down the change in the expectation:

$$
\begin{aligned}
&\mathbf{E}_{H' \sim I_k[f_k]}[\delta_{S,S'}(H' \setminus D)|f_k] - \delta_{S,S'}(I_k \setminus D) \\
&= \mathtt{levcng}_{I_k \to (I_k \setminus D)/S}(f_k) \sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k}} \right) \left(\frac{b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_e}{r_e} \right) \\
&\mathtt{levcng}_{I_k \to (I_k \setminus D)/(S,S')}(f_k) \times \\
&\left(\sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{sw} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{r_{f_k} r_e} \right) \right) \\
&+ \left(\frac{\mathtt{lev}_{I_k}(f_k)}{\mathtt{lev}_{(I_k \setminus D)/S}(f_k) \mathtt{lev}_{(I_k \setminus D)/(S,S')}(f_k)} + \frac{1 - \mathtt{lev}_{I_k}(f_k)}{(1 - \mathtt{lev}_{(I_k \setminus D)/S}(f_k))(1 - \mathtt{lev}_{(I_k \setminus D)/(S,S')}(f_k))} \right) \\
&\sum_{w \in S'} \sum_{e \in \partial_{I_k} w} \left(\frac{(b_{sw}^T L^+_{(I_k \setminus D)/S} b_{f_k})^2}{r_{f_k}} \right) \left(\frac{(b_{ss'}^T L^+_{(I_k \setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k \setminus D)/(S,S')} b_e)}{r_{f_k} r_e} \right)
\end{aligned}
$$

By Proposition B.5.7,

$$
\begin{aligned}
&|\mathbf{E}_{H'\sim I_k[f_k]}[\delta_{S,S'}(H'\setminus D)|f_k] - \delta_{S,S'}(I_k\setminus D)| \\
&\le |\texttt{levcng}_{I_k\to(I_k\setminus D)/S}(f_k)| \sum_{w\in S'}\sum_{e\in\partial_{I_k}w}\left(\frac{(b_{sw}^T L^+_{(I_k\setminus D)/S} b_{f_k})^2}{r_{f_k}}\right)\left(\frac{b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_e}{r_e}\right) \\
&+ |\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)|\times \\
&\left(\sum_{w\in S'}\sum_{e\in\partial_{I_k}w}\left(b_{sw}^T L^+_{(I_k\setminus D)/S} b_{sw}\right)\left(\frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_e|}{r_{f_k} r_e}\right)\right) \\
&+ 128\sum_{w\in S'}\sum_{e\in\partial_{I_k}w}\left(\frac{(b_{sw}^T L^+_{(I_k\setminus D)/S} b_{f_k})^2}{r_{f_k}}\right)\left(\frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_e|}{r_{f_k} r_e}\right)
\end{aligned}
$$

Let $\alpha_{f_k} = \max_{w\in S'} \frac{(b_{sw}^T L^+_{(I_k\setminus D)/S} b_{f_k})^2}{(b_{sw}^T L^+_{(I_k\setminus D)/S} b_{sw}) r_{f_k}}$. By the "$S-S'$-normalized degree change stability" guarantees of `Oracle`,

$$
\begin{aligned}
&|\mathbf{E}_{H'\sim I_k[f_k]}[\delta_{S,S'}(H'\setminus D)|f_k] - \delta_{S,S'}(I_k\setminus D)| \\
&\le |\texttt{levcng}_{I_k\to(I_k\setminus D)/S}(f_k)|\left(\frac{\rho}{|W|}\delta_{S,S'}(I\setminus D)\right) \\
&+ \left(|\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)| + 128\alpha_{f_k}\right)\left(\frac{\rho}{|W|}\delta_{S,S'}(I\setminus D)\right)
\end{aligned}
$$

Now, we exploit the fact that f_k is chosen randomly from all remaining edges in W. $K \le |W|/4$, so there are always at least $|W|/4$ remaining edges in W by the "Size of Z" guarantee. By Lemma 4.10.12 applied twice, 4.10.13 applied once, and 4.10.14 (for α_{f_k}) applied to the graph $(I_k\setminus D)/S$ with $X \leftarrow U_S$ and $Y \leftarrow S'$,

$$
\begin{aligned}
&|\mathbf{E}_{H'\sim I_k[f_k]}[\delta_{S,S'}(H'\setminus D)] - \delta_{S,S'}(I_k\setminus D)| \\
&\le \mathbf{E}_{f_k}\left[|\mathbf{E}_{H'\sim I_k[f_k]}[\delta_{S,S'}(H'\setminus D)|f_k] - \delta_{S,S'}(I_k\setminus D)|\right] \\
&\le \frac{2}{|W|}\sum_{f_k\in W}\left(|\texttt{levcng}_{I_k\to(I_k\setminus D)/S}(f_k)| + |\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)| + 128\alpha_{f_k}\right)\times \\
&\left(\frac{\rho}{|W|}\delta_{S,S'}(I\setminus D)\right) \\
&\le \frac{2\rho(2\Delta^{I''_k\setminus D}(U_S,S') + 2\Delta^{I''_k\setminus D}(U_{S'},S) + 2|D| + 128\Delta^{I''_k\setminus D}(U_S,S'))}{|W|^2}\delta_{S,S'}(I\setminus D)
\end{aligned}
$$

Applying the result of the "Degree of midpoints" part shows that

$$|\mathbf{E}_{H'\sim I_k[f_k]}[\delta_{S,S'}(H' \setminus D)] - \delta_{S,S'}(I_k \setminus D)| \leq \frac{520\rho\Delta_k}{p|W|^2}\delta_{S,S'}(I \setminus D)$$

as desired. □

Proposition 4.10.17 (Stability with respect to sums of deferred potentials)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), r_{min}/n^4)$-stable subset of W for the electrical function*

$$\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H\setminus D)/(S,S')}(b_{su} + b_{sv})\right) + \left(\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H\setminus D)/(S,S')}(b_{us'} + b_{vs'})\right)$$

of H with high probability.

Proof of Proposition 4.10.17. Well-definedness. The function given is an electrical function of H.

Degree of midpoints. Same as in the proof of Proposition 4.10.17.

Lipschitz contractions. As in the previous stability proof, let I_k denote the graph after splitting an edge, one of whose copies is f_k. By Sherman-Morrison, the triangle inequality, Proposition B.5.7 with Rayleigh monotonicity, and the "Deferred endpoint potential change stability" guarantee of `Oracle` in that order,

$$
\begin{aligned}
&\left| \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')/f_k}(b_{su}+b_{sv}) \right) + \left(\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')/f_k}(b_{us'}+b_{vs'}) \right) \right. \\
&\left. - \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) \right) - \left(\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) \right| \\
&= \left| \sum_{\{u,v\}\in A} \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}))}{b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \right. \\
&\left. + \sum_{\{u,v\}\in B} \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'}))}{b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \right| \\
&\leq \sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv})|}{b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&+ \sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&\leq 8 \sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_{f_k}} \\
&+ 8 \sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_{f_k}} \\
&\leq \frac{8\rho}{|W|} \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) + \frac{8 r_{min}}{n^4}
\end{aligned}
$$

as desired.

Lipschitz deletions. This part is similar to the contraction part. By Sherman-Morrison, the triangle inequality, Proposition B.5.7 with Rayleigh monotonicity, and the "Deferred endpoint potential change stability" guarantee of `Oracle` in that order,

$$
\begin{aligned}
&\left| \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')\setminus f_k}(b_{su}+b_{sv}) \right) + \left(\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')\setminus f_k}(b_{us'}+b_{vs'}) \right) \right. \\
&\left. - \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) \right) - \left(\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) \right| \\
&= \left| \sum_{\{u,v\}\in A} \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}))}{r_{f_k} - b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \right. \\
&\left. + \sum_{\{u,v\}\in B} \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'}))}{r_{f_k} - b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \right| \\
&\leq \sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_{f_k} - b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&+ \sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_{f_k} - b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&\leq 8 \sum_{\{u,v\}\in A} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_{f_k}} \\
&+ 8 \sum_{\{u,v\}\in B} \frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_{f_k}} \\
&\leq \frac{8\rho}{|W|} \left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'}) \right) + \frac{8 r_{min}}{n^4}
\end{aligned}
$$

as desired.

Change in expectation. Compute the expected change using Sherman-Morrison:

$$\mathbf{E}_{H'\sim I_k[f_k]}\left[\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\middle| f_k\right]$$
$$- \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) - \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})$$
$$= \texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)\left(\sum_{\{u,v\}\in A}\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}))}{r_{f_k}}\right.$$
$$\left.+\sum_{\{u,v\}\in B}\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}b_{f_k})(b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'}))}{r_{f_k}}\right)$$

By the "Deferred endpoint potential change stability" guarantee,

$$\left|\mathbf{E}_{H'\sim I_k[f_k]}\left[\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\middle| f_k\right]\right.$$
$$\left.- \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) - \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right|$$
$$\leq |\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)|\left(\sum_{\{u,v\}\in A}\frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv})|}{r_{f_k}}\right.$$
$$\left.+\sum_{\{u,v\}\in B}\frac{|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}b_{f_k}||b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{r_{f_k}}\right)$$
$$\leq |\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)|\left(\frac{\rho}{|W|}\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv})\right.\right.$$
$$\left.\left.+\sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right)+\frac{r_{min}}{n^4}\right)$$

By Lemmas 4.10.12 and 4.10.13 and the "Size of Z" guarantee,

$$\left|\mathbf{E}_{H'\sim I_k[f_k]}\left[\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right.\right.$$
$$\left.\left. - \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) - \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right]\right|$$
$$\leq \mathbf{E}_{f_k}\left[\left|\mathbf{E}_{H'\sim I_k[f_k]}\left[\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right.\right.\right.$$
$$\left.\left.\left. - \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) - \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right| f_k\right]\right]$$
$$\leq \frac{2}{|W|}\left(\sum_{f_k\in W} |\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)|\right)\left(\frac{\rho}{|W|}\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv})\right.\right.$$
$$\left.\left. + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right) + \frac{r_{min}}{n^4}\right)$$
$$\leq \frac{2}{|W|}\left(\Delta^{I_k''\setminus D}(U_S,S') + \Delta^{I_k''\setminus D}(U_{S'},S) + |D|\right)\left(\frac{\rho}{|W|}\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv})\right.\right.$$
$$\left.\left. + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right) + \frac{r_{min}}{n^4}\right)$$

By "Degree of midpoints," $\Delta^{I_k''\setminus D}(U_S,S') + \Delta^{I_k''\setminus D}(U_{S'},S) + |D| \leq (2/p)\Delta_k$, so

$$\left|\mathbf{E}_{H'\sim I_k[f_k]}\left[\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(H'\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right.\right.$$
$$\left.\left. - \sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{su}+b_{sv}) - \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right]\right|$$
$$\leq \frac{4\rho\Delta_k}{p|W|^2}\left(\sum_{\{u,v\}\in A} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{su}+b_{sv}) + \sum_{\{u,v\}\in B} b_{ss'}^T L^+_{(I\setminus D)/(S,S')}(b_{us'}+b_{vs'})\right) + \frac{r_{min}}{n^4}$$

as desired.

□

Proposition 4.10.18 (Stability with respect to the main objective)**.** *For all $k \in \{0, 1, \ldots, K(|W|) - 1\}$, the set Z_k is a $(\tilde{O}(\rho), \tilde{O}(\rho\Delta_k/p), 0)$-stable subset of W for the electrical function*

$$b_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_{ss'}$$

of H with high probability.

Proof of Proposition 4.10.18. Well-definedness. $b_{ss'}^T L_{H\setminus D} b_{ss'}$ is an electrical function of H.

Degree of midpoints. Same as in the proof of Proposition 4.10.17.

Lipschitz contractions. As in the previous stability proofs, let I_k denote the graph after splitting an edge, one of whose copies is f_k. By Sherman-Morrison, Proposition B.5.7 with Rayleigh monotonicity, and the "Main objective change stability" guarantee of `Oracle` in that order,

$$\begin{aligned}
|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')/f_k} b_{ss'} - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}| &= \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&\leq 8\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{r_{f_k}} \\
&\leq \frac{8\rho}{|W|} b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}
\end{aligned}$$

as desired.

Lipschitz deletions. This is similar to the contraction proof. By Sherman-Morrison, Proposition B.5.7 with Rayleigh monotonicity, and the "Main objective change stability" guarantee of `Oracle` in that order,

$$\begin{aligned}
|b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')\setminus f_k} b_{ss'} - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}| &= \frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{r_{f_k} - b_{f_k}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k}} \\
&\leq 8\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{r_{f_k}} \\
&\leq \frac{8\rho}{|W|} b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}
\end{aligned}$$

as desired.

Change in expectation. The change in expectation conditioned on a choice of f_k can be written as

$$\mathbf{E}_{H'\sim I_k[f_k]}[b_{ss'}^T L^+_{(H'\setminus D)/(S,S')} b_{ss'}|f_k] - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}$$
$$= \texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{r_{f_k}}$$

Therefore, by Lemma 4.10.12 applied twice, Lemma 4.10.13 applied once, and the "Size of Z" guarantee,

$$\begin{aligned}
&|\mathbf{E}_{H'\sim I_k[f_k]}[b_{ss'}^T L^+_{(H'\setminus D)/(S,S')} b_{ss'}] - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}| \\
&\le \mathbf{E}_{f_k}\left[|\mathbf{E}_{H'\sim I_k[f_k]}[b_{ss'}^T L^+_{(H'\setminus D)/(S,S')} b_{ss'}|f_k] - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}|\right] \\
&\le \frac{2}{|Z|}\sum_{f_k\in W}|\texttt{levcng}_{I_k\to(I_k\setminus D)/(S,S')}(f_k)|\frac{(b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{f_k})^2}{r_{f_k}} \\
&\le \frac{4\rho}{|W|^2}(\Delta^{I_k''\setminus D}(U_S,S') + \Delta^{I_k''\setminus D}(U_{S'},S) + |D|)b_{ss'}^T L^+_{(I\setminus D)/(S,S')} b_{ss'}
\end{aligned}$$

By "Degree of midpoints,"

$$|\mathbf{E}_{H'\sim I_k[f_k]}[b_{ss'}^T L^+_{(H'\setminus D)/(S,S')} b_{ss'}] - b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}| \le \frac{4\rho\Delta_k}{p|W|^2} b_{ss'}^T L^+_{(I_k\setminus D)/(S,S')} b_{ss'}$$

as desired.

□

B.6 Deferred proofs for Section 4.11

B.6.1 Fast stability concentration inequalities

Proposition 4.11.1. *Let $\{M^{(k)}\}_k \in \mathbb{R}^{n\times n}$ be a sequence of symmetric, nonnegative random matrices, $\{Z^{(k)}\}_k \in \{0,1\}^n$, and $S^{(k)} \subseteq [n]$ with the following properties:*

- *For all $i \in [n]$, $\sum_{j=1,j\neq i}^n M_{ij}^{(0)} \le \sigma$ where $\sigma \le \sigma_0$. Furthermore, $S^{(0)} = \emptyset$.*
- *The random variables $\{Z^{(k)}\}_k$ are defined by making $Z^{(k+1)}$ the indicator of a uniformly random choice $w^{(k+1)} \in [n] \setminus S^{(k)}$. Let $S^{(k+1)} := S^{(k)} \cup \{w^{(k+1)}\}$.*
- *For all $i, j \in [n]$ and k, $M_{ij}^{(k+1)} \le M_{ij}^{(k)} + \gamma \sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)} M_{lj}^{(k)}$.*

With probability at least $1 - 1/n^8$,

$$\sum_{j \neq i, j \notin S^{(k)}} M_{ij}^{(k)} \leq \sigma_1$$

for all $i \notin S^{(k)}$ *and all* $k \leq n/2$.

Proof of Proposition 4.11.1. Inductively assume that for all $i \notin S^{(k)}$ and for all $k \leq n/2$, $\sum_{j \neq i, j \notin S^{(k)}} M_{ij}^{(k)} \leq \sigma_1$. We now use Theorem 4.9.11 to validate this assumption. The third given condition along with the inductive assumption shows that

$$\begin{aligned}
\sum_{j \neq i, j \notin S^{(k+1)}} M_{ij}^{(k+1)} &\leq \left(\sum_{j \neq i, j \notin S^{(k+1)}} M_{ij}^{(k)} \right) + \gamma \left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)} \left(\sum_{j \neq i, j \notin S^{(k+1)}} M_{lj}^{(k)} \right) \right) \\
&= \left(\sum_{j \neq i, j \notin S^{(k+1)}} M_{ij}^{(k)} \right) + \gamma \left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)} \left(\sum_{j \neq i, j \neq w^{(k+1)}, j \notin S^{(k)}} M_{lj}^{(k)} \right) \right) \\
&= \left(\sum_{j \neq i, j \notin S^{(k+1)}} M_{ij}^{(k)} \right) + \gamma \left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)} \left(\sum_{j \neq i, j \neq l, j \notin S^{(k)}} M_{lj}^{(k)} \right) \right) \\
&\leq \left(\sum_{j \neq i, j \notin S^{(k)}} M_{ij}^{(k)} \right) + \gamma \sigma_1 \left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)} \right)
\end{aligned}$$

for $i \notin S^{(k+1)}$. To apply Theorem 4.9.11, we need bounds on the mean, variance, and maximum deviation of each increment. We start with the mean:

$$
\begin{aligned}
&\mathbf{E}\left[\sum_{j\neq i, j\notin S^{(k+1)}} M_{ij}^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&\leq \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} + \gamma\sigma_1 \mathbf{E}\left[\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&\leq \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} + \gamma\sigma_1 \mathbf{E}\left[\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&= \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} + \gamma\sigma_1 \sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} \mathbf{E}\left[Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&= \left(1 + \frac{\gamma\sigma_1}{n-k-1}\right) \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \\
&\leq \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} + \frac{2\gamma\sigma_1^2}{n}
\end{aligned}
$$

Next, we bound the variance:

$$
\begin{aligned}
&\mathbf{Var}\left(\sum_{j\neq i, j\notin S^{(k+1)}} M_{ij}^{(k+1)} \mid S^{(k)}, i\notin S^{(k+1)}\right)\\
&= \mathbf{Var}\left(\sum_{j\neq i, j\notin S^{(k+1)}} M_{ij}^{(k+1)} - \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \mid S^{(k)}, i\notin S^{(k+1)}\right)\\
&\leq \mathbf{E}\left[\left(\sum_{j\neq i, j\notin S^{(k+1)}} M_{ij}^{(k+1)} - \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)}\right)^2 \mid S^{(k)}, i\notin S^{(k+1)}\right]\\
&\leq \gamma^2\sigma_1^2\mathbf{E}\left[\left(\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} Z_l^{(k+1)}\right)^2 \mid S^{(k)}, i\notin S^{(k+1)}\right]\\
&= \gamma^2\sigma_1^2\mathbf{E}\left[\sum_{l\neq i, l\notin S^{(k)}} (M_{il}^{(k)})^2 Z_l^{(k+1)} \mid S^{(k)}, i\notin S^{(k+1)}\right]\\
&\leq \frac{\gamma^2\sigma_1^2}{n-k-1}\sum_{l\neq i, l\notin S^{(k)}} (M_{il}^{(k)})^2\\
&\leq \frac{2\gamma^2\sigma_1^4}{n}
\end{aligned}
$$

Finally, we bound the maximum change:

$$
\begin{aligned}
\sum_{j\neq i, j\notin S^{(k+1)}} M_{ij}^{(k+1)} - \sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} &\leq \gamma\sigma_1\left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)}\right)\\
&\leq \gamma\sigma_1\left(\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)}\right)\\
&\leq \gamma\sigma_1^2
\end{aligned}
$$

conditioned on $i\notin S^{(k+1)}$. Applying Theorem 4.9.11 to the random variables $\{\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)}\}_k$ before the stopping time $\{k : i\in S^{(k)}\}$ shows that

$$\begin{aligned}
&\Pr\left[\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} - \mathbf{E}\left[\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \mid i \notin S^{(k)}\right] \geq \lambda \mid i \notin S^{(k)}\right] \\
&\leq \exp\left(-\frac{\lambda^2}{(n/2)(2\gamma^2\sigma_1^4/n) + (\lambda/3)\gamma\sigma_1^2}\right) \\
&\leq \exp\left(-\frac{\lambda^2}{\gamma^2\sigma_1^4 + (\lambda/3)\gamma\sigma_1^2}\right)
\end{aligned}$$

given the inductive assumption and $k \leq n/2$. Substituting $\lambda = (8\log n)\gamma\sigma_1^2 \leq \sigma_1/4$ gives a probability bound of $1/n^8$. Now, we just have to bound the change in the expectation. Do this by summing up increments:

$$\begin{aligned}
&\mathbf{E}\left[\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \mid i \notin S^{(k)}\right] - \sum_{j\neq i} M_{ij}^{(0)} \\
&= \sum_{r=0}^{k-1} \mathbf{E}\left[\sum_{j\neq i, j\notin S^{(r+1)}} M_{ij}^{(r+1)} - \sum_{j\neq i, j\notin S^{(r)}} M_{ij}^{(r)} \mid i \notin S^{(k)}\right] \\
&= \sum_{r=0}^{k-1} \mathbf{E}\left[\sum_{j\neq i, j\notin S^{(r+1)}} M_{ij}^{(r+1)} - \sum_{j\neq i, j\notin S^{(r)}} M_{ij}^{(r)} \mid i \notin S^{(r+1)}\right] \\
&= \sum_{r=0}^{k-1} \mathbf{E}_{S^{(r)}}\left[\mathbf{E}_{w^{(r+1)}}\left[\sum_{j\neq i, j\notin S^{(r+1)}} M_{ij}^{(r+1)} - \sum_{j\neq i, j\notin S^{(r)}} M_{ij}^{(r)} \mid i \notin S^{(r+1)}, S^{(r)}\right]\right] \\
&\leq 2k\gamma\sigma_1^2/n \\
&\leq \gamma\sigma_1^2 \\
&\leq \sigma_1/4
\end{aligned}$$

By the first given condition and the fact that $\sigma_0 \leq \sigma_1/2$,

$$\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \leq \sigma_1/4 + \sigma_1/4 + \sigma_1/2 \leq \sigma_1$$

with probability at least $1 - 1/n^8$. This completes the verification of the inductive hypothesis and the desired result. □

Proposition 4.11.2. *Let $\{M^{(k)}\}_k \in \mathbb{R}^{n\times n}$ be a sequence of symmetric, nonnegative random matrices, $\{v^{(k)}\}_k \in \mathbb{R}^n$ be a sequence of nonnegative random vectors, $\{Z^{(k)}\}_k \in \{0,1\}^n$, and $\{S^{(k)}\}_k \subseteq [n]$ with the following properties:*

- *For all $i \in [n]$ and all k, $\sum_{j\neq i, j\notin S^{(k)}} M_{ij}^{(k)} \leq \sigma_1$.*
- *The random variables $\{Z^{(k)}\}_k$ are defined by making $Z^{(k+1)}$ the indicator of a uniformly random choice $w^{(k+1)} \in [n] \setminus S^{(k)}$. Let $S^{(k+1)} := S^{(k)} \cup \{w^{(k+1)}\}$.*
- *For all $i \in [n]$, $v_i^{(0)} \leq \tau$.*
- *For all $i \in [n]$ and k, $v_i^{(k+1)} \leq v_i^{(k)} + \gamma \sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)}(v_l^{(k)} + v_i^{(k)})$.*

With probability at least $1 - 1/n^8$,

$$v_i^{(k)} \leq \tau_1$$

for all $i \notin S^{(k)}$ and all $k \leq n/2$.

Proof of Proposition 4.11.2. Inductively assume that for all $i \notin S^{(k)}$ and for all $k \leq n/2$, $v_i^{(k)} \leq \tau_1$. We now use Theorem 4.9.11 to validate this assumption. The fourth given condition along with the inductive assumption shows that

$$\begin{aligned} v_i^{(k+1)} &\leq v_i^{(k)} + \gamma \left(\sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)} (v_l^{(k)} + v_i^{(k)}) \right) \\ &\leq v_i^{(k)} + 2\gamma\tau_1 \left(\sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)} \right) \end{aligned}$$

for $i \notin S^{(k+1)}$. To apply Theorem 4.9.11, we need bounds on the mean, variance, and maximum deviation of each increment. We start with the mean:

$$\begin{aligned} \mathbf{E}\left[v_i^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)} \right] &\leq v_i^{(k)} + 2\gamma\tau_1 \mathbf{E}\left[\sum_{l=1}^n M_{il}^{(k)} Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)} \right] \\ &\leq v_i^{(k)} + 2\gamma\tau_1 \mathbf{E}\left[\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)} \right] \\ &= v_i^{(k)} + 2\gamma\tau_1 \sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} \mathbf{E}\left[Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)} \right] \\ &\leq v_i^{(k)} + \frac{4\gamma\tau_1\sigma_1}{n} \end{aligned}$$

Next, we bound the variance:

$$\begin{aligned}
\mathbf{Var}\left(v_i^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right) &= \mathbf{Var}\left(v_i^{(k+1)} - v_i^{(k)} \mid S^{(k)}, i \notin S^{(k+1)}\right) \\
&\leq \mathbf{E}\left[\left(v_i^{(k+1)} - v_i^{(k)}\right)^2 \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&\leq 4\gamma^2\tau_1^2\mathbf{E}\left[\left(\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)} Z_l^{(k+1)}\right)^2 \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&= 4\gamma^2\tau_1^2\mathbf{E}\left[\sum_{l\neq i, l\notin S^{(k)}} (M_{il}^{(k)})^2 Z_l^{(k+1)} \mid S^{(k)}, i \notin S^{(k+1)}\right] \\
&\leq \frac{4\gamma^2\tau_1^2}{n-k-1}\sum_{l\neq i, l\notin S^{(k)}} (M_{il}^{(k)})^2 \\
&\leq \frac{8\gamma^2\tau_1^2\sigma_1^2}{n}
\end{aligned}$$

Finally, we bound the maximum change:

$$\begin{aligned}
v_i^{(k+1)} - v_i^{(k)} &\leq 2\gamma\tau_1\left(\sum_{l=1}^{n} M_{il}^{(k)} Z_l^{(k+1)}\right) \\
&\leq 2\gamma\tau_1\left(\sum_{l\neq i, l\notin S^{(k)}} M_{il}^{(k)}\right) \\
&\leq 2\gamma\tau_1\sigma_1
\end{aligned}$$

conditioned on $i \notin S^{(k+1)}$. Applying Theorem 4.9.11 to the random variables $\{v_i^{(k)}\}_k$ before the stopping time $\{k : i \in S^{(k)}\}$ shows that

$$\begin{aligned}
\Pr\left[v_i^{(k)} - \mathbf{E}\left[v_i^{(k)} \mid i \notin S^{(k)}\right] \geq \lambda \mid i \notin S^{(k)}\right] &\leq \exp\left(-\frac{\lambda^2}{(n/2)(8\gamma^2\sigma_1^2\tau_1^2/n) + (\lambda/3)2\gamma\sigma_1\tau_1}\right) \\
&\leq \exp\left(-\frac{\lambda^2}{4\gamma^2\sigma_1^2\tau_1^2 + (\lambda/3)2\gamma\sigma_1\tau_1}\right)
\end{aligned}$$

given the inductive assumption and $k \leq n/2$. Substituting $\lambda = (16\log n)\gamma\sigma_1\tau_1 \leq \tau_1/4$ gives a probability bound of $1/n^8$. Now, we just have to bound the change in the expectation. Do this by summing up increments:

$$\begin{aligned}
\mathbf{E}\left[v_i^{(k)} \mid i \notin S^{(k)}\right] - v_i^{(0)} &= \sum_{r=0}^{k-1} \mathbf{E}\left[v_i^{(r+1)} - v_i^{(r)} \mid i \notin S^{(k)}\right] \\
&= \sum_{r=0}^{k-1} \mathbf{E}\left[v_i^{(r+1)} - v_i^{(r)} \mid i \notin S^{(r+1)}\right] \\
&= \sum_{r=0}^{k-1} \mathbf{E}_{S^{(r)}}\left[\mathbf{E}_{w^{(r+1)}}\left[v_i^{(r+1)} - v_i^{(r)} \mid S^{(r)}, i \notin S^{(r+1)}\right]\right] \\
&\leq 4k\gamma\sigma_1\tau_1/n \\
&\leq 2\gamma\sigma_1\tau_1 \\
&\leq \tau_1/4
\end{aligned}$$

By the first given condition and the fact that $\tau \leq \tau_1/2$,

$$v_i^{(k)} \leq \tau_1/4 + \tau_1/4 + \tau_1/2 \leq \tau_1$$

with probability at least $1 - 1/n^8$. This completes the verification of the inductive hypothesis and the desired result. □

Proposition 4.11.5. *Consider a random sequence $\{v^{(k)}\}_{k\geq 0}$ generated as follows. Given $v^{(k)}$,*

- *Pick $\{u_i^{(k)}\}_{i=1}^{\ell_k}$ and $\{w_i^{(k)}\}_{i=1}^{\ell_k}$, with $\sum_{i=1}^{\ell_k} u_i^{(k)} w_i^{(k)} \leq \eta v^{(k)}$*
- *Let $Z^{(k+1)} \in \{0,1\}^{\ell_k}$ denote the indicator of a uniformly random choice over $[\ell_k]$*
- *Pick $v^{(k+1)} \leq v^{(k)} + \gamma \sum_{i=1}^{\ell_k} u_i^{(k)} Z_i^{(k+1)} w_i^{(k)}$*

Let $m_0 = \min_k \ell_k$ and $M_0 = \max_k \ell_k$. Then with probability at least $1 - 2\tau$,

$$v^{(k')} \leq (2\gamma\eta)^\rho v^{(0)}$$

for all $k' \leq m_0 \min(\frac{1}{(\log(M_0^2/\tau))\eta^2\gamma^2}, \frac{1}{200\eta\gamma^2\log(M_0^2/\tau)}(\tau/M_0^2)^{1/\rho})$

Proof. Let $K = m_0 \min(\frac{1}{(\log(M_0^2/\tau))\eta^2\gamma^2}, \frac{1}{200\eta\gamma^2\log(M_0^2/\tau)}(\tau/M_0^2)^{1/\rho})$. Let $\{k_i\}_i$ be the (random variable) subsequence of superscripts for which $v^{(k_i)} - v^{(k_i-1)} > \frac{v^{(k_{i-1})}}{100\log(M_0^2/\tau)}$, with $k_0 := 0$. For all $k \in [k_{i-1}, k_i - 1]$, by Theorem 4.9.11,

$$\Pr[v^{(k)} - v^{(k_{i-1})} \geq \lambda] \leq e^{-\frac{\lambda^2}{K\eta^2\gamma^2(v^{(k_{i-1})})^2/m_0 + \lambda v^{(k_{i-1})}/(100\log(M_0^2/\tau))}}$$

so each such interval can only increase $v^{(k)}$ by a factor of 2 with probability at least $1 - \tau/(M_0^2)$. Therefore, to show the desired concentration bound, we just need to bound the number of k_is. For any $k \in [k_{i-1}, k_i]$, only $100\log(M_0^2/\tau)\eta\gamma^2$ different $u_j^{(k)}w_j^{(k)}$ products can be greater than $v^{(k-1)}/(200\gamma\log(M_0^2/\tau)) \leq v^{(k_{i-1})}/(100\gamma\log(M_0^2/\tau))$. Therefore, the probability that $v_k - v_{k-1} \geq \frac{v^{(k_{i-1})}}{100\log(M_0^2/\tau)}$ is at most

$$\frac{200\log(M_0^2/\tau)\eta\gamma^2}{m_0}$$

This means that the probability that more than ρ k_is is at most

$$\begin{aligned}\sum_{a=(\rho+1)}^{K}\binom{K}{a}\left(\frac{200\log(M_0^2/\tau)\eta\gamma^2}{m_0}\right)^a &\leq 2K^\rho\left(\frac{200\log(M_0^2/\tau)\eta\gamma^2}{m_0}\right)^\rho \\ &\leq 2\tau/M_0^2\end{aligned}$$

For each i, we know that

$$v^{(k_i)} \leq \eta\gamma v^{(k_i-1)}$$

With probability at least $1 - \frac{2\tau}{M_0^2}$, there are at most ρ is, as discussed above. The value can increase by a factor of at most a $(2\gamma\eta)^\rho$ factor in total with probability at least $1 - M_0^2\frac{2\tau}{M_0^2} = 1 - 2\tau$, as desired. □

Proposition 4.11.6. *For any graph J with $S, S' \subseteq V(J)$ and $A, B, D, X \subseteq E(J)$, the families of functions*

- $\mathcal{F}_0 := (\{g_e^{(0)}(H)\}_{e\in X}, \phi^{(0)})$
- $\mathcal{F}_{1,s} := (\{g_e^{(1),X\cap E(H),s}(H)\}_{e\in X}, \phi^{(1)})$
- $\mathcal{F}_{1,s'} := (\{g_e^{(1),X\cap E(H),s'}(H)\}_{e\in X}, \phi^{(1)})$
- $\mathcal{F}_2 := (\{g_e^{(2)}(H)\}_{e\in X}, \phi^{(2)})$
- $\mathcal{F}_{3,s} := (\{g_e^{(3),s}(H)\}_{e\in X}, \phi^{(3)})$
- $\mathcal{F}_{3,s'} := (\{g_e^{(3),s'}(H)\}_{e\in X}, \phi^{(3)})$
- $\mathcal{F}_{4,s} := (\{g_e^{(4),s}(H)\}_{e\in X}, \phi^{(4),s})$
- $\mathcal{F}_{4,s'} := (\{g_e^{(4),s'}(H)\}_{e\in X}, \phi^{(4),s'})$

are flexible for the graph J.

Proof. Consider any minor H of J and consider some edge $f \in X \cap E(H)$ as described in Definition 4.11.3. Fix an edge $e \in X \cap E(H)$. Throughout all proofs, we focus on the contraction case, as the deletion case is exactly the same except with $\texttt{lev}_{\phi(H)}(f)$ replaced with $1 - \texttt{lev}_{\phi(H)}(f)$.

$g_e^{(0)}$ **flexiblity.** By Sherman-Morrison (Propositions 4.9.4 and 4.9.5) and the triangle inequality,

$$\begin{aligned} g_e^{(0)}(H/f) &= \frac{|b_{ss'}^T L^+_{(H/f\setminus D)/(S,S')} b_e|}{\sqrt{r_e}} \\ &\leq \frac{|b_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_e|}{\sqrt{r_e}} + \frac{|b_{ss'}^T L^+_{(H\setminus D)/(S,S')} b_f||b_f^T L^+_{(H\setminus D)/(S,S')} b_e|}{\texttt{lev}_{(H\setminus D)/(S,S')}(f) r_f \sqrt{r_e}} \\ &= g_e^{(0)}(H) + \frac{1}{\texttt{lev}_{\phi^{(0)}(H)}(f)} \frac{|b_e^T L^+_{\phi^{(0)}(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} g_f^{(0)}(H) \end{aligned}$$

as desired.

$g_e^{(1),X\cap E(H),s}$ **and** $g_e^{(1),X\cap E(H),s'}$ **flexibility.** The proof for $g_e^{(1),X,s'}$ is the same as the proof for $g_e^{(1),X,s}$ with s, S and s', S' swapped. Therefore, we focus on $g_e^{(1),X,s}$. By Sherman-Morrison and the triangle inequality,

$$\begin{aligned} g_e^{(1),X\cap E(H/f),s}(H/f) &= \sum_{\{u,v\}\in(X\cap E(H/f))\setminus\{e\}} \frac{|b_e^T L^+_{(H/f\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \\ &= \sum_{\{u,v\}\in(X\cap E(H))\setminus\{e,f\}} \frac{|b_e^T L^+_{(H/f\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \\ &\leq \sum_{\{u,v\}\in(X\cap E(H))\setminus\{e,f\}} \left(\frac{|b_e^T L^+_{(H\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\sqrt{r_e}} \right. \\ &\quad \left. + \frac{|b_e^T L^+_{(H\setminus D)/(S,S')} b_f||b_f^T L^+_{(H\setminus D)/(S,S')}(b_{su} + b_{sv})/2|}{\texttt{lev}_{(H\setminus D)/(S,S')}(f) r_f \sqrt{r_e}} \right) \\ &\leq g_e^{(1),X\cap E(H),s}(H) + \frac{1}{\texttt{lev}_{\phi^{(1)}(H)}(f)} \frac{|b_e^T L^+_{\phi^{(1)}(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} g_f^{(1),X\cap E(H),s}(H) \end{aligned}$$

as desired.

$g_e^{(2)}$ **flexibility.** By Sherman-Morrison and the triangle inequality,

$$
\begin{aligned}
g_e^{(2)}(H/f) &= \sum_{\{u,v\}\in A} \frac{|b_e^T L^+_{(H/f\setminus D)/(S,S')}(b_{su}+b_{sv})|}{\sqrt{r_e}} + \sum_{\{u,v\}\in B} \frac{|b_e^T L^+_{(H/f\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{\sqrt{r_e}} \\
&\le g_e^{(2)}(H) + \sum_{\{u,v\}\in A} \frac{|b_e^T L^+_{(H\setminus D)/(S,S')} b_f||b_f^T L^+_{(H\setminus D)/(S,S')}(b_{su}+b_{sv})|}{\texttt{lev}_{(H\setminus D)/(S,S')}(f) r_f \sqrt{r_e}} \\
&+ \sum_{\{u,v\}\in B} \frac{|b_e^T L^+_{(H\setminus D)/(S,S')} b_f||b_f^T L^+_{(H\setminus D)/(S,S')}(b_{us'}+b_{vs'})|}{\texttt{lev}_{(H\setminus D)/(S,S')}(f) r_f \sqrt{r_e}} \\
&= g_e^{(2)}(H) + \frac{1}{\texttt{lev}_{\phi^{(2)}(H)}(f)} \frac{|b_e^T L^+_{\phi^{(2)}(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} g_f^{(2)}(H)
\end{aligned}
$$

as desired.

$g_e^{(3),s}$ and $g_e^{(3),s'}$ flexibility. We focus on $g_e^{(3),s}$, as $g_e^{(3),s'}$ is the same except that s', S' is swapped with s, S. By Sherman-Morrison and the triangle inequality,

$$
\begin{aligned}
g_e^{(3),s}(H/f) &= \sum_{w\in S'} b_{sw}^T L^+_{(J\setminus D)/S} b_{sw} \sum_{e'\in\partial_H w} \frac{|b_e^T L^+_{(H/f\setminus D)/(S,S')} b_{e'}|}{\sqrt{r_e} r_{e'}} \\
&\le g_e^{(3),s}(H) + \sum_{w\in S'} b_{sw}^T L^+_{(J\setminus D)/S} b_{sw} \sum_{e'\in\partial_H w} \frac{|b_e^T L^+_{(H\setminus D)/(S,S')} b_f||b_f^T L^+_{(H\setminus D)/(S,S')} b_{e'}|}{\sqrt{r_e}\texttt{lev}_{(H\setminus D)/(S,S')}(f) r_f r_{e'}} \\
&= g_e^{(3),s}(H) + \frac{1}{\texttt{lev}_{\phi^{(3)}(H)}(f)} \frac{|b_e^T L^+_{\phi^{(3)}(H)} b_f|}{\sqrt{r_e}\sqrt{r_f}} g_f^{(3),s}(H)
\end{aligned}
$$

as desired.

$g_e^{(4),s}$ and $g_e^{(4),s'}$ flexibility. We focus on $g_e^{(4),s}$, as $g_e^{(4),s'}$ is the same except that s', S' is swapped with s, S. By Sherman-Morrison and the triangle inequality,

$$
\begin{aligned}
g_e^{(4),s}(H/f) &= \sum_{w\in S'}\left(\frac{|b_{sw}^T L^+_{(H/f\setminus D)/S} b_e|}{\sqrt{r_e}}\right)^2 \sum_{e'\in\partial_J w}\frac{b_{ss'}^T L^+_{(J\setminus D)/(S,S')} b_{e'}}{r_{e'}} \\
&\le \sum_{w\in S'}\left(\frac{|b_{sw}^T L^+_{(H\setminus D)/S} b_e|}{\sqrt{r_e}} + \frac{|b_{sw}^T L^+_{(H\setminus D)/S} b_f||b_f^T L^+_{(H\setminus D)/S} b_e|}{\texttt{lev}_{(H\setminus D)/S}(f) r_f\sqrt{r_e}}\right)^2 \sum_{e'\in\partial_J w}\frac{b_{ss'}^T L^+_{(J\setminus D)/(S,S')} b_{e'}}{r_{e'}} \\
&= g_e^{(4),s}(H) + \frac{|b_f^T L^+_{(H\setminus D)/S} b_e|}{\texttt{lev}_{(H\setminus D)/S}(f)\sqrt{r_f}\sqrt{r_e}}\sum_{w\in S'}\left(2\frac{|b_{sw}^T L^+_{(H\setminus D)/S} b_e||b_{sw}^T L^+_{(H\setminus D)/S} b_f|}{\sqrt{r_e}\sqrt{r_f}}\right. \\
&+ \left.\left(\frac{|b_{sw}^T L^+_{(H\setminus D)/S} b_f|}{\sqrt{r_f}}\right)^2 \frac{|b_f^T L^+_{(H\setminus D)/S} b_e|}{\texttt{lev}_{(H\setminus D)/S}(f)\sqrt{r_f}\sqrt{r_e}}\right)\sum_{e'\in\partial_J w}\frac{b_{ss'}^T L^+_{(J\setminus D)/(S,S')} b_{e'}}{r_{e'}} \\
&\le g_e^{(4),s}(H) + \frac{3|b_f^T L^+_{\phi^{(4),s}(H)} b_e|}{(\texttt{lev}_{\phi^{(4),s}(H)}(f))^2\sqrt{r_f}\sqrt{r_e}}(g_e^{(4),s}(H) + g_f^{(4),s}(H))
\end{aligned}
$$

where the last inequality follows from AM-GM and $\frac{|b_e^T L^+_{(H\setminus D)/S} b_f|}{\sqrt{r_e}\sqrt{r_f}} \le 1$. This is the desired result.

□

B.7 Parameters

Throughout this paper, there are many parameters and constants that are hidden in $m^{o(1)}$ and $\alpha^{o(1)}$ dependencies. For any values of σ_0 and σ_1, our main algorithm takes

$$f(\sigma_0,\sigma_1)m^{1+1/\sigma_1}\alpha^{1/(\sigma_0+1)}$$

time, where $f(\sigma_0,\sigma_1) \le 2^{\sqrt{\log n}\sigma_1^{100\sigma_0}}$. Setting $\sigma_0 = \sigma_1 = (\log\log n)/(400\log\log\log n)$ yields an algorithm with runtime $m^{o(1)}m^{1+(\log\log\log n)/(400\log\log n)}\alpha^{1/(\log\log n)} = m^{1+o(1)}\alpha^{o(1)}$, as desired.

To establish this runtime, we need bounds on the values of each of the following parameters that do not depend on m^{1/σ_1} or $\alpha^{1/(\sigma_0+1)}$:

- The conductivity parameter ζ
- The modifiedness parameter τ
- The boundedness parameter κ
- The number of clans ℓ

- The carving parameter $\mu_{\text{carve}} = (2000\mu_{\text{app}})^{\sigma_1^{4\sigma_0}}$
- Smaller parameters:
  - The appendix bounding parameter $\mu_{\text{app}} = 2^{100(\log n)^{3/4}}$
  - The fixing bounding parameter $\mu_{\text{con}} = 2^{100\sqrt{\log n}}$
  - The empire radius bounding parameter $\mu_{\text{rad}} = 2^{1000(\log n)^{3/4}}$
  - Lemma 4.4.18 parameter: $\mu_{\text{mod}} = 200\mu_{\text{con}}\mu_{\text{app}}\mu_{\text{carve}}$

The parameter μ_{app} is used for simplicity to bound all $m^{o(1)}\alpha^{o(1)}$ terms that appear in the appendix. μ_{con} is used in the "Size" bound of Lemma 4.8.2. The carving parameter μ_{carve} arises from growing a ball around each $\mathcal{Q}_i$ in the conditioning digraph with radius $\sigma_1^i \le \sigma_1^{\sigma_0}$. By the "Temporary condition" of edges in conditioning digraphs, a part Q with an edge from P is only $8\mu_{\text{app}}$ times farther from $\mathcal{Q}_i$ than P is, so all parts within distance $\sigma_1^{\sigma_0}$ in the conditioning digraph must be within distance $(8\mu_{\text{app}})^{\sigma_1^{\sigma_0}}\alpha^{i/(\sigma_0+1)}r_{min} \le \mu_{\text{carve}}\alpha^{i/(\sigma_0+1)}r_{min}$. For more details, see Section 4.7.

The conductivity, modifiedness, boundedness, and number of clan parameters increase during each iteration of the while loop in `ExactTree`. We designate maximum values $\zeta_{\max}$, $\tau_{\max}$, $\kappa_{\max}$, and $\ell_{\max}$ for each. ℓ does not depend on ζ, τ, and κ because it only increases by more than a $(\log m)$-factor in `RebuildEmpire`, which increases it by a factor of μ_{app}. κ increases by at most a factor of $\ell \le \ell_{\max}$ during each iteration. τ and ζ depend on one another. τ additively increases by $\tau\mu_{\text{app}} + \zeta\mu_{\text{carve}}\mu_{\text{app}}$ during each iteration, while ζ multiplicatively increases by a constant factor and additively increases by $\ell\mu_{\text{app}}$. Since the while loop in `ExactTree` only executes for $\sigma_1^{\sigma_0}$ iterations, values of

- $\zeta_{\max} = ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{8\sigma_0}}$
- $\tau_{\max} = ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{8\sigma_0}}$
- $\kappa_{\max} = ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{4\sigma_0}}$
- $\ell_{\max} = ((\log m)\mu_{\text{app}})^{(2\sigma_1)^{2\sigma_0}}$

suffice. Therefore, the runtime of the partial sampling step (the bottleneck) is at most

$$2^{\sqrt{\log n}\sigma_1^{100\sigma_0}} m^{1+1/\sigma_1}\alpha^{1/(\sigma_0+1)} \le m^{1+o(1)}\alpha^{o(1)}$$

as desired.

B.8 List of Definitions

B.9 Almost-linear time, ϵ-approximate random spanning tree generation for arbitrary weights

In this section, we obtain a $m^{1+o(1)}\epsilon^{-o(1)}$-time algorithm for generating an ϵ-approximate random spanning tree. Specifically, we output each spanning tree according to a distribution with total variation distance at most ϵ from the real one.

We do this by reducing the problem to our main result; an $m^{1+o(1)}\alpha^{1/(\sigma_0+1)}$-time algorithm for generating an exact random spanning tree. The reduction crudely buckets edges by their resistance and computes connected components of low-resistance edges. Deleting edges with very high resistances does not affect the marginal of the random spanning tree distribution with respect to low-resistance edges very much. Therefore, sampling a random spanning tree in the graph with high-resistance edges deleted yields an ϵ-approximation random tree for that marginal in the entire graph.

To make this approach take $m^{1+o(1)}\epsilon^{-o(1)}$ time, we just need to make sure that conditioning on a connected component F of low-resistance edges does not take much longer longer than $O(|F|m^{o(1)})$ time. In other words, at most $O(|F|m^{o(1)})$ medium-resistance edges should exist. This can be assured using ball growing on $\sqrt{\sigma_0}$ different scales. This results in a graph being conditioned on with $\alpha \leq m^{10\sqrt{\sigma_0}}$ and size at most $|F|m^{1/\sqrt{\sigma_0}}$, where F is the set being conditioned on. Therefore, to condition on $|F|$ edges, the algorithm only needs to do $|F|^{1+o(1)}m^{1/\sqrt{\sigma_0}}m^{10\sqrt{\sigma_0}/(\sigma_0+1)}$ work, as desired.

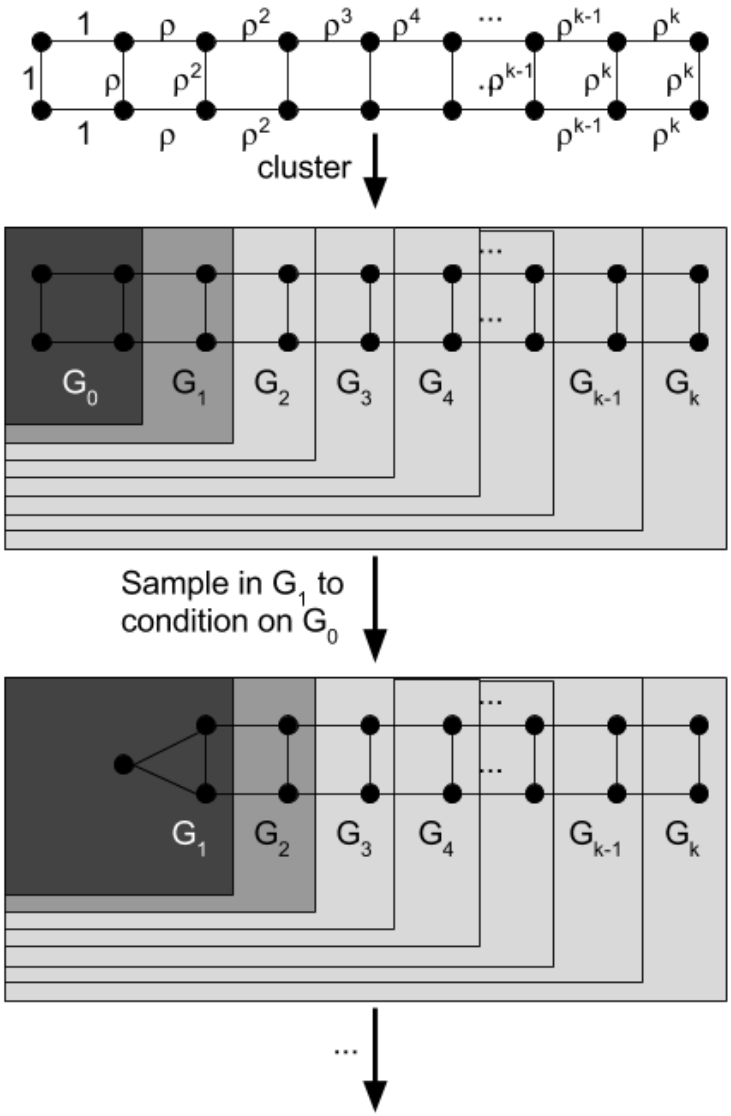

Figure B.9.1: The first iteration of the while loop of the `ApxTree` algorithm. The amount of work done during each iteration is not much larger than the amount of progress (reduction in graph size) made.

Now, we implement this approach as follows:

Algorithm 37: `ApxTree`(G)

$\rho \leftarrow (m/\epsilon)^{10}$
$T_{return} \leftarrow \emptyset$
// can compute connected components of all G_is in near-linear time with union-find
$G_i \leftarrow$ the subgraph of G consisting of edges with resistance at most $\rho^i r_{min}$
$i^* \leftarrow 0$
while $E(G) \neq \emptyset$ **do**
  while $|E(G_{i^*+1})| > m^{1/\sqrt{\sigma_0}}|E(G_{i^*})|$ **do**
    $i^* \leftarrow i^* + 1$
  end
  foreach *connected component* C *of* G_{i^*+1} **do**
    $T \leftarrow$ `ExactTree`$(G_{i^*+1}[C])$
    contract edges in G in the intersection of T with $E(G_{i^*})$ and add these edges to T_{return}
    delete edges from G that are in $E(G_{i^*})$ but not T
  end
end
return T_{return}

The correctness of this algorithm relies on the following claim:

Proposition B.9.1. *Consider a graph I and an edge $e \in E(I)$. Let I' be the subgraph of I of edges with resistance at most ρr_e^I. Then*

$$\mathtt{Reff}_I(e) \leq \mathtt{Reff}_{I'}(e) \leq (1 + 2\epsilon/m^9)\mathtt{Reff}_I(e)$$

Proof. The lower bound follows immediately from Rayleigh monotonicity, so we focus on the upper bound. Let f be the unit electrical flow between the endpoints of e. It has energy $\mathtt{Reff}_{I'}(e)$. Each edge $g \in E(I) \setminus E(I')$ has

$$f_g \leq \frac{b_e^T L_I^+ b_e}{r_g^I} \leq r_e^I/(\rho r_e^I) \leq \epsilon/m^{10}$$

Therefore, deleting edges in $I \setminus I'$ and removing the associated flow results in a flow between the endpoints of e with energy at most $b_e^T L_I^+ b_e$ that ships at least $1 - m(\epsilon/m^{10}) = 1 - \epsilon/m^9$ units of flow. Scaling the flow up by $1/(1 - \epsilon/m^9)$ yields the desired result. □

Theorem 4.1.2. *Given a weighted graph G and $\epsilon \in (0, 1)$, a random spanning tree T of G can be sampled from a distribution with total variation distance at most ϵ from the uniform distribution in time $m^{1+o(1)}\epsilon^{-o(1)}$ time.*

Proof of Theorem 4.1.2 given that `ExactTree` *satisfies Theorem 4.1.1.*
Correctness. At each step, the algorithm contracts a forest in G, so it maintains the fact that G is connected. Since the algorithm terminates only when $E(G) = \emptyset$, T_{return} is a tree. We now compute the total variation distance between the output distribution and the real spanning tree distribution for G.

To do this, we set up a hybrid argument. Let $(e_i)_{i=1}^m$ be an ordering of the edges in G in increasing order of resistance. Notice that the subset $E(G_j) \subseteq E(G)$ is a prefix of this list for all j, which means that conditioning on a G_j eliminates a prefix of $(e_i)_{i=1}^m$.

For $k \in \{0, 1, \ldots, m\}$, let $\mathcal{D}_k$ denote the distribution over spanning trees of G obtained by sampling a tree T using `ApxTree`, conditioning on its intersection with the set $\{e_1, e_2, \ldots, e_k\}$ to obtain a graph G', and sampling the rest of T using the real uniform random spanning tree distribution of the graph G'. $\mathcal{D}_0$ is the uniform distribution over spanning trees of G, while $\mathcal{D}_m$ is the distribution over trees output by `ApxTree`. Therefore, to complete the correctness proof, we just need to show that the total variation distance between $\mathcal{D}_0$ and $\mathcal{D}_m$ is at most ϵ. We do this by bounding the total variation distance between $\mathcal{D}_k$ and $\mathcal{D}_{k+1}$ for all $k \leq m-1$.

Consider a spanning tree T' of G. We now compute its probability mass in the distributions $\mathcal{D}_k$ and $\mathcal{D}_{k+1}$. Consider the i^* in the `ApxTree` algorithm for which $e_{k+1} \in E(G_{i^*}) \setminus E(G_{i^*_{prev}})$, where i^*_{prev} is the previous value for which `ApxTree` conditions on $G_{i^*_{prev}}$ in Lines 11 and 12. Line 10 extends the partial sample from $G_{i^*_{prev}}$ to G_{i^*} by sampling from the true uniform distribution of G_{i^*+1} conditioned on the partial sample for $G_{i^*_{prev}}$ matching T'. By Theorem 4.3.8, one could equivalently sample from this distribution by sampling edges with probability equal to their leverage score in G_{i^*+1} conditioned on the samples for the previous edges. Applying this reasoning shows that

$$\begin{aligned}\Pr_{T_a \sim \mathcal{D}_k}[T_a = T'] = &\Pr_{T_b \sim \texttt{ApxTree}(G)}[E(T_b) \cap \{e_i\}_{i=1}^k = E(T') \cap \{e_i\}_{i=1}^k] \\ &\Pr_{T_c \sim H_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}] \\ &\Pr_{T_d \sim H_{k+1}}[E(T_d) \cap \{e_i\}_{i=k+2}^m = E(T') \cap \{e_i\}_{i=k+2}^m]\end{aligned}$$

and that

$$\begin{aligned}\Pr_{T_a \sim \mathcal{D}_{k+1}}[T_a = T'] = &\Pr_{T_b \sim \texttt{ApxTree}(G)}[E(T_b) \cap \{e_i\}_{i=1}^k = E(T') \cap \{e_i\}_{i=1}^k] \\ &\Pr_{T_c \sim H'_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}] \\ &\Pr_{T_d \sim H_{k+1}}[E(T_d) \cap \{e_i\}_{i=k+2}^m = E(T') \cap \{e_i\}_{i=k+2}^m]\end{aligned}$$

where H_k is the graph obtained by conditioning on the partial sample of T' in $\{e_1, e_2, \ldots, e_k\}$ and H'_k is obtained from H_k by removing all edges of H_k that are not in G_{i^*+1}. By definition of G_{i^*+1} and the fact that $e_k \in E(G_{i^*})$, H'_k contains all edges in H_k

with resistance at most $\rho r_{e_{k+1}}$. Therefore, by Proposition B.9.1 applied after using Theorem 4.3.8 to write probabilities as leverage and nonleverage scores,

$$|\Pr_{T_c \sim H_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}] - \Pr_{T_c \sim H'_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}]| \leq 2\epsilon/m^9$$

Therefore, the total variation distance between $\mathcal{D}_k$ and $\mathcal{D}_{k+1}$ is

$$\begin{aligned}
&\sum_{\text{spanning trees } T' \text{ of } G} |\Pr_{T_a \sim \mathcal{D}_k}[T_a = T'] - \Pr_{T_a \sim \mathcal{D}_{k+1}}[T_a = T']| \\
&= \sum_{\text{spanning trees } T' \text{ of } G} \Bigg(\\
&\left| \Pr_{T_c \sim H_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}] - \Pr_{T_c \sim H'_k}[E(T_c) \cap \{e_{k+1}\} = E(T') \cap \{e_{k+1}\}] \right| \\
&\times \Pr_{T_b \sim \texttt{ApxTree}(G)}[E(T_b) \cap \{e_i\}_{i=1}^k = E(T') \cap \{e_i\}_{i=1}^k] \\
&\times \Pr_{T_d \sim H_{k+1}}[E(T_d) \cap \{e_i\}_{i=k+2}^m = E(T') \cap \{e_i\}_{i=k+2}^m] \Bigg) \\
&\leq \frac{2\epsilon}{m^9} \sum_{\text{spanning trees } T' \text{ of } G} \Bigg(\Pr_{T_b \sim \texttt{ApxTree}(G)}[E(T_b) \cap \{e_i\}_{i=1}^k = E(T') \cap \{e_i\}_{i=1}^k] \\
&\Pr_{T_d \sim H_{k+1}}[E(T_d) \cap \{e_i\}_{i=k+2}^m = E(T') \cap \{e_i\}_{i=k+2}^m] \Bigg) \\
&\leq \frac{4\epsilon}{m^9} \sum_{\text{restrictions of spanning trees } T' \text{ of } G \text{ to } \{e_i\}_{i=1}^k} \left(\Pr_{T_b \sim \texttt{ApxTree}(G)}[E(T_b) \cap \{e_i\}_{i=1}^k = E(T') \cap \{e_i\}_{i=1}^k] \right) \\
&\leq \frac{4\epsilon}{m^9}
\end{aligned}$$

Therefore, the total variation distance between $\mathcal{D}_0$ and $\mathcal{D}_m$ is at most $m(4\epsilon/m^9) \leq \epsilon$, as desired.

Runtime. When the algorithm samples a tree in G_{i^*+1}, it contracts or deletes (removes) all edges in G_{i^*}. Line 6 ensures that $|E(G_{i^*+1})| \leq m^{1/\sqrt{\sigma_0}}|E(G_{i^*})|$ whenever conditioning occurs. Furthermore, since $G_i \subseteq G_{i+1}$ for all i, the innermost while loop only executes $\sqrt{\sigma_0}$ times for each iteration of the outer while loop. This means that G_{i^*+1} has $\beta \leq \rho^{\sqrt{\sigma_0}}$ since all lower edges where in a prior G_{i^*}. By Theorem 4.1.1, `ExactTree` takes at most

$$|E(G_{i^*+1})|^{1+o(1)}(\rho^{\sqrt{\sigma_0}})^{1/\sigma_0} \leq |E(G_{i^*})|(m/\epsilon)^{o(1)+11/\sqrt{\sigma_0}}$$

time. Since the G_{i*}s that are conditioned on each time are edge-disjoint (as they are contracted/deleted immediately), the total size of the G_{i*}s over the course of the algorithm is m, which means that the total runtime of the outer while loop is at most $m^{1+o(1)+11/\sqrt{\sigma_0}}\epsilon^{-o(1)} = m^{1+o(1)}\epsilon^{-o(1)}$, as desired. □

Appendix C

Spectral Subspace Sparsification Appendix

C.1 Bounds on eigenvalues of Laplacians and SDDM matrices

We first give upper bounds on the traces of the inverses of Laplacians and their submatrices.

Lemma C.1.1. *For any Laplacian L_G and $S \subset V(G)$,*

$$\mathrm{Tr}\left(L_G^+\right) \leq n^2/w_{\min}, \tag{C.1}$$
$$\mathrm{Tr}\left((L_G)_{S,S}^{-1}\right) \leq n^2/w_{\min}. \tag{C.2}$$

Proof of Lemma C.1.1. Let $T := V(G) \setminus S$. The first upper bound follows by

$$\mathrm{Tr}\left(L_G^+\right) = \frac{1}{n}\sum_{u,v\in V} b_{u,v}^T L_G^+ b_{u,v} \leq \frac{1}{n}(n^3\frac{1}{w_{\min}}) \leq \frac{n^2}{w_{\min}}. \tag{C.3}$$

The second upper bound follows by

$$\mathrm{Tr}\left((L_G)_{S,S}^{-1}\right) = \sum_{u\in S} b_{u,T}^T L_{G/T}^+ b_{u,T} \leq n \cdot \frac{n}{w_{\min}} \leq \frac{n^2}{w_{\min}}. \tag{C.4}$$

The first inequalities of (C.3) and (C.4) both follow from the fact that the effective resistance is at most the shortest path. □

Lemma 5.2.7. *For any Laplacian L_G and $S \subset V(G)$,*

$$\lambda_2(L_G) \geq w_{\min}/n^2, \tag{5.1}$$
$$\lambda_{\min}\left((L_G)_{S,S}\right) \geq w_{\min}/n^2, \tag{5.2}$$
$$\lambda_{\max}\left((L_G)_{S,S}\right) \leq \lambda_{\max}(L_G) \leq n w_{\max}. \tag{5.3}$$

Proof of Lemma 5.2.7. For the upper bounds, we have

$$\lambda_{\max}\left((L_G)_{S,S}\right) \leq \lambda_{\max}\left(L\right) \leq \lambda_{\max}\left(w_{\max}L_{K_n}\right) \leq nw_{\max},$$

where the first inequality follows from Cauchy interlacing, and K_n denotes the complete graph of n vertices.

For the lower bounds, we have

$$\lambda_2(L_G) \geq 1/\mathrm{Tr}\left(L_G^+\right) \geq w_{\min}/n^2,$$
$$\lambda_{\min}\left(L_{S,S}\right) \geq 1/\mathrm{Tr}\left((L_G)_{S,S}^{-1}\right) \geq w_{\min}/n^2.$$

□

C.2 Bounds on 2-norms of some useful matrices

Lemma 5.2.8. *The following upper bounds on the largest singular values/eigenvalues hold:*

$$\sigma_{\max}(W_G^{1/2}B_G) \leq (nw_{\max})^{1/2}, \tag{5.4}$$
$$\lambda_{\max}(SC(L_G,S)) \leq nw_{\max}, \tag{5.5}$$
$$\sigma_{\max}((L_G)_{S,T}) = \sigma_{\max}((L_G)_{T,S}) \leq nw_{\max}, \tag{5.6}$$

where $T := V(G) \setminus S$.

Proof of Lemma 5.2.8. The largest singular value of $W_G^{1/2}B_G$ follows by

$$\sigma_{\max}(W_G^{1/2}B_G) \leq (\lambda_{\max}(L_G))^{1/2} \leq (nw_{\max})^{1/2} \qquad \text{by (5.3)}.$$

The largest eigenvalue of Schur complements follows by

$$\lambda_{\max}(SC(L_G,S)) \leq \lambda_{\max}((L_G)_{S,S}) \leq nw_{\max} \qquad \text{by (5.3)}.$$

The largest singular value of $(L_G)_{S,T}$ follows by

$$\begin{aligned}
\sigma_{\max}((L_G)_{S,T}) \leq& \left(\lambda_{\max}\left((L_G)_{S,T}^T(L_G)_{T,S}\right)\right)^{1/2} \\
\leq& \left(nw_{\max}\cdot\lambda_{\max}\left((L_G)_{S,T}^T(L_G)_{T,T}^{-1}(L_G)_{T,S}\right)\right)^{1/2} \qquad \text{by (5.3)} \\
\leq& \left(nw_{\max}\cdot\lambda_{\max}\left((L_G)_{S,S}\right)\right)^{1/2} \quad \text{since } SC(L_G,S) \text{ is positive semi-definite} \\
\leq& nw_{\max} \qquad \text{by (5.3)}.
\end{aligned}$$

□

C.3 Bounds on errors of LaplSolve using ℓ_2 norms

Lemma 5.2.10. *For any Laplacian L_G, vectors $x, \tilde{x} \in \mathbb{R}^n$ both orthogal to 1, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_{L_G} \leq \epsilon \|x\|_{L_G},$$

the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\|.$$

Proof of Lemma 5.2.10. The error follows by

$$\begin{aligned}\|x - \tilde{x}\| \leq & n w_{\min}^{-1/2} \|x - \tilde{x}\|_{L_G} \qquad \text{by (5.1)} \\ \leq & n w_{\min}^{-1/2} \epsilon \|x\|_{L_G} \leq n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \qquad \text{by (5.3)}\end{aligned}$$

□

Lemma 5.2.11. *For any Laplacian L_G, $S \subset V$, vectors $x, \tilde{x} \in \mathbb{R}^{|S|}$, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_M \leq \epsilon \|x\|_M,$$

where $M := (L_G)_{S,S}$, the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\|.$$

Proof of Lemma 5.2.11. The error follows by

$$\begin{aligned}\|x - \tilde{x}\| \leq & n w_{\min}^{-1/2} \|x - \tilde{x}\|_M \qquad \text{by (5.2)} \\ \leq & n w_{\min}^{-1/2} \epsilon \|x\|_M \leq n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \qquad \text{by (5.3)}\end{aligned}$$

□

C.4 Split subroutines

Proposition 5.3.4. *There is a linear-time algorithm $(I, \mathcal{P}) \leftarrow \texttt{Split}(H)$ that, given a graph H, produces a graph I with $V(H) \subseteq V(I)$ and a set of pairs of edges $\mathcal{P}$ with the following additional guarantees:*

- *(Electrical equivalence) For all* $x \in \mathbb{R}^{V(I)}$ *that are supported on* $V(H)$, $x^T L_I^+ x = x_H^T L_H^+ x_H$.
- *(Bounded leverage scores) For all* $e \in E(I)$, $\texttt{lev}_I(e) \in [3/16, 13/16]$
- *($\mathcal{P}$ description) Every edge in I is in exactly one pair in $\mathcal{P}$. Furthermore, there is a bijection between pairs $(e_0, e_1) \in \mathcal{P}$ and edges $e \in E(H)$ for which either (a) e_0, e_1 and e have the same endpoint pair or (b) $e_0 = \{u, w\}$, $e_1 = \{w, v\}$, and $e = \{u, w\}$ for some degree 2 vertex w.*

Algorithm 38: `Split`(H)

Input: a graph H
Output: a graph I with a pair of edges for each edge in H and a set of paired edges in $\mathcal{P}$

```
I ← H
P ← ∅
foreach edge e ∈ E(H) do
    if 1/16-JL-approximation to lev_H(e) ≥ 1/2 then
        Replace e = {u, v} ∈ E(I) with two edges e_0 = {u, v} and e_1 = {u, v} with
        r_{e_0} = r_{e_1} = 2 r_e
        Add the pair (e_0, e_1) to P
    else
        Add a vertex w to V(I)
        Replace e = {u, v} ∈ E(I) with two edges e_0 = {u, w} and e_1 = {w, v} with
        r_{e_0} = r_{e_1} = r_e / 2
        Add the pair (e_0, e_1) to P
    end
end
return (I, P)
```

Proof. **Electrical equivalence.** Two parallel edges with resistance $2r_e$ are electrically equivalent to one edge with resistance r_e. Two edges with resistance $r_e/2$ in series are equivalent to one edge with resistance r_e. Therefore, both ways of replacing edges in H with pairs of edges in I result in an electrically equivalent graph.

Bounded leverage scores. For an edge e that is replaced with two series edges e_0 and e_1,

$$\texttt{lev}_I(e_0) = \texttt{lev}_I(e_1) = \frac{1}{2} + \frac{\texttt{lev}_H(e)}{2} \in [1/2, 3/4]$$

since $\texttt{lev}_H(e) \in [0, 1/2(1 + 1/16)]$. For an edge e that is replaced with two parallel edges e_0 and e_1,

$$\texttt{lev}_I(e_0) = \texttt{lev}_I(e_1) = \texttt{lev}_H(e)/2 \in [1/4, 1/2]$$

since $\texttt{lev}_H(e) \in [1/2(1-1/16), 1]$. Since all edges in I result from one of these operations, they all have leverage score in $[3/16, 13/16]$, as desired.

$\mathcal{P}$ **description.** (a) describes edges resulting from parallel replacements, while (b) describes edges reesulting from series replacements.

Runtime. Estimating the leverage scores takes near-linear time [90]. Besides this, the algorithm just does linear scans of the graph. Therefore, it takes near-linear time.

□

Proposition 5.3.5. *There is a linear-time algorithm* $H \leftarrow \texttt{Unsplit}(I, \mathcal{P})$ *that, given a graph* I *and a set of pairs* $\mathcal{P}$ *of edges in* I*, produces a minor* H *with* $V(H) \subseteq V(I)$ *and the following additional guarantees:*

- *(Electrical equivalence) For all* $x \in \mathbb{R}^{V(I)}$ *that are supported on* $V(H)$*,* $x^T L_I^+ x = x_H^T L_H^+ x_H$*.*
- *(Edges of* H*) There is a surjective map* $\phi : E(I) \to E(H)$ *from non-self-loop,non-leaf edges of* I *such that for any pair* $(e_0, e_1) \in \mathcal{P}$*,* $\phi(e_0) = \phi(e_1)$*. Furthermore, for each* $e \in E(H)$*, either (a)* $\phi^{-1}(e) = e$*, (b)* $\phi^{-1}(e) = \{e_0, e_1\}$*, with* $(e_0, e_1) \in \mathcal{P}$ *and* e_0, e_1 *having the same endpoints as* e *or (c)* $\phi^{-1}(e) = \{e_0, e_1\}$*, with* $(e_0, e_1) \in \mathcal{P}$ *and* $e_0 = \{u, w\}, e_1 = \{w, v\}$*, and* $e = \{u, v\}$ *for a degree 2 vertex* w*.*

Algorithm 39: $\texttt{Unsplit}(I, \mathcal{P})$

Input: a graph I and a set of nonintersecting pairs of edges $\mathcal{P}$
Output: a graph H with each pair unsplit to a single edge

```
H ← I
foreach pair (e_0, e_1) ∈ P do
    if e_0 and e_1 have the same endpoints {u, v} and e_0, e_1 ∈ E(I) then
        Replace e_0 and e_1 in H with one edge e = {u, v} with r_e = 1/(1/r_{e_0} + 1/r_{e_1})
    else if e_0 = {u, w}, e_1 = {w, v}, w has degree 2, and e_0, e_1 ∈ E(I) then
        Replace e_0 and e_1 in H with one edge e = {u, v} with r_e = r_{e_0} + r_{e_1}
    end
end
```

Proof. **Electrical equivalence.** Two parallel edges with resistance r_{e_0} and r_{e_1} are electrically equivalent to one edge with resistance $1/(1/r_{e_0} + 1/r_{e_1})$. Two edges with resistance r_{e_0} and r_{e_1} in series are equivalent to one edge with resistance $r_{e_0} + r_{e_1}$. Therefore, both ways of replacing pairs of edges in I with single edges in H result in an electrically equivalent graph.

Edges of H**.** Since the pairs in $\mathcal{P}$ do not intersect, the map $\phi(e_i) = e$ that maps an edge e_i, $i \in \{0, 1\}$ to the e as described in the foreach loop is well-defined. Since each $(e_0, e_1) \in \mathcal{P}$ pair is assigned to the same edge e, $\phi(e_0) = \phi(e_1) = e$. Each edge in the output graph H originates from the initialization of H to I, the if statement, or the else statement. These are type (a),(b), and (c) edges respectively. Therefore, ϕ satisfies the required conditions.

Runtime. The algorithm just requires a constant number of linear scans over the graph. □