Chennai Mathematical Institute

Masters Thesis

Random Spanning Trees

Author:
Bhishmaraj S

Supervisor:
Dr. Samir Datta

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Computer Science at CMI

June 14, 2020

Declaration of Authorship

I, Bhishmaraj S, declare that this thesis titled, "Random Spanning Trees" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself .

Signed:

Date:

CHENNAI MATHEMATICAL INSTITUTE

Abstract

Samir Datta
Computer Science at CMI

Master of Science

Random Spanning Trees

by Bhishmaraj S

We study the problem of sampling a spanning tree of a graph with uniform probability. We survey some of the existing algorithms in the literature and explain a recent paper Harvey and Xu [18] in detail. Our initial motivation was to study the dynamic complexity of uniformly sampling spanning trees.

Acknowledgements

I would like to thank my advisor Prof. Samir Datta for his patience and guidance in advicing me . I would also like to thank my family members and friends without whose support this would have not been possible. I would also like to thank Kishlaya for the disscussions on random walks and spectral graph theory.

Contents

List of Figures

1 Introduction

Spanning trees have been a central object of study in graph theory for a long time. Kirchhoff [21] established a linear algebraic relationship between spanning trees of graphs and determinants while studying electric networks. We are interested in the following problem.

Problem 1 (Uniform random spanning tree)**.** *Given an undirected connected graph $G = (V, E)$, sample a spanning tree T with probability $\frac{1}{|\mathcal{T}|}$ where $\mathcal{T}$ denotes the set of all spanning trees of G.*

Sampling spanning trees happens to be a primitive used in various problems such as

- Constructing expanders ([15], [12])
- Approximation algorithms for the travelling salesman problem([14], [2])
- Graph Sparcification ([13], [10])
- Analysis of network reliability ([5],[24], [7])
- Sequence shuffling problem in Bioinformatics ([19])

Bar-Ilan and Zernik [3] proposed a distributed algorithm for this problem. Recently Russo, Teixeira, and Francisco [26] implemented some of the random walk algorithms and have compared their efficiencies.

1.0.1 Algorithms for sampling random spanning trees

There has been a lot of progress in this problem for the past 40 years. These algorithms can be broadly classified into 3 categories.

Matrix Based

These algorithms are based on Kirchoff Matrix Tree theorem and involve computing determinants of the laplacian matrix of the graph to sample spanning trees. These notions would be made clear in later part of the thesis.

- Random Spanning Tree, [16]
- Unranking and ranking spanning trees of a graph, [6]
- Generating random combinatorial objects, [22]
- Two Algorithms for Unranking Arborescences, [8]
- Generating random spanning trees via fast matrix multiplication, [18]

Random Walk Based

These algorithms simulate variants of random walk on the input graph and consutructs a spanning tree from this simulation.

- Generating random spanning trees, [4]
- The random walk construction of uniform spanning trees and uniform labelled trees, [1]
- Generating random spanning trees more quickly than the cover time, [31]
- How to Get a Perfectly Random Sample from a Generic Markov Chain and Generate a Random Spanning Tree of a Directed Graph, [25]

Approximation Algorithms

These are more recent algorithms which sample a uniform spanning tree with a high probability. The main theme here is to employ tools from approximation algorithms.

- Faster generation of random spanning trees, [20]
- Fast Generation of Random Spanning Trees and the Effective Resistance Metric, [23]
- Sampling Random Spanning Trees Faster than Matrix Multiplication, [11]
- An Almost-Linear Time Algorithm for Uniform Random Spanning Tree Generation, [27]

1.0.2 Motivation

The main motivation with which we started exploring this problem was to extend sampling random spanning trees to a dynamic setting. We explored some ideas on how this could be done faster than computing it from scratch. This is explained in the **Conclusion**.

1.0.3 Structure of the thesis

First we define the basic ideas which would be used later. Then we give a brief summary of the algorithms of some of the papers listed above. Then we explain the Harvey and Xu [18] algorithm in detail.

2 Background

To make the exposition self contained I have chosen the results which are relevant to understanding the Harvey, Xu algorithm. Most of these results are taken from Vishnoi [29]

2.1 Preliminary Linear Algebra

Fact 1. *If A is a $n \times n$ real symmetric matrix, then all it's eigenvalues are real.*

Fact 2 (Eigenvectors of different eigenvalues are orthogonal)**.** *Let λ_i and λ_j be two eigenvalues of a symmetric matrix A and $\boldsymbol{u}_i, \boldsymbol{u}_j$ be it's corresponding eigenvectors. If $\lambda_i \neq \lambda_j$ then $\langle \boldsymbol{u}_i, \boldsymbol{u}_j \rangle = 0$*

Fact 3 (Min-Max Characterizations of Eigenvalues)**.** *If A is a $n \times n$ real symmetric matrix, then the largest eigenvalue of A is*

$$\lambda_n(A) = \max_{v \in \mathbb{R}^n \setminus \{0\}} \frac{v^T A v}{v^T v}$$

$$\lambda_1(A) = \min_{v \in \mathbb{R}^n \setminus \{0\}} \frac{v^T A v}{v^T v}$$

Fact 4 (Courant-Weyl-Fisher min-max principle)**.**

$$\lambda_k(A) = \min_{\substack{v \in \mathbb{R}^n \setminus \{0\} \\ v^T u_i = 0 \ \forall i \in \{1,\ldots,k-1\}}} \frac{v^T A v}{v^T v}$$

$$\lambda_k(A) = \max_{\substack{v \in \mathbb{R}^n \setminus \{0\} \\ v^T u_i = 0 \ \forall i \in \{1,\ldots,k-1\}}} \frac{v^T A v}{v^T v}$$

Fact 5 (Positive Semi Definite)**.** *A matrix A is said to be positive semi definite (PSD) if $\lambda_1(A) \geq 0$. A is said to be positive definite if $\lambda_1(A) > 0$*

2.2 Laplacian of a Graph

Definition 1 (Laplacian)**.** *For an undirected unweighted graph $G = (V, E)$ the Laplacian L_G is a $|V| \times |V|$ matrix defined as*

$$L_G(i,j) = \begin{cases} -1 & \text{if } (i,j) \in E \\ deg(i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

It can also be seen that

$$L_G = D - A$$

where D is a diagonal matrix with diagonal entries as degree of the corresponding vertex.

Definition 2 (Weighted Laplacian)**.** *For an undirected weighted graph $G = (V, E)$ and a weight function $\boldsymbol{w} : E \to \mathbb{R}_{\geq 0}$ the Laplacian L_G is a $|V| \times |V|$ matrix defined as*

$$L_G(i,j) = \begin{cases} -w(i,j) & \text{if } (i,j) \in E \\ \sum_{(i,v)\in E} w(i,v) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

2.2.1 Properties of Laplacian

Fact 6. *The Laplacian of a graph is PSD*

Fact 7. *ker(L) = span(**1**)*

Fact 8. *Let L be the Laplacian of a graph $G = (V, E)$, then*

$$\lambda_2(L) > 0 \iff G \text{ is connected}$$

2.2.2 Kirchoff Matrix Tree Theorem

Fact 9. *The number of spanning trees in a graph G is **det**$(L_G[i])$ (for any i) where L_G denotes the Laplacian of G and $L_G[i]$ denotes the matrix with i^{th} row and column removed.*

2.3 Electric Networks

2.3.1 Unweighted Graph

Incidence Matrix

Given an undirected unweighted graph $G = (V, E)$ with arbitrary orientation of edges. Let $B \in \mathbb{M}_{n\times m}$ [1] called the edge-vertex incidence matrix defined as

$$B(i,e) = \begin{cases} 1 & \text{if } i \text{ is tail of } e \\ -1 & \text{if } i \text{ is head of } e \\ 0 & \text{otherwise} \end{cases}$$

Fact 10. *For a graph G with arbitrarily chosen incidence matrix B and Laplacian L,*

$$B \cdot B^T = L$$

Given an unweighted graph $G = (V, E)$ we associate a electrical network by replacing each edge with a resistor with resistance 1 Ω. A current source is introduced in each vertex, denoted as $\mathbf{c}_{\text{ext}} \in \mathbb{R}^n$. This induces a voltage at each vertex and current at each edge. Let's denote it as $\mathbf{v} \in \mathbb{R}^n, \mathbf{i} \in \mathbb{R}^m$

Let i_{xy} denote the current from vertex x to y for edge $(x, y) \in E$. And v_x denote the potential at a vertex $x \in V$.

[1] I have used a transposed version compared to Vishnoi [29] so that it's consistent with the notation used later

Kirchoff's current law

Kirchoff's current law states that the algebraic sum of current into any vertex equals zero.

$$B \cdot \mathbf{i} = \mathbf{c}_{\text{ext}}$$

Ohm's law

Ohm's law states that electric current through an edge is directly proportional to the potential different across an edge and inversely proportional to the resistance of the edge.

$$i_{xy} = \frac{v_x - v_y}{r_{xy}}$$

Since the resistance in the unweighted graph is 1 Ω we have

$$\mathbf{i} = B^T \cdot \mathbf{v}$$

Combining Ohm's law and Kirchoff's law we get

$$L \cdot \mathbf{v} = \mathbf{c}_{\text{ext}}$$

Laplacian pseudoinverse

Effective Resistance

Effective resistance across 2 vertices x, y is the resistance between x, y if we treat the graph as a single resistor connected between x and y.

In our case we are mainly interested in the effective resistance across an edge $e = (x, y) \in E$.

Definition 3. *Effective Resistance is the potential difference across an edge $e = (x, y)$ when $1A$ is inducted at x and taken out at y. It is denoted as R_e^{eff}*

So we have $\mathbf{c}_{\text{ext}} = e_x - e_y$ where e_i denotes a vector with 1 in the i^{th} index and 0 elsewhere. By definition of effective resistance we have

$$\begin{aligned} R_e^{\text{eff}} &= (e_x - e_y)^T \cdot \mathbf{v} \\ R_e^{\text{eff}} &= (e_x - e_y)^T \cdot L^+ \cdot \mathbf{c}_{\text{ext}} \\ R_e^{\text{eff}} &= (e_x - e_y)^T \cdot L^+ \cdot (e_x - e_y) \end{aligned}$$

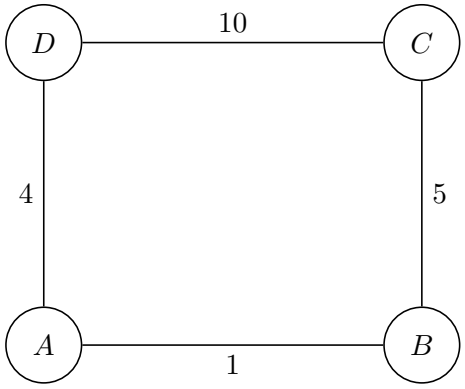
2.3.2 Weighted Graph

There are a few subtle changes which needs to be incorporated for the weighted graph setting. Suppose $G = (V, E)$ be a undirected weighted graph with weight function $\mathbf{w} : E \to \mathbb{R}_{\geq 0}$. Now the electric network of G coressponds has edges replaced by a resistor with resistance $r_e = 1/\mathbf{w}(e), \forall e \in E$. The intuition is that lower weight for an edge in graph G means it's barely there hence it coressponds to higher resistance. And having no edge coressponds to infinite resistance.

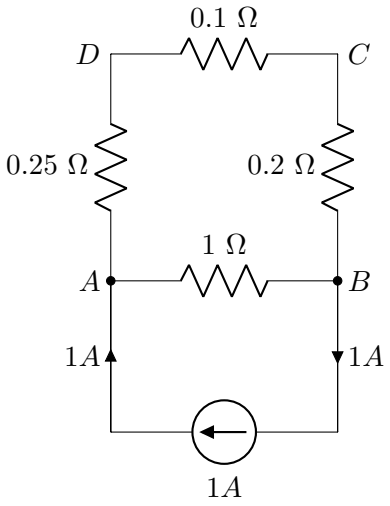
For deriving the other relations let $W \in \mathbb{M}_{m \times m}$ be a diagonal matrix such that $W(e, e) = \mathbf{w}(e)$. Now $L = B\ W\ B^T$ and Ohm's law becomes $\mathbf{i} = W\ B^T\ \mathbf{v}$. As it can be seen the formula for effective resistance remains the same

2.3.3 An Example

Consider the following weighted graph G in **Figure 2.1**



(A) The original graph G



(B) The electric network version of G

FIGURE 2.1: An example of a graph and it's corresponding electric network

Following the convention such that current going inside a vertex as negative and out as positive.

By Ohm's law we have

$$\begin{aligned} i_{AD} &= 4\,(v_A - v_D) \\ i_{DC} &= 10\,(v_D - v_C) \\ i_{CB} &= 5\,(v_C - v_B) \\ i_{AB} &= 1\,(v_A - v_B) \end{aligned}$$

By Kirchoff's current law we have

$$\begin{aligned} i_{AD} + i_{AB} &= 1 \\ -i_{AD} + i_{DC} &= 0 \\ -i_{DC} + i_{CB} &= 0 \\ -i_{AB} - i_{CB} &= -1 \end{aligned}$$

Now combining these two we get

$$\begin{aligned} 5v_A - 1v_B - 0v_C - 1v_D &= 1 \\ -1v_A + 6v_B - 5v_C - 0v_D &= -1 \\ 0v_A - 5v_B + 15v_C - 10v_D &= 0 \\ -4v_A - 0v_B - 10v_C + 14v_D &= 0 \end{aligned}$$

And this is exactly what we would have gotten using $L \cdot \mathbf{v} = \mathbf{c}_{\text{ext}}$

3 Random Walk Approach

3.1 Aldous, Broder

Aldous [1] and Broder [4] independently invented the following simple random walk based algortihm.

Input: $G = (V, E)$
Output: A random spanning tree
Choose a starting vertex s arbitrarily
$T_V \leftarrow \{s\}, T_E \leftarrow \emptyset$
while $|T_V| < |V$ **do**
  $next =_{u.a.r} N(s)$
  if $next \notin T_V$ **then**
    $T_V = T_V \cup \{next\}$
    $T_E = T_E \cup \{(s, next)\}$
  end
  $s = next$
end
return $T = (T_V, T_E)$

Algorithm 1: Aldous-Broder Algorithm

3.1.1 Running Time

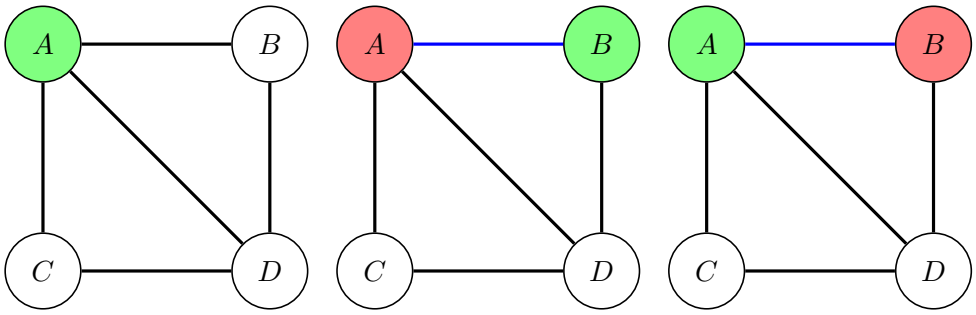
Cover Time

$cov_G(u)$:= The expected number of steps for a random walk starting at u to visit all the vertices in G

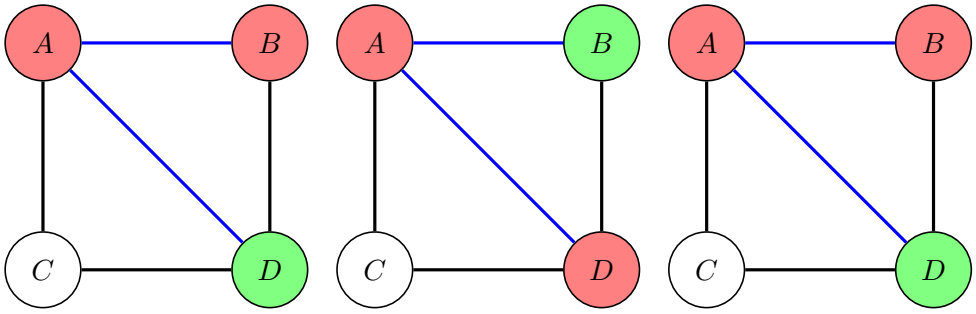
Cover Time of G

$cov_G := \max_{u \in V_G} cov_G(u)$
It is known that $cov_G = \mathcal{O}(|V|\ |E|) = \mathcal{O}(|V|^3)$

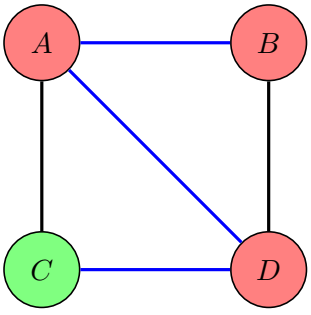
3.1.2 An Example



(A) Iteration xxx (B) Iteration xxy (C) Iteration xxz



(D) Iteration xyx (E) Iteration xyy (F) Iteration xyz



(G) Iteration xzx

4 Matrix Approach

4.1 Harvey, Xu

Harvey and Xu [18] proposed a $\mathcal{O}(N^\omega)$ (where ω is the exponent of fast matrix multiplication) algorithm for sampling a uniform spanning tree which is much simpler compared to the one proposed earlier by Colbourn, Myrvold, and Neufeld [8] which also has the same running time. The initial starting point for the algorithm is using the relationship between effective resistance and the probability that an edge belongs to a spanning tree. And also the fact that sampling an edge corresponds to contracting it and discarding the edge corresponds to deleting it. The following naive chain rule algorithm works on the same principle.

Naive chain rule algorithm

Input: $G = (V, E)$ and L_G^+
Output: Set of edges corresponding to a random spanning tree
for $e = (u, v) \in E$ **do**
  $R_e^{\text{eff}} = (\chi_u - \chi_v)^T \; L_G^+ \; (\chi_u - \chi_v)$;
  if $(X \sim Bernoulli(R_e^{eff})) = 1$ **then**
    Add edge e to the spanning tree;
    $G = G/e$;
  else
    $G = G \setminus e$;
  end
  Update L_G^+ ;
end

Algorithm 1: Sampling uniform spanning tree using chain rule

Computing L_G^+ takes $\mathcal{O}(N^3)$ hence the overall running time is $\mathcal{O}(MN^3)$. The main bottleneck of this algorithm is to modify the graph each time after making a decision to sample an edge.

4.1.1 Summary of the paper

On a high level the main ideas used in the paper are

1. Use a divide and conquer algorithm to break the graph into smaller parts and sample on each parts separately and update the pseudoinverse of the laplacian lazily only on the subgraph when needed

2. The important insight here is that the sampling probability of an edge depends only on 4 entries of the pseudoinverse of the laplacian. Hence we don't need to update all the entries of the matrix when the graph is modified

3. A well known method to compute inverse of a matrix with updates is to use the Sherman-Morrison-Woodbury formula. But in this case the formula has to be modified to work for the case where only a submatrix is modified.

4. Since while contracting an edge the number of vertices decreases it would get cumbersome to modify the dimension of the matrix evertime. So they overcome this issue by considering the formula on the limit case. When the graph is considered as a electric network then increasing the weight of an edge corresponds to shorting that link, hence in the limit case we get the same result as contracting the edge

5. One of the main improvements over the previous algorithms of the same complexity (Colbourn, Myrvold, and Neufeld [8]) is that the intricacies of LU decomposition is avoided since the current algorithm uses only matrix inversion.

6. All the formulas are derived for the submatrix case hence the complexity of a sub problem depends only on the size of the subproblem.

4.1.2 Harvey, Xu Algorithm

Given a graph $G = (V_G, E_G)$. The algortihm proceeds by splitting the vertex set into 2 equal halves $V_G = S \uplus R$. And now they define subsets of edges $E[S] = (S \times S) \cap E_G$ and $E[R, S] = (S \times R) \cap E_G$.

Suppose $S = S_1 \uplus S_2$ and $R = R_1 \uplus R_2$. Then we can see that

$$E[S] = E[S_1] \cup E[S_2] \cup E[S_1, S_2]$$

$$E[R, S] = \bigcup_{i,j \in \{1,2\}} E[R_i, S_j]$$

This formula is used to recurse on the subproblems for each subset. In this manner each edge will be visited exactly once by the algorithm

Notation

- N - This is an auxiliary matrix which starts with L_G^+ and gets updated lazily as the algorithm progresses.

- **function** $SampleSpanningTree(G = (V, E))$ - This is the main function from which the execution of the algorithm starts. It takes input a graph and outputs a set of edges corresponding to a random spanning tree.

- **function** $SampleEdgesWithin(S)$ - This function takes input a set of vertices S and returns sets F and D which are set of edges contracted and deleted respectively in the subgraph induced by S on G.

- **function** $SampleEdgesCrossing(R, S)$ - This function works similar to the above one but samples edges crossing the sets R and S. Also the base case of the entire algorithm (case where $|R| = |S| = 1$) of making the decision to sample an edge is handled here.

- **procedure** $Update(S, F, D)$ - This function updates the sub matrix $N_{S,S}$ based on the formulas derived in **Corollary 1** and **Theorem 3**

Algorithm 2. A Recursive Algorithm

function SAMPLESPANNINGTREE($G = (V_G, E_G)$)
  $N \leftarrow L_G^+$
  SAMPLEEDGESWITHIN(V_G)
  return the uniform spanning tree T
function SAMPLEEDGESWITHIN(S)
  if $|S| = 1$ **then return**
  Divide S in half: $S = S_1 \cup S_2$
  for $i \in \{1, 2\}$ **do**
    SAMPLEEDGESWITHIN(S_i)
    Restore N_{S_i,S_i} to its value before entering the recursion
    $F \leftarrow$ the set of edges contracted in SAMPLEEDGESWITHIN(S_i)
    $D \leftarrow$ the set of edges deleted in SAMPLEEDGESWITHIN(S_i)
    UPDATE(S, F, D)
  SAMPLEEDGESCROSSING(S_1, S_2)
function SAMPLEEDGESCROSSING(R, S)
  if $|R| = 1$ **then**
    Let $r \in R$ and $s \in S$, $R^{\text{eff}} \leftarrow (\mathcal{X}_r - \mathcal{X}_s)^T N (\mathcal{X}_r - \mathcal{X}_s)$
    Flip a biased coin that turns head with probability R^{eff}
    if head **then**
      Add $e_{r,s}$ to the uniform spanning tree T and the set of contracted edges
    else
      Add $e_{r,s}$ to the set of deleted edges
  else
    Divide R and S each in half: $R = R_1 \cup R_2$ and $S = S_1 \cup S_2$
    for $i \in \{1, 2\}$ and $j \in \{1, 2\}$ **do**
      SAMPLEEDGESCROSSING(R_i, S_j)
      Restore $N_{R_i \cup S_j, R_i \cup S_j}$ to its value before entering the recursion
      $F \leftarrow$ the set of edges contracted in SAMPLEEDGESCROSSING(R_i, S_j)
      $D \leftarrow$ the set of edges deleted in SAMPLEEDGESCROSSING(R_i, S_j)
      UPDATE($R \cup S, F, D$)
procedure UPDATE(S, F, D)
  Let V be the incidence matrix for F
  Let L_D be the Laplacian matrix for D
  $N_{S,S} \leftarrow N_{S,S} - N_{S,S} V_{S,*} (V_{S,*}^T N_{S,S} V_{S,*})^{-1} V_{S,*}^T N_{S,S}$
  $N_{S,S} \leftarrow N_{S,S} - N_{S,S} (L_{D_{S,S}} N_{S,S} - I)^{-1} L_{D_{S,S}} N_{S,S}$

FIGURE 4.1: Harvey, Xu algorithm [a]

[a] This image is exactly reproduced from the paper.

An Example

The below diagram shows how the algorithm proceeds by recursively splitting the graph.

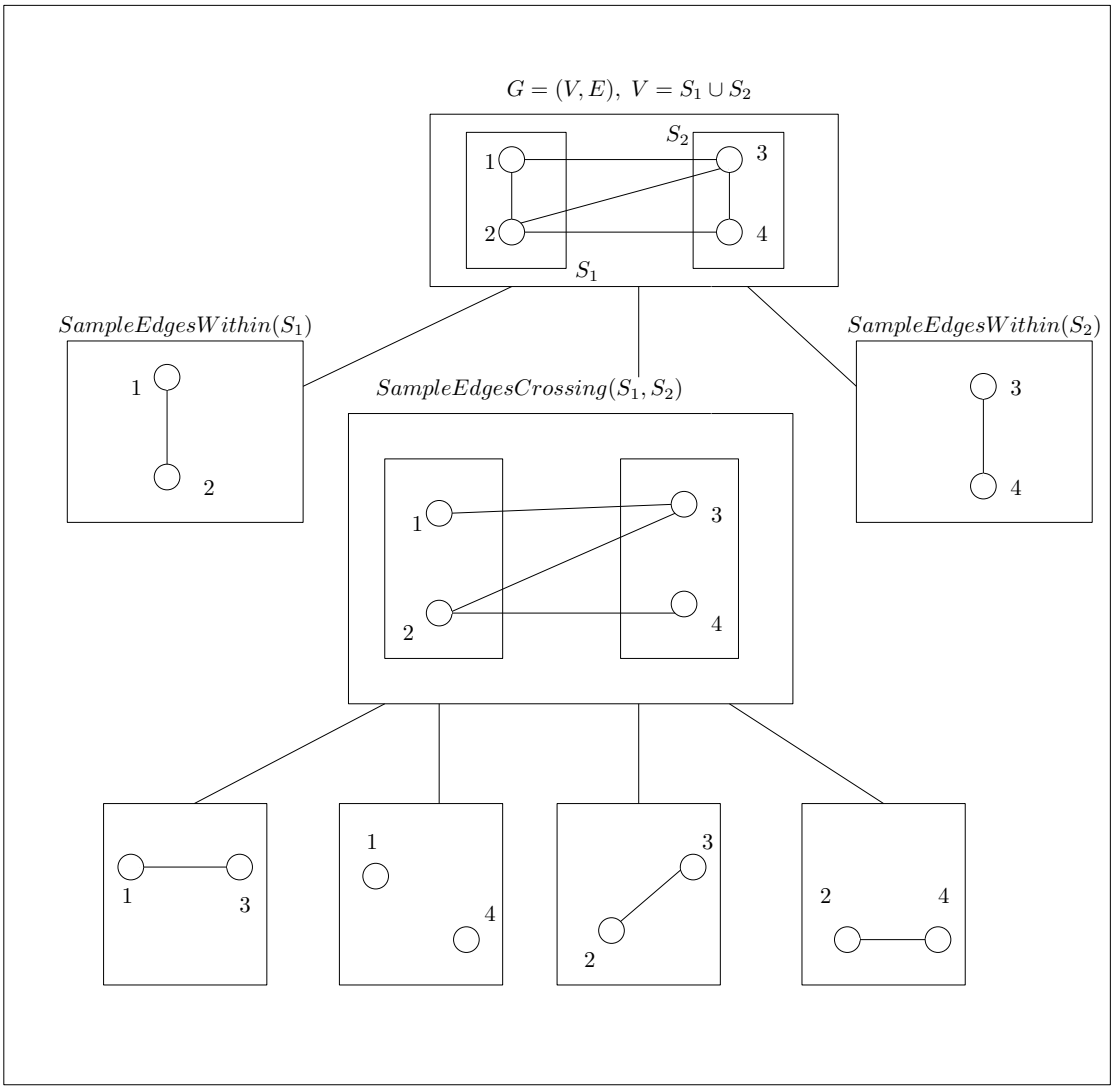


FIGURE 4.2: An example of Harvey, Xu algorithm [a]

[a] The function calls to $SampleEdgesWithin(S_1)$ and $SampleEdgesWithin(S_2)$ in turn would call their respective $SampleEdgesCrossing$ functions. But it would be the base case and would look similar to the one portrayed here and hence omitted for the sake of brevity.

4.1.3 Structure of the paper

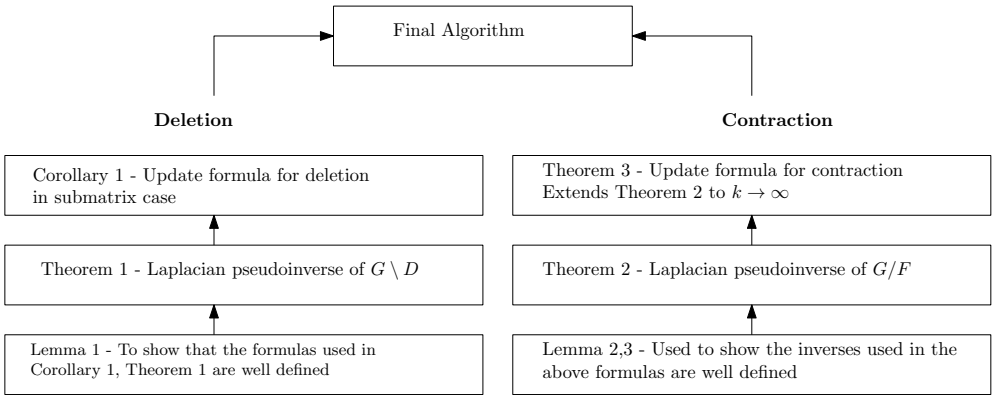


FIGURE 4.3: Dependencies between the results

4.1.4 Facts used

Tao [28] has a nice derivation of the Sherman-Morrison formula.

Fact 1 (Woodbury matrix identity)**.** *Let $M \in \mathbb{M}_{n\times n}, U \in \mathbb{M}_{n\times k}, V \in \mathbb{M}_{n\times k}$. Suppose M is non-singular then $M+UV^T$ is non-singular $\iff$ $I+V^TM^{-1}U$ is non-singular. If $M+UV^T$ is non-singular, then*

$$(M+UV^T)^{-1} = M^{-1} - \left(M^{-1}\cdot U\cdot (I+V^TM^{-1}U)^{-1}\cdot V^T\cdot M^{-1}\right)$$

Fact 2. *For any $L \in \mathbb{M}_{n\times n}$ with $ker(L) = span(\mathbf{1})$, we have $LL^+ = I - \frac{\mathbf{1}\cdot\mathbf{1}^T}{n}$ and $P := I - \frac{\mathbf{1}\cdot\mathbf{1}^T}{n}$ is called the **projection matrix**.*

The following set of facts are about the properties of matrix operations (addition, multiplication, etc.) on sub-matrices. The first 4 are easy to see, so I haven't derived them. For the last one I have written a derivation using Shur's complement from Wikipedia

Fact 3 (Sub-matrices)**.** *For all the results below, S denotes a index set and S^c denotes it's complement.*

1. *For any $A, B \in \mathbb{M}_{n\times n}, (A+B)_{S,S} = A_{S,S} + B_{S,S}$*
2. *If $C = D\cdot E\cdot F$ then $C_{S,S} = D_{S,*}\cdot E\cdot F_{*,S}$*
3. *For $A \in \mathbb{M}_{m\times n}, B \in \mathbb{M}_{n\times l}$, If $A_{S^c,S^c} = 0$ or $B_{S^c,S^c} = 0$ then $(AB)_{S,S} = A_{S,S}\cdot B_{S,S}$*
4. *For any matrix C where $C = D\cdot E\cdot F$. If $D_{*,S^c} = 0$ and $F_{S^c,*} = 0$, then $C = D_{*,S}\cdot E_{S,S}\cdot F_{S,*}$*
5. *$D = \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}$, and $E = \begin{bmatrix} A & B \\ X & Y \end{bmatrix}$ where $M, A \in \mathbb{M}_{n\times n}$ and If $(MA-I)$ is invertible Then,*

$$(DE-I)^{-1} = \begin{bmatrix} (MA-I)^{-1} & (MA-I)^{-1}\cdot M\cdot B \\ 0 & -I \end{bmatrix}$$

Proof. $(DE - I)^{-1}$ can be computed using Shur Complement([30]) .

Suppse $N = \begin{bmatrix} P & Q \\ R & S \end{bmatrix}$ and Shur's complement of block S and P is

$$N/S := P - QS^{-1}R \qquad N/P := S - RP^{-1}Q$$

Then

$$N^{-1} = \begin{bmatrix} P^{-1} + (P^{-1}Q(N/P)^{-1}RP^{-1}) & -(P^{-1}Q(N/P)^{-1}) \\ -((N/P)^{-1}RP^{-1}) & (N/P)^{-1} \end{bmatrix}$$

In our case $N = \begin{bmatrix} MA - I & MB \\ 0 & -I \end{bmatrix}$ and $(N/P) = -I$. From this it follows that

$$N^{-1} = (DE - I)^{-1} = \begin{bmatrix} (MA - I)^{-1} & (MA - I)^{-1} \cdot M \cdot B \\ 0 & -I \end{bmatrix}$$

■

Fact 4. *Let $A, B \in \mathbb{M}_{n\times n}$ with B being symmetric PSD. Suppose x is an eigenvector of AB corresponding to eigenvalue λ. Then $\sqrt{B}x$ is an eigenvector of $\sqrt{B}A\sqrt{B}$ corresponding to eigenvalue λ*

Fact 5 (Laplacian and graph connectivity (Fiedler value))**.** *Let G be a graph with n vertices. Suppose $(\lambda_1, \lambda_2 \cdots \lambda_n)$ be the eigenvalues corresponding to the eigenvectors $(v_1, v_2 \cdots v_n)$ of the Laplacian of G denoted as L_G. L_G is symmetric PSD with $\lambda_1 = 0$ and $v_1 = \mathbf{1}$. The following properties relate the eigenvalues of L_G with the connectivity of G :*

1. *$\lambda_2 > 0 \iff G$ is connected*
2. *G is disconnected $\iff \exists z$ with $z^T \mathbf{1} = 0$ and $z^T L_G z = 0$*

The above is true for L_G^+ also

Definition 1 (χ_u)**.** *χ_u is a vector of size $|V|$*

$$\chi_u(i) = \begin{cases} 1, & \text{if } i = u \\ 0, & \text{otherwise} \end{cases}$$

Definition 2 (Uniform random spanning tree)**.** *Let $\hat{T}$ be the random variable denoting a uniformly random spanning tree, then $\mathbb{P}(\hat{T} = T) = \frac{1}{|\mathcal{T}|}$, where $\mathcal{T}$ is the set of all spanning trees of G.*

Fact 6. *Given a graph $G = (V, E)$ with laplacian L_G, the effective resistance of an edge $e = \{u, v\} \in E$ is*

$$R_e^{\mathit{eff}} = (\chi_u - \chi_v)^T \; L_G^+ \; (\chi_u - \chi_v)$$

Then for any $e \in E$ we have

$$\mathbb{P}(e \in \hat{T}) = R_e^{\mathit{eff}}$$

4.1.5 Technical details of the results

Deletion

The first step in obtaining a updation formula for deletion is to make sure $(I - L_D L_G^+)$ is invertible. As the inverse of this term would be used in the expansion of $(L_G - L_D)^+$

For the first direction of Lemma 1, the main result used is **Fact 5** (G is disconnected $\iff \exists z$ with $z^T\mathbf{1} = 0$ and $z^T L_G^+ z = 0$). So if we can show this for a suitable z we are done. Now using the hypothesis that $(I - L_D L_G^+)$ is singular and **Fact 4** they derive the following $y^T \cdot \sqrt{L_G^+} \cdot L_{G\setminus D} \cdot \sqrt{L_G^+} \cdot y = 0$. As we an see the remaining part is to show $(z = \sqrt{L_G^+} y) \perp \mathbf{1}$

Lemma 1 (Formulas in **Theorem 1** are well defined)**.** *Let $G = (V, E)$ be a connected graph and $D \subseteq E$ then*
$\left(I - L_D L_G^+\right)$ *is invertible* $\iff$ *$G \setminus D$ is connected*

Proof. First let's show that If $(I - L_D L_G^+)$ is singular then $G \setminus D$ is disconnected

- Since $(I - L_D L_G^+)$ is singular $\exists x \neq 0$ s.t. $(I - L_D L_G^+)x = 0$

$$\implies L_D L_G^+ \, x = x \tag{4.1}$$
$$\implies 1 \in eigenvalues(L_D L_G^+) \tag{4.2}$$
$$\implies 1 \in eigenvalues((L_G - L_{G\setminus D}) L_G^+) \tag{4.3}$$

- Let $x \perp \mathbf{1}$ be an eigenvector of $(L_G - L_{G\setminus D})L_G^+$ with eigenvalue 1.

- By **Fact 4**, $y = \frac{\sqrt{L_G^+}x}{\left\|\sqrt{L_G^+}x\right\|}$ is an eigenvector of $\sqrt{L_G^+}(L_G - L_{G\setminus D})\sqrt{L_G^+}$

$$= y^T \cdot \sqrt{L_G^+}(L_G - L_{G\setminus D})\sqrt{L_G^+} \cdot y = 1 \tag{4.4}$$
$$= y^T \sqrt{L_G^+} L_G \sqrt{L_G^+} y = y^T L_G^+ L_G y = y^T P y \tag{4.5}$$
$$= y^T \left(I - \frac{\mathbf{1}^T\mathbf{1}}{n}\right) y = y^T y - \left(\frac{y^T\mathbf{1}^T\mathbf{1}y}{n}\right) = y^T y = 1 \tag{4.6}$$

- $\therefore y^T \sqrt{L_G^+} L_{G\setminus D} \sqrt{L_G^+} y = 0$ now if we consider $z = \sqrt{L_G^+} y$ and show that $z^T\mathbf{1} = 0$ then we can use **Fact 5** to complete the proof

$$y^T \sqrt{L_G^+}\mathbf{1} = x^T \sqrt{L_G^+}\sqrt{L_G^+}\mathbf{1} = 0 L_G^+ \tag{4.7}$$
$$G \setminus D \text{ is disconnected} \tag{4.8}$$

Now to prove the converse, If $G \setminus D$ is disconnected then $I - L_D L_G^+$ is singular

- If $G \setminus D$ is disconnected then $\exists y \perp \mathbf{1}, ||y|| = 1$ we have

  1. $y^T \cdot \sqrt{L_G^+} \cdot L_{G\setminus D} \cdot \sqrt{L_G^+} \cdot y = 0$
  2. $y^T \cdot \sqrt{L_G^+} \cdot L_G \cdot \sqrt{L_G^+} \cdot y = y^T y = 1$

- From (1) and (2) we get $y^T\sqrt{L_G^+}(L_G - L_{G\setminus D})\sqrt{L_G^+}y = 1$

$$\implies y^T \cdot \sqrt{L_G^+} \cdot L_D \cdot \sqrt{L_G^+} \cdot y = 1 \tag{4.9}$$
$$\implies 1 \in \text{eigenvalues}(L_D L_G^+) \tag{4.10}$$
$$\implies (I - L_D L_G^+) \text{ is singular} \tag{4.11}$$

■

In **Theorem 1** they just show that the formula for the updated pseudoinverse is indeed true. This is shown using the following identity $LL^+ = P$

Theorem 1 (Update identity for Deletion)**.** *Let $G = (V, E)$ be a connected graph and $D \subseteq E$. If $G \setminus D$ is connected then*

$$(L_G - L_D)^+ = L_G^+ - \left(L_G^+ \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)$$

Proof. If R.H.S is indeed true then it should satisfy the property of $LL^+ = P$

$$(L_G - L_D) \cdot (L_G - L_D)^+$$
$$(L_G - L_D) \cdot \left(L_G^+ - \left(L_G^+ \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)\right)$$
$$\left[P - L_D L_G^+\right] - \left[(L_G L_G^+ - L_D L_G^+) \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right]$$
$$\left[P - L_D L_G^+\right] + \left[\left((L_D L_G^+ - I) + \frac{\mathbf{1}\mathbf{1}^T}{n}\right) \cdot \left((L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)\right]$$
$$\left[P - L_D L_G^+\right] + \left[(L_D L_G^+) + \left(\frac{\mathbf{1}\mathbf{1}^T}{n} \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)\right]$$

We can see that $-\mathbf{1}^T(L_D L_G^+ - I) = -\mathbf{1}^T L_D L_G^+ + (I\mathbf{1})^T = 0 + \mathbf{1}^T = \mathbf{1}^T$. Hence $\mathbf{1}^T(L_D L_G^+ - I)^{-1} = -\mathbf{1}^T$. And also $\mathbf{1}^T L_D = 0$. Hence,

$$P - L_D L_G^+ + L_D L_G^+ + \mathbf{1}^T L_D L_G^+ = P$$

■

Definition 3 (Submatrix)**.** *A submatrix of a martix A containing rows S and columns T is denoted as $A_{S,T}$*

Corollary 1 modifies the update formula in **Theorem 1** to work for submatrices and hence reduce the compliexity to $\mathcal{O}(|S|^\omega)$. They do this by first applying the facts related to submatrices **Fact 3.3, 3.5**.

Corollary 1 (Improved **Theorem 1** for submatrix)**.** *Let $G = (V, E)$ be a connected graph and $D \subseteq G$. For $S \subseteq V$ define $E[S]$ as $(S \times S) \cap E$. Suppose $E_D \subseteq E[S]$ and $G \setminus D$ is connected then*

$$(L_G - L_D)^+_{S,S} = (L_G^+)_{S,S} - \left((L_G^+)_{S,S} \cdot ((L_D)_{S,S}\ (L_G^+)_{S,S}\ -\ I)^{-1} \cdot (L_D)_{S,S} \cdot (L_G^+)_{S,S}\right)$$

Proof. From **Theorem 1** we know that $(L_G - L_D)^+ = L_G^+ - \left(L_G^+ \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)$. If we apply **Fact 3.1** to $(L_G - L_D)^+$ we get

$$(L_G^+)_{S,S} - \left(L_G^+ \cdot (L_D L_G^+ - I)^{-1} \cdot L_D \cdot L_G^+\right)_{S,S}$$

Applying **Fact 3.3** we get

$$(L_G^+)_{S,S} - \left((L_G^+)_{S,S} \cdot (L_D L_G^+ - I)^{-1}_{S,S} \cdot (L_D)_{S,S} \cdot (L_G^+)_{S,S}\right)$$

Now applying **Fact 3.5** to $(L_D L_G^+ - I)^{-1}_{S,S}$

Fact 3.5 states that If $D = \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}$, and $E = \begin{bmatrix} A & B \\ X & Y \end{bmatrix}$ where $M, A \in \mathbb{M}_{n\times n}$ and If $(MA - I)$ is invertible Then,

$$(DE - I)^{-1} = \begin{bmatrix} (MA-I)^{-1} & (MA-I)^{-1} \cdot M \cdot B \\ 0 & -I \end{bmatrix}$$

Here we have $L_D = \begin{bmatrix} (L_D)_{S,S} & 0 \\ 0 & 0 \end{bmatrix}$, and $L_G = \begin{bmatrix} (L_G)_{S,S} & (L_G)_{S,S^c} \\ (L_G)_{S^c,S} & (L_G)_{S^c,S^c} \end{bmatrix}$

$$\therefore\ (L_D L_G - I)^{-1} = \begin{bmatrix} ((L_D)_{S,S}(L_G)_{S,S} - I)^{-1} & (L_D)_{S,S}(L_G)_{S,S^c} \\ 0 & -I \end{bmatrix}$$

Hence we get the required result

$$(L_G - L_D)^+_{S,S} = (L_G^+)_{S,S} - \left((L_G^+)_{S,S} \cdot ((L_D)_{S,S}\ (L_G^+)_{S,S}\ -\ I)^{-1} \cdot (L_D)_{S,S} \cdot (L_G^+)_{S,S}\right)$$

■

Contraction

The main approach proposed to tackle contraction updates is to increase the weight of the edges that are to be contracted to a large value k.

Definition 4 (Incidence Matrix)**.** *Let $G = (V, E)$, given an edge $e = u, v \in E$ the incidence vector of e is defined as $v_e = (\chi_u - \chi_v)$. Given a set of edges $D = \{e_1, e_2 \cdots e_m\} \subseteq E$, the incidence matrix of D is defined as $B_D = [v_{e_1} | v_{e_2} \cdots | v_{e_m}]$*

Note - *I have used a different notation for the incidence matrix compared to the original paper as I found it to be a bit confusing. And B is the common notation for incidence matrices in other resources such as Vishnoi [29].*

Definition 5 ($G+ke$)**.** *$G+ke$ is the weighted graph obtained by increasing e's weight by k*

Lemma 2 (Formulas in **Theorem 2** are well defined)**.** *Let $G = (V, E)$ be a connected graph. Given $F \subseteq E$ with $|F| = r$ and let B_F be the incidence matrix of F.*

$$B_F^T\ L_G^+\ B_F \text{ is invertible} \iff F \text{ is a forest}$$

Proof. First they show
F is a forest $\implies$ $B_F^T\ L_G^+\ B_F$ is invertible

So the main idea of this proof is to show that $B_F^T\ L_G^+\ B_F$ is positive definite. This is enough because positive definite matrices are non singular (if not then they will have 0 as an eigenvalue). Now using the following claim

Claim 1. *The incidence matrix of a acyclic graph has full column rank*

Hence for any $x \in \mathbb{R}^r, x \neq 0$, let $y = B_F x$ and $y \neq 0$. Also $y^T \mathbf{1} = x^T B_F^T \mathbf{1} = 0$. Hence $y \perp ker(L_G^+)$. Now since G is connected we have $\lambda_2(L_G) > 0$. Now since y corresponds to all vectors perpendicular to $ker(L_G^+)$ we can say that $y^T L_G^+ y > 0$ now expanding $y = B_F x$ we get $x^T(B_F^T\ L_G^+\ B_F)x > 0$. Hence $B_F^T\ L_G^+\ B_F$ is positive definite and hence invertible.

■

Lemma 3 (Formulas in **Theorem 2** are well defined)**.** *Let $G = (V, E)$ be a connected graph. Given $F \subseteq E$ and let B_F be the incidence matrix of F. For any $k > 0$,*

$$\textit{If } F \textit{ is a forest then } \left(\frac{I}{k} + B_F^T\ L_G^+\ B_F\right) \textit{ is invertible for any } k > 0$$

Proof. Suppose A, B are positive definite matrices then $A+B$ is also positive definite. Since A, B are positive definite we have $x^T Ax > 0, x^T Bx > 0$ for any x. Combining these two identity we get $x^T(A + B)x > 0$. Hence $A + B$ is also positive definite.

By **Lemma 2** $B_F^T\ L_G^+\ B_F$ is positive definite. And I/k is also positive definite because all the eigenvalues are $1/k$ and we have $k > 0$. Since positive definite matrices are non-singular, $\left(\frac{I}{k} + B_F^T\ L_G^+\ B_F\right)$ is invertible ■

Theorem 2 uses **Lemma 2** to show that the contraction update formula for finite k is well defined.

Theorem 2 (Contraction update formula for finite k)**.** *Let $G = (V, E)$ be a connected graph. Given $F \subseteq E$ and let B_F be the incidence matrix of F. For any $k > 0$,*

$$(L_G + k\ L_F)^+ = L_G^+ - \left(L_G^+ \cdot B_F \cdot (\frac{I}{k} + B_F^T\ L_G^+\ B_F)^{-1} \cdot B_F^T \cdot L_G^+\right)$$

Proof. They use the same strategy used in **Theorem 1**. Also note that $B_F B_F^T = L_F$

$$\left[L_G + kB_F B_F^T\right] \cdot \left[L_G^+ - \left(L_G^+\ B_F\ (\frac{I}{k} + B_F^T\ L_G^+\ B_F)^{-1}\ B_F^T\ L_G^+\right)\right]$$

$$= P + kB_F B_F^T L_G^+ - \left((L_G L_G^+ B_F + kB_F B_F^T L_G^+ B_F)\ (\frac{I}{k} + B_F^T L_G^+ B_F)^{-1}\ B_F^T\ L_G^+\right)$$

Here $L_G L_G^+ B_F = (I - \frac{\mathbf{1}\mathbf{1}^T}{n})B_F = B_F$. Because each column sum of B_F is 0.

$$= P + kB_F B_F^T L_G^+ - \left(kB_F\ (\frac{I}{k} + B_F^T L_G^+ B_F)\ (\frac{I}{k} + B_F^T L_G^+ B_F)^{-1}\ B_F^T\ L_G^+\right)$$

$$= P + kB_F B_F^T L_G^+ - kB_F B_F^T L_G^+ = P$$

■

This coroallry is a direct consequence of the facts pertaining to submatrices (**Fact 3**)

Corollary 2 (Improves **Theorem 2** for sub-matrices)**.** *Let $G = (V, E)$ be a connected graph. Given $F \subseteq E$ and let B_F be the incidence matrix of F. Suppose $F \subseteq E[S]$, where $S \subseteq V$. For any $k > 0$,*

$$(L_G + k\ L_F)_{S,S}^+ = (L_G^+)_{S,S} - \left((L_G^+)_{S,S}\ (B_F)_{S,*}\ (\frac{I}{k} + (B_F^T)_{S,*}\ (L_G^+)_{S,S}\ (B_F)_{S,*})^{-1}\ (B_F^T)_{S,*}\ (L_G^+)_{S,S}\right)$$

In **Theorem 3** they extend the contraction formula from **Theorem 2** to the case $k \to \infty$. The main idea of the proof is to define $M_k = L^+_{G+kF}$ [1]
$M = L^+_G - \left(L^+_G\ B_F\ (B^T_F L^+_G B_F)^{-1}\ B^T_F\ L^+_G\right)$ and to show that $\lim_{k\to\infty} \|M_k - M\| = 0$

Theorem 3 (Extends **Theorem 2** to $k \to \infty$ case)**.** *For a forest $F_1 \subseteq E$, let $G(k) = G + k\ F_1$ as defined in **Definition 3** . Let $F_2 \subseteq E$ be disjoint from F_1 such that $F_1 \cup F_2$ is a forest. Let B_{F_2} be the incidence matrix of F_2. For $k > 0$ define $N = \lim_{k\to\infty} L^+_{G(k)}$*

$$\lim_{k\to\infty} L^+_{G(k)+kF_2} = N - \left(N \cdot B_{F_2} \cdot (B^T_{F_2}\ N\ B_{F_2}) \cdot B^T_{F_2} \cdot N\right)$$

Also $ker\left(\lim_{k\to\infty} L^+_{G(k)+kF_2}\right) = span\ (B_{F_1\cup F_2} \cup \mathbf{1})$

Proof. From **Theorem 2** we have $M_k = L^+_G - \left(L^+_G \cdot B_F \cdot (\frac{I}{k} + B^T_F\ L^+_G\ B_F)^{-1} \cdot B^T_F \cdot L^+_G\right)$ and they define $A = B^T_F L^+_G B_F$

$$\|M_k - M\| = \left\| L^+_G\ B_F \left((\frac{I}{k} + A)^{-1} - A^{-1} \right)\ B^T_F\ L^+_G \right\|$$

Using the property of matrix norm $\|X\ Y\| \le \|X\|\ \|Y\|$, they get

$$\|M_k - M\| \le \left\|L^+_G\right\|^2\ \|B_F\|\ \left\|B^T_F\right\|\ \left\| \left(\frac{I}{k} + A\right)^{-1} - A^{-1} \right\| \qquad (1)$$

Now they expand the last term in the above inequality using the Sherman-Morrison-Woodbury identity and apply the matrix norm inequality and with some straightforward manipulations they show that

$$\lim_{k\to\infty} \left\| \left(\frac{I}{k} + A\right)^{-1} - A^{-1} \right\| = 0$$

Now applying this to (1) they prove that $\lim_{k\to\infty} \|M_k - M\| = 0$ ∎

4.1.6 Summary

The ideas discussed here such as lazy updates were used in Harvey [17] to get a randomized algorithm for non-bipartite matching in $\mathcal{O}(N^\omega)$.

[1] I have changed the notation to M as the proof used in the paper used N for defining 2 different formulas

5 Conclusion

We have reviewed some of the algorithms used for sampling a uniform spanning tree. The main impetus for going into the details of Harvey and Xu [18] is due to the fact that it uses the Sherman-Morrison-Woodbury identity for updating the laplacian pseudoinverse. Recently Datta et al. [9] also used the same identity for showing that **REACHABILITY** problem is in $\texttt{DynFO} + \texttt{Mod}\ 2(\leq, +, \times)$. Hence we explored the possibility of using the same framework for sampling spanning trees in the dynamic setting. But it turns out that there are a lot of subtleties in coming up with a proper formulation.

Bibliography

[1] David J Aldous. "The random walk construction of uniform spanning trees and uniform labelled trees". In: *SIAM Journal on Discrete Mathematics* 3.4 (1990), pp. 450–465.

[2] Arash Asadpour et al. "An O(log n/log log n)-Approximation Algorithm for the Asymmetric Traveling Salesman Problem". In: *Operations Research* 65.4 (2017), pp. 1043–1061. DOI: 10.1287/opre.2017.1603. eprint: https://doi.org/10.1287/opre.2017.1603. URL: https://doi.org/10.1287/opre.2017.1603.

[3] Judit Bar-Ilan and Dror Zernik. "Random leaders and random spanning trees". In: *Distributed Algorithms*. Ed. by Jean-Claude Bermond and Michel Raynal. Berlin, Heidelberg: Springer Berlin Heidelberg, 1989, pp. 1–12. ISBN: 978-3-540-46750-2.

[4] A. Broder. "Generating random spanning trees". In: *30th Annual Symposium on Foundations of Computer Science*. 1989, pp. 442–447.

[5] Charles J. Colbourn. *The Combinatorics of Network Reliability*. USA: Oxford University Press, Inc., 1987. ISBN: 0195049209.

[6] Charles J Colbourn, Robert P.J Day, and Louis D Nel. "Unranking and ranking spanning trees of a graph". In: *Journal of Algorithms* 10.2 (1989), pp. 271–286. ISSN: 0196-6774. DOI: https://doi.org/10.1016/0196-6774(89)90016-3. URL: http://www.sciencedirect.com/science/article/pii/0196677489900163.

[7] Charles J Colbourn, Bradley M Debroni, and Wendy J Myrvold. "Estimating the coefficients of the reliability polynomial". In: *Congressus Numerantium* 62 (1988), pp. 217–223.

[8] Charles J. Colbourn, Wendy J. Myrvold, and Eugene Neufeld. "Two Algorithms for Unranking Arborescences". In: *Journal of Algorithms* 20.2 (1996), pp. 268 –281. ISSN: 0196-6774. DOI: https://doi.org/10.1006/jagm.1996.0014. URL: http://www.sciencedirect.com/science/article/pii/S0196677496900140.

[9] Samir Datta et al. "Dynamic complexity of Reachability: How many changes can we handle?" In: *CoRR* abs/2004.12739 (2020). arXiv: 2004.12739. URL: https://arxiv.org/abs/2004.12739.

[10] Shlomi Dolev and Daniel Khankin. "Random Spanning Trees for Expanders, Sparsifiers, and Virtual Network Security". In: *arXiv preprint arXiv:1612.02569* (2016).

[11] 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*. STOC 2017. Montreal, Canada: Association for Computing Machinery, 2017, 730–742. ISBN: 9781450345286. DOI: 10.1145/3055399.3055499. URL: https://doi.org/10.1145/3055399.3055499.

[12] Alan Frieze et al. “Expanders via Random Spanning Trees”. In: *SIAM Journal on Computing* 43.2 (2014), pp. 497–513. DOI: 10.1137/120890971. eprint: https://doi.org/10.1137/120890971. URL: https://doi.org/10.1137/120890971.

[13] Wai Shing Fung and Nicholas J. A. Harvey. “Graph Sparsification by Edge-Connectivity and Random Spanning Trees”. In: *CoRR* abs/1005.0265 (2010). arXiv: 1005.0265. URL: http://arxiv.org/abs/1005.0265.

[14] S. O. Gharan, A. Saberi, and M. Singh. “A Randomized Rounding Approach to the Traveling Salesman Problem”. In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*. 2011, pp. 550–559.

[15] 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, 576–585.

[16] A Guénoche. “Random spanning tree”. In: *Journal of Algorithms* 4.3 (1983), pp. 214 –220. ISSN: 0196-6774. DOI: https://doi.org/10.1016/0196-6774(83)90022-6. URL: http://www.sciencedirect.com/science/article/pii/0196677483900226.

[17] Nicholas J. A. Harvey. “Algebraic Algorithms for Matching and Matroid Problems”. In: *SIAM J. Comput.* 39.2 (July 2009), 679–702. ISSN: 0097-5397. DOI: 10.1137/070684008. URL: https://doi.org/10.1137/070684008.

[18] Nicholas JA Harvey and Keyulu Xu. “Generating random spanning trees via fast matrix multiplication”. In: *LATIN 2016: Theoretical Informatics*. Springer. 2016, pp. 522–535.

[19] D. Kandel et al. “Shuffling biological sequences”. In: *Discrete Applied Mathematics* 71.1 (1996), pp. 171 –185. ISSN: 0166-218X. DOI: https://doi.org/10.1016/S0166-218X(97)81456-4. URL: http://www.sciencedirect.com/science/article/pii/S0166218X97814564.

[20] Jonathan A. Kelner and Aleksander Madry. “Faster Generation of Random Spanning Trees”. In: *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’09. USA: IEEE Computer Society, 2009, 13–21. ISBN: 9780769538501.

[21] G. Kirchhoff. “Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird”. In: *Annalen der Physik* 148.12 (1847), pp. 497–508. DOI: 10.1002/andp.18471481202. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.18471481202. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.18471481202.

[22] V.G Kulkarni. “Generating random combinatorial objects”. In: *Journal of Algorithms* 11.2 (1990), pp. 185 –207. ISSN: 0196-6774. DOI: https://doi.org/10.1016/0196-6774(90)90002-V. URL: http://www.sciencedirect.com/science/article/pii/019667749090002V.

[23] Aleksander Mądry, 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 ’15. San Diego, California: Society for Industrial and Applied Mathematics, 2015, 2019–2036.

[24] Louis D. Nel and Charles J. Colbourn. "Combining monte carlo estimates and bounds for network reliability". In: *Networks* 20.3 (1990), pp. 277–298. DOI: 10.1002/net.3230200303. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230200303. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230200303.

[25] James Gary Propp and David Bruce Wilson. "How to Get a Perfectly Random Sample from a Generic Markov Chain and Generate a Random Spanning Tree of a Directed Graph". In: *J. Algorithms* 27 (1998), pp. 170–217.

[26] Luís M. S. Russo, Andreia Sofia Teixeira, and Alexandre P. Francisco. "Linking and Cutting Spanning Trees". In: *Algorithms* 11.4 (2018). ISSN: 1999-4893. DOI: 10.3390/a11040053. URL: https://www.mdpi.com/1999-4893/11/4/53.

[27] 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: Association for Computing Machinery, 2018, 214–227. ISBN: 9781450355599. DOI: 10.1145/3188745.3188852. URL: https://doi.org/10.1145/3188745.3188852.

[28] Terrance Tao. *Matrix identities as derivatives of determinant identities*. https://terrytao.wordpress.com/2013/01/13/matrix-identities-as-derivatives-of-determinant-identities/.

[29] Nisheeth K. Vishnoi. "Lx = b". In: *Foundations and Trends® in Theoretical Computer Science* 8.1–2 (2013), pp. 1–141. ISSN: 1551-305X. DOI: 10.1561/0400000054. URL: http://dx.doi.org/10.1561/0400000054.

[30] Wikipedia contributors. *Schur complement — Wikipedia, The Free Encyclopedia*. [Online; accessed 28-May-2020]. 2020. URL: https://en.wikipedia.org/w/index.php?title=Schur_complement&oldid=947233275.

[31] 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: Association for Computing Machinery, 1996, 296–303. ISBN: 0897917855. DOI: 10.1145/237814.237880. URL: https://doi.org/10.1145/237814.237880.