A Unified Approach to Scalable Spectral Sparsification of Directed Graphs

Ying Zhang
Stevens Institute of Technology
Hoboken, New Jersey
yzhan232@stevens.edu

Zhiqiang Zhao
Michigan Technological University
Houghton, Michigan
qzzhao@mtu.edu

Zhuo Feng
Stevens Institute of Technology
Hoboken, New Jersey
zhuo.feng@stevens.edu

ABSTRACT

Recent spectral graph sparsification research allows constructing nearly-linear-sized subgraphs that can well preserve the spectral (structural) properties of the original graph, such as the first few eigenvalues and eigenvectors of the graph Laplacian, leading to the development of a variety of nearly-linear time numerical and graph algorithms. However, there is not a unified approach that allows for truly-scalable spectral sparsification of both directed and undirected graphs. In this work, we prove the existence of linear-sized spectral sparsifiers for general directed graphs and introduce a practically-efficient and unified spectral graph sparsification approach that allows sparsifying real-world, large-scale directed and undirected graphs with guaranteed preservation of the original graph spectra. By exploiting a highly-scalable (nearly-linear complexity) spectral matrix perturbation analysis framework for constructing nearly-linear sized (directed) subgraphs, it enables us to well preserve the key eigenvalues and eigenvectors of the original (directed) graph Laplacians. The proposed method has been validated using various kinds of directed graphs obtained from public domain sparse matrix collections, showing promising results for solving directed graph Laplacians, spectral embedding, and partitioning of general directed graphs, as well as approximately computing (personalized) PageRank vectors.

KEYWORDS

Spectral graph theory, directed graphs, PageRank, Laplacian solver, graph partitioning

ACM Reference Format:
Ying Zhang, Zhiqiang Zhao, and Zhuo Feng. 2020. A Unified Approach to Scalable Spectral Sparsification of Directed Graphs. In Proceedings of ACM Conference (Conference’17). ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnn.nnnnnn

1 INTRODUCTION

Many research problems for simplifying large graphs leveraging spectral graph theory have been extensively studied by mathematicians and theoretical computer science (TCS) researchers in the past decade [1, 7, 8, 16, 19, 24, 27]. Recent spectral graph sparsification research allows constructing nearly-linear-sized 1 subgraphs that can well preserve the spectral (structural) properties of the original graph, such as the first few eigenvalues and eigenvectors of the graph Laplacian. The related results can potentially lead to the development of a variety of nearly-linear time numerical and graph algorithms for solving large sparse matrices and partial differential equations (PDEs), graph-based semi-supervised learning (SSL), computing the stationary distributions of Markov chains and personalized PageRank vectors, spectral graph partitioning and data clustering, max flow and multi-commodity flow of undirected graphs, nearly-linear time circuit simulation and verification algorithms, etc [5, 7, 8, 12, 13, 15, 17, 27, 28, 31, 32].

However, there is not a unified approach that allows for truly-scalable spectral sparsification of both directed and undirected graphs. For example, the state-of-the-art sampling-based methods for spectral sparsification are only applicable to undirected graphs [17, 25, 28]; the latest algorithmic breakthrough in spectral sparsification of directed graphs [7, 8] can only handle strongly-connected directed graphs 2, which inevitably limits its applications when confronting real-world graphs, since many directed graphs may not be strongly connected, such as the graphs used in chip design automation (e.g. timing analysis) tasks as well as the graphs used in machine learning and data mining tasks. Consequently, there is still a pressing need for the development of highly-robust (theoretically-rigorous) and truly-scalable (nearly-linear complexity) algorithms for reducing real-world large-scale (undirected and directed) graphs while preserving key graph spectral (structural) properties.

This paper proves the existence of linear-sized spectral sparsifiers for general directed graphs, and introduces a practically-efficient and unified spectral sparsification approach that allows simplifying real-world, large-scale directed and undirected graphs with guaranteed preservation of the original graph spectra. More specifically, we exploit a highly-scalable (nearly-linear complexity) spectral matrix perturbation analysis framework for constructing ultra-sparse (directed) subgraphs that can well preserve the key eigenvalues and eigenvectors of the original graph Laplacians. Unlike the prior state-of-the-art methods that are only suitable for handling specific types of graphs (e.g. undirected or strongly-connected directed graphs [8, 25]), the proposed approach is more general and thus will allow for truly-scalable spectral sparsification of a much wider range of real-world complex graphs that may involve billions of elements. The spectrally-sparsified directed graphs constructed by the proposed approach will potentially lead to the development of much faster numerical and graph-related algorithms. For example,

1The number of edges is close to the number of nodes in the sparsifier.
2A strongly connected directed graph is a directed graph in which any node can be reached from any other node along with direction.
spectrum-sparsified social (data) networks allow for more efficient modeling and analysis of large social (data) networks; spectrum-sparsified neural networks allow for more scalable model training and processing in emerging machine learning tasks; spectrum-sparsified web-graphs allow for much faster computations of personalized PageRank vectors; spectrum-sparsified integrated circuit networks will lead to more efficient partitioning, modeling, simulation, optimization and verification of large chip designs, etc.

The rest of this paper is organized as follows. Section 2 provides a brief introduction to graph Laplacians and spectral sparsification of directed graphs. In Section 3, a scalable and unified spectral sparsification framework for general graphs is described in detail. Section 4 describes a practically-efficient spectral sparsification approach, while Section 5 introduces potential applications of the proposed graph sparsification framework. Section 6 demonstrates extensive experimental results for a variety of real-world, large-scale directed graphs, which is followed by the conclusion of this work in Section 7.

2 PRELIMINARIES

2.1 Laplacians for (un)directed graphs

Consider a directed graph \( G = (V, E_G, w_G) \) with \( V \) denoting the set of vertices, \( E_G \) representing the set of directed edges, and \( w_G \) denoting the associated edge weights. In the following, we denote the diagonal matrix by \( D_G \) with \( D_G(i, i) = \text{outdegree of node } i \) as well as the adjacency matrix of \( G \) by \( A_G \):

\[
A_G(i, j) = \begin{cases} 
    w_{ij} & \text{if } (i, j) \in E_G \\
    0 & \text{otherwise}.
\end{cases}
\]

Then the directed Laplacian matrix can be constructed as follows [8]:

\[
L_G = D_G - A_G^T.
\]

Let \( n = |V|, m = |E_G| \), and undirected graphs can be converted into equivalent directed graphs by replacing each undirected edge with two opposite directed edges. While for most direct graphs \( L_G \) may not be a symmetric matrix.

It can be shown that any directed (undirected) graph Laplacian constructed using (2) will satisfy the following properties: 1) Each column (and row) sum is equal to zero; 2) All off-diagonal elements are non-positive; 3) The Laplacian matrix is asymmetric (symmetric and indefinite) (positive semidefinite).

2.2 Spectral sparsification of undirected graphs

Graph sparsification aims to find a subgraph (sparsifier) \( S = (V, E_S, w_S) \) that has the same set of vertices but much fewer edges than the original graph \( G \). There are two types of sparsification methods: the cut sparsification methods preserve cuts in the original graph through random sampling of edges [2], whereas spectral sparsification methods preserve the graph spectral (structural) properties, such as distances between vertices, effective resistances, cuts in the graph, as well as the stationary distributions of Markov chains [7, 8, 27]. Therefore, spectral graph sparsification is a much stronger notion than cut sparsification.

For undirected graphs, spectral sparsification aims to find an ultra-sparse subgraph proxy that is spectrally-similar to the original one. \( G \) and \( S \) are said to be \( \sigma \)-spectrally similar if the following condition holds for all real vectors \( x \in \mathbb{R}^V \):

\[
\frac{\lambda_{\text{max}}}{\sigma} \leq L_G x^T L_S x \leq \sigma x^T L_G x,
\]

where \( L_G \) and \( L_S \) denote the symmetric diagonally dominant (SDD) Laplacian matrices of graphs \( G \) and \( S \), respectively. By defining the ratio condition number to be \( \kappa(L_G, L_S) = \lambda_{\text{max}}/\lambda_{\text{min}} \), where \( \lambda_{\text{max}} \) (\( \lambda_{\text{min}} \)) denotes the largest (smallest) eigenvalues of \( L_G^T L_G \) and \( L_S^T L_S \) denotes the Moore-Penrose pseudoinverse of \( L_S \). If it is further shown that \( \kappa(L_G, L_S) \leq \sigma^2 \), implying that a smaller relative condition number or \( \sigma^2 \) corresponds to a higher (better) spectral similarity between two graphs.

2.3 Spectral sparsification of directed graphs

A significant progress has been made for spectral analysis of directed graphs in [6], which for the first time has proved the Cheeger inequality for directed graphs and shown the connection between directed graph partitioning and the smallest (nontrivial) eigenvalue of directed Laplacian. More specifically, the transition probability matrix and the stationary distributions of Markov chains have been exploited for constructing the undirected Laplacians for strongly-connected directed graphs. The latest algorithmic breakthrough in spectral sparsification for strongly-connected directed graphs has been introduced based on the results in [6], which proposes to first convert strongly connected graphs into Eulerian graphs via Eulerian scaling, and subsequently sparsify the undirected graphs obtained via Laplacian symmetrization [6] by leveraging existing spectral graph theory for undirected graphs [8]. It has been shown that such an approach for directed graphs can potentially lead to the development of almost-linear-time algorithms for solving asymmetric linear systems, computing the stationary distribution of a Markov chain, computing expected commute times in a directed graph, etc [8].

For directed graphs, the subgraph \( S \) can be considered spectrally similar to the original graph \( G \) if the condition number or the ratio between the largest and smallest singular values of \( L_S^T L_G \) is close to 1 [7, 8]. Since the singular values of \( L_S^T L_G \) correspond to the square roots of eigenvalues of \((L_S^T L_G)^T (L_S^T L_G)\), spectral sparsification of directed graphs is equivalent to finding an ultra-sparse subgraph \( S \) such that the condition number of \((L_S^T L_G)^T (L_S^T L_G)\) is small enough.

3 A UNIFIED SPARSIFICATION FRAMEWORK

3.1 Overview of our approach

We introduce a unified spectral graph sparsification framework that allows handling both directed and undirected graphs in nearly-linear time. The core idea of our approach is to leverage a novel spectrum-preserving Laplacian symmetrization procedure to convert directed graphs into undirected ones (as shown in Figure 1). Then existing spectral sparsification methods for undirected graphs [1, 12, 13, 19, 27] can be exploited for directed graph spectral sparsification tasks.

Our approach for symmetrizing directed graph Laplacians is motivated by the following fact: the eigenvalues of \((L_S^T L_G)^T (L_S^T L_G)\) will always correspond to the eigenvalues of \((L_S L_S^T)^{-1} L_G L_G^T \) under the condition that \( L_G \) and \( L_S \) are diagonalizable. It can be shown that \( L_G L_G^T \) and \( L_S L_S^T \) can be considered as special graph Laplacian matrices corresponding to undirected graphs that may contain
negative edge weights. Consequently, as long as a directed subgraph $S$ can be found such that the undirected graphs corresponding to negative edge weights.

Since the eigenspace related to outgoing edges of directed graphs $L$ leads to strongly connected, and thus can be applied sparsification [7, 8], our approach does not require the underlying directed graph to be spectrally similar to the original directed graph $G$. Consequently, as long as a directed subgraph $S$ and $G$ after symmetrization will have the all-one vector as its eigenvector $u_i$ of $L_S$, that is also a Symmetric Positive Semi-definite (SPS) matrix.

In the following, assume that $G = (V, E_G, w_G)$ is a weighted directed graph, whereas $S = (V, E_S, w_S)$ is its initial spectral sparsifier (subgraph), such as a spanning subgraph. Define $L_{G_u} = L_G^T L_G$ and $L_S = L_S^T L_S$ to be the undirected graph Laplacians obtained via the proposed symmetrization procedure for $G$ and $S$.

### 3.2 Spectrum-preserving symmetrization

Performing singular value decomposition (SVD) on $L_G$ leads to $L_G = \sum_i \sigma_i \mathbf{e}_i \mathbf{e}_i^T$, where $\mathbf{e}_i$ and $\sigma_i$ are the left and right eigenvectors of $L_G$, respectively. $^3$ It should be noted that $\mathbf{e}_i$ and $\mathbf{e}_i$ with $i = 1, \ldots, n$ span the eigenspace of $L_G^T L_G$ and $L_G^T L_G$, respectively. Since the eigenspace related to outgoing edges of directed graphs needs to be preserved, we will only focus on the Laplacian symmetrization matrix $L_{G_u} = L_G^T L_G$ that is also a Symmetric Positive Semi-definite (SPS) matrix.

**Theorem 3.1.** For any directed Laplacian $L_G$, its undirected graph Laplacian $L_{G_u}$ after symmetrization will have the all-one vector as its null space and correspond to an undirected graph that may include negative edge weights.

$^3$The pseudoinverse of $L_G$ is $L_G^{-1} = \sum_i \frac{1}{\sigma_i} \mathbf{e}_i \mathbf{e}_i^T$

**Proof.** Each element $(i, j)$ in $L_{G_u}$ can be written as follows:

$$
L_{G_u} = \begin{cases} D_{G_{ii}}^2 + \sum_k A_{G_{ki}}^2 & i = j \\ \sum_k (-A_{G_{ki}}A_{G_{kj}} + A_{G_{ki}}D_{G_{kj}} + D_{G_{ki}}A_{G_{kj}}) & i \neq j. \end{cases}
$$

(3)

It can be shown that the following is always true:

$$
L_{G_u} = \sum_{i,j} L_{G_u} = \sum_k L_{G_{ik}} L_{G_{kj}} + \sum_{i,j \neq i} L_{G_{ik}} L_{G_{jk}} = \sum_k L_{G_{ik}} \left( L_{G_{ik}} + \sum_{j \neq i} L_{G_{jk}} \right) = 0,
$$

(4)

which indicates the all-one vector is the null space of $L_{G_u}$. For directed graphs, it can be shown that if a node has more than one outgoing edge, in the worst case the neighboring nodes pointed by such outgoing edges will form a clique possibly with negative edge weights in the corresponding undirected graph after symmetrization.

As an example shown in Figure 2, when edge $e_2$ is added into the initial graph $G$ that includes a single edge $e_1$, an extra edge (shown in red dashed line) coupling with $e_1$ will be created in the resultant undirected graph $G_u$. Similarly, when an edge $e_3$ is further added, two extra edges coupling with $e_1$ and $e_2$ will be created in $G_u$. When the last edge $e_4$ is added, it forms a clique.

It can be shown that $G_u$ will contain negative edge weights under the following condition:

$$
\sum_k (A_{G_{ki}} D_{G_{kj}} + D_{G_{ki}} A_{G_{kj}}) > \sum_k A_{G_{ki}} A_{G_{kj}}.
$$

(5)

3.3 Existence of linear-sized spectral sparsifier

It has been shown that every undirected graph with positive edge weights has a Twice-Ramanujan spectral sparsifier with positive edge weights to spectrally-approximate the original graph [1, 19]. In this work, we extend the above theory to deal with the undirected graphs obtained through the proposed Laplacian symmetrization procedure that may introduce negative weights.

**Theorem 3.2.** For a given directed graph $G$ and its undirected graph $G_u = (V, E_{G_u}, w_{G_u})$ obtained via Laplacian symmetrization, there exists a $(1+\epsilon)$-spectral sparsifier $S$ with $O(\epsilon^2)$ edges such that its undirected graph $S_u = (V, E_S, w_{S_u})$ after symmetrization satisfies the following condition for any $x \in \mathbb{R}^n$:

$$
(1-\epsilon)x^T L_{G_u} x \leq x^T L_{S_u} x \leq (1+\epsilon)x^T L_{G_u} x.
$$

(6)

We will need the following the lemma [33] to prove our theorem.

**Lemma 3.3.** Let $\epsilon > 0$, and $u_1, u_2, \ldots, u_m$ denote a set of vectors in $\mathbb{R}^n$ that allow expressing the identity decomposition as:

$$
\sum_{i=1}^m u_i u_i^T = I_{m \times m}.
$$

(7)
where $I_{n \times n}$ denotes an identity matrix. Then there exists an $O(m/\epsilon^2)$-time algorithm [19] that can find non-negative coefficients $\{t_i\}_{i=1}^m$ such that at most $\left|\{t_i | t_i > 0\}\right| = O(n/\epsilon^2)$ and for any $x \in \mathbb{R}^n$:

$$(1 - \epsilon)x^TI_{n \times n}x \leq \sum_{i} t_i x^T u_i u_i^T x \leq (1 + \epsilon)x^TI_{n \times n}x. $$

(8)

**Proof.** Any directed graph Laplacian can also be written as:

$$L_G = B^T W C,$$

(9)

where $B_{m \times n}$ and $C_{n \times n}$ are the edge-vertex incidence matrix and the injection matrix defined below:

$$B(i, v) = \begin{cases} 1 & \text{if $v$ is $i$-th edge’s head} \\ -1 & \text{if $v$ is $i$-th edge’s tail}, \end{cases}$$

(10)

$$C(i, v) = \begin{cases} 1 & \text{if $v$ is $i$-th edge’s head} \\ 0 & \text{if $v$ is $i$-th edge’s tail}, \end{cases}$$

(11)

and $W_{m \times m}$ is the diagonal matrix with $W(i, i) = w_i$. We show how to construct the vectors $u_i$ for $i = 1, \ldots, m$ in (8), which will suffice for proving the existence of linear-sized spectral sparsifiers for directed graphs. (9) allows writing the undirected Laplacian after symmetrization as $L_G = B^T W C C^T W B$ and setting $W_\theta = W C \theta$. Since $W_\theta$ is an SPS matrix, we can always construct a $U$ matrix with $u_i$ for $i = 1, \ldots, m$ as its column vectors:

$$U_{n \times m} = [u_1, \ldots, u_m] = L_{G_\theta}^{1/2} B^T W_\theta^{1/2}.$$

(12)

$U$ contains all the information of the directed edges in $G$. It can be shown that $U$ satisfies the following equation:

$$U U^T = \sum_{i} u_i u_i^T = L_{G_\theta}^{1/2} B^T W_\theta B L_{G_\theta}^{1/2} = L_{G_\theta}^{1/2} L_{G_\theta} L_{G_\theta}^{1/2} = I_{r \times r},$$

(13)

where $r$ is the rank of the $L_{G_\theta}$. According to Lemma 3.3, we can always construct a diagonal matrix $T \in \mathbb{R}^{m \times m}$ with $t_i$ as its $i$-th diagonal element. Then there will be at most $O(n/\epsilon^2)$ positive diagonal elements in $T$, which allows constructing $L_{s_\theta} = B^T W_\theta^{1/2} T W_\theta^{1/2} B$ that corresponds to the directed subgraph $S$ for achieving $(1 + \epsilon)$-spectral approximation of $G$ as required by (6). It can be shown that each $u_i$ with a nonzero $t_i$ coefficient corresponds to the outgoing edges pointed by the same node. Consequently, for directed graphs with bounded degrees, there will be $O(n/\epsilon^2)$ total number of directed edges in the $(1 + \epsilon)$-spectral sparsifier $S$. □

### 4.1 Initial subgraph sparsifier

The idea of using subgraphs as preconditioners for more efficiently solving linear system of equations has been first introduced in [29], showing that a maximum-spanning-tree (MST) subgraph can be leveraged as an $m\epsilon$-proxy of the original undirected graph. Recent nearly-linear time spectral sparsification algorithms for undirected graphs exploit similar ideas based on low-stretch spanning tree subgraphs [11–13]. In this work, we exploit ideas that are closely related to spanning-tree based subgraphs as well as the Markov chains of random walks. The following procedure for constructing the initial subgraph sparsifiers for directed graphs has been developed:

1. Compute the transition matrix $P_{t u m} = D_{t u m}^{-1} A_{t u m}$ from $A_{t u m} = A_G + A_G^T$, where, $L_{t u m}$ and $D_{t u m}$ are Laplacian matrix and diagonal matrix for graph $G_{t u m}$ respectively;
2. Construct an undirected graph $G'_{t u m}$ with $P_{t u m}$ as its adjacency matrix, and find an MST subgraph $S_{m s t}$ of $G'_{t u m}$;
3. Construct a directed subgraph $S^0$ according to the $S_{m s t}$, and check every node in $S^0$, to which the nodes that have at least one outgoing edge in $G$ but none in $S^0$, include the outgoing edge with the largest weight into $S^0$;
4. Return the latest subgraph $S$ as the initial spectral sparsifier.

Step 3) will make sure that the graph Laplacians of directed graphs $G$ and $S^0$ share the same rank and nullity. As aforementioned, if a node has more than one outgoing edge, in the worst case the neighboring nodes pointed by such outgoing edges will form a clique in the corresponding undirected graph after symmetrization. Consequently, when constructing the initial subgraphs from the original directed graphs, it is important to limit the number of outgoing edges for each node so that the resultant undirected graph after Laplacian symmetrization will not be too dense. To this end, emerging graph transformation techniques that allow splitting high-degree nodes into multiple low-degree ones can be exploited. For example, recent research shows such split (e.g. uniform-degree tree) transformations can dramatically reduce graph irregularity while preserving critical graph connectivity, distance between node pairs, the minimal edge weight in the path, as well as outdegrees and indegrees when using push-based and pull-based vertex-centric programming [23].

### 4.2 Edge spectral sensitivity

Denote the descending eigenvalues and eigenvectors of $L_{s_\theta}^{-1} L_{G_\theta}$ by $\mu_{max} = \mu_1 \geq \mu_2 \geq \cdots \geq \mu_n > 0$ and $v_1, v_2, \ldots, v_n$, respectively. In addition, let matrix $V = [v_1, v_2, \ldots, v_n]$. Consider the following first-order generalized eigenvalue perturbation problem:

$$L_{G_\theta}(v_1 + \delta v_1) = (\mu_1 + \delta \mu_1)(L_{s_\theta} + \delta L_{s_\theta})(v_1 + \delta v_1),$$

(14)

where a small perturbation $\delta L_{s_\theta}$ in $s_\theta$ is introduced and subsequently leads to the perturbed generalized eigenvalues and eigenvectors $\mu_1 + \delta \mu_1$ and $v_1 + \delta v_1$. Then the task of spectral sparsification of general (un)directed graphs can be formulated as follows: recover as few as possible extra edges back to the initial subgraph $S$ such that the largest eigenvalues or the condition number of $L_{s_\theta}^{-1} L_{G_\theta}$ can be dramatically reduced. Expanding (14) by only keeping the...
first-order terms simply leads to:

\[ L_{G_n} \delta v_i = \mu_i L_{S_n} \delta v_i + \delta \mu_i L_{S_n} v_i + \mu_i \delta L_{S_n} v_i, \quad \delta v_i = \sum_{j=1}^{n} \delta_{ij} v_j. \quad (15) \]

Since both \( L_{G_n} \) and \( L_{S_n} \) are SPS matrices, \( L_{S_n} \)-orthogonal generalized eigenvectors \( v_i \) for \( i = 1, \ldots, n \) can be found to satisfy:

\[ v_i^T L_{S_n} v_j = \begin{cases} 1, & i = j \\ 0, & i \neq j. \end{cases} \quad (16) \]

By expanding \( \delta L_{S_n} \) with only the first-order terms, the spectral perturbation for each off-subgraph edge can be expressed as:

\[ \frac{\delta \mu_i}{\mu_i} = -v_i^T \delta L_{S_n} v_i = -v_i^T (\delta L_{S_n} L_{S_n}^T + L_{S_n} \delta L_{S_n}) v_i, \quad (17) \]

where \( \delta L_{S_n} = w_{p,q} \epsilon_{p,q} \epsilon_{p}^T \) for \((p, q) \in E_G \setminus E_S \). \( \epsilon_{p,q} \in \mathbb{R}^n \) denotes the vector with only the \( p \)-th element being 1 and others being 0, and \( \epsilon_{p,q} = \epsilon_p - \epsilon_q \). The spectral sensitivity \( \delta \mu_{p,q} \) for the off-subgraph edge \((p, q)\) can be computed by:

\[ \delta \mu_{p,q} = v_i^T \delta L_{S_n} v_i. \quad (18) \]

Equation (18) allows computing the spectral sensitivity of the dominant generalized eigenvalue with respect to the Laplacian perturbation due to adding extra off-subgraph edge into \( G \), which thus can be leveraged to rank the spectral importance of each edge. As a result, spectral sparsification of general (un)directed graphs can be achieved by only recovering the top few off-subgraph edges that have the largest spectral sensitivities into \( G \).

Since the above framework is based on spectral matrix perturbation analysis, compared to existing spectral sparsification methods that are limited to specific types of graphs, such as undirected graphs or strongly-connected directed graphs [7, 8], the proposed graph sparsification framework is more universal and thus will be applicable to a much broader range of graph problems.

### 4.3 Approximate dominant eigenvectors

A generalized power iteration method is proposed to allow much faster computation of the dominant generalized eigenvectors for spectral sparsification of directed graphs. Starting from any initial random vector expressed as \( h_0 = \sum_{i} \alpha_i v_i \), the dominant generalized eigenvector \( v_1 \) can be approximately computed by performing the following \( t \)-step power iterations:

\[ v_1 = h_1 = (L_{S_n}^T L_{G_n})^t h_0 = \sum_{i} \alpha_i^t v_i. \quad (19) \]

When the number of power iterations is small (e.g., \( t \leq 3 \)), \( h_1 \) will be a linear combination of the first few dominant generalized eigenvectors corresponding to the largest few eigenvalues. Then the spectral sensitivity for the off-subgraph edge \((p, q)\) can be approximately computed by

\[ \delta \mu_{p,q} = h_1^T \delta L_{S_n} h_1, \quad (20) \]

which will allow us to well approximate the spectral sensitivity in (17) for ranking off-subgraph edges during spectral sparsification. The key to fast computation of \( h_1 \) using generalized power iterations is to quickly solve the linear system of equations \( L_{G_n} x = h_0 \), which requires to explicitly construct \( L_{G_n} \) rather than \( L_{G_n} \). To this end, we leverage the latest Lean Algebraic Multigrid (LAMG) algorithm that is capable of handling the undirected graphs with negative edge weights as long as the Laplacian matrix is SPS. The LAMG algorithm also enjoys an empirical \( O(m) \) runtime complexity for solving large scale graph Laplacian matrices [20].

### 4.4 Lean algebraic multigrid (LAMG)

The setup phase of LAMG contains two main steps: First, a nodal elimination procedure is performed to eliminate disconnected and low-degree nodes. Next, a node aggregation procedure is applied for aggregating strongly connected nodes according to the following affinity metric \( c_{uv} \) for nodes \( u \) and \( v \):

\[ c_{uv} = \frac{|| (X_u, X_v) ||_2^2}{(X_u, X_u)(X_v, X_v)}, \quad (X, Y) = \sum_{k=1}^{K} x^{(k)} y^{(k)} \quad (21) \]

where \( X_u = (x_u^{(1)} \ldots x_u^{(K)}) \) is computed by applying a few Gauss-Seidel (GS) relaxations using \( K \) initial random vectors to the linear system equation \( L_{G_n} x = 0 \). Let \( x \) represent the approximation of the true solution \( x \) after applying several GS relaxations to \( L_{G_n} x = 0 \). Due to the smoothing property of GS relaxation, the latest error can be expressed as \( e_x = x - \hat{x} \), which will only contain the smooth components of the initial error, while the highly oscillating modes will be effectively damped out [4]. It has been shown that the node affinity metric \( c_{uv} \) can effectively reflect the distance or strength of connection between nodes in a graph: a larger \( c_{uv} \) value indicates a stronger connection between nodes \( u \) and \( v \) [20]. Therefore, nodes \( u \) and \( v \) are considered strongly connected to each other if \( x_u \) and \( x_v \) are highly correlated for all the \( K \) test vectors, which thus should be aggregated to form a coarse level node.

Once the multilevel hierarchical representations of the original graph (Laplacians) have been created according to the above scheme, algebraic multigrid (AMG) solvers can be built and subsequently leveraged to solve large Laplacian matrices efficiently.

### 4.5 Edge spectral similarities

The proposed spectral sparsification algorithm will first sort all off-subgraph edges according to their spectral sensitivities in descending order \((p_1, q_1), (p_2, q_2), \ldots \) and then select top few off-subgraph edges to be recovered to the initial subgraph. To avoid recovering redundant edges into the subgraph, it is indispensable to check the edge spectral similarities: only the edges that are not similar to each other will be added to the initial sparsifier. To this end, we exploit the following spectral embedding of off-subgraph edges using approximate dominant generalized eigenvectors \( h_0 \) computed by (19):

\[ \psi_{p,q}(h_t) = \sum_{k} w_{p,q,k} h_t^T (\epsilon_{p,q}^T \epsilon_{p,q} + \epsilon_{p,q}^T \epsilon_{p,q}) h_0, \quad (22) \]

where \( (p, q) \) are the directed edges sharing the same head with \((p, q)\) but different tails. Then the proposed scheme for checking spectral similarity of two off-subgraph edges will include the following steps:

1. Perform \( t \)-step power iterations with \( r = O(\log n) \) initial random vectors \( h_0^{(1)} \ldots h_0^{(r)} \) to compute \( r \) approximate dominant generalized eigenvectors \( h_1^{(1)} \ldots h_1^{(r)} \).
2. For each edge \((p, q)\), compute a \( r \)-dimensional spectral embedding vector \( s_{p,q} \in \mathbb{R}^r \) with \( s_{p,q}(r) = \psi_{p,q}(h_t^{(r)}) \).
Algorithm 1 Edge Similarities Checking

\textbf{Input:} \(E_{\text{list}}, L_G, L_S, d_{\text{out}}, \epsilon\)
1: Perform \(t\)-step power iterations with \(r = O(\log n)\) initial random vectors \(h_0, \ldots, h_r\) to compute \(r\) approximate dominant generalized eigenvectors \(h_1, \ldots, h_r\);  
2: Choose each edge \((p, q)\) whose starting node has out-degree less than \(d_{\text{out}}\) into a new \(E_{\text{list}}\);  
3: Compute a \(r\)-dimensional edge similarity vector \(s_p, q \in \mathbb{R}^r\) for \(\forall (p, q) \in E_{\text{list}}: s_p, q(r) = \psi_{p, q}(h_r)^T\);  
4: Let \(E_{\text{addlist}} = \{(p, q, 1)\}\);  
5: \textbf{for} \(i = 1: E_{\text{list}}\) \textbf{do}  
6: \textbf{if} \(1 - \frac{\|s_p, q - s_p, q\|}{\max (\|s_p, q\|, \|s_p, q\|)} < \epsilon\), for \(\forall (p, q) \in E_{\text{addlist}}\) then  
7: \(E_{\text{addlist}} = E_{\text{addlist}} \cup \{(p, q, t)\}\);  
8: \textbf{end if}  
9: \textbf{end for}  
10: Return graph \(E_{\text{addlist}}\).

4.6 Algorithm flow and complexity
The algorithm flow for directed graph spectral sparsification is described in Algorithm 2, while its complexity has been summarized as follows:

(a) Generate an initial subgraph \(S\) from the original directed graph in \(O(m \log n)\) or \(O(m + n \log n)\) time;
(b) Compute the approximate dominant eigenvector \(h_1\) and the spectral sensitivity of each off-subgraph edge in \(O(m)\) time;
(c) Recover a small amount of spectrally-dissimilar off-subgraph edges into the latest subgraph \(S\) according to their spectral sensitivities and similarities in \(O(m)\) time;
(d) Repeat steps (b) and (c) until the desired condition number or spectral similarity is achieved.

5 APPLICATIONS OF DIRECTED GRAPH SPARSIFICATION
Spectral graph sparsification algorithms can be potentially applied to accelerate many graph and numerical algorithms [8]. In this work, we demonstrate the applications of the proposed sparsification algorithm in solving directed Laplacian problems (e.g., \(L x = b\)), [8, 9], computing personalized PageRank vectors [7], as well as spectral graph partitioning.

5.1 PageRank and personalized PageRank
The idea of PageRank is to give a measurement of the importance for each web page. For example, PageRank algorithm aims to find the most popular web pages, while the personalized PageRank algorithm aims to find the pages that users will most likely to visit. To state it mathematically, the PageRank vector \(p\) satisfies the following equations:
\[
p = A_G^T D^{-1} p.
\]
where \(p\) is also the eigenvector of \(A_G^T D^{-1}\) that corresponds to the eigenvalue equal to 1. Meanwhile, \(p\) represents the stable distribution of random walks on graph \(G\). However, \(D^{-1}\) cannot be defined if there exists nodes that have no outgoing edges. To deal with such situation, a self-loop with a small edge weight can be added for each node.

The stable distributions of (un)directed graphs may not be unique. For example, the undirected graphs that have multiple strongly-connected components, or the directed graphs that have nodes without any outgoing edges, may have non-unique distributions. In addition, it may take very long time for a random walk to converge to a stable distribution on a given (un)directed graph.

To avoid such situation in PageRank, a jumping factor \(\alpha\) that describes the possibility at \(\alpha\) to jump to a uniform vector can be added, which is shown as follows:
\[
p = (1 - \alpha) A_G^T D^{-1} p + \frac{\alpha}{n} 1, \quad (25)
\]
where \(\alpha \in [0, 1]\) is a jumping constant. After applying Taylor expansions, we can obtain that \(p = \frac{\alpha}{n} \sum_i ((1 - \alpha) A_G^T D^{-1})^i 1\). By setting the proper value of \(\alpha\) (e.g., \(\alpha = 0.15\)), the term \(1 - (1 - \alpha)^2\) will be quickly reduced with increasing \(i\). Instead of starting with a uniform vector \(\frac{1}{n}\), a nonuniform personalization vector \(p_r\) can be applied:
\[
p = (1 - \alpha) A_G^T D^{-1} p + \alpha p_r. \quad (27)
\]
In this work, we show that the PageRank vector obtained with the sparsified graph can preserve the original PageRank information. After obtaining the PageRank vector computed using the sparsifier, a few GS relaxations will be applied to further improve the solution quality.
5.2 Directed Laplacian solver

Consider the solution of the following linear systems of equations:

$$Lx = b.$$  

(28)

Recent research has been focused on more efficiently solving the above problem when $L$ is a Laplacian matrix of an undirected graph [15, 18]. In this work, we will mainly focus on solving nonsymmetric Laplacian matrices that correspond to directed graphs.

**Lemma 5.1.** When solving (28), the right preconditioning system is applied, leading to the following alternative linear system of equations:

$$L_Gy = b,$$  

(29)

where vector $b$ will lie in the left singular vector space. When the solution of (29) is obtained, the solution of (28) is given by $L_G^\top y = x$.

It is obvious that solving the above equation is equivalent to solving the problem of $L_GL_G^\top x = b$. In addition, $L_G$ is a Laplacian matrix of an undirected graph that can be much denser than $G$. Therefore, we propose to solve the linear system of $L_S\tilde{y} = b$ instead to effectively approximate (29) since $G_{S_u}$ is sparser than $G$, and more efficient to solve in practice.

We analyze the solution errors based on the generalized eigenv-value problem of $L_G$ and $L_U$. We have $VL_G V^\top = \mu$ and $VL_S V^\top = I$, where $V = [v_1, v_2, \ldots, v_n]$, $\mu$ is the diagonal matrix with its generalized eigenvectors $\mu_i \geq 1$ on its diagonal. Since the errors can be calculated from the following procedure:

$$L_G y - L_S \tilde{y} = L_G (y - \tilde{y}) + (L_G - L_S) \tilde{y} = 0,$$  

(30)

we can write the error term as follows:

$$(y - \tilde{y}) \approx L_G^+ (L_G - L_S) \tilde{y}.$$  

(31)

Since $\tilde{y} = \sum_i a_i v_i$, the error can be further expressed as

$$(y - \tilde{y}) \approx \sum_i a_i (1 - \frac{1}{\mu_i}) |v_i|.$$  

(32)

Therefore, the error term (32) can be generally considered as a combination of high-frequency errors (generalized eigenvectors with respect to high generalized eigenvalues) and low-frequency errors (generalized eigenvectors with respect to low generalized eigenvalues). After applying GS relaxations, the high-frequency error terms can be efficiently removed (smoothed), while the low-frequency errors tend to become zero if the generalized eigenvalues approach 1 considering $(1 - \frac{1}{\mu_i})$ tends to be approaching zero. As a result, the error can be effectively eliminated using the above solution smoothing procedure.

In summary, in the proposed directed Laplacian solver, the following steps are needed:

(a) We will first extract a spectral sparsifier $L_S$ of a given (undirected) directed graph $L_G$. Then, it is possible to compute an approximate solution by exploiting its spectral sparsifier $L_S = L_S L_G^\top$ via solving $\tilde{y} = L_G^\top b$ instead.

(b) Then we improve the approximate solution $\tilde{y}$ by getting rid of the high-frequency errors via applying a few steps of GS iterations [3].

(C) The final solution is obtained from $x = L_G^\top \tilde{y}$.

5.3 Directed graph partitioning

It has been shown that partitioning and clustering of directed graphs can play very important roles in a variety of applications related to machine learning [21], data mining and circuit synthesis and optimization [22], etc. However, the efficiency of existing methods for partitioning directed graphs strongly depends on the complexity of the underlying graphs [21].

In this work, we propose a spectral method for directed graph partitioning problems. For an undirected graph, the eigenvectors corresponding to the first few smallest eigenvalues can be utilized for the spectral partitioning purpose [26]. For a directed graph $G$ on the other hand, the left singular vectors of Laplacian $L_G$ will be required for directed graph partitioning. The eigen-decomposition of its symmetrization $L_{G_u}$ can be written as

$$L_{G_u} = \sum_1^n \mu_i v_i v_i^\top.$$  

(33)

where $0 = \mu_1 \leq \ldots \mu_k$ and $v_1, \ldots, v_k$, with $k \leq n$ denote the Laplacian eigenvalues and eigenvectors, respectively. There may not be $n$ eigenvalues if when there are some nodes without any outgoing edges. In addition, the spectral properties of $L_{G_u}$ are more complicated since the eigenvalues always have multiplicity (either algebraic or geometric multiplicities). For example, the eigenvalues according to the symmetrization of the directed graph in Figure 4 have a few multiplicities: $\mu_2 = \mu_3$, $\mu_4 = \mu_5 = \mu_6 = \mu_7$, $\mu_9 = \mu_{10}$.

Therefore, we propose to exploit the eigenvectors (left singular vectors of directed Laplacian) corresponding to the first few different eigenvalues (singular values of directed Laplacian) for directed graph partitioning. For example, the partitioning result of directed graph in Figure 4 will depend on the eigenvectors of $v_1, v_2, v_4, v_6$ that correspond to eigenvalues of $\mu_1, \mu_2, \mu_4, \mu_6$. As shown in Fig (4), the spectral partitioning results can quite different between the directed and undirected graph with the same set of nodes and edges.

![Figure 3: Eigenvalues distribution of $L_{G_u}$ for the directed graph in Figure 4](image)

In general, it is possible to first extract a spectrally-similar directed graph before any of the prior partitioning algorithms are applied. Since the proposed spectral sparsification algorithm can well preserve the structural (global) properties of the original graphs, the partitioning results obtained from the sparsified graphs will be very similar to the original ones.

6 EXPERIMENTAL RESULTS

The proposed algorithm for spectral sparsification of directed graphs has been implemented using MATLAB and C++. Extensive experiments have been conducted to evaluate the proposed method with

---

\[ \text{Code available at: https://github.com/web1799/directed_graph_sparsification} \]
Figure 4: Spectral partitioning of directed (left) and undirected graphs (right). The nodes within the same cluster are assigned the same color.

Figure 5: The spectral sensitivities (scores) of off-subgraph edges (e2 to e19) for the directed (left) and undirected graph (right).

Table 1: Results of directed graph spectral sparsification

| Test Cases | \(|V_G|\) | \(|E_G|\) | \(|E_S|\) | \(|E_G|\) | time (s) | \(\mu_{max}\) |
|------------|----------|----------|----------|----------|---------|------------|
| gre_115    | 1.1E2    | 4.2E2    | 0.46     | 0.71     | 0.05    | 8.2E4X     |
| gre_185    | 1.8E2    | 1.0E3    | 0.33     | 0.46     | 0.14    | 9.8E3X     |
| harvard500 | 0.5E3    | 2.6E3    | 0.31     | 0.40     | 0.64    | 1.2E5X     |
| cell1      | 0.7E4    | 3.0E4    | 0.31     | 0.37     | 3.10    | 1.0E5X     |
| pesa       | 1.2E4    | 8.0E4    | 0.27     | 0.51     | 8.80    | 5.3E8X     |
| big        | 1.3E4    | 9.0E5    | 0.27     | 0.49     | 12.86   | 4.1E11X    |
| gre_1107   | 1.1E3    | 5.6E3    | 0.26     | 0.39     | 0.24    | 58X        |
| wordnet3   | 0.8E5    | 1.3E5    | 0.64     | 0.84     | 50.00   | 12X        |
| p2p-Gnutella31 | 0.6E5 | 1.5E5    | 0.35     | 0.43     | 11.90   | 6X         |
| p2p-Gnutella05 | 8.8E3 | 3.2E4    | 0.23     | 0.65     | 27.59   | 35X        |
| mathworks100 | 1.0E2 | 5.5E2    | 0.20     | 0.50     | 0.04    | 30X        |
| ibm32      | 3.2E1    | 1.3E2    | 0.46     | 0.57     | 0.02    | 12X        |

eigenvector \(h_s\). We report the total runtime for the eig solver using either the LAMG solver or *eigs* function. \(\mu_{max}\) denotes the reduction rate of the largest generalized eigenvalue of \(L^T_G L_G\). We also plot the detailed reduction rates of the largest generalized eigenvalue when adding different number of off-subgraph edges to the sparsifiers of graph ‘gre_115’ and ‘peta’ in Figure 7. It shows that the largest generalized eigenvalue can be effectively reduced if sufficient off-subgraph edges are included into the sparsifier.

Table 2: the errors between the exact and approximate solutions of \(L_G x = b\) with or without Gauss Seidel smoothing

| Test Cases | gre_115 | gre_185 | cell1 | pesa | big | gre_1107 | wordnet3 |
|------------|---------|---------|-------|------|-----|----------|----------|
| w/o smooth.| 0.41    | 0.42    | 0.44  | 2.1E-6 | 4.3E-3 | 0.6      | 0.72     |
| w/ smooth. | 0.04    | 0.12    | 0.07  | 8.0E-9 | 1.1E-4 | 0.10     | 0.07     |

Figure 6: The correlation of the Personalized PageRank between itself and its sparsifier for the ‘gre_115.mtx’ graph (left) and graph ‘gre_185.mtx’ (right) w/o smoothing.

Table 2 shows the results of the directed Laplacian solver on different directed graphs. It reports relative errors between the exact solution and the solution calculated by the proposed solver with and without smoothing. It shows that errors can be dramatically reduced after smoothing, and our proposed solver can well approximate the true solution of \(L_G x = b\).
This paper proves the existence of linear-sized spectral sparsifiers for general directed graphs, and proposes a practically-efficient and unified spectral graph sparsification framework. Such a novel spectral sparsification approach allows sparsifying real-world, large-scale directed and undirected graphs with guaranteed preservation of the original graph spectral properties. By exploiting a highly-scalable (nearly-linear complexity) spectral matrix perturbation analysis framework for constructing nearly-linear sized (directed) subgraphs, it enables to well preserve the key eigenvalues and eigenvectors of the original (directed) graph Laplacians. The proposed method has been validated using various kinds of directed graphs obtained from public domain sparse matrix collections, showing promising spectral sparsification and partitioning results for general directed graphs.

7 CONCLUSIONS

This paper proves the existence of linear-sized spectral sparsifiers for general directed graphs, and proposes a practically-efficient and unified spectral graph sparsification framework. Such a novel spectral sparsification approach allows sparsifying real-world, large-scale directed and undirected graphs with guaranteed preservation of the original graph spectral properties. By exploiting a highly-scalable (nearly-linear complexity) spectral matrix perturbation analysis framework for constructing nearly-linear sized (directed) subgraphs, it enables to well preserve the key eigenvalues and eigenvectors of the original (directed) graph Laplacians. The proposed method has been validated using various kinds of directed graphs obtained from public domain sparse matrix collections, showing promising spectral sparsification and partitioning results for general directed graphs.

REFERENCES

[1] J. Batson, D. Spielman, and N. Srivastava. Twice-Ramanujan Sparsifiers. SIAM Journal on Computing, 41(6):1704–1721, 2012.
[2] A. A. Benczur and D. R. Karger. Approximating st minimum cuts in o (n 2) time. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (STOC), pages 47–55. ACM, 1996.
[3] W. Briggs. A multigrid tutorial. SIAM Press, 1987.
[4] W. L. Briggs, V. E. Henson, and S. F. McCormick. A multigrid tutorial. volume 72. Siam, 2000.
[5] P. Christiano, J. Kelner, A. Madry, D. Spielman, and S. Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In Proc. ACM STOC, pages 273–282, 2011.
[6] F. Chung. Laplacians and the cheeger inequality for directed graphs. Annals of Combinatorics, 9(1):1–19, 2005.
[7] M. B. Cohen, J. Kelner, R. Kyng, J. Peebles, R. Peng, A. B. Rao, and A. Sidford. Solving Directed Laplacian Systems in Nearly-Linear Time through Sparse LU Factorizations. In Foundations of Computer Science (FOCS), 2018 59th Annual IEEE Symposium on, pages 898–909. IEEE, 2018.
[8] M. B. Cohen, J. Kelner, J. Peebles, R. Peng, A. B. Rao, A. Sidford, and A. Vladu. Almost-linear-time algorithms for markov chains and new spectral primitives for directed graphs. In Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pages 410–419. ACM, 2017.
[9] M. B. Cohen, J. Kelner, J. Peebles, R. Peng, A. Sidford, and A. Vladu. Faster algorithms for computing the stationary distribution, simulating random walks, and more. In 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), pages 583–592. IEEE, 2016.
[10] T. Davis and Y. Hu. The university of Florida sparse matrix collection. ACM Trans. on Math. Soft. (TOMS), 38(1):1, 2011.
[11] M. Elkin, Y. Emek, D. Spielman, and S. Teng. Lower-stretch spanning trees. SIAM Journal on Computing, 38(2):608–628, 2008.
[12] Z. Feng. Spectral graph sparsification in nearly-linear time leveraging efficient spectral perturbation analysis. In Design Automation Conference (DAC), 2016 53rd ACM/EDAC/IEEE, pages 1–6. IEEE, 2016.
[13] Z. Feng. Similarity-aware spectral sparsification by edge filtering. In Design Automation Conference (DAC), 2018, pages 55–55. IEEE, 2018.
[14] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. Deep learning, volume 1. MIT press Cambridge, 2016.
[15] J. A. Kelner, Y. T. Lee, L. Orecchia, and A. Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. In Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms, pages 217–226. SIAM, 2014.
[16] P. Kolev and K. Mehlhorn. A note on spectral clustering. arXiv preprint arXiv:1509.09788, 2015.
[17] I. Koutis, G. Miller, and R. Peng. Approaching Optimality for Solving SDD Linear Systems. In Proc. ACM FOCS, pages 235–244, 2010.
[18] I. Koutis, G. L. Miller, and R. Peng. Approaching optimality for solving sdd linear systems. In Foundations of Computer Science (FOCS), 2009 50th Annual IEEE Symposium on, pages 235–244. IEEE, 2010.
[19] Y. T. Lee and H. Sun. An SDD-based Algorithm for Linear-Sized Spectral Sparsification. In Proceedings of the 69th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, pages 678–687, New York, NY, USA, 2017. ACM.
[20] O. Livne and A. Brandt. Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver. SIAM Journal on Scientific Computing, 34(4):B499–B522, 2012.
[21] F. D. Malliaros and M. Vazirgiannis. Clustering and community detection in directed networks: A survey. Physics Reports, 533(4):95–142, 2013.
[22] G. D. Micheli. Synthesis and optimization of digital circuits. McGraw-Hill Higher Education, 1994.
[23] A. H. Nodheh Sabot, J. Qiu, and Z. Zhao. Tigr: Transforming irregular graphs for gpu-friendly graph processing. In Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems, pages 622–636. ACM, 2018.
[24] R. Peng, H. Sun, and L. Zanetti. Partitioning well-clustered graphs: Spectral clustering works. In Proceedings of The 28th Conference on Learning Theory (COLT), pages 1423–1455, 2015.
[25] D. Spielman and N. Srivastava. Graph sparsification by effective resistances. SIAM Journal on Computing, 40(6):1913–1926, 2011.
[26] D. Spielman and S. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In Foundations of Computer Science (FOCS), 1996. Proceedings., 37th Annual Symposium on, pages 96–105. IEEE, 1996.
[27] D. Spielman and S. Teng. Spectral sparsification of graphs. SIAM Journal on Computing, 40(4):981–1025, 2011.
[28] D. Spielman and S. Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. SIAM Journal on Matrix Analysis and Applications, 35(3):835–885, 2014.
[29] P. M. Vaidya. Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners. A talk based on this manuscript.
[30] R. Yang, R. He, K. Chen, P. Elskenschatzi, W. L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD ’18, 2018.
[31] Z. Zhao and Z. Feng. A spectral graph sparsification approach to scalable vectorless power grid integrity verification. In Proceedings of the 54th Annual Design Automation Conference 2017, page 68. ACM, 2017.
[32] Z. Zhao, Y. Wang, and Z. Feng. SAMG: Sparsified Graph Theoretic Algebraic Multigrid for Solving Large Symmetric Diagonally Dominant (SDD) Matrices. In Proceedings of the 36th International Conference on Computer-Aided Design (ICCAD). ACM, 2017.