Multi-way sparsest cut problem on trees with a control on the number of parts and outliers

Ramin Javadi∗† Saleh Ashkboos‡

Abstract

Given a graph, the sparsest cut problem asks for a subset of vertices whose edge expansion (the normalized cut given by the subset) is minimized. In this paper, we study a generalization of this problem seeking for \( k \) disjoint subsets of vertices (clusters) whose all edge expansions are small and furthermore, the number of vertices remained in the exterior of the subsets (outliers) is also small. We prove that although this problem is \( \text{NP} \)-hard for trees, it can be solved in polynomial time for all weighted trees, provided that we restrict the search space to subsets which induce connected subgraphs. The proposed algorithm is based on dynamic programming and runs in the worst case in \( O(k^2n^3) \), when \( n \) is the number of vertices and \( k \) is the number of clusters. It also runs in linear time when the number of clusters and the number of outliers is bounded by a constant.

Key words: sparsest cut problem, isoperimetric number, Cheeger constant, normalized cut, graph partitioning, computational complexity, weighted trees.

Subject classification: 05C85, 68Q25, 68R10.

1 Introduction

Data clustering is definitely among the main topics of modern computer science with an indispensable role in data mining, image and signal processing, network and data analysis, and data summarization (e.g. see [13] and references therein). Considering the current status of data science, one may name some fundamental challenges in this field, among many others, as follows:

- Clustering huge and usually high-dimensional data.
- Clustering in presence of outliers and anomalies.
- Clustering non-geometric (usually non-Euclidean) data.
- Clustering with no prior information about the number of clusters or other features of data (as model of the source etc.).
Needless to say, in each case, efficiency and time-complexity of the proposed algorithms are global parameters with a decisive role in applicability. The subject of this article falls into the setup of clustering in an unsupervised and static graph-based data presentation. It is instructive to note that the graph-based approach essentially provides data presentation in a very general (not necessarily Euclidean) setting in terms of similarity kernels. In this respect, one of the main well-studied criteria is the “sparsest cut problem” which apart from tremendous real-world applications in the context of spectral clustering (see e.g. [19, 21]), has played a crucial role in the development of many subjects in theoretical computer science (see e.g. [7, 22]).

Our main objective in this article is to improve this approach, which is essentially based on solving a suitable subpartitioning problem on a corresponding minimum spanning tree, by providing an algorithm that not only gives rise to a fast clustering procedure, but also provides good control on determining the number of clusters and outliers. The procedure is based on a dynamic programming which runs in the worst case in $O(k^2n^3)$, where $n$ is the data size and $k$ is the number of clusters. Also, the algorithm runs in linear time in terms of the data size when the number of clusters $k$ and the upper bound on the number of outliers are both constant (which is the case in the most prevalent applications). To the best of our knowledge, the partitioning problem solved by the proposed algorithm (Algorithm 3) is among the most challenging problems in this literature which is efficiently solvable, while we will also dwell on some important consequences in what follows.

### 1.1 A formal setup and the main result

Partitioning problems are essentially as old as graph theory itself, with wide applications in science and technology. In particular, one may refer to the unnormalized partitioning problems that usually are considered as different versions of minimum cut problems as well as the normalized versions which are more plausible in real applications, however, are much harder to resolve. One of the main problems in the category of normalized cut criteria is the sparsest cut problem which is defined as follows. Given a graph $G$, the sparsest cut problem asks for a cut (a subset of vertices) which has the minimum edge expansion, i.e.

$$ \phi(G) := \min_{S \subseteq V(G), S \neq \emptyset} \max \left\{ \frac{|\partial S|}{|S|}, \frac{|\partial S|}{|S|} \right\}, $$

where $\overline{S} := V(G) \setminus S$ and $\partial S$ is the set of all edges with exactly one end in $S$. The sparsest cut problem is known to be an $NP$–hard problem on general graphs [18, 21]. Efforts to find an efficient algorithm for a good approximation of this problem have triggered off the development of many subfields of computer science and have had a significant influence on algorithm design and complexity theory. It is amazing to see that recent advances in computer science have given rise to a culmination of ideas not only from the classical graph theoretic point of view but also from the more geometric point of view discussed in the theory of Riemannian manifolds and stochastic processes [23]. Up to now, the best known approximation result for the sparsest cut problem is due to Arora, Rao, and Vazirani [3] which gives an $O(\sqrt{\log n})$ approximation algorithm.

It is also worth noting that the invariant defined in (1) has an intimate connection with the second eigenvalue of the associated Laplacian operator. In fact, relaxation of the minimization problem in (1) to the Euclidean norm for real functions (i.e. changing the edge expansion to the Euclidean 2-norm of the gradient of real functions which is the energy representable by the Laplacian operator) gives rise to an eigenvalue problem which is efficiently solvable, while estimating the approximation ratio of this relaxation has led to some fundamental contributions
Given a graph $G = (V, E, \omega, c)$ in which $\omega : V \to \mathbb{Q}^+$ and $c : E \to \mathbb{Q}^+$ are the vertex and edge weight functions, respectively. Note that in the literature close to applications the function $c$ is sometimes referred to as the kernel or the similarity, while from a geometric point of view the graph can also be considered as a discrete metric-measure space, where the distance function is usually chosen to be proportional to some inverse function of $c$. In this setting, by an unweighted graph we mean a graph in which all the vertex and edge weights are equal to 1.

A partition (or a subpartition in particular) is said to be connected each of its parts is a connected subgraph of $G$.

From a geometric point of view, the conductance can be interpreted as a normalized norm of a gradient function or a normalized energy (e.g., see [5, 6] for more on the geometric interpretations).

The set $\mathcal{D}_k(V)$ is defined to be the set of all $k$-subpartitions $\{A_1, \ldots, A_k\} := \{A_i\}_1^k$ of $V$, in which $A_i$’s are nonempty disjoint subsets of $V$. The residue of a subpartition $\{A_1, \ldots, A_k\}$ is defined to be the set $R := V - \bigcup_{i=1}^k A_i$. The set of all $k$-partitions of $V$, which is denoted by $\mathcal{P}_k(V)$, is the subclass of $\mathcal{D}_k(V)$ containing all $k$-subpartitions $\{A_i\}_1^k$ for which $\bigcup_{i=1}^k A_i = V$ (i.e., $R = \emptyset$). A subpartition (or a partition in particular) is said to be connected if the subgraph induced on each of its parts is a connected subgraph of $G$. A generalization of the sparsest cut problem can be formulated as follows.

**Definition 1.** Given a weighted graph $G = (V, E, \omega, c)$ and a positive integer $k$, $1 \leq k \leq |V|$, the $k$th isoperimetric number is defined as,

$$\iota_k(G) := \min_{\{A_i\}_1^k \in \mathcal{P}_k(V)} \max_{1 \leq i \leq k} \phi_G(A_i).$$

Furthermore, considering the partitions, the $k$th minimum normalized cut number is defined as,

$$\tilde{i}_k(G) := \min_{\{A_i\}_1^k \in \mathcal{P}_k(V)} \max_{1 \leq i \leq k} \phi_G(A_i).$$

A vertex $v \in V$ is called a $k$-outlier, if there exists a minimizing subpartition achieving $\iota_k(G)$, while $v$ lies in its residue. It is well-known that $\iota_2 = \iota_2$ (see [8]) and the common value is usually called the Cheeger constant or edge expansion in the literature.

In this regard, Louis et al. in [17] provide a polynomial time approximation algorithm which outputs a $(1 - \epsilon)k$-partition of the vertex set such that each piece has expansion at most $O(\sqrt{\log n \log k})$ times $\tilde{i}_k(G)$ (for every positive number $\epsilon$). Also, in [16], higher-order Cheeger’s
inequalities have been proved which relate the above parameters to the eigenvalues of the associated Laplacian Matrix (see also [8, 10]).

Prior to formulating our problem, let us discuss some facts. First, one may note that as an imprecise rule of thumb, changing the cost function of a partitioning problem, from the normalized form to the unnormalized form, from partitions to subpartitions, or from the mean (i.e. 1-norm) to the max (i.e. ∞-norm) generally makes the problem more tractable in the sense that finding more efficient algorithms to solve the problem become more probable. One of our major observations in this article is the fact that the restriction of the search space to “connected” subpartitions reduces the complexity of the problem too. In particular, this distinction is much comprehensible when the graph is a tree where the restriction on subpartitions to be connected reduces the complexity of the problem from \( NP \)-hard to polynomial time. Also, note that this restriction is to the best of our advantage in the sense that a cluster is more expected to be represented by a connected subgraph than a disconnected one (based on intra-similarity of the objects within a cluster). Hence, as far as clustering is concerned, this can be considered as an acceptable assumption. As a matter of fact, in what follows, we show that such a change to the better will give rise to an efficient algorithm for clustering with a control on the number of parts and outliers.

We denote the main problem, i.e. the multi-way sparsest cut problem with a control on the residue number, by the acronym “MSC problem” which is defined as follows.

**MSC Problem.**

**INSTANCE:** A weighted graph \( G = (V, E, \omega, c) \), nonnegative integers \( \kappa \in \mathbb{Z}^+ \) and \( \lambda \in \mathbb{Z}^+ \) and a positive rational number \( \xi \in \mathbb{Q}^+ \).

**QUERY:** Does there exist a \( \kappa \)-subpartition of \( V \) such as \( \{A_i\}_{1 \leq i \leq \kappa} \in \mathcal{D}_\kappa(V) \) such that \( \max_{1 \leq i \leq \kappa} \{\phi_G(A_i)\} \leq \xi \) and its residue number is at most \( \lambda \), i.e. \( |V \setminus \bigcup_{1 \leq i \leq \kappa} A_i| \leq \lambda \)?

The MSC problem is known to be a hard problem even when the graph is of its simplest form, i.e. a tree. When the graph \( G \) is a tree, it is proved in [9] that MSC problem is \( NP \)-complete even when the tree is unweighted and \( \lambda \) is constant (e.g. \( \lambda = 0 \)). Nonetheless, it is shown there that the problem is solvable in linear time for weighted trees when we drop the restriction on the residue number (i.e. \( \lambda = |V| \)). An improvement of this result has effectively been applied to real clustering problems for large data-sets [11].

The main contribution of this article (Algorithm 3) is to show that although MSC problem is \( NP \)-complete for trees, it becomes tractable when the search space is restricted to connected subpartitions. In other words, the following problem abbreviated by CMSC can be solved in polynomial time for weighted trees.

**CMSC Problem.**

**INSTANCE:** A weighted graph \( G = (V, E, \omega, c) \), nonnegative integers \( \kappa \in \mathbb{Z}^+ \) and \( \lambda \in \mathbb{Z}^+ \) and a positive rational number \( \xi \in \mathbb{Q}^+ \).

**QUERY:** Does there exist a connected \( \kappa \)-subpartition of \( V \) such as \( \{A_i\}_{1 \leq i \leq \kappa} \in \mathcal{D}_\kappa(V) \) such that \( \max_{1 \leq i \leq \kappa} \{\phi_G(A_i)\} \leq \xi \) and its residue number is at most \( \lambda \), i.e. \( |V \setminus \bigcup_{1 \leq i \leq \kappa} A_i| \leq \lambda \)?

This result along with the fact that the minimum spanning tree of a geometric metric-measure space inherits a large part of the geometry of the space, can lead to a good approximation for MSC problem for general graphs. This can justify the importance of the problem on weighted trees when applications are concerned. Let us consider some consequences of this result.

Firstly, note that given a weighted tree \( T \) and integers \( \kappa \) and \( \lambda \), finding the minimum number \( \xi \) for which there exists a connected \( \kappa \)-subpartition with the residue number at most \( \lambda \)
and $\max_{1 \leq i \leq k}\{\phi_G(A_i)\} \leq \xi$ (as well as finding the minimizing subpartition) can be done in polynomial time by applying our algorithm iteratively along with a simple binary search.

Secondly, given a weighted tree $T$ and numbers $\xi, \lambda$ (the worst edge expansion of the clusters), we can obtain a number $k_{\text{max}}(T, \xi)$, denoting the maximum number of parts for which the answer to CMSC problem is positive. This by itself is an important piece of information when one considers the large existing literature discussing how to determine the number of clusters for a clustering algorithm (e.g. see [20] for $k$-means).

Thirdly, from another point of view, CMSC problem can be considered as a problem of outlier-robust clustering where a solution will provide information on the number of outliers. It is well-known that detection of outliers and anomalies in data-sets are among the most challenging problems in the field, not just because of the hardness of the problem itself, but since the concepts themselves are quite fuzzy and depend on many different parameters as scaling or distribution of the source (e.g. see [4,12] for the background). These facts, and in particular, lack of a universal sound and precise definition, is among the first obstacles when one is dealing with these kinds of problems. In [11] some evidence has been discussed that how the data remained in the exterior of the clusters in MSC problem can be justified to be actual outliers in some sense.

Finally, our method can be extended to handle some more general semi-supervised settings where a number of training samples are given by the user which are forced or forbidden to lie in outliers (see Section 4).

The organization of forthcoming sections is as follows. In Section 2, we give required definitions and notations as well as the lemmas which justify our algorithm. In Section 3, we present the main algorithm and explain how it can find the optimal subpartition. We also compute the time complexity of our algorithm. Finally, in Section 4, we discuss some extensions which handle more realistic models.

## 2 Preliminaries

Let $T$ be a rooted tree with root $r$. There is a natural partial order induced through the root on the vertices and edges of $T$ defined as $u \leq v$ for two vertices $u$ and $v$ whenever there is a path $P(r, v, u)$ in $T$ starting from $r$ and ending at $u$ which contains $v$. Similarly, $e \leq e'$ whenever there is a path $P(r, e', e)$ in $T$ starting from $r$ and containing $e$ and $e'$ such that $e'$ is closer than $e$ to $r$ on $P$. In this setting, note that for any $u \neq r$ there exists a unique minimal vertex $v$, with $v \geq u$ and an edge $e_u := uv$, where $v$ and $e_u$ are called the parent vertex and the parent edge of $u$, respectively (and also $u$ is called the child of $v$). Also, for a given edge $e = uv$ with $u \leq v$ we may refer to $e^- = u$ and $e^+ = v$, intermittently. For some technical reasons, we add one new vertex $r'$ to $T$ and connect it to $r$ and define the parent edge of $r$, $e_r$, as the edge $rr'$. Also, we set $\omega(r') = c(e_r) = 0$.

If $F$ is a subset of edges of $T$, then $M(F)$ is the set of maximal elements of $F$ with respect to the natural partial order of $T$. Given a vertex $u$ with the parent edge $e_u$, the subtree $T_u = T_{e_u}$ refers to the subtree induced on the set $\{v \in V(T) : v \leq u\}$. Therefore, $T_r = T_{e_r} = T$.

Let $T = (V,E,\omega,c)$ be a weighted tree and $\xi$ be a fixed positive number. For every integer $0 \leq k \leq |V|$, define $\mathcal{C}_k(T)$ to be the class of all $k$-subpartitions $A = \{A_i\}_1^k$ such that for each $1 \leq i \leq k$, $A_i \subseteq V(T)$ and the induced subgraph of $T$ on $A_i$ is connected (i.e. $A_i$ is a subtree of $T$). Also, given a subpartition $A = \{A_i\}_1^k \in \mathcal{C}_k(T)$, its residue set is defined as $R(A,T) := V(T) \setminus \bigcup_{i=1}^k A_i$. We also define,$$
\phi_T(A_i) := \frac{c(\partial A_i)}{\omega(A_i)}, \quad \phi_T(A) := \max_{1 \leq i \leq k} \phi_T(A_i) \quad \text{and} \quad \iota_k^T(T) := \min_{A \in \mathcal{C}_k(T)} \phi_T(A).
$$

In the following we describe the idea that our algorithm is based on and also prove the correctness
of the algorithm. First, note that since we are looking for subsets with small edge expansion, when we cut an edge \( e \), the subset containing \( e^- \) sustains a loss in its edge expansion. The cause of this deficiency is that the numerator of the edge expansion is added by \( c(e) \) and the denominator is subtracted by \( \omega(T_e) \). With this intuition, for every edge \( e \in E \), define

\[
\varepsilon_\xi(e) := \xi \omega(T_e) + c(e).
\]  

Now, let \( \kappa \) and \( \lambda \) be two nonnegative integers and for every integers \( 1 \leq k \leq \kappa \), \( 0 \leq l \leq \lambda \) and vertex \( u \in V(T) \), define \( C_\xi(u, k, l) \) to be the set of all \( k \)-subpartitions \( A = \{ A_i \}_{i=1}^k \) in \( C_k(T_u) \) such that \( u \in A_1 \) and \( |R(A,T_u)| \leq l \) and for each \( 2 \leq i \leq k \), we have \( \phi_T(A_i) \leq \xi \). For each such subpartition \( A \), let \( F_A := \partial A_1 \setminus \{ e_u \} \). Note that any pair of edges in \( F_A \) are incomparable and define,

\[
\gamma_\xi(A) := \sum_{e \in F_A} \varepsilon_\xi(e).
\]

We will shortly see that minimizing the edge expansion \( \phi_T(A_1) \), in some sense, is equivalent to minimizing \( \gamma_\xi(A) \) (see (6)). Thus, define,

\[
\Gamma_\xi(u, k, l) := \min_{A \in C_\xi(u,k,l)} \gamma_\xi(A).
\]

On the other hand, for every integers \( 0 \leq k \leq \kappa \) and \( 0 \leq l \leq \lambda \) and vertex \( u \in V(T) \), define \( \mu_\xi(u,k,l) \) to be equal to \( 1 \) if there exists a connected \( k \)-subpartition \( A = \{ A_i \}_{i=1}^k \) in \( C_k(T_u) \) such that \( \phi_T(A) \leq \xi \) and \( |R(A,T_u)| \leq l \) and it is equal to \( 0 \), otherwise. Note that, although \( A_i \)'s are subsets of \( V(T_u) \), \( \phi_T(A_i) \) is computed in the whole tree \( T \). Also, note that for every vertex \( u \in V(T) \) and integer \( l \), we have

\[
\mu_\xi(u,0,l) = \begin{cases} 
1 & \text{if } |V(T_u)| \leq l, \\
0 & \text{o.w.}
\end{cases}
\]

In fact, our main goal is to compute the parameter \( \mu_\xi(k,\kappa,\lambda) \), since evidently the answer to CMSC problem is yes if and only if \( \mu_\xi(r,\kappa,\lambda) = 1 \). In the sequel, we are going to show that the parameters \( \Gamma_\xi(u,k,l) \) and \( \mu_\xi(u,k,l) \) can be computed recursively in a breadth-first scanning of vertices towards the root. First, in the following, we explain how one can compute \( \mu_\xi(u,k,l) \) recursively in terms of the values \( \Gamma_\xi(u,k,l) \), \( 1 \leq k \leq \kappa \), \( 0 \leq l \leq \lambda \). For this, let \( \xi, \kappa, \lambda \) be fixed and given a vertex \( u \), let \( (u_1, \ldots, u_d) \) be an ordering of all of its children. Now, for every integers \( 0 \leq k \leq \kappa \), \( 0 \leq l \leq \lambda \), define

\[
U(1,k,l) := \mu_\xi(u_1, k, l),
\]

and for every \( 2 \leq i \leq d \), define

\[
U(i,k,l) := \begin{cases} 
1 & \text{if there exist } 0 \leq k' \leq k \text{ and } 0 \leq l' \leq l - 1 \text{ such that } U(i-1,k',l') = \mu_\xi(u_i, k - k', l - 1 - l') = 1, \\
0 & \text{o.w.}
\end{cases}
\]

In the following lemma, we show how one can use the recursion in (5) to compute the function \( \mu_\xi \).

**Lemma 2.** Let \( u \) be a vertex in a rooted tree \( T \), \( \xi \geq 0 \) be a number and \( \lambda \geq 0, \kappa \geq 1 \) be two integers. Also, let \( u_1, \ldots, u_d \) be the children of \( u \) in \( T \). For every integers \( 0 \leq k \leq \kappa \), \( 0 \leq l \leq \lambda \), \( \mu_\xi(u,k,l) = 1 \) if and only if either \( \Gamma_\xi(u,k,l) \leq \xi \omega(T_u) - c(e_u) \), or \( U(d,k,l) = 1 \).
Proof. Suppose that \( \mu_\xi(u, k, l) = 1 \) and let \( A = \{ A_i \}_{i=1}^k \in \mathcal{C}_k(T_u) \) be a connected subpartition where \( R(A, T_u) \leq l \) and \( \phi_T(A) \leq \xi \). First, assume that \( u \in R(A, T_u) \). Thus, \( A \) itself can be partitioned into \( d \) connected subpartitions \( A_1, \ldots, A_d \) such that \( A_i \in \mathcal{C}_k(T_{u_i}) \), for some integers \( k_i \), where \( k_1 + \cdots + k_d = k \). Also, let \( l_i = |R(A_i, T_{u_i})| \). Therefore, by definition \( \mu_\xi(u_i, k_i, l_i) = 1 \) and \( l_1 + \cdots + l_d = |R(A, T_u) \setminus \{ u \}| \leq l - 1 \). Thus, again by definition \( U(d, k, l) = 1 \). Next, suppose that \( u \notin R(A, T_u) \) and so, without loss of generality, assume that \( u \in A_1 \). Then,

\[
\phi_T(A_1) \leq \xi \iff c(\partial A_1) \leq \xi \omega(A_1) \iff c(e_u) + \sum_{e \in F_A} c(e) \leq \xi \left( \omega(T_u) - \sum_{e \in F_A} \omega(T_e) \right) \leq \gamma_\xi(A) \leq \xi \omega(T_u) - c(e_u).
\]

Therefore, \( \Gamma_\xi(u, k, l) \leq \xi \omega(T_u) - c(e_u) \). This implies that if \( \mu_\xi(u, k, l) = 1 \), then either \( \Gamma_\xi(u, k, l) \leq \xi \omega(T_u) - c(e_u) \), or \( U(d, k, l) = 1 \).

Now, suppose that \( U(d, k, l) = 1 \). Then, there exist integers \( l_1, \ldots, l_d \) and \( k_1, \ldots, k_d \) such that \( \sum_{i=1}^d l_i = l - 1 \), \( \sum_{i=1}^d k_i = k \) and \( \mu_\xi(u_i, k_i, l_i) = 1 \), for all \( 1 \leq i \leq d \). Thus, for each \( 1 \leq i \leq d \), there exists \( A_i \in \mathcal{C}_k(T_{u_i}) \) such that \( R(A_i, T_{u_i}) \leq l_i \) and \( \phi_T(A_i) \leq \xi \). Define \( A = \bigcup_{i=1}^d A_i \). Thus, \( A \in \mathcal{C}_k(T_u) \) and \( R(A, T_u) \leq l \). Hence, \( \mu_\xi(u, k, l) = 1 \).

Finally, suppose that \( \Gamma_\xi(u, k, l) \leq \xi \omega(T_u) - c(e_u) \). Also, let \( A \in \mathcal{C}_k(u, k, l) \) be a minimizer with \( \gamma_\xi(A) = \Gamma_\xi(u, k, l) \). Then, by definition, for every \( 2 \leq i \leq k \), \( \phi_T(A) \leq \xi \) and \( |R(A, T_u)| \leq l \) and by (6), \( \phi_T(A_1) \leq \xi \). Hence, \( \mu_\xi(u, k, l) = 1 \). This completes the proof.}

As we see in Lemma 2, in order to obtain the value of \( \mu_\xi(u, k, l) \), we require to have the value of \( \Gamma_\xi(u, k, l) \). In the next step, we show that, given \( \xi, \kappa, \lambda \), how one may compute \( \Gamma_\xi(u, k, l) \) efficiently for all vertices \( u \) and integers \( 1 \leq k \leq \kappa, 0 \leq l \leq \lambda \). For this, let \( \xi, \kappa, \lambda \) be fixed and given a vertex \( u \), let \( (u_1, \ldots, u_d) \) be an ordering of all of its children. Now, for every integers \( 1 \leq k \leq \kappa, 0 \leq l \leq \lambda \) and \( 1 \leq i \leq d \), define

\[
X_\xi(i, k, l) := \begin{cases} \min\{\varepsilon_\xi(uu_i), \Gamma_\xi(u_i, k, l)\} & \text{if } \mu_\xi(u_i, k - 1, l) = 1, \\ \Gamma_\xi(u_i, k, l) & \text{o.w.} \end{cases}
\]

Also, define

\[
Y_\xi(1, k, l) := X_\xi(1, k, l),
\]

and for every \( 2 \leq i \leq d \), define

\[
Y_\xi(i, k, l) := \min\{Y_\xi(i - 1, k', l'), X_\xi(i, k + 1 - k', l - l') : 1 \leq k' \leq k, 0 \leq l' \leq l\}.
\]

The following lemma shows how to compute the function \( \Gamma_\xi \) using recursion (9).

Lemma 3. Let \( T \) be a rooted tree, \( \xi \geq 0 \) be a number and \( \lambda \geq 0, \kappa \geq 1 \) be two integers. Then, for every vertex \( u \in V(T) \) with \( d \) children \( (u_1, \ldots, u_d) \) and every integers \( 1 \leq k \leq \kappa \) and \( 0 \leq l \leq \lambda \), we have

\[
\Gamma_\xi(u, k, l) = Y_\xi(d, k, l).
\]

Proof. We prove the lemma by induction on the number \( d \). Let \( A = \{ A_i \}_{i=1}^k \in \mathcal{C}_k(u, k, l) \) be a \( k \)-subpartition. First, suppose that \( d = 1 \). If \( uu_1 \in F_A \), then \( \gamma_\xi(A) = \varepsilon_\xi(uu_1) \) and \( A_2, \ldots, A_k \subseteq V(T_{u_1}) \), so \( \mu_\xi(u_1, k - 1, l) = 1 \). Also, if \( uu_1 \notin F_A \), then \( u_1 \in A_1 \) and \( \gamma_\xi(A) \geq \Gamma_\xi(u_1, k, l) \). Therefore, \( \Gamma_\xi(u, k, l) = X_\xi(1, k, l) = Y_\xi(1, k, l) \) as in (7) and (8).

Now, suppose that \( d \geq 2 \). Let \( T' := T \setminus T_{ud} \) and \( T'' := T \setminus (\bigcup_{i=1}^{d-1} T_{u_i}) \) and \( \Gamma'(k, l) \) and \( \Gamma''(k, l) \) be the values of \( \Gamma_\xi(u, k, l) \) for the trees \( T, T' \) and \( T'' \), respectively. Also, let \( l' = |R(A, T) \cap V(T')|, l'' = l - l' \) and let \( k' \) (resp. \( k'' \)) be the number of sets \( A_i \) which intersect
\( V(T') \) (resp. \( V(T''') \)). Then, evidently we have \( k' + k'' = k + 1 \) (note that \( A_1 \) intersects both \( V(T') \) and \( V(T''') \)) and \( \gamma(\mathcal{A}) \geq \Gamma''(k', l') + \Gamma''(k'', l'') \). Therefore,
\[
\Gamma(k, l) = \min\{\Gamma'(k', l') + \Gamma''(k'', l'') : k' + k'' = k + 1, l' + l'' = l\}.
\]

On the other hand, by the induction hypothesis, we have \( \Gamma'(k', l') = Y_\xi(d - 1, k', l') \) and \( \Gamma''(k'', l'') = X_\xi(d, k'', l'') \). Hence, by (9), we have \( \Gamma(k, l) = Y_\xi(d, k, l) \) and we are done. ■

3 The algorithm

In this section, using Lemmas 2 and 3, we provide an algorithm to solve the CMSC problem for all weighted trees. The cores of the algorithm are two dynamic programmings. The final solution to the problem is given in Algorithm 3 which scans the vertices in a BFS order towards the root \( r \) and computes recursively the values of \( \Gamma(u, k, l) \) and \( \mu(u, k, l) \), for \( 1 \leq k \leq \kappa \) and \( 0 \leq l \leq \lambda \). The structure of Algorithm 3 which deploys Algorithms 1 and 2 as two subroutines, is as follows.

First, for all leaves \( u \) (vertices with no children), it computes the values of \( \Gamma(u, k, l) \) and \( \mu(u, k, l) \) (Lines 8-13 in Algorithm 3). Next, for a vertex \( u \), with children \((u_1, \ldots, u_d)\), according to Lemma 3, it applies a dynamic programming (Algorithm 1) based on the recursion given in Equations (8) and (9), to obtain the value of \( \Gamma(u, k, l) \), assuming the values of \( \mu(u_1, k, l) \) and \( \Gamma(u_i, k, l) \) are given. Finally, according to Lemma 2, it applies another dynamic programming (Algorithm 2) based on the recursion given in (5) to obtain the value of \( \mu(u, k, l) \), assuming the values of \( \Gamma(u_i, k, l) \) and \( \mu(u_i, k, l) \) are given. The backtracking ends up outputting the value of \( \mu_r(r, \kappa, \lambda) \) which is equal to 1 if and only if there exists a connected \( \kappa \)-subpartition \( \mathcal{A} \) with \( \phi_T(\mathcal{A}) \leq \xi \) and \( |R(\mathcal{A}, T)| \leq \lambda \). This completes the solution.

3.1 Time complexity

The time complexity of the provided algorithms can be computed as follows. In Algorithm 1, Lines 2-12 can be done in \( O(d(\lambda + 1)\kappa) \). Also, Lines 14-25 can be performed in \( O(d(\lambda + 1)^2\kappa^2) \).

In Algorithm 2, Lines 2-9 run in \( O((\lambda + 1)\kappa) \) and Lines 11-26 run in \( O(d(\lambda + 1)^2\kappa^2) \). Hence, the runtime of Algorithm 3 is in \( O((\lambda + 1)^2\kappa^2n) \). Since in real applications, the values of \( \kappa \) and \( \lambda \) are mostly much smaller than \( n \), we can assume that the algorithm runs in linear time with respect to the number of nodes.

3.2 Constructing the optimal subpartition

Now, we show that during the execution of Algorithm 3, how one can construct a subpartition \( \mathcal{A} \in \mathcal{C}_\xi(T) \) with \( \phi_T(\mathcal{A}) \leq \xi \) and \( |R(\mathcal{A}, T)| \leq \lambda \) (if there exists). Let \( \xi, \kappa \) and \( \lambda \) be fixed and for every vertex \( u \in V(T) \) and \( 1 \leq k \leq \kappa \) and \( 0 \leq l \leq \lambda \), if \( \mu(u, k, l) = 1 \), then let \( \mathcal{A}_u = \mathcal{A}_u(u, k, l) \) be a \( k \)-subpartition in \( \mathcal{C}_k(T_u) \) such that \( \phi_T(\mathcal{A}_u) \leq \xi \) and \( |R(\mathcal{A}_u, T_u)| \leq l \). Also, if \( \mu(u, k, l) = 0 \), let \( \mathcal{A}_u = \mathcal{A}_u(u, k, l) := \emptyset \). Then, the subpartition \( \mathcal{A}_r(r, \kappa, \lambda) \) is what we are looking for. Also, let \( \mathcal{A}_T = \mathcal{A}_T(u, k, l) \) be a subpartition in \( \mathcal{C}_\xi(u, k, l) \) which minimizes (3).

Now, let \( u \) be a vertex with children \((u_1, \ldots, u_d)\). First, according to Algorithm 2 and assuming that we have all the subpartitions \( \mathcal{A}_T(u, k, l) \) and \( \mathcal{A}_u(u_1, k, l) \), we explain how to obtain \( \mathcal{A}_u(u, k, l) \). For this, throughout the execution of Algorithm 2, in Line 5, if \( \Gamma(u, k, l) \leq \xi \omega(T_u) - c(e_u) \), then set \( \mathcal{A}_u := \mathcal{A}_T(u, k, l) \), otherwise set \( \mathcal{A}_u(u, k, l) := \emptyset \). Also, in Line 10, if \( \mu(u_1, k, l) = 1 \), then set \( \mathcal{U}(k, l) := \mathcal{A}_u(u_1, k, l) \) and in Line 18, if \( U(i - 1, k', l') = \mu(u_i, k - k', l - 1 - l') = 1 \), then set \( \mathcal{U}(k, l) := \mathcal{U}(k', l') \cup \mathcal{A}_u(u_i, k - k', l - 1 - l') \). Finally, in Line 27, if \( U(d, k, l) = 1 \), then set \( \mathcal{A}_u := \mathcal{U}(k, l) \).
and assuming that we have all the subpartitions $A_u \subseteq 1 \leq l \leq \lambda$.

Output: The numbers $\Gamma_\xi(u, k, l)$ for all $1 \leq k \leq \kappa$ and $0 \leq l \leq \lambda$.

Algorithm 1

Input: A weighted tree $(T, \omega, c)$, a rational number $\xi$ and integers $\kappa \geq 1$ and $\lambda \geq 0$. A vertex $u \in V(T)$ with children $(u_1, \ldots, u_d)$. The numbers $\mu_\xi(u_i, k, l)$ and $\Gamma_\xi(u_i, k, l)$, for all $1 \leq i \leq d$, $1 \leq k \leq \kappa$ and $0 \leq l \leq \lambda$.

Output: The numbers $\Gamma_\xi(u, k, l)$ for all $1 \leq k \leq \kappa$ and $0 \leq l \leq \lambda$.

1. Set $\varepsilon_i := \xi \omega(T_{u_i}) + c(u_{u_i})$, $\forall 1 \leq i \leq d$;
2. for $i = 1 : d$ do
3.     for $l = 0 : \lambda$ do
4.         for $k = 1 : \kappa$ do
5.             if $\mu_\xi(u_i, k-1, l) = 1$ and $\varepsilon_i \leq \Gamma_\xi(u_i, k, l)$ then
6.                 $X(i, k, l) := \varepsilon_i$;
7.             else
8.                 $X(i, k, l) := \Gamma_\xi(u_i, k, l)$;
9.         end if
10.     end for
11. end for
12. end for
13. Set $Y(1, k, l) := X(1, k, l)$, $\forall 1 \leq k \leq \kappa, 0 \leq l \leq \lambda$;
14. for $i = 2 : d$ do
15.     for $l = 0 : \lambda$ do
16.         for $k = 1 : \kappa$ do
17.             $Y(i, k, l) := +\infty$;
18.         end for
19.     end for
20.     for $k' = 1 : k$ do
21.         $Y(i, k, l) := \min\{Y(i, k, l), Y(i-1, k', l') + X(i, k + 1 - k', l - l')\}$;
22.     end for
23. end for
24. end for
25. end for
26. Set $\Gamma_\xi(u, k, l) := Y(d, k, l)$, $\forall 1 \leq k \leq \kappa, 0 \leq l \leq \lambda$;
27. return $\Gamma_\xi(u, k, l)$, for all $1 \leq k \leq \kappa$ and $0 \leq l \leq \lambda$;

Next, according to Algorithm 1 and assuming that we have all the subpartitions $A_\mu(u_i, k, l)$ and $A_\Gamma(u_i, k, l)$, we explain how to obtain $A_\Gamma(u, k, l)$. First, throughout the execution of Algorithm 1, in Line 5, if $\mu_\xi(u_i, k-1, l) = 1$ and $\varepsilon_i \leq \Gamma_\xi(u_i, k, l)$, then set $X(i, k, l) := A_\mu(u_i, k-1, l) \cup \{\{u\}\}$, otherwise let $X(i, k, l)$ be the subpartition obtained from $A_\Gamma(u_i, k, l)$ by adding the vertex $u$ to the set containing $u_i$. Also, in Line 13, set $Y(k, l) := X(1, k, l)$. Next, in Line 20, if $Y(i-1, k', l') + X(i, k + 1 - k', l - l') \leq Y(i, k, l)$, then let $Y(k, l)$ be obtained from the disjoint union of $Y(k', l')$ and $X(i, k + 1 - k', l - l')$ by merging two sets containing the vertex $u$. Finally, in Line 26, set $A_\Gamma(u, k, l) := Y(k, l)$.

4 Towards more extensions

In this section, we show that our presented scheme can be generalized to solve the following more realizable problems efficiently:

1. Solving CMSC problem on trees with potentials.
Algorithm 2

**Input:** A weighted tree \((T, \omega, c)\), a rational number \(\xi\) and integers \(\kappa \geq 1\) and \(\lambda \geq 0\). A vertex \(u \in V(T)\) with children \((u_1, \ldots, u_d)\). The numbers \(\Gamma_\xi(u, k, l)\), for all \(1 \leq k \leq \kappa\) and \(0 \leq l \leq \lambda\). The numbers \(\mu_\xi(u, k, l)\) for all \(1 \leq i \leq d\), \(1 \leq k \leq \kappa\) and \(0 \leq l \leq \lambda\).

**Output:** The numbers \(\mu_\xi(u, k, l)\) for all \(1 \leq k \leq \kappa\) and \(0 \leq l \leq \lambda\).

1: Let \(e_u\) be the parent edge of \(u\) and \(u_1, \ldots, u_d\) be the children of \(u\).
2: for \(l = 0 : \lambda\) do
  3:   for \(k = 1 : \kappa\) do
  4:     \(\mu_\xi(u, k, l) := 0\);
  5:     if \(\Gamma_\xi(u, k, l) \leq \xi \omega(T_u) - c(e_u)\) then
  6:       \(\mu_\xi(u, k, l) := 1\);
  7:   end if
  8: end for
9: end for
10: Define \(U(1, k, l) := \mu_\xi(u_1, k, l)\), for every \(0 \leq k \leq \kappa, 0 \leq l \leq \lambda\);
11: for \(i = 2 : d\) do
12:   for \(l = 0 : \lambda\) do
13:     for \(k = 0 : \kappa\) do
14:       if \(\mu_\xi(u, k, l) = 0\) then
15:         \(U(i, k, l) := 0\);
16:       end if
17:     end for
18:     for \(k = 0 : \kappa\) do
19:       if \(U(i - 1, k', l') = \mu_\xi(u_i, k - k', l - 1 - l') = 1\) then
20:         \(U(i, k, l) := 1\) and go to Line 22;
21:       end if
22:   end for
23: end if
24: end for
25: end for
26: end for
27: For every \(0 \leq k \leq \kappa\) and \(0 \leq l \leq \lambda\), if \(U(d, k, l) = 1\), then set \(\mu_\xi(u, k, l) := 1\).
28: return \(\mu_\xi(u, k, l)\), for all \(0 \leq l \leq \lambda\);

2. Solving CMSC problem on forests.

3. Solving the following semi-supervised problem: Given a weighted graph \(G = (V, E, \omega, c)\) (not necessarily a forest), two disjoint subsets \(S_1, S_2 \subseteq V\), rational number \(\xi\) and integers \(\kappa, \lambda\), such that the induced subgraph of \(G\) on \(V(G) \setminus S_1\) is a forest. Does there exist a connected subpartition \(A \in \mathcal{D}_k(V)\) such that \(\phi_G(A) \leq \xi\), \(|R(A, G)| \leq \lambda\), \(S_1 \subseteq R(A, G)\) and \(S_2 \cap R(A, G) = \emptyset\)?

In the following, we elaborate on the modifications that should be made to tackle the above settings.

1. In the setting of trees with potentials, each vertex \(v \in V(T)\) is endowed with a potential weight, say \(p(v)\), which is a nonnegative number and the goal is to determine whether there exists a connected \(k\)-subpartition \(A \in \mathcal{D}_k(V)\) such that

\[
\phi_T(A) = \max_{1 \leq i \leq k} \left\{ \phi_T(A_i) = \frac{c(\partial A_i) + p(A_i)}{\omega(A_i)} \right\} \leq \xi
\]
Algorithm 3

Input: A weighted tree \( (T, \omega, e) \), a rational number \( \xi \) and integers \( \kappa \geq 1 \) and \( \lambda \geq 0 \).
Output: Decide if there exists \( A \in \mathcal{G}_k(T) \) where \( \phi_T(A) \leq \xi \) and \( |R(A, T)| \leq \lambda \).

1: Root \( T \) with an arbitrary node \( r \);
2: Order all nodes in BFS order with respect to \( r \), as \( v_1, \ldots, v_n = r \);
3: Set \( i := 1 \);
4: while \( i \leq n \) do
5: \hspace{1em} Let \( u := v_i \) and \( e_u \) be the parent edge of \( u \) and \( u_1, \ldots, u_d \) be the children of \( u \).
6: \hspace{1em} Initialize \( \mu_e(u, k, l) := 0, \forall 0 \leq k \leq \kappa, 0 \leq l \leq \lambda \);
7: \hspace{1em} Initialize \( \Gamma^e(u, k, l) := +\infty, \forall 1 \leq k \leq \kappa, 0 \leq l \leq \lambda \);
8: \hspace{1em} if \( d = 0 \) then
9: \hspace{2em} Set \( \mu_e(u, 0, l) := 1, \forall 1 \leq l \leq \lambda \).
10: \hspace{2em} if \( c(e_u) \leq \xi \omega(u) \) then
11: \hspace{3em} Set \( \mu_e(u, 1, l) := 1, \forall 0 \leq l \leq \lambda \).
12: \hspace{1em} end if
13: \hspace{1em} Set \( \Gamma^e(u, 1, l) := 0, \forall 0 \leq l \leq \lambda \);
14: \hspace{1em} else
15: \hspace{2em} Using Algorithm 1, find the value of \( \Gamma^e(u, k, l) \), for all \( 0 \leq l \leq \lambda \) and \( 1 \leq k \leq \kappa \);
16: \hspace{2em} Using Algorithm 2, find the value of \( \mu_e(u, k, l) \), for all \( 0 \leq l \leq \lambda \) and \( 0 \leq k \leq \kappa \);
17: \hspace{1em} end if
18: \hspace{1em} \( i \leftarrow i + 1 \);
19: end while
20: If \( \mu_e(u, \kappa, \lambda) = 1 \), then return Yes. Otherwise, return No.

and \( |R(A, T)| \leq \lambda \). We can extend our method to solve this problem using Algorithm 3. First, for each edge \( e \in E(T) \), amend the definition of \( \varepsilon_e \) in (2) as follows

\[
\varepsilon_e := \xi \omega(T_e) + c(e) - p(T_e).
\]

Also, define the functions \( \mu_e \) and \( \Gamma^e \) analogously. Next, with a similar argument as in Lemma 2, one may prove that \( \mu_e(u, k, l) = 1 \) if and only if either \( \Gamma^e(u, k, l) \leq \xi \omega(T_u) - c(e_u) - p(T_u) \), or \( U(d, k, l) = 1 \). Moreover, Lemma 3 is still valid. So, we should just change Line 1 in Algorithm 1 and Line 5 in Algorithm 2, accordingly and then Algorithm 3 works for the new setting.

2. Suppose that the forest \( F \) consists of \( c \) disjoint trees \( T_1, \ldots, T_c \) rooted at \( r_1, \ldots, r_c \) respectively. Also, let \( \xi, \kappa, \lambda \) be fixed. First, using Algorithm 3, compute the value of \( \mu_e(r_i, k, l) \), for every integers \( 0 \leq l \leq \lambda \), \( 0 \leq k \leq \kappa \) and \( 1 \leq i \leq c \). Also, define

\[
Z_{\xi}(1, k, l) = \mu_e(r_1, k, l).
\]

The following recursion helps us to solve the problem on \( F \). For every \( 2 \leq i \leq c \), define

\[
Z(i, k, l) := \begin{cases} 
1 & \text{if there exists } 0 \leq k' \leq k \text{ and } 0 \leq l' \leq l, \text{ s.t.} \\
Z(i-1, k', l') = \mu_e(r_i, k-k', l-l') = 1, & \text{o.w.} 
\end{cases}
\]

Then, the solution to CMSC problem is yes if and only if \( Z_{\xi}(c, \kappa, \lambda) = 1 \). Furthermore, One may easily extend this recursion to solve the corresponding problem on forests with potentials.
3. In this setting some vertices should be or should not be in the residue set. The problem can be solved in the following steps:

- First, for each vertex \( v \in V(G) \), define a potential as follows
  \[
p(v) := \sum_{e \in E \setminus \{v\}, S_1} c(e).
  \]

- Now, let \( F \) be a forest obtained from \( G \) by deleting all vertices in \( S_1 \). Also, let \( \lambda' := \lambda - |S_1| \).

- If \( S_2 \) is empty, then the solution can be obtained by performing the method given in 2 on the forest \( F \) with the potential weight \( p \) and the numbers \( \xi, \kappa, \lambda' \). If \( S_2 \) is non-empty, we have to make the following additional modifications to handle the problem.

Suppose that \( T \) is a tree and \( S \subseteq V(T) \) is a subset of vertices. Also, numbers \( \xi, \kappa, \lambda \) are given. We are looking for a connected subpartition \( A \in \mathcal{D}_\kappa(V) \) such that \( \phi_T(A) \leq \xi \), \( |R(A,T)| \leq \lambda \) and \( S \cap R(A,T) = \emptyset \). Note that Lemma 3 is still valid in this setting. However, in the computation of \( \mu_\xi(u,k,l) \), for each \( u \in V(T) \), in Lemma 2, if \( u \in S \), then \( u \) is not allowed to be in the residue set. So, the value of \( \mu_\xi(u,k,l) \) is equal to 1 if and only if \( \Gamma_\xi(u,k,l) \leq \xi \omega(T_u) - c(e_u) \). Thus, with a similar proof as in Lemma 2, we can prove that

\[
\mu_\xi(u,k,l) = \begin{cases} 
1 & \text{if } u \in S \text{ and } \Gamma_\xi(u,k,l) \leq \xi \omega(T_u) - c(e_u), \\
1 & \text{if } u \notin S \text{ and either } \Gamma_\xi(u,k,l) \leq \xi \omega(T_u) - c(e_u) \text{ or } U(d,k,l) = 1, \\
0 & \text{o.w.}
\end{cases}
\]

Then, Algorithm 2 can be modified accordingly to compute the value of \( \mu_\xi(u,k,l) \).

5 Concluding remarks and future work

In this paper, a multi-way sparsest cut problem has been investigated for weighted trees and it was shown that although the problem is \( NP \)-complete for trees, it becomes tractable when the search space is confined to connected subdomains. One of the strengths of the method is that it has a control on the number of outliers and can manage semi-supervised settings when some data points are forced or forbidden to be outlier. Besides the theoretical importance of the sparsest cut problem, when our method is applied to the minimum spanning tree, it can steer several applications in both unsupervised and semi-supervised clustering.

One may also consider an analogous problem when we are seeking for a subpartition minimizing “the average” (instead of the maximum) of the edge expansions of the parts (e.g. as in [21]). This objective function is more sensitive and exquisite and are more likely to produce high-quality clustering results. Nevertheless, the problem unfortunately turns out to be \( NP \)-complete on trees even when the search space is restricted to connected subpartitions (or partitions) [9]. Finding a good approximation algorithm for this problem is an interesting and challenging task that can be the purpose of future work in this line of research.

Acknowledgment. We would like to express our sincere thanks to Amir Daneshgar whose valuable comments were crucial in preparing and improving the present article.
References

[1] N. Alon, *Eigenvalues and expanders*, Combinatorica 6 (1986), no. 2, 83–96. Theory of computing (Singer Island, Fla., 1984). MR 875835

[2] N. Alon and V. D. Milman, *λ₁, isoperimetric inequalities for graphs, and superconcentrators*, J. Combin. Theory Ser. B 38 (1985), no. 1, 73–88. MR 782626

[3] Sanjeev Arora, Satish Rao, and Umesh Vazirani, *Expander flows, geometric embeddings and graph partitioning*, J. ACM 56 (2009), no. 2, Art. 5, 37. MR 2535878

[4] James C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*, Plenum Press, New York-London, 1981, With a foreword by L. A. Zadeh, Advanced Applications in Pattern Recognition. MR 631231

[5] Peter Buser, *Geometry and spectra of compact Riemann surfaces*, Progress in Mathematics, vol. 106, Birkhäuser Boston, Inc., Boston, MA, 1992. MR 1183224

[6] Isaac Chavel, *Eigenvalues in Riemannian geometry*, Pure and Applied Mathematics, vol. 115, Academic Press, Inc., Orlando, FL, 1984. Including a chapter by Burton Randol, With an appendix by Jozef Dodziuk. MR 768584

[7] Fan R. K. Chung, *Spectral graph theory*, CBMS Regional Conference Series in Mathematics, vol. 92, Published for the Conference Board of the Mathematical Sciences, Washington, DC; by the American Mathematical Society, Providence, RI, 1997. MR 1421568

[8] Amir Daneshgar, Hossein Hajiabolhassan, and Ramin Javadi, *On the isoperimetric spectrum of graphs and its approximations*, J. Combin. Theory Ser. B 100 (2010), no. 4, 390–412. MR 2644242

[9] Amir Daneshgar and Ramin Javadi, *On the complexity of isoperimetric problems on trees*, Discrete Applied Mathematics 160 (2012), no. 1, 116–131.

[10] Amir Daneshgar, Ramin Javadi, and Laurent Miclo, *On nodal domains and higher-order Cheeger inequalities of finite reversible Markov processes*, Stochastic Process. Appl. 122 (2012), no. 4, 1748–1776. MR 2914771

[11] Amir Daneshgar, Ramin Javadi, and SB Shariat Razavi, *Clustering and outlier detection using isoperimetric number of trees*, Pattern Recognition 46 (2013), no. 12, 3371–3382.

[12] Rajesh N Davé and Raghuram Krishnapuram, *Robust clustering methods: a unified view*, IEEE Transactions on fuzzy systems 5 (1997), no. 2, 270–293.

[13] Guojun Gan, Chaoqun Ma, and Jianhong Wu, *Data clustering*, ASA-SIAM Series on Statistics and Applied Probability, vol. 20, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; American Statistical Association, Alexandria, VA, 2007, Theory, algorithms, and applications. MR 2331172

[14] Shlomo Hoory, Nathan Linial, and Avi Wigderson, *Expander graphs and their applications*, Bull. Amer. Math. Soc. (N.S.) 43 (2006), no. 4, 439–561. MR 2247919

[15] Mark Jerrum and Alistair Sinclair, *The markov chain monte carlo method: an approach to approximate counting and integration*, Approximation algorithms for NP-hard problems (1996), 482–520.
[16] James R Lee, Shayan Oveis Gharan, and Luca Trevisan, *Multiway spectral partitioning and higher-order cheeger inequalities*, Journal of the ACM (JACM) 61 (2014), no. 6, 37.

[17] Anand Louis and Konstantin Makarychev, *Approximation algorithm for sparsest k-partitioning*, Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, 2014, pp. 1244–1255. MR 3376452

[18] Bojan Mohar, *Isoperimetric numbers of graphs*, J. Combin. Theory Ser. B 47 (1989), no. 3, 274–291. MR 1026065

[19] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al., *On spectral clustering: Analysis and an algorithm*, Advances in neural information processing systems 2 (2002), 849–856.

[20] Dan Pelleg, Andrew W Moore, et al., *X-means: Extending k-means with efficient estimation of the number of clusters*, ICML, vol. 1, 2000, pp. 727–734.

[21] Jianbo Shi and Jitendra Malik, *Normalized cuts and image segmentation*, IEEE Transactions on pattern analysis and machine intelligence 22 (2000), no. 8, 888–905.

[22] Vijay V. Vazirani, *Approximation algorithms*, Springer-Verlag, Berlin, 2001. MR 1851303

[23] Shing Tung Yau, *Isoperimetric constants and the first eigenvalue of a compact Riemannian manifold*, Ann. Sci. École Norm. Sup. (4) 8 (1975), no. 4, 487–507. MR 0397619