ℓ_p-Norm Multiway Cut

Karthekeyan Chandrasekaran
University of Illinois at Urbana-Champaign, IL, USA

Weihang Wang
University of Illinois at Urbana-Champaign, IL, USA

Abstract

We introduce and study ℓ_p-NORM-MULTIWAY-CUT: the input here is an undirected graph with non-negative edge weights along with k terminals and the goal is to find a partition of the vertex set into k parts each containing exactly one terminal so as to minimize the ℓ_p-norm of the cut values of the parts. This is a unified generalization of min-sum multiway cut (when $p = 1$) and min-max multiway cut (when $p = \infty$), both of which are well-studied classic problems in the graph partitioning literature. We show that ℓ_p-NORM-MULTIWAY-CUT is NP-hard for constant number of terminals and is NP-hard in planar graphs. On the algorithmic side, we design an $O(\log^2 n)$-approximation for all $p \geq 1$. We also show an integrality gap of $\Omega(k^{1-1/p})$ for a natural convex program and an $O(k^{1-1/p-\epsilon})$-inapproximability for any constant $\epsilon > 0$ assuming the small set expansion hypothesis.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases multiway cut, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.29

Related Version *Full Version*: https://arxiv.org/abs/2106.14840

Funding *Karthekeyan Chandrasekaran*: Supported in part by NSF grants CCF 1814613 and 1907937.
Weihang Wang: Supported in part by NSF grants CCF 1814613 and 1907937.

1 Introduction

MULTIWAY-CUT is a fundamental problem in combinatorial optimization with both theoretical as well as practical motivations. The input here is an undirected graph $G = (V, E)$ with non-negative edge weights $w : E \to \mathbb{R}_+$ along with k specified terminals $T = \{t_1, t_2, \ldots, t_k\} \subseteq V$. The goal is to find a partition $\mathcal{P} = (P_1, P_2, \ldots, P_k)$ of the vertex set with $t_i \in P_i$ for each $i \in [k]$ so as to minimize the sum of the cut values of the parts, i.e., the objective is to minimize $\sum_{i=1}^{k} w(\delta(P_i))$, where $\delta(P_i)$ denotes the set of edges with exactly one end-vertex in P_i and $w(\delta(P_i)) := \sum_{e \in \delta(P_i)} w(e)$. On the practical side, MULTIWAY-CUT has been used to model file-storage in networks as well as partitioning circuit elements among chips – see [14, 22]. On the theoretical side, MULTIWAY-CUT generalizes the min (s, t)-cut problem which is polynomial-time solvable. In contrast to min (s, t)-cut, MULTIWAY-CUT is NP-hard for $k \geq 3$ terminals [14]. The algorithmic study of MULTIWAY-CUT has led to groundbreaking rounding techniques and integrality gap constructions in the field of approximation algorithms [2, 4–7, 12, 16, 17, 21] and novel graph structural techniques in the field of fixed-parameter algorithms [18]. It is known that MULTIWAY-CUT does not admit a $(1.20016 - \epsilon)$-approximation for any constant $\epsilon > 0$ assuming the Unique Games Conjecture [4] and the currently best known approximation factor is 1.2965 [21].

Motivated by its connections to partitioning and clustering, Svitkina and Tardos [22] introduced a local part-wise min-max objective for MULTIWAY-CUT– we will denote this problem as MIN-MAX-MULTIWAY-CUT: The input here is the same as MULTIWAY-CUT while the goal is to find a partition $\mathcal{P} = (P_1, P_2, \ldots, P_k)$ of the vertex set with $t_i \in P_i$ for each $i \in [k]$ so as to minimize $\max_{i=1}^{k} w(\delta(S))$. We note that MULTIWAY-CUT and MIN-MAX-MULTIWAY-CUT differ only in the objective function – the objective function in MULTIWAY-CUT is to minimize

the sum of the cut values of the parts while the objective function in MIN-MAX-MULTIWAY-CUT is to minimize the max of the cut values of the parts. MIN-MAX-MULTIWAY-CUT can be viewed as a fairness inducing multiway cut as it aims to ensure that no part pays too much in cut value. Svitkina and Tardos showed that MIN-MAX-MULTIWAY-CUT is NP-hard for $k \geq 4$ terminals and also that it admits an $O(\log^3 n)$-approximation. Bansal, Feige, Krauthgamer, Makarychev, Nagarajan, Naor, and Schwartz subsequently improved the approximation factor to $O(\sqrt{\log n \log k})$ (which is $O(\log n)$) [3].

In this work, we study a unified generalization of MULTIWAY-CUT and MIN-MAX-MULTIWAY-CUT that we term as ℓ_p-NORM-MULTIWAY-CUT: In this problem, the input is the same as MULTIWAY-CUT, i.e., we are given an undirected graph $G = (V, E)$ with non-negative edge weights $w : E \to \mathbb{R}_+$ along with k specified terminal vertices $T = \{t_1, t_2, \ldots, t_k\} \subseteq V$. The goal is to find a partition $\mathcal{P} = (P_1, P_2, \ldots, P_k)$ of the vertex set with $t_i \in P_i$ for each $i \in [k]$ so as to minimize the ℓ_p-norm of the cut values of the k parts – formally, we would like to minimize

$$\left(\sum_{i=1}^{k} \left(\sum_{e \in \delta(P_i)} w(e) \right)^p \right)^{\frac{1}{p}}.$$

Throughout, we will consider $p \geq 1$. We note that ℓ_p-NORM-MULTIWAY-CUT for $p = 1$ corresponds to MULTIWAY-CUT and for $p = \infty$ corresponds to MIN-MAX-MULTIWAY-CUT. We emphasize that ℓ_p-NORM-MULTIWAY-CUT could also be viewed as a multiway cut that aims for a stronger notion of fairness than MULTIWAY-CUT but a weaker notion of fairness than MIN-MAX-MULTIWAY-CUT. For $k = 2$ terminals, ℓ_p-NORM-MULTIWAY-CUT reduces to min (s, t)-cut for all $p \geq 1$ and hence, can be solved in polynomial time.

1.1 Our Results

We begin by remarking that there is a fundamental structural difference between MULTIWAY-CUT and ℓ_p-NORM-MULTIWAY-CUT for $p > 1$ (i.e., between $p = 1$ and $p > 1$). The optimal partition to MULTIWAY-CUT satisfies a nice structural property: assuming that the input graph is connected, every part in an optimal partition for MULTIWAY-CUT will induce a connected subgraph. Consequently, MULTIWAY-CUT is also phrased as the problem of deleting a least weight subset of edges so that the resulting graph contains k connected components with exactly one terminal in each component. However, this nice structural property does not hold for ℓ_p-NORM-MULTIWAY-CUT for $p > 1$ as illustrated by the example in Figure 1. We remark that Svitkina and Tardos made a similar observation suggesting that the nice structural property fails for MIN-MAX-MULTIWAY-CUT, i.e., for $p = \infty$ – in contrast, our example in Figure 1 shows that the nice structural property fails to hold for every $p > 1$.

We now discuss our hardness results for ℓ_p-NORM-MULTIWAY-CUT.

▶ **Theorem 1.** *We have the following hardness results for ℓ_p-NORM-MULTIWAY-CUT.*

1. *ℓ_p-NORM-MULTIWAY-CUT is NP-hard for every $p > 1$ and every $k \geq 4$.*
2. *ℓ_p-NORM-MULTIWAY-CUT in planar graphs is NP-hard for every $p > 1$.*

We note that the case of $p = 1$ and $p = \infty$ are already known to be hard: MULTIWAY-CUT is NP-hard for $k = 3$ terminals and is NP-hard in planar graphs when k is arbitrary (i.e., when k is not a fixed constant) [14]; MIN-MAX-MULTIWAY-CUT is NP-hard for $k = 4$ terminals and is NP-hard in trees when k is arbitrary [22]. Our NP-hardness in planar graphs result also requires k to be arbitrary.

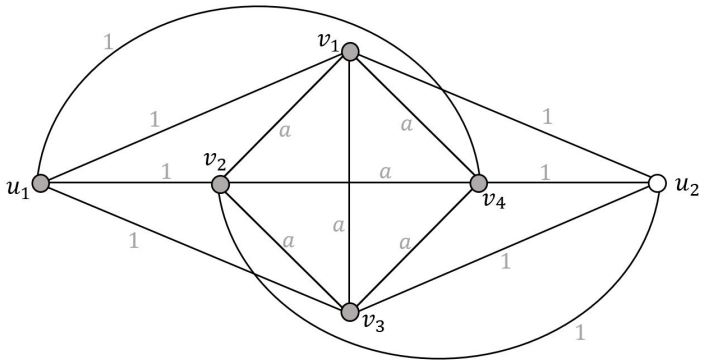

Figure 1 An example where the unique optimum partition for ℓ_p-NORM-MULTIWAY-CUT for $k = 5$ induces a disconnected part for every $p > 1$. The edge weights are as shown with $a := 8^{p/(p-1)}$ and the set of terminals is $\{u_1, v_1, v_2, v_3, v_4\}$. A partition that puts u_2 with one of the terminals in $\{v_1, v_2, v_3, v_4\}$ (and isolates the remaining terminals) has ℓ_p-norm objective value $((3a+3)^p + 3(3a+2)^p + 4^p)^{1/p}$ and the partition that puts u_2 with u_1 (and isolates the remaining terminals) has ℓ_p-norm objective value $(4(3a+2)^p + 8^p)^{1/p}$ – the latter is strictly cheaper by the choice of a.

Given that the problem is NP-hard, we focus on designing approximation algorithms. We show the following result:

▶ **Theorem 2.** *There exists a polynomial-time $O(\log^{1.5} n \log^{0.5} k)$-approximation for ℓ_p-NORM-MULTIWAY-CUT for every $p \geq 1$, where n is the number of vertices and k is the number of terminals in the input instance.*

We note that our approximation factor is $O(\log^2 n)$ since $k \leq n$. While it might be tempting to design an approximation algorithm by solving a convex programming relaxation for ℓ_p-NORM-MULTIWAY-CUT and rounding it, we rule out this approach: the natural convex programming relaxation has an integrality gap of $\Omega(k^{1-1/p})$ – see Section 4. Hence, our approach for the approximation algorithm is not based on a convex program but instead based on combinatorial techniques.

For comparison, we state the currently best known approximation factors for $p = 1$ and $p = \infty$: MULTIWAY-CUT admits a 1.2965-approximation via an LP-based algorithm [21] and MIN-MAX-MULTIWAY-CUT admits an $O(\sqrt{\log n \log k})$-approximation based on a bicriteria approximation for the small-set expansion problem [3].

As a final result, we show that removing the dependence on the number n of vertices in the approximation factor of ℓ_p-NORM-MULTIWAY-CUT is hard assuming the small set expansion hypothesis [20]. In particular, we show that achieving a $(k^{1-1/p-\epsilon})$-approximation for any constant $\epsilon > 0$ is hard. We note that there is a trivial $O(k^{1-1/p})$-approximation for ℓ_p-NORM-MULTIWAY-CUT.

1.2 Outline of techniques

We briefly outline the techniques underlying our results.

Hardness results

We show hardness of ℓ_p-NORM-MULTIWAY-CUT for $k = 4$ terminals by a reduction from the graph bisection problem. Our main tool to control the ℓ_p-norm objective in our hardness reduction is the Mean Value Theorem and its consequences. In order to show NP-hardness of ℓ_p-NORM-MULTIWAY-CUT in planar graphs, we reduce from the 3-partition problem. We do a gadget based reduction where the gadget is planar. We note that the number of terminals in this reduction is not a constant and is $\Omega(n)$, where n is the number of vertices. Once again, we rely on the Mean Value Theorem and its consequences to control the ℓ_p-norm objective in the reduction. We mention that the starting problems in our hardness reductions are inspired by the hardness results shown by Svitkina and Tardos for MIN-MAX-MULTIWAY-CUT: they showed that MIN-MAX-MULTIWAY-CUT is NP-hard for $k = 4$ terminals by a reduction from the graph bisection problem and that MIN-MAX-MULTIWAY-CUT is NP-hard in trees by a reduction from the 3-partition problem. We also use these same starting problems, but our reductions are more involved owing to the ℓ_p-norm nature of the objective.

Approximation algorithm

For the purposes of the algorithm, we will assume knowledge of the optimum value, say OPT – such a value can be guessed within a factor of 2 via binary search. Our approximation algorithm proceeds in three steps. We describe these three steps now.

In the first step of the algorithm, we obtain a collection $\mathcal{S}$ of subsets of the vertex set satisfying four properties: (1) each set S in the collection $\mathcal{S}$ has at most one terminal, (2) the ℓ_p-norm of the cut values of the sets in the collection raised to the pth power is small, i.e., $\sum_{S\in\mathcal{S}} w(\delta(S))^p = (\beta^p \log n)\mathrm{OPT}^p$ where $\beta = O(\sqrt{\log n \log k})$, (3) the number of sets in the collection $\mathcal{S}$ is $O(k \log n)$, and (4) the union of the sets in the collection $\mathcal{S}$ is V. We perform this first step via a multiplicative updates method. For this, we use a bicriteria approximation algorithm for the unbalanced terminal cut problem which was given by Bansal et al [3] (see Section 2 for a description of the unbalanced terminal cut problem and the bicriteria approximation).

Although property (2) gives a bound on the ℓ_p-norm of the cut values of the sets in the collection $\mathcal{S}$ relative to the optimum, the collection $\mathcal{S}$ does not correspond to a feasible multiway cut: recall that a feasible multiway cut is a partition $\mathcal{P} = (P_1, \ldots, P_k)$ of the vertex set where each P_i contains exactly one terminal. The objective of the next two steps is to refine the collection $\mathcal{S}$ to achieve feasibility without blowing up the ℓ_p-norm of the cut values of the parts.

In the second step of the algorithm, we uncross the sets in the collection $\mathcal{S}$ to obtain a partition $\mathcal{Q}$ without increasing the cut values of the sets. We crucially exploit the posimodularity property of the graph cut function to achieve this: posimodularity implies that for all subsets $A, B \subseteq V$ of vertices, either $w(\delta(A)) \geq w(\delta(A - B))$ or $w(\delta(B)) \geq w(\delta(B - A))$. We iteratively consider all pairs of crossing subsets A, B in the collection $\mathcal{S}$ and replace A with $A - B$ if $w(\delta(A)) \geq w(\delta(A - B))$ or replace B with $B - A$ if $w(\delta(B)) \geq w(\delta(B - A))$. The outcome of this step is a partition $\mathcal{Q}$ of the vertex set V satisfying three properties: (i) each part Q in the partition $\mathcal{Q}$ has at most one terminal, (ii) the ℓ_p-norm of the cut values of the parts in the partition $\mathcal{Q}$ raised to the pth power is still small, i.e., $\sum_{Q\in\mathcal{Q}} w(\delta(Q))^p = (\beta^p \log n)\mathrm{OPT}^p$, and (iii) the number of parts in the partition $\mathcal{Q}$ is $O(k \log n)$.

Once again, we observe that the partition $\mathcal{Q}$ at the end of the second step may not correspond to a feasible multiway cut: we could have more than k parts in $\mathcal{Q}$ with some of the parts having no terminals. We address this issue in the third step by a careful aggregation.

For the third step of the algorithm, let Q_i be the part in $\mathcal{Q}$ that contains terminal t_i – we have k such parts by property (i) – and let $R_1, \ldots, R_t$ be the remaining parts in $\mathcal{Q}$ that contain no terminals. We will aggregate the remaining parts of $\mathcal{Q}$ into the k parts $Q_1, \ldots, Q_k$ without blowing up the ℓ_p-norm of the cut value of the parts. By property (iii), the number of remaining parts t is $O(k \log n)$. We create k disjoint buckets $B_1, \ldots, B_k$ where B_i contains the union of $O(\log n)$ many parts among $R_1, \ldots, R_t$. Finally, we merge B_i with Q_i. This results in a partition $\mathcal{P} = (Q_1 \cup B_1, \ldots, Q_k \cup B_k)$ of V with terminal t_i being in the ith part $Q_i \cup B_i$. The key now is to control the blow-up in the pth power of the ℓ_p-norm of the cut values of the parts in $\mathcal{P}$: we bound this by a $O(\log^{p-1} n)$-factor relative to the pth power of the ℓ_p-norm of the cut values of the parts in $\mathcal{Q}$ via Jensen's inequality; while using Jensen's inequality, we exploit the fact that each bucket contained $O(\log n)$ many parts. Consequently, using property (ii), the ℓ_p-norm objective value of the cut values of the parts in the partition $\mathcal{P}$ raised to the pth power is still small – we show that $\sum_{P \in \mathcal{P}} w(\delta(P))^p = \beta^p \log^p n \mathrm{OPT}^p$ and hence, we have an approximation factor of $O(\beta \log n)$.

The first step of our algorithm is inspired by the $O(\log n)$-approximation algorithm for MIN-MAX-MULTIWAY-CUT due to Bansal et al [3] – we modify the multiplicative weights update method and adapt it for ℓ_p-NORM-MULTIWAY-CUT. Our second and third steps differ from that of Bansal et al. We mention that the second and third steps of our algorithm can be adapted to achieve an $O(\beta \log n)$-approximation factor for ℓ_p-NORM-MULTIWAY-CUT for $p = \infty$, but the resulting approximation factor is only $O(\log^2 n)$ which is weaker than the $O(\log n)$-factor achieved by Bansal et al. The additional loss of $\log n$-factor in our algorithm comes from the third step (i.e., the aggregation step). The aggregation step designed in [3] is randomized and saves the $\log n$-factor in expectation, but it does not generalize to ℓ_p-NORM-MULTIWAY-CUT. As mentioned before, the second step of our algorithm relies on posimodularity. The posimodularity property of the graph cut function has been used in previous works for MIN-MAX-MULTIWAY-CUT in an implicit fashion by a careful and somewhat tedious edge counting argument [3, 22]. We circumvent the edge counting argument here by the clean posimodularity abstraction. Moreover, the posimodularity abstraction makes the counting considerably easier for our more general problem of ℓ_p-NORM-MULTIWAY-CUT.

1.3 Related Work

ℓ_p-NORM-MULTIWAY-CUT can be viewed as a fairness inducing objective in the context of multiway partitioning problems. Recent works have proposed and studied various fairness inducing objectives for graph cuts and partitioning that are different from ℓ_p-NORM-MULTIWAY-CUT. We briefly discuss these works here. All of the works mentioned in this subsection consider a more general problem known as *correlation clustering* – we discuss these works by specializing to cut and partitioning problems since these specializations are the ones related to our work.

Puleo and Milenkovic [19] introduced a local vertex-wise min-max objective for min (s,t)-cut – here, the goal is to partition the vertex set V of the given edge-weighted undirected graph into two parts $(S, V \setminus S)$ each containing exactly one of the terminals in $\{s,t\}$ so as to minimize $\max_{v \in V} w(\delta(v) \cap \delta(S))$. The motivation behind this objective is that the cut should be fair to every vertex in the graph, i.e., no vertex should pay a lot for the edges in the cut. A result of Chvátal [13] implies that this problem is $(2-\epsilon)$-inapproximable for every constant $\epsilon > 0$. Charikar, Gupta, and Schwartz [10] gave an $O(\sqrt{n})$-approximation for this problem. Reducing the approximability vs inapproximability gap for this problem remains an intriguing open problem. Kalhan, Makarychev, and Zhou [15] considered an ℓ_p-norm

version of the objective where the goal is to minimize $(\sum_{v\in V} w(\delta(v)\cap\delta(S))^p)^{1/p}$ and gave an $O(n^{\frac{1}{2}-\frac{1}{2p}}\log^{\frac{1}{2}-\frac{1}{2p}} n)$-approximation, thus interpolating the best known results for $p=1$ and $p=\infty$.

Ahmadi, Khuller, and Saha [1] introduced a min-max version of multicut: the input consists of an undirected graph $G=(V,E)$ with edge weights $w: E\to\mathbb{R}_+$ along with source-sink terminal pairs $(s_1,t_1),\ldots,(s_k,t_k)$. The goal is to find a partition $\mathcal{P}=(P_1,\ldots,P_r)$ of the vertex set with all source-sink pairs separated by the partition so as to minimize $\max_{i\in[r]} w(\delta(P_i))$. We emphasize that the number of parts here – namely, r – is not constrained by the input and hence, could be arbitrary. Ahmadi, Khuller, and Saha gave an $O(\sqrt{\log n \max\{\log k, \log T\}})$-approximation for this problem, where T is the number of parts in the optimal solution. Kalhan, Makarychev, and Zhou [15] improved the approximation factor to $2+\epsilon$.

Organization

We begin with preliminaries in Section 2. We present the complete details of our approximation algorithm and prove Theorem 2 in Section 3. We discuss a convex program and its integrality gap in Section 4. Due to space limitations, the proofs of the rest of the results appear in the complete version of this work [9]. We conclude with a few open problems in Section 5.

2 Preliminaries

We start with notations that will be used throughout. Let $G=(V,E)$ be an undirected graph with edge weight function $w: E\to\mathbb{R}_+$ and vertex weight function $y: V\to\mathbb{R}_+$. For every subset $S\subseteq V$, we use $\delta_G(S)$ to denote the set of edges that have exactly one end-vertex in S (we will drop the subscript G when the graph is clear from context), and we write $w(\delta(S)) := \sum_{e\in\delta(S)} w(e)$. Moreover, we will use $y(S)$ to refer to $\sum_{v\in S} y(v)$. We will denote an instance of ℓ_p-NORM-MULTIWAY-CUT by (G,w,T), where $G=(V,E)$ is the input graph, $w: E\to\mathbb{R}_+$ is the edge weight function, and $T\subseteq V$ is the set of terminal vertices. We will call a partition $\tilde{\mathcal{P}}=(P_1,\ldots,P_r)$ of the vertex set to be a multiway cut if $r=k$ and $t_i\in P_i$ for each $i\in[k]$ and denote the ℓ_p-norm of the cut values of the parts (i.e., $(\sum_{i=1}^k w(\delta(P_i))^p)^{1/p}$) as the *$\ell_p$-norm objective value of the multiway cut $\tilde{\mathcal{P}}$*.

We note that the function $\mu:\mathbb{R}\to\mathbb{R}$ defined by $\mu(x):=x^p$ is convex for every $p\geq 1$. We will use Jensen's inequality as stated below in our approximation algorithm as well as our hardness reductions.

▶ **Lemma 3** (Jensen). *Let $\mu:\mathbb{R}\to\mathbb{R}$ be a convex function. For arbitrary $x_1,\ldots,x_t\in\mathbb{R}$, we have*

$$\mu\left(\frac{1}{t}\sum_{i=1}^{t} x_i\right) \leq \frac{1}{t}\sum_{i=1}^{t}\mu(x_i).$$

Our algorithm relies on the graph cut function being symmetric and submodular. We recall that the graph cut function $f: 2^V\to\mathbb{R}_+$ is given by $f(S):=w(\delta(S))$ for all $S\subseteq V$. Let $f: 2^V\to\mathbb{R}_+$ be a set function. The function f is symmetric if $f(S)=f(V\setminus S)$ for all $S\subseteq V$, submodular if $f(A)+f(B)\geq f(A\cap B)+f(A\cup B)$ for all $A,B\subseteq V$, and posimodular if $f(A)+f(B)\geq f(A-B)+f(B-A)$ for all $A,B\subseteq V$. Symmetric submodular functions are also posimodular (see Proposition 4) – this fact has been used implicitly [3, 22] and explicitly [8, 11] before.

▶ **Proposition 4.** *Symmetric submodular functions are posimodular.*

Proof. Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular set function on a set V, and let $A, B \subseteq V$ be two arbitrary subsets. Then, we have

$$\begin{aligned} f(A) + f(B) &= f(V - A) + f(B) \geq f((V - A) \cup B) + f((V - A) \cap B) \\ &= f(V - (A - B)) + f(B - A) = f(A - B) + f(B - A). \end{aligned}$$

In the above, the first and last equations follow by symmetry and the inequality follows by submodularity. ◀

Our algorithm for ℓ_p-NORM-MULTIWAY-CUT relies on an intermediate problem, namely the Unbalanced Terminal Cut problem that we introduce now. In Unbalanced Terminal Cut (UTC), the input (G, w, y, τ, T) consists of an undirected graph $G = (V, E)$, an edge weight function $w : E \to \mathbb{R}_+$, a vertex weight function $y : V \to \mathbb{R}_+$, a real value $\tau \in [0, 1]$, and a set $T \subseteq V$ of terminal vertices. The goal is to compute

$$\text{UTC}(G, w, y, \tau, T) := \min \{w(\delta(S)) : S \subseteq V, y(S) \geq \tau \cdot y(V), |S \cap T| \leq 1\}.$$

Bansal et al. gave a bicriteria approximation for UTC that is summarized in the theorem below.

▶ **Theorem 5.** *[3] There exists an algorithm UTC-BICRIT-ALGO that takes as input (G, w, y, τ, T) consisting of an undirected graph $G = (V, E)$, an edge weight function $w : E \to \mathbb{R}_+$, a vertex weight function $y : V \to \mathbb{R}_+$, a number $\tau \in [0, 1]$, and a set $T \subseteq V$ of terminal vertices and runs in polynomial time to return a set $S \subseteq V$ such that*

1. *$|S \cap T| \leq 1$,*
2. *$y(S) = \Omega(\tau) y(V)$, and*
3. *$w(\delta(S)) \leq \alpha\, UTC(G, w, y, \tau, T)$, where $\alpha = O(\sqrt{\log n \log(1/\tau)})$ and $n = |V|$.*

3 Approximation Algorithm

Let OPT be the optimal ℓ_p-norm objective value of a multiway cut in the given instance. For the purposes of the algorithm, we will assume knowledge of a value D such that $D \geq \text{OPT}^p$ – such a value can be guessed via binary search.

Our approximation algorithm to prove Theorem 2 involves three steps. In the first step of the algorithm, we will obtain a collection $\mathcal{S}$ of $O(k \log n)$ sets whose union is the vertex set V such that each set in the collection has at most one terminal, the cut value of each set is not too large relative to D, and the ℓ_p-norm of the cut values of the sets in the collection is within a polylog(n) factor of D (see Lemma 6). Although the collection $\mathcal{S}$ has low ℓ_p-norm value relative to D, the collection $\mathcal{S}$ may not be a feasible multiway cut. In the second step of the algorithm, we uncross the sets in the collection $\mathcal{S}$ without increasing the ℓ_p-norm of the cut values of the sets in the collection (see Lemma 10). After uncrossing, we obtain a partition, but we could have more than k sets. We address this in our third step, where we aggregate parts to ensure that we obtain exactly k parts (see Lemma 11). We rely on Jensen's inequality to ensure that the aggregation does not blow-up the ℓ_p-norm of the cut values of the sets in the partition.

We begin with the first step of the algorithm in Lemma 6.

▶ **Lemma 6.** *There exists an algorithm that takes as input an undirected graph $G = (V, E)$, an edge weight function $w : E \to \mathbb{R}_+$, k distinct terminal vertices $T := \{t_1, \ldots, t_k\} \subseteq V$ and a value $D > 0$ such that there exists a partition $(P_1^*, \ldots, P_k^*)$ of V with $t_i \in P_i^*$ for all $i \in [k]$ and $\sum_{i=1}^k w(\delta(P_i^*))^p \leq D$, and runs in polynomial time to return a collection of sets $\mathcal{S} \subseteq 2^V$ that satisfies the following:*

1. $|S \cap T| \leq 1$ *and* $w(\delta(S)) \leq \beta(2D)^{1/p}$ *for every* $S \in \mathcal{S}$,
2. $\sum_{S \in \mathcal{S}} w(\delta(S))^p = \beta^p (\log n) D$, *and*
3. $|\mathcal{S}| = O(k \log n)$ *and* $|\{S \in \mathcal{S} : v \in S\}| \geq \log n$ *for each* $v \in V$,

where $\beta = O(\sqrt{\log n \log k})$.

Proof. We will use Algorithm 1 to obtain the desired collection $\mathcal{S}$. We will show the correctness of Algorithm 1 based on Claims 7, 8 and 9.

Algorithm 1 Multiplicative weights update.

Initialize $t \leftarrow 1$, $\mathcal{S} \leftarrow \emptyset$, $y^1(v) = 1$ for each $v \in V$, $Y^1 = \sum_{v \in V} y^1(v)$ and $\beta = O(\sqrt{\log n \log k})$
while $Y^t > \frac{1}{n}$ **do**
  for $i = 1, 2, \ldots, \log(2k)$ **do**
    Execute UTC-BICRIT-ALGO$(G, w, y^t, 2^{-i}, T)$ to obtain a subset $S^t(i) \subseteq V$
    if $w(\delta(S^t(i))) \leq \beta(\frac{4D}{2^i})^{1/p}$ **then**
      Set $S^t = S^t(i)$ and BREAK
    end if
  end for
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{S^t\}$.
  for $v \in V$ **do**
    Set $y^{t+1}(v) = \begin{cases} y^t(v)/2 & \text{if } v \in S^t, \\ y^t(v) & \text{if } v \in V \setminus S^t. \end{cases}$
  end for
  Set $Y^{t+1} = \sum_{v \in V} y^{t+1}(v)$.
  Set $t \leftarrow t + 1$.
end while
Return $\mathcal{S}$

Our first claim will help in showing that the set S^t added in each iteration of the while loop satisfies certain nice properties.

▷ Claim 7. For every iteration t of the while loop of Algorithm 1, there exists $i \in \{1, 2, \ldots, \log(2k)\}$ such that the set $S^t(i)$ satisfies the following conditions:
1. $|S^t(i) \cap T| \leq 1$,
2. $y^t(S^t(i)) = \Omega(\frac{Y^t}{2^i})$, and
3. $w(\delta(S^t(i))) \leq \beta(\frac{4D}{2^i})^{1/p}$.

Proof. We have that $\sum_{i=1}^{k} y^t(P_i^*) = y^t(V)$ and

$$\sum_{i=1}^{k} w(\delta(P_i^*))^p \leq D.$$

Let L be the subset of indices of parts for which the cut value is relatively low:

$$L := \left\{ j \in [k] : w(\delta(P_j^*))^p \leq \frac{2y^t(P_j^*)}{Y^t} \cdot D \right\}.$$

It follows that

$$\sum_{j \in [k] \setminus L} y^t(P_j^*) < \sum_{j \in [k] \setminus L} \frac{w(\delta(P_j^*))^p Y^t}{2D} \leq \frac{Y^t}{2}$$

and hence,

$$\sum_{j\in L} y^t(P_j^*) = Y^t - \sum_{j\in[k]\setminus L} y^t(P_j^*) > Y^t - \frac{Y^t}{2} = \frac{Y^t}{2}.$$

Since $|L| \leq k$, there exists an index $q \in L$ such that $y^t(P_q^*) > Y^t/(2k)$. Let us fix i_0 to be an integer such that $y^t(P_q^*) \in (Y^t \cdot 2^{-i_0}, Y^t \cdot 2^{-i_0+1}]$. Then, we must have $i_0 \leq \log(2k)$. We note that the set P_q^* satisfies $|P_q^* \cap T| = 1$ and $y^t(P_q^*) > Y^t/(2k) = y^t(V)/(2k)$. This implies P_q^* is feasible to the UTC problem on input $(G, w, y^t, 1/2^{i_0}, T)$. Therefore, according to Theorem 5, the set $S^t(i_0)$ has the following properties: Firstly, $|S^t(i_0) \cap T| \leq 1$. Secondly, $y^t(S^t(i_0)) = \Omega(1/2^{i_0})y^t(V) = \Omega(Y^t/2^{i_0})$. Finally,

$$\begin{aligned}
w(\delta(S^t(i_0))) &= O(\sqrt{\log n \log(2k)}) \cdot \mathrm{UTC}\left(G, w, y^t, \frac{1}{2^{i_0}}, T\right) \\
&= O(\sqrt{\log n \log k}) \cdot w(\delta(P_q^*)) \\
&= O(\sqrt{\log n \log k}) \cdot \left(\frac{2y^t(P_q^*)}{Y^t} \cdot D\right)^{\frac{1}{p}} \\
&= O(\sqrt{\log n \log k}) \cdot \left(\frac{2 \cdot Y^t \cdot 2^{-i_0+1}}{Y^t} \cdot D\right)^{\frac{1}{p}} \\
&= O(\sqrt{\log n \log k}) \cdot \left(\frac{4D}{2^{i_0}}\right)^{\frac{1}{p}}.
\end{aligned}$$

This completes the proof of Claim 7. ◁

For the rest of the proof, we will use the following notation: In the t'th iteration of the while loop of Algorithm 1, we will fix $i_t \in \{1, 2, \ldots, \log(2k)\}$ to be the integer such that $S^t = S^t(i_t)$. We will use ℓ to denote the total number of iterations of the while loop. For each $v \in V$, We define $N_v := |\{t \in [\ell] : v \in S^t\}|$ to be the number of sets in the collection $\mathcal{S}$ that contain the vertex v.

We observe that for each $v \in V$, we have $y^{\ell+1}(v) = 2^{-N_v}$. Claim 7 and Theorem 5 together imply that the t'th iteration of the while loop leads to a set S^t being added to the collection $\mathcal{S}$ such that

1. $|S^t \cap T| \leq 1$,
2. $y^t(S^t) = \Omega(\frac{Y^t}{2^{i_t}})$, and
3. $w(\delta(S^t)) \leq \beta(\frac{4D}{2^{i_t}})^{1/p}$.

Our next claim shows that the number of iterations of the while loop executed in Algorithm 1 is small. Moreover, the union of the sets in the collection $\mathcal{S}$ is the vertex set V.

▷ Claim 8. The number of iterations ℓ of the while loop satisfies $\ell = O(k \log n)$. Moreover, $N_v \geq \log n$ for each $v \in V$.

Proof. Upon termination of Algorithm 1, we must have $Y^{\ell+1} \leq 1/n$. Combining with the earlier observation that $y^{\ell+1}(v) = 2^{-N_v}$ for every $v \in V$, we have that

$$2^{-N_v} = y^{\ell+1}(v) \leq Y^{\ell+1} \leq \frac{1}{n},$$

which implies that $N_v \geq \log n$ for every $v \in V$.

It remains to show that $\ell = O(k \log n)$. Consider the tth iteration of the while loop for an arbitrary $t \in [\ell]$. By property 2 of the set S^t stated above, we have that $y^t(S^t) \geq cY^t/2^{i_t} \geq cY^t/(2k)$ for some constant $c > 0$. Consequently,

$$Y^{t+1} = Y^t - \frac{y^t(S^t)}{2} \leq Y^t - \frac{cY^t}{4k} = \left(1 - \frac{c}{4k}\right) Y^t.$$

Due to the termination condition of the while loop, we know that $Y^\ell > 1/n$. Hence,

$$\frac{1}{n} < Y^\ell \leq \left(1 - \frac{c}{4k}\right)^{\ell-1} Y^1 = \left(1 - \frac{c}{4k}\right)^{\ell-1} n \leq \exp\left(-\frac{c(\ell-1)}{4k}\right) n.$$

Therefore, $\frac{c(\ell-1)}{4k} = O(\log n)$ which implies that $\ell = O(k \log n)$. This completes the proof of Claim 8. ◁

The next claim bounds the ℓ_p-norm of the cut values of the sets in the collection $\mathcal{S}$.

▷ Claim 9. The collection $\mathcal{S}$ returned by Algorithm 1 satisfies $\sum_{S \in \mathcal{S}} w(\delta(S))^p = O(\beta^p \log n) \cdot D$.

Proof. Consider the tth iteration of the while loop for an arbitrary $t \in [\ell]$. By property 3 of the set S^t stated above, we have that $w(\delta(S^t)) \leq \beta(4D/2^{i_t})^{1/p}$ and consequently, $2^{i_t} \leq 4D\beta^p \cdot w(\delta(S^t))^{-p}$. Moreover, by property 2 of the set S^t stated above, we have that $y^t(S^t) \geq cY^t/2^{i_t}$ for some constant $c > 0$. Hence,

$$y^t(S^t) \geq \frac{cY^t}{2^{i_t}} \geq \frac{cY^t \cdot w(\delta(S^t))^p}{\beta^p \cdot 4D}.$$

Therefore,

$$Y^{t+1} = Y^t - \frac{y^t(S^t)}{2} \leq \left(1 - \frac{c \cdot w(\delta(S^t))^p}{\beta^p \cdot 8D}\right) Y^t.$$

Using the fact that $Y^\ell > 1/n$, we observe that

$$\begin{aligned}\frac{1}{n} < Y^\ell &\leq Y^1 \cdot \prod_{t=1}^{\ell-1} \left(1 - \frac{c \cdot w(\delta(S^t))^p}{\beta^p \cdot 8D}\right) = n \cdot \prod_{t=1}^{\ell-1} \left(1 - \frac{c \cdot w(\delta(S^t))^p}{\beta^p \cdot 8D}\right) \\ &\leq n \cdot \prod_{i=1}^{\ell-1} \exp\left(-\frac{c \cdot w(\delta(S^t))^p}{\beta^p \cdot 8D}\right) = n \cdot \exp\left(-\frac{c \cdot \sum_{i=1}^{\ell-1} w(\delta(S^t))^p}{\beta^p \cdot 8D}\right).\end{aligned}$$

This implies that $\frac{c \cdot \sum_{i=1}^{\ell-1} w(\delta(S^t))^p}{\beta^p \cdot 8D} = O(\log n)$, and hence $\sum_{i=1}^{\ell-1} w(\delta(S^t))^p = O(\beta^p \log n) \cdot D$.

In the ℓ'th iteration of the while loop, we have $w(\delta(S^\ell)) \leq \beta(4D/2^{i_\ell})^{1/p}$ by property 3 of the set S^t stated above and hence $w(\delta(S^\ell))^p \leq \beta^p \cdot 4D/2^{i_\ell} \leq O(\beta^p D)$. Consequently, $\sum_{i=1}^{\ell} w(\delta(S^t))^p = O(\beta^p \log n) \cdot D$. This completes the proof of Claim 9. ◁

We now show correctness of our algorithm to complete the proof of Lemma 6. Firstly, we note that every $S \in \mathcal{S}$ satisfies $|S \cap T| \leq 1$ by property 1 of the set S^t stated above. Moreover, we have $w(\delta(S)) \leq \beta(4D/2^i)^{1/p} \leq \beta(2D)^{1/p}$, which implies conclusion 1 in Lemma 6. Secondly, Conclusion 2 in Lemma 6 is implied by Claim 9. Finally, conclusion 3 of Lemma 6 is implied by Claim 8 because each iteration of the while loop adds exactly one new set to the collection $\mathcal{S}$.

We now bound the run time of Algorithm 1. Each iteration of the while loop takes polynomial time due to Theorem 5, and the number of iterations of the while loop is $O(k \log n)$. This implies that the total run time of Algorithm 1 is indeed polynomial in the size of the input. ◀

The collection $\mathcal{S}$ that we obtain in Lemma 6 may not be a partition. Our next lemma will uncross the collection $\mathcal{S}$ obtained from Lemma 6 to obtain a partition without increasing the cut values of the sets.

▶ **Lemma 10.** *There exists an algorithm that takes as input a collection $\mathcal{S} \subseteq 2^V$ of subsets of vertices satisfying the conclusions in Lemma 6 and runs in polynomial time to return a partition $\tilde{\mathcal{Q}}$ of V such that*

1. $|Q \cap T| \leq 1$ *for each* $Q \in \tilde{\mathcal{Q}}$,
2. $\sum_{Q \in \tilde{\mathcal{Q}}} w(\delta(Q))^p \leq \sum_{S \in \mathcal{S}} w(\delta(S))^p$, *and*
3. *the number of parts in $\tilde{\mathcal{Q}}$ is $O(k \log n)$.*

Proof. For convenience, we will define $f : 2^V \to \mathbb{R}_+$ by $f(S) := w(\delta(S))$ for all $S \subseteq V$. We will use Algorithm 2 to obtain the desired partition $\tilde{\mathcal{Q}}$ of V.

Algorithm 2 Uncrossing.

```
Initialize Q̃ ← S
while there exist distinct sets A, B ∈ Q̃ such that A ∩ B ≠ ∅ do
  if f(A) ≥ f(A − B) then
    Set A ← A − B
  else
    Set B ← B − A
  end if
end while
Return Q̃
```

We now prove the correctness of Algorithm 2. We begin by observing that Algorithm 2 indeed outputs a partition of the vertex set: Firstly, the while loop enforces that the output $\tilde{\mathcal{Q}}$ satisfies $A \cap B = \emptyset$ for all distinct $A, B \in \tilde{\mathcal{Q}}$. Secondly, during each iteration of the while loop, the set $\bigcup_{Q \in \tilde{\mathcal{Q}}} Q$ remains unchanged: In the iteration of the while loop that uncrosses $A, B \in \tilde{\mathcal{Q}}$, let A' and B' denote the updated sets at the end of the while loop, respectively. Then we must have $A' \cup B' = (A - B) \cup B = A \cup B$ or $A' \cup B' = A \cup (B - A) = A \cup B$. In either case, since $A' \cup B' = A \cup B$, the set $\bigcup_{Q \in \tilde{\mathcal{Q}}} Q$ remains unchanged after the update. Therefore, we have $\bigcup_{Q \in \tilde{\mathcal{Q}}} Q = \bigcup_{S \in \mathcal{S}} S$. We recall that $\bigcup_{S \in \mathcal{S}} S = V$ by conclusion 3 of Lemma 6. Hence, $\tilde{\mathcal{Q}}$ is indeed a partition of V.

Furthermore, each set Q in the output $\tilde{\mathcal{Q}}$ is a subset of some set $S \in \mathcal{S}$. This implies $|Q \cap T| \leq |S \cap T| \leq 1$, thus proving the first conclusion.

To prove the second conclusion, we use posimodularity of f as shown in Proposition 4. Namely, for every $A, B \subseteq V$,

$$f(A) + f(B) \geq f(A - B) + f(B - A).$$

Therefore, at least one of the following two hold: either $f(A) \geq f(A-B)$ or $f(B) \geq f(B-A)$. This implies that, by the choice of the algorithm, $\sum_{Q \in \tilde{\mathcal{Q}}} f(Q)^p$ does not increase.

To see the third conclusion, we note that after each iteration of the while loop, the size of $\tilde{\mathcal{Q}}$ is unchanged. Therefore, at the end Algorithm 2, we have $|\tilde{\mathcal{Q}}| = |\mathcal{S}| = O(k \log n)$ by Lemma 6.

Finally, we bound the run time as follows. At initialization, there are $O((k \log n)^2)$ pairs $(A, B) \in \tilde{\mathcal{Q}}^2$ such that $A \cap B \neq \emptyset$. After each iteration of the while loop, the number of such pairs decreases by at least 1. Therefore, the total number of iterations of the while loop is $O((k \log n)^2)$. Hence, Algorithm 2 indeed runs in polynomial time. ◀

The partition $\tilde{\mathcal{Q}}$ that we obtain in Lemma 10 may contain more than k parts and hence, some of the parts may not contain any terminals. Our next lemma will aggregate the parts in $\tilde{\mathcal{Q}}$ from Lemma 10 to obtain a k-partition that contains exactly one terminal in each part while controlling the increase in the ℓ_p-norm of the cut value of the parts.

▶ **Lemma 11.** *There exists an algorithm that takes as input a partition $\tilde{\mathcal{Q}}$ of V satisfying the conclusions in Lemma 10 and runs in polynomial time to return a partition $(P_1, P_2, \ldots, P_k)$ of V such that*

1. *$t_i \in P_i$ for each $i \in [k]$, and*
2. *$\sum_{i=1}^{k} w(\delta(P_i))^p = O((\beta \log n)^p) \cdot D$.*

Proof. We will use Algorithm 3 on input $\tilde{\mathcal{P}}$ to obtain the desired partition.

Algorithm 3 Aggregating.

Let $\mathcal{F} = \{Q \in \tilde{\mathcal{Q}} : Q \cap T = \emptyset\}$.
Let $\mathcal{P}' = \{Q \in \tilde{\mathcal{Q}} : Q \cap T \neq \emptyset\} = \{Q'_1, \ldots, Q'_k\}$, where $t_i \in Q'_i$ for each $i \in [k]$.
Partition the sets in $\mathcal{F}$ into k buckets $B_1, \ldots, B_k$ such that $|B_i| = O(\log n)$ for each $i \in [k]$ (arbitrarily).
for $i = 1, 2, \ldots, k$ **do**
  Set $P_i \leftarrow Q'_i \cup \left(\bigcup_{A \in B_i} A\right)$
end for
Return $(P_1, \ldots, P_k)$.

The run time of Algorithm 3 is linear in its input size. We now argue the correctness. We note that the third step in Algorithm 3 is possible because $|\mathcal{F}| \leq |\tilde{\mathcal{Q}}| = O(k \log n)$.

Since $|Q \cap T| \leq 1$ for each $Q \in \tilde{\mathcal{Q}}$, the tuple $(P_1, \ldots, P_k)$ returned by Algorithm 3 is indeed a partition of V satisfying $t_i \in P_i$ for all $i \in [k]$. We will now bound $\sum_{i=1}^{k} f(P_i)^p$, where $f : 2^V \to \mathbb{R}_+$ is given by $f(S) := w(\delta(S))$ for all $S \subseteq V$. We have that

$$\sum_{i=1}^{k} f(P_i)^p = \sum_{i=1}^{k} f\left(Q'_i \cup \left(\bigcup_{A \in B_i} A\right)\right)^p \leq \sum_{i=1}^{k} \left(f(Q'_i) + \sum_{A \in B_i} f(A)\right)^p .$$

Since the number of sets in B_i is $O(\log n)$, we have the following using Jensen's inequality (Lemma 3) for each $i \in [k]$:

$$\begin{aligned}\left(f(Q'_i) + \sum_{A \in B_i} f(A)\right)^p &\leq (|B_i| + 1)^{p-1} \left(f(Q'_i)^p + \sum_{A \in B_i} f(A)^p\right) \\ &= O(\log^{p-1} n) \left(f(Q'_i)^p + \sum_{A \in B_i} f(A)^p\right).\end{aligned}$$

Hence,

$$\begin{aligned}\sum_{i=1}^{k} f(P_i)^p &= \sum_{i=1}^{k} O(\log^{p-1} n) \left(f(Q'_i)^p + \sum_{A \in B_i} f(A)^p\right) = O(\log^{p-1} n) \sum_{Q \in \tilde{\mathcal{Q}}} f(Q)^p \\ &= O(\log^{p-1} n) \sum_{S \in \mathcal{S}} f(S)^p = \beta^p O(\log^p n) D.\end{aligned}$$

The last but one equality above is due to conclusion 2 of Lemma 10bbb and the last equality is due to conclusion 2 of Lemma 6. Hence, $\sum_{i=1}^{k} w(\delta(P_i))^p = \sum_{i=1}^{k} f(P_i)^p = O((\beta \log n)^p) D$. ◀

Lemmas 6, 10, and 11 together lead to an algorithm that takes as input an undirected graph $G = (V, E)$, an edge weight function $w : E \to \mathbb{R}_+$, k distinct terminal vertices $T := \{t_1, \ldots, t_k\} \subseteq V$, and a value $D > 0$ such that there exists a partition $(P_1^*, \ldots, P_k^*)$ of V with $t_i \in P_i^*$ for all $i \in [k]$ such that $\sum_{i=1}^k w(\delta(P_i^*))^p \leq D$, and runs in polynomial time to return a multiway cut $\mathcal{P} = (P_1, \ldots, P_k)$ such that

$$\left(\sum_{i=1}^k w(\delta(P_i))^p\right)^{\frac{1}{p}} = (O((\beta \log n)^p)D)^{\frac{1}{p}} = O(\beta \log n)D^{\frac{1}{p}} = O(\log^{1.5} n \log^{0.5} k)D^{\frac{1}{p}}.$$

In order to prove Theorem 2, we may use binary search to guess $D \in [\mathrm{OPT}^p, (2\mathrm{OPT})^p]$ and run the above algorithm to obtain a multiway cut $\mathcal{P} = (P_1, \ldots, P_k)$ such that

$$\left(\sum_{i=1}^k w(\delta(P_i))^p\right)^{\frac{1}{p}} = O(\log^{1.5} n \log^{0.5} k)D^{\frac{1}{p}} = O(\log^{1.5} n \log^{0.5} k)\mathrm{OPT}.$$

This completes the proof of Theorem 2.

4 Convex program and integrality gap

The following is a natural convex programming relaxation for ℓ_p-NORM-MULTIWAY-CUT on instance (G, w, T) where $T = \{t_1, \ldots, t_k\}$ are the terminal vertices (the objective function can be convexified by introducing additional variables and constraints):

$$\text{Minimize } \left(\sum_{i=1}^k \left(\sum_{uv \in E} w(uv) \cdot |x(u,i) - x(v,i)|\right)^p\right)^{1/p} \text{ subject to} \tag{1}$$

$$\begin{aligned} \sum_{i=1}^k x(v,i) &= 1 \quad \forall v \in V, \\ x(t_i, i) &= 1 \quad \forall i \in [k], \\ x(v,i) &\geq 0 \quad \forall v \in V,\ \forall i \in [k]. \end{aligned}$$

▶ **Lemma 12.** *The convex program in* (1) *has an integrality gap of at least* $k^{1-1/p}/2$.

Proof. Consider the star graph that has k leaves $\{t_1, \ldots, t_k\}$ and a center vertex v with all edge weights being 1. Let the terminal vertices be the k leaves. The optimum ℓ_p-norm objective value of a multiway cut is

$$((k-1)^p + k - 1)^{\frac{1}{p}},$$

and it corresponds to the partition $(\{t_1, v\}, \{t_2\}, \{t_3\}, \ldots, \{t_k\})$. A feasible solution to the convex program (1) is given by $x(v,i) = 1/k$ for all $i \in [k]$, which yields an objective of

$$\left(k \cdot \left(\frac{k-1}{k} + (k-1) \cdot \frac{1}{k}\right)^p\right)^{\frac{1}{p}} = \frac{2k-2}{k} \cdot k^{\frac{1}{p}}.$$

This results in an integrality gap of at least

$$\frac{((k-1)^p + k - 1)^{\frac{1}{p}}}{\frac{2k-2}{k} \cdot k^{\frac{1}{p}}} \geq \frac{k-1}{\frac{2k-2}{k} \cdot k^{\frac{1}{p}}} = \frac{k^{1-\frac{1}{p}}}{2}.$$ ◀

Bansal et al. give an SDP relaxation for MIN-MAX-MULTIWAY-CUT and show that the star graph has an integrality gap of $\Omega(k)$ for this SDP relaxation. This SDP relaxation can be generalized in a natural fashion to ℓ_p-NORM-MULTIWAY-CUT. The star graph still exhibits an integrality gap of $\Omega(k^{1-1/p})$ for the generalized SDP relaxation for ℓ_p-NORM-MULTIWAY-CUT.

5 Conclusion

In this work, we introduced ℓ_p-NORM-MULTIWAY-CUT for $p \geq 1$ as a unified generalization of MULTIWAY-CUT and MIN-MAX-MULTIWAY-CUT. We showed that ℓ_p-NORM-MULTIWAY-CUT is NP-hard for constant number of terminals or in planar graphs for every $p \geq 1$. The natural convex program for ℓ_p-NORM-MULTIWAY-CUT has an integrality gap of $\Omega(k^{1-1/p})$ and the problem is $(k^{1-1/p-\epsilon})$-inapproximable for any constant $\epsilon > 0$ assuming the small set expansion hypothesis, where k is the number of terminals in the input instance. The inapproximability result suggests that a dependence on n in the approximation factor is unavoidable if we would like to obtain an approximation factor that is better than the trivial $O(k^{1-1/p})$-factor. On the algorithmic side, we gave an $O(\sqrt{\log^3 n \log k})$-approximation (i.e., an $O(\log^2 n)$-approximation), where n is the number of vertices in the input graph. Our results suggest that the approximability behaviour of ℓ_p-NORM-MULTIWAY-CUT exhibits a sharp transition from $p = 1$ to $p > 1$. Our work raises several open questions. We mention a couple of them: (1) Can we achieve an $O(\log n)$-approximation for ℓ_p-NORM-MULTIWAY-CUT for every $p \geq 1$? We recall that when $p = \infty$, the current best approximation factor is indeed $O(\log n)$ [3]. (2) Is there a polynomial-time algorithm for ℓ_p-NORM-MULTIWAY-CUT for any given p that achieves an approximation factor that smoothly interpolates between the best possible approximation for $p = 1$ and the best possible approximation for $p = \infty$ – e.g., is there an $O(\log^{1-1/p} n)$-approximation?

References

1. S. Ahmadi, S. Khuller, and B. Saha. Min-max correlation clustering via multicut. In *Integer Programming and Combinatorial Optimization*, IPCO, pages 13–26, 2019.
2. H. Angelidakis, Y. Makarychev, and P. Manurangsi. An improved integrality gap for the Călinescu-Karloff-Rabani relaxation for multiway cut. In *Integer Programming and Combinatorial Optimization*, IPCO, pages 39–50, 2017.
3. N. Bansal, U. Feige, R. Krauthgamer, K. Makarychev, V. Nagarajan, J. Naor, and R. Schwartz. Min-max graph partitioning and small set expansion. *SIAM Journal on Computing*, 43(2):872–904, 2014.
4. K. Bérczi, K. Chandrasekaran, T. Király, and V. Madan. Improving the integrality gap for multiway cut. *Mathematical Programming*, 183:171–193, 2020.
5. N. Buchbinder, J. Naor, and R. Schwartz. Simplex partitioning via exponential clocks and the multiway cut problem. In *Proceedings of the forty-fifth annual ACM Symposium on Theory of Computing*, STOC, pages 535–544, 2013.
6. N. Buchbinder, R. Schwartz, and B. Weizman. Simplex transformations and the multiway cut problem. In *Proceedings of the twenty-eighth annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 2400–2410, 2017.
7. G. Călinescu, H. Karloff, and Y. Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564–574, 2000.
8. K. Chandrasekaran and C. Chekuri. Min-max partitioning of hypergraphs and symmetric submodular functions. In *Proceedings of the thirty-second annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 1026–1038, 2021.
9. K. Chandrasekaran and W. Wang. ℓ_p-norm Multiway Cut. Preprint in arXiv: 2106.14840, 2021.
10. M. Charikar, N. Gupta, and R. Schwartz. Local guarantees in graph cuts and clustering. In *Integer Programming and Combinatorial Optimization*, IPCO, pages 136–147, 2017.
11. C. Chekuri and A. Ene. Submodular cost allocation problem and applications. In *International Colloquium on Automata, Languages and Programming*, ICALP, pages 354–366, 2011.

12 K. Cheung, W. Cunningham, and L. Tang. Optimal 3-terminal cuts and linear programming. *Mathematical Programming*, 106(1):1–23, March 2006.

13 V. Chvátal. Recognizing decomposable graphs. *Journal of Graph Theory*, 8:51–53, 1984.

14 E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

15 S. Kalhan, K. Makarychev, and T. Zhou. Correlation clustering with local objectives. In *Advances in Neural Information Processing Systems 32*, pages 9346–9355, 2019.

16 D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research*, 29(3):436–461, 2004.

17 R. Manokaran, J. Naor, P. Raghavendra, and R. Schwartz. SDP gaps and UGC hardness for multiway cut, 0-extension, and metric labeling. In *Proceedings of the fortieth annual ACM Symposium on Theory of Computing*, STOC, pages 11–20, 2008.

18 D. Marx. Parameterized graph separation problems. *Theoretical Computer Science*, 351(3):394–406, 2006.

19 G. Puleo and O. Milenkovic. Correlation clustering and biclustering with locally bounded errors. *IEEE Transactions on Information Theory*, 64:4105–4119, 2018.

20 P. Raghavendra, D. Steurer, and M. Tulsiani. Reductions between expansion problems. In *IEEE Conference on Computational Complexity*, CCC, pages 64–73, 2012.

21 A. Sharma and J. Vondrák. Multiway cut, pairwise realizable distributions, and descending thresholds. In *Proceedings of the forty-sixth annual ACM Symposium on Theory of Computing*, STOC, pages 724–733, 2014.

22 Z. Svitkina and É. Tardos. Min-max multiway cut. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, APPROX, pages 207–218, 2004.