Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Hypergraph k-Cut for Fixed k in Deterministic Polynomial Time

Karthekeyan Chandrasekaran, Chandra Chekuri

informs
http://pubsonline.informs.org/journal/moor
MATHEMATICS OF OPERATIONS RESEARCH
Articles in Advance, pp. 1–20
ISSN 0364-765X (print), ISSN 1526-5471 (online)

Hypergraph *k*-Cut for Fixed *k* in Deterministic Polynomial Time

Karthekeyan Chandrasekaran,[a,*] Chandra Chekuri[a]

[a] University of Illinois Urbana-Champaign, Champaign, Illinois 61820
*Corresponding author
Contact: karthe@illinois.edu, https://orcid.org/0000-0002-3421-7238 (KC); chekuri@illinois.edu (CC)

Received: December 13, 2020
Revised: August 12, 2021
Accepted: November 2, 2021
Published Online in Articles in Advance: March 22, 2022

MSC2020 Subject Classification: 68R10

https://doi.org/10.1287/moor.2021.1250

Abstract. We consider the Hypergraph-k-Cut problem. The input consists of a hypergraph $G = (V, E)$ with nonnegative hyperedge-costs $c : E \rightarrow \mathbb{R}_+$ and a positive integer k. The objective is to find a minimum cost subset $F \subseteq E$ such that the number of connected components in $G - F$ is at least k. An alternative formulation of the objective is to find a partition of V into k nonempty sets $V_1, V_2, \ldots, V_k$ so as to minimize the cost of the hyperedges that cross the partition. Graph-k-Cut, the special case of Hypergraph-k-Cut obtained by restricting to graph inputs, has received considerable attention. Several different approaches lead to a polynomial-time algorithm for Graph-k-Cut when k is fixed, starting with the work of Goldschmidt and Hochbaum (Math of OR, 1994). In contrast, it is only recently that a randomized polynomial time algorithm for Hypergraph-k-Cut was developed (Chandrasekaran, Xu, Yu, Math Programming, 2019) via a subtle generalization of Karger's random contraction approach for graphs. In this work, we develop the first deterministic algorithm for Hypergraph-k-Cut that runs in polynomial time for any fixed k. We describe two algorithms both of which are based on a divide and conquer approach. The first algorithm is simpler and runs in $n^{O(k^2)}m$ time while the second one runs in $n^{O(k)}m$ time, where n is the number of vertices and m is the number of hyperedges in the input hypergraph. Our proof relies on new structural results that allow for efficient recovery of the parts of an optimum k-partition by solving minimum (S, T)-terminal cuts. Our techniques give new insights even for Graph-k-Cut.

Funding: This work was supported in part by the National Science Foundation [Grant CCF-1907937].

Keywords: hypergraphs • k-cut • algorithms

1. Introduction

A hypergraph $G = (V, E)$ consists of a finite set V of vertices and a finite set E of hyperedges where each $e \in E$ is a subset of V. In this work, we consider the Hypergraph-k-Cut problem, in particular when k is a fixed constant. The input to this problem consists of a hypergraph $G = (V, E)$ with nonnegative hyperedge costs $c : E \rightarrow \mathbb{R}_+$ and a positive integer k. The objective is to find a minimum-cost subset of hyperedges whose removal results in at least k connected components. An equivalent partitioning formulation turns out to be quite important. In this formulation, the objective is to find a partition of V into k nonempty sets $V_1, V_2, \ldots, V_k$ so as to minimize the cost of the hyperedges that cross the partition. A hyperedge $e \in E$ crosses a partition $(V_1, V_2, \ldots, V_k)$ if it has vertices in more than two parts; that is, there exist distinct $i, j \in [k]$ such that $e \cap V_i \neq \emptyset$ and $e \cap V_j \neq \emptyset$.

Cut and partitioning problems in graphs, hypergraphs, and related structures, including submodular functions, are extensively studied in algorithms and combinatorial optimization literature for their theoretical importance and numerous applications. Hypergraph-k-Cut is of interest for its applications and simplicity and also because of its close connections to a special case, namely in graphs, and to a generalization, namely in submodular functions. For this reason, the complexity of Hypergraph-k-Cut has been an intriguing open problem for several years, with some important recent progress. First, we describe these closely related problems and some prior work on them.

Graph-k-Cut: This is a special case of Hypergraph-k-Cut where the input is a graph instead of a hypergraph. When $k = 2$, Graph-k-Cut is the global minimum cut problem (Graph-MinCut), which is a fundamental and well-known problem. It is easy to see that Graph-MinCut can be solved in polynomial time via reduction to min s-t cuts; however, there is more structure in Graph-MinCut, and this can be exploited to obtain faster deterministic and randomized algorithms [23, 24, 33, 40]. The complexity of Graph-k-Cut for $k \geq 3$ has also been extensively investigated with substantial recent work. Goldschmidt and Hochbaum [16, 17] showed that Graph-k-Cut is NP-Hard when k is part of the input and that it is polynomial-time solvable when k is any fixed constant (this is not obvious even for $k = 3$). They used a divide-and-conquer approach for Graph-k-Cut that resulted in an algorithm with a running time of $n^{O(k^2)}$. We will describe the technical aspects of this approach in more detail later. This approach has been refined over several papers, culminating in an algorithm of Kamidoi et al. [22] that ran in $n^{(4+o(1))k}$ time. Two very different approaches also give polynomial-time algorithms for fixed k. The random contraction approach of Karger [23], via the improvement in Karger and Stein's [24] work, led to a Monte Carlo randomized algorithm with a running time of $\tilde{O}(n^{2k-2})$. Very recently, Gupta and colleagues [20, 21] showed that the Karger-Stein algorithm in fact runs in $O(n^k)$ time; $n^{(1-o(1))k}$ appears to be lower bound on the run-time via a reduction from the problem of finding a maximum-weight clique of size k (see [28]). Another approach is via tree packings, which was introduced by Karger for Graph-MinCut. Thorup [41] showed that tree packings can also be used to obtain a polynomial-time algorithm for Graph-k-Cut. Thorup's [41] algorithm is deterministic and runs in $n^{2k+O(1)}$ time and was clarified in [9] via an LP relaxation, and this also resulted in a slight improvement in the runtime and currently yields the fastest deterministic algorithm. We defer discussion of polynomial-time approximation algorithms for Graph-k-Cut, when k is part of the input, to the related work section.

Submodular Partition Problems: Graph and hypergraph cut functions are submodular, and one can view Graph-k-Cut and Hypergraph-k-Cut as special cases of a more general problem called Submodular-k-Partition (abbreviated to Submod-k-Part) that we define now. We recall that a real-valued set function $f: 2^V \to \mathbb{R}$ is submodular iff $f(A \cap B) + f(A \cup B) \le f(A) + f(B)$ for all $A, B \subseteq V$. Zhao et al. [43] defined Submod-k-Part as follows; given f specified via a value oracle and a positive integer k, partition V into nonempty sets $V_1, V_2, \ldots, V_k$ so as to minimize $\sum_{i=1}^{k} f(V_i)$. A special case of Submod-k-Part is Sym-Submod-k-Part when f is symmetric (that is $f(A) = f(V \setminus A)$ for all $A \subseteq V$). It is not hard to see that Graph-k-Cut is a special case of Sym-Submod-k-Part. However, Hypergraph-k-Cut is not a special case of Sym-Submod-k-Part, even though the hypergraph cut function is itself symmetric.[1] As observed in [34], one can reduce Hypergraph-k-Cut to Submod-k-Part. Submod-k-Part and Sym-Submod-k-Part are very general problems. For $k = 2$, they can be solved in polynomial-time via submodular function minimization. It is a very interesting open problem to decide whether they admit polynomial-time algorithms for all fixed k. Okumoto et al. [34] showed that Submod-k-Part is polynomial-time solvable for $k = 3$. They generalized the work of Xiao [42], who showed that Hypergraph-k-Cut is polynomial-time solvable for $k = 3$. Queyranne [36] claimed, in 1999, a polynomial-time algorithm for Sym-Submod-k-Part when k is fixed; however, the claim was retracted subsequently. This is reported in [18], where it is also shown that Sym-Submod-k-Part has a polynomial-time algorithm for $k \le 4$.

Multiterminal variants: We also mention that Graph-k-Cut, Hypergraph-k-Cut, and Submod-k-Part have natural variants involving separating specified terminal vertices $s_1, s_2, \ldots, s_k$. These versions are NP-hard for $k \ge 3$. We discuss approximation algorithms for these problems in the related work section.

Hypergraph-*k*-Cut and main result: The complexity of Hypergraph-k-Cut for fixed k has been open since the work of Goldschmidt and Hochbaum [16] for graphs (1988). For $k = 2$, this is the Hypergraph-MinCut problem and can be solved via reduction to min s-t cuts in directed graphs [27] or via other approaches that take advantage of the submodularity structure of the hypergraph cut function (see [8] and references therein). For $k \ge 3$ and bounded rank hypergraphs, Fukunaga [14] generalized Thorup's [41] tree-packing approach to solve Hypergraph-k-Cut for fixed k — the run-time depends exponentially in the rank (rank is the maximum cardinality of a hyperedge in the input hypergraph). It was also observed that Karger's random contraction approach for graphs easily extends to give a randomized algorithm for bounded rank hypergraphs. As we noted earlier, Xiao [42] obtained a polynomial-time algorithm for Hypergraph-k-Cut when $k = 3$. In fairly recent work, Chandrasekaran et al. [5] obtained the first randomized polynomial-time algorithm for Hypergraph-k-Cut for any fixed k; their Monte Carlo algorithm runs in $\tilde{O}(pn^{2k-1})$ time, where $p = \sum_{e \in E} |e|$ is the representation size of the input hypergraph. Subsequently, Fox et al. [13] improved the randomized runtime to $\tilde{O}(mn^{2k-2})$, where m is the number of hyperedges in the input hypergraph. Both these randomized algorithms are based on random contraction of hyperedges and are inspired partly by earlier work in [15] for Hypergraph-MinCut.

The existence of a randomized algorithm for Hypergraph-k-Cut raises the question of the existence of a deterministic algorithm. Random contraction-based algorithms do not lend themselves naturally to derandomization. Perhaps more pertinent is our interest in addressing the complexity of Submod-k-Part. There is no natural random contraction approach for this more general problem. For Graph-k-Cut, two distinct approaches lead to deterministic algorithms, and among these, the tree-packing approach, like the random contraction approach, does not appear to apply to Submod-k-Part. This leaves the divide-and-conquer approach initiated in the papers by Goldschmidt and Hochbaum [16, 17]. Is there a variant of this approach that works for Hypergraph-k-Cut and Submod-k-Part? We discovered certain structural properties of Hypergraph-k-Cut (that do not hold for other submodular functions) to prove our main result stated below.

Theorem 1. *There is a deterministic polynomial-time algorithm for* Hypergraph-k-Cut *for any fixed k.*

We believe that our work gives additional impetus to finding a polynomial-time algorithm for Submod-k-Part when k is fixed.

1.1. Technical Overview and Structural Results

We focus on the unit-cost variant of the problem in the rest of this work for the sake of notational simplicity. We note that this is without loss of generality since we allow multigraphs. All of our algorithms extend in a straightforward manner to arbitrary hyperedge costs. They rely only on minimum (s,t)-cut computations, and hence, they are strongly polynomial-time algorithms.

A key algorithmic tool will be the use of terminal cuts. We need some notation. Let $G=(V,E)$ be a hypergraph. For a subset U of vertices, we will use $\overline{U}$ to denote $V\backslash U$, $\delta(U)$ to denote the set of hyperedges crossing U and $d(U):=|\delta(U)|$ to denote the value of U. More generally, given a partition $(V_1,V_2,\ldots,V_h)$, we denote the number of hyperedges crossing the partition by $\mathrm{cost}(V_1,V_2,\ldots,V_h)$. Let S,T be disjoint subsets of vertices. A 2-partition $(U,\overline{U})$ is an (S,T)-terminal cut if $S\subseteq U\subseteq V\backslash T$. Here, the set U is known as the source set and the set $\overline{U}$ is known as the sink set. A minimum valued (S,T)-terminal cut is known as a minimum (S,T)-terminal cut. Because there could be multiple minimum (S,T)-terminal cuts, we will be interested in *source maximal* minimum (S,T)-terminal cuts and *source minimal* minimum (S,T)-terminal cuts. These cuts are unique and can be found in polynomial-time via standard maxflow algorithms. In fact, these definitions extend to general submodular functions. Given $f:2^V\to\mathbb{R}$ and disjoint sets $S,T\subseteq V$, we can define a minimum (S,T)-terminal cut for f as $\min_{U:S\subseteq U,T\subseteq\overline{U}} f(U)$. Uniqueness of source maximal and source minimal (S,T)-terminal cuts follow from submodularity, and one can also find these in polynomial time via submodular function minimization.

Our algorithm follows the divide-and-conquer approach that was first used by Goldschmidt and Hochbaum [16, 17] for Graph-k-Cut and in a more general fashion by Kamidoi et al. [22] to improve the running time for Graph-k-Cut. The goal in this approach is to identify one part of some fixed optimum k-partition $(V_1,V_2,\ldots,V_k)$, say V_1 without loss of generality, and then recursively find a $(k-1)$ partition of $V\backslash V_1$. How do we find such a part? Goldschmidt and Hochbaum [16, 17] proved a key structural lemma for Graph-k-Cut. Suppose $(V_1,V_2,\ldots,V_k)$ is an optimum k-partition such that V_1 is the part with the smallest cut value (i.e., $|\delta(V_1)|\le|\delta(V_i)|$ for all $i\in[k]$) and V_1 is maximal subject to this condition. Then, either $|V_1|\le k-2$ or there exist disjoint sets S,T such that $S\subseteq V_1, T\subseteq\overline{V_1}$ with $|S|\le k-1$ and $|T\cap V_j|=1$ for every $j\in\{2,\ldots,k\}$ so that the source maximal minimum (S,T)-terminal cut is $(V_1,\overline{V_1})$. One can guess/enumerate all small-sized (S,T)-pairs to find an $O(n^{2k-2})$-sized collection of sets so that the collection contains V_1 and recursively finds an optimum $(k-1)$-partition of $V\backslash U$ for each set U in the collection. This leads to an $n^{O(k^2)}$-time algorithm for Graph-k-Cut.

Queyranne [36] claimed that a natural generalization of the preceding structural lemma holds in the more general setting of Sym-Submod-k-Part. However, as reported in Guinez and Queyranne [18], the claimed proof was incorrect, and it was proved only for $k=3,4$. More importantly, as also noted in Guinez and Queyranne [18], this structural lemma (even if true for arbitrary k) is not useful for Sym-Submod-k-Part because one cannot recurse on $V\backslash V_1$; the function f restricted to $V\backslash V_1$ is no longer symmetric. The reader might now wonder how the approach works for Graph-k-Cut. Interestingly, Graph-k-Cut has the very nice property that the graph cut function restricted to $V\backslash V_1$ is still symmetric.

However, Hypergraph-k-Cut, the problem of interest here, is not a special case of Sym-Submod-k-Part. Nevertheless, we are able to prove a strong structural characterization that we state below. We consider the partition viewpoint of Hypergraph-k-Cut. We will denote a k-partition by an ordered tuple. A k-partition is a minimum k-partition if it has the minimum number of crossing hyperedges among all possible k-partitions. Because there could be multiple minimum k-partitions, we will be interested in the k-partition $(V_1,\ldots,V_k)$ for which V_1 is maximal; formally, we define a minimum k-partition $(V_1,\ldots,V_k)$ to be a *maximal minimum k-partition* if there is no other minimum k-partition $(V_1',\ldots,V_k')$ such that V_1 is strictly contained in V_1'. The following is our main structural result.

Theorem 2. *Let $G=(V,E)$ be a hypergraph, and let $(V_1,\ldots,V_k)$ be a maximal minimum k-partition in G for an integer $k\ge 2$. Suppose $|V_1|\ge 2k-2$. Then, for every subset $T\subseteq\overline{V_1}$ such that T intersects V_j for every $j\in\{2,\ldots,k\}$, there exists a subset $S\subseteq V_1$ of size $2k-2$ such that $(V_1,\overline{V_1})$ is the source maximal minimum (S,T)-terminal cut.*

Some important remarks regarding the preceding theorem are in order. First, this is surprising; for instance, if the optimum k-partition $V_1,\ldots,V_k$ is unique, then the theorem allows us to find any part V_i of the optimum k-partition $V_1,\ldots,V_k$ by solving minimum (S,T)-terminal cuts for S and T of bounded sizes (by noting that the reordered k-partition $(V_i,V_1,\ldots,V_{i-1},V_{i+1},\ldots,V_k)$ is also a maximal minimum k-partition due to uniqueness and by applying Theorem 2 to this reordered k-partition). Such a result was not known even for graphs. Secondly, our structural theorem differs crucially from the structural lemma of Goldschmidt and Hochbaum [17] for Graph-k-Cut in that it does not rely on V_1 being the part with the smallest cut value. This also explains why we need S to be of size $2k-2$ instead of $k-1$; one can show that $2k-2$ is tight for our structural theorem if we want to identify an arbitrary part even when considering Graph-k-Cut. Third, our structural theorem does not hold for general submodular functions. The theorem statement was partly inspired by experiments on small-sized instances, and the proof is partly inspired by a structural theorem in Kamidoi et al. [22] for graphs.

Theorem 2 implies, relatively easily, an $n^{O(k^2)}m$-time algorithm for Hypergraph-k-Cut, where n and m are the number of vertices and hyperedges in the input hypergraph. We improve the running time to $n^{O(k)}m$ using a similar but more involved structural result that allows us to recover the union of $k/2$ parts of an optimum k-partition. This high-level approach of recovering the union of $k/2$ parts of an optimum k-partition was developed in Kamidoi et al. [22] for Graph-k-Cut. As we already mentioned in the preceding paragraph, a proof of a key structural lemma in Kamidoi et al. [22] was an inspiration for our proofs, although the precise statement of our structural theorem is different from the structural lemma of Kamidoi et al. [22] and more subtle. We clarify this subtlety; the structural lemma in Kamidoi et al. [22] for graphs is that any 2-partition whose cut value is strictly smaller than half the optimum k-cut value can be recovered as a minimum (S,T)-terminal cut for S and T of sizes at most $k-1$. In contrast, our structural theorem (Theorem 2) states that V_1 — whose cut value need not necessarily be smaller than half the optimum k-cut value — can be recovered as a minimum (S,T)-terminal cut for S and T of sizes at most $2k-2$. We emphasize that the factor 2 in the conclusion of our structural result (i.e., in the size of S) is not simply a consequence of weakening the hypothesis by a factor of 2 compared with that of Kamidoi et al. [22].

1.1.1. Organization. In Section 2, we formally describe and analyze the basic recursive algorithm that utilizes our main structural theorem (Theorem 2). We prove an important uncrossing property of the hypergraph cut function in Section 3 and use it to prove Theorem 2 in Section 4. In Section 5, we prove a refined structural theorem and use it in Section 6 to derive a faster algorithm based on divide-and-conquer.

1.2. Other Related Work

Our main focus is on Hypergraph-k-Cut and Graph-k-Cut when k is fixed. As we mentioned already, Graph-k-Cut is NP-hard when k is part of the input [16]. A $2(1-1/k)$ approximation is known for Graph-k-Cut [39]; several other approaches also give a 2-approximation (see [9, 35] and references therein). Manurangsi [31] showed that there is no polynomial-time $(2-\epsilon)$-approximation for any constant $\epsilon > 0$ assuming the *small set expansion hypothesis* [37]. In contrast, Hypergraph-k-Cut was shown [7] to be at least as hard as the *densest k-subgraph* problem. Combined with results in Manurangsi [30], this shows that Hypergraph-k-Cut is unlikely to have a subpolynomial factor approximation ratio and illustrates that Hypergraph-k-Cut differs significantly from Graph-k-Cut when k is part of the input.

As we mentioned earlier, terminal versions of Submod-k-Part and its special cases such as Multiway-Cut in graphs have been studied extensively. The most general version here is Submod-Multiway-Cut; given a submodular function $f: 2^V \to \mathbb{R}$ (by value oracle) and terminals $\{s_1, s_2, \dots, s_k\} \subset V$, the goal is to find a partition $(V_1, \dots, V_k)$ to minimize $\sum_{i=1}^{k} f(V_i)$ subject to the constraint that $s_i \in V_i$ for every $i \in [k]$. These problems are NP-hard even for $k = 3$, and the main focus has been on approximation algorithms. We refer the reader to [2, 6, 12, 43] for further references. We mention that for nonnegative f and fixed k, the best approximation algorithms for Submod-k-Part and Sym-Submod-k-Part are via the terminal versions, a $(1.5-1/k)$ for Sym-Submod-k-Part and a $2(1-1/k)$-approximation for Submod-k-Part [6, 12].

Fixed parameter tractability of Graph-k-Cut has also been investigated. It is known that Graph-k-Cut is $W[1]$-hard (and hence, not likely to be FPT) parameterized by k [11], whereas it is FPT when parameterized by k and the solution size [26]. We observed, via a simple reduction from a result of Marx [32] on vertex separators, that Hypergraph-k-Cut is $W[1]$ hard even when parameterized by k *and* the solution size. This also demonstrates that Hypergraph-k-Cut differs in complexity from Graph-k-Cut. A recent work [29] (also see [19, 25]) has shown that Graph-k-Cut admits a fixed-parameter approximation scheme when parameterized by k. A fixed-parameter approximation scheme is also known for min-max graph k-cut[2] when parameterized by k [4].

Another problem closely related to Hypergraph-k-Cut is the Hypergraph-k-Partition problem. The input to Hypergraph-k-Partition is a hypergraph $G=(V,E)$ and a positive integer k, and the goal is to partition V into k nonempty sets $V_1, \dots, V_k$, but the objective is to minimize $\sum_{i=1}^{k} |\delta_G(V_i)|$; this means that a hyperedge e that crosses $h \geq 2$ parts pays h instead of only once (as is the case in Hypergraph- k-Cut). Hypergraph-k-Partition is a special case of Sym-Submod-k-Part, and its complexity status for fixed $k \geq 5$ is open. Hypergraph-k-Partition in constant rank hypergraphs is solvable in polynomial-time by relying on the fact that the number of constant-approximate minimum k-cuts in a constant rank hypergraph is polynomial.

2. Recursive Algorithm

Theorem 2 allows us to design a recursive algorithm for HYPERGRAPH-k-CUT. We need some notation in order to describe the recursive algorithm. For a hypergraph $G=(V,E)$ and for a subset $U \subseteq V$, let $G[U]$ denote the hypergraph obtained from G by discarding the vertices in $\overline{U}$ and by discarding all hyperedges $e \in E$ that intersect $\overline{U}$. The formal algorithm is described in Figure 1. It follows the high-level outline given in the technical overview. It enumerates $n^{O(k)}$ minimum (S,T)-terminal cuts, one of which is guaranteed to identify one part of an optimum k-partition, and then recursively finds an optimum $(k-1)$-partition after removing the found part. The runtime guarantee is given in Theorem 3. Theorem 1 follows from Theorem 3 by observing that the source maximal minimum (s,t)-terminal cut in a hypergraph can be computed in deterministic polynomial time; for example, it can be computed in time $O(np\log n) = O(n^2 m \log n)$, where n, m, and p are the number of vertices, hyperedges, and the size of the hypergraph respectively [8].

Theorem 3. *Let $G=(V,E)$ be a n-vertex hypergraph of size p, and let $k \geq 1$ be an integer. Then, algorithm CUT(G, k) given in Figure 1 returns a partition corresponding to a minimum k-cut in G, and it can be implemented to run in $n^{O(k^2)}T(n,p)$ time, where T(n, p) denotes the time complexity for computing the source maximal minimum (s, t)-terminal cut in a n-vertex hypergraph of size p.*

Proof. We first show the correctness of the algorithm. All candidates considered by the algorithm correspond to a k-partition, so we have to show only that the algorithm returns a k-partition corresponding to a minimum k-cut. We show this by induction on k. The base case of $k=1$ is trivial. We show the induction step. Assume that $k \geq 2$. Let $(V_1,\ldots,V_k)$ be a maximal minimum k-partition with cost OPT_k. By Theorem 2, the 2-partition $(V_1,\overline{V_1})$ is in $\mathcal{C}$. By induction hypothesis, the algorithm will return a minimum $(k-1)$-partition $(Q_1,\ldots,Q_{k-1})$ of $G[\overline{V_1}]$. Hence,

$$\mathrm{cost}_{G[\overline{V_1}]}(Q_1,\ldots,Q_{k-1}) \leq \mathrm{cost}_{G[\overline{V_1}]}(V_2,\ldots,V_k).$$

Therefore, the cost of the k-partition $(V_1,Q_1,\ldots,Q_{k-1})$ is

$$d(V_1) + \mathrm{cost}_{G[\overline{V_1}]}(Q_1,\ldots,Q_{k-1}) \leq d(V_1) + \mathrm{cost}_{G[\overline{V_1}]}(V_2,\ldots,V_k) = OPT_k.$$

Moreover, the k-partition $(V_1,Q_1,\ldots,Q_{k-1})$ is in $\mathcal{R}$. Hence, the algorithm returns a k-partition with cost at most OPT_k.

Next, we bound the runtime of the algorithm. Let $N(k, n)$ denote the number of source maximal minimum (s,t)-terminal cut computations executed by the algorithm CUT(G, k) on an n-vertex hypergraph G. We note that $|\mathcal{R}| = |\mathcal{C}| = O(n^{3k-3})$. Therefore,

$$N(k,n) \leq O(n^{3k-3})(1+N(k-1,n)) \text{ and}$$
$$N(1,n) = O(1).$$

Hence, $N(k,n) = O(n^{3k(k-1)/2})$. The total runtime is dominated by the time to implement these minimum (s,t)-terminal cuts, and hence, it is $O(n^{3k(k-1)/2})T(n,p)$. □

Figure 1. Algorithm to compute minimum k-cut in hypergraphs.

```
Algorithm CUT(G, k)
  Input: Hypergraph G = (V, E) and an integer k ≥ 1
  Output: A k-partition corresponding to a minimum k-cut in G
  If k = 1
      Return V
  else
      Initialize C ← {U ⊂ V : |U| ≤ 2k − 3} and R ← ∅
      For every disjoint S, T ⊂ V with |S| = 2k − 2 and |T| = k − 1
          Compute the source maximal minimum (S, T)-terminal cut (U, Ū)
          C ← C ∪ {U}
      For each U ∈ C
          P_Ū := CUT(G[Ū], k − 1)
          P := Partition of V obtained by concatenating U with P_Ū
          R ← R ∪ {P}
      Among all k-partitions in R, pick the one with minimum cost and return it
```

3. Uncrossing Properties of the Hypergraph Cut Function

In this section, we show the following uncrossing theorem, which will be useful to prove the main structural theorem. See Figure 2 for an illustration of the sets that appear in the statement of Theorem 4. The motivation for the statement of this uncrossing theorem will be clearer in the proof of Theorem 2. The reader may want to skip the rather long and technical proof of the uncrossing theorem in the first reading and come back to it after seeing its use in the proof of Theorem 2.

Theorem 4. *Let $G=(V,E)$ be a hypergraph, $k \geq 2$ be an integer, and $\emptyset \neq R \subsetneq U \subsetneq V$. Let $S=\{u_1,\ldots,u_p\} \subseteq U \backslash R$ for $p \geq 2k-2$. Let $(\overline{A_i}, A_i)$ be a minimum $((S \cup R)\backslash\{u_i\}, \overline{U})$-terminal cut. Suppose that $u_i \in A_i \backslash (\cup_{j \in [p]\backslash\{i\}} A_j)$ for every $i \in [p]$. Then, there exists a k-partition $(P_1,\ldots,P_k)$ of V with $\overline{U} \subsetneq P_k$ such that*

$$\mathrm{cost}(P_1,\ldots,P_k) \leq \frac{1}{2}\min\{d(A_i)+d(A_j) : i,j \in [p], i \neq j\}.$$

The rest of the section is devoted to the proof of Theorem 4. We begin with some background on the hypergraph cut function. Let $G=(V,E)$ be a hypergraph. For a subset A of vertices, we recall that $d(A)$ denotes the number of hyperedges that intersect both A and $\overline{A}$. The function $d: 2^V \to \mathbb{R}_+$ is known as the hypergraph cut function. The hypergraph cut function is symmetric, that is,

$$d(A) = d(\overline{A}) \text{ for all } A \subseteq V,$$

and submodular, that is,

$$d(A)+d(B) \geq d(A \cap B) + d(A \cup B) \text{ for all subsets } A, B \subseteq V.$$

For our purposes, it will help to count the hyperedges more accurately than employ the submodularity inequality. We define some notation that will help in more accurate counting. Let $(Y_1,\ldots,Y_p,W,Z)$ be a partition of V. We recall that $\mathrm{cost}(Y_1,\ldots,Y_p,W,Z)$ denotes the number of hyperedges that cross the partition; when considering these hyperedges, it is convenient to visualize each part of the partition as a single vertex obtained by contracting the part. We define the following quantities:

1. Let $\mathrm{cost}(W,Z) = |\{e \mid e \subseteq W \cup Z, e \cap W \neq \emptyset, e \cap Z \neq \emptyset\}|$ be the number of hyperedges contained in $W \cup Z$ that intersect both W and Z.
2. Let $\alpha(Y_1,\ldots,Y_p,W,Z)$ be the number of hyperedges that intersect Z and at least two of the sets in $\{Y_1,\ldots,Y_p,W\}$.
3. Let $\beta(Y_1,\ldots,Y_p,Z)$ be the number of hyperedges that are disjointed from Z but intersect at least two of the sets in $\{Y_1,\ldots,Y_p\}$.

For a partition $(Y_1,\ldots,Y_p,W,Z)$, we will be interested in the sum of $\mathrm{cost}(Y_1,\ldots,Y_p,W,Z)$ and the three quantities defined above, which we denote as $\sigma(Y_1,\ldots,Y_p,W,Z)$, that is,

$$\sigma(Y_1,\ldots,Y_p,W,Z) := \mathrm{cost}(Y_1,\ldots,Y_p,W,Z) + \mathrm{cost}(W,Z) + \alpha(Y_1,\ldots,Y_p,W,Z) + \beta(Y_1,\ldots,Y_p,Z).$$

We note that $\sigma(Y_1,\ldots,Y_p,W,Z)$ counts every hyperedge that crosses the partition twice, except for those hyperedges that intersect exactly one of the sets in $\{Y_1,\ldots,Y_p\}$ and exactly one of the sets in $\{W,Z\}$ that are counted exactly once (see Figure 3).

Figure 2. Illustration of the sets that appear in Theorem 4 and Lemma 2.

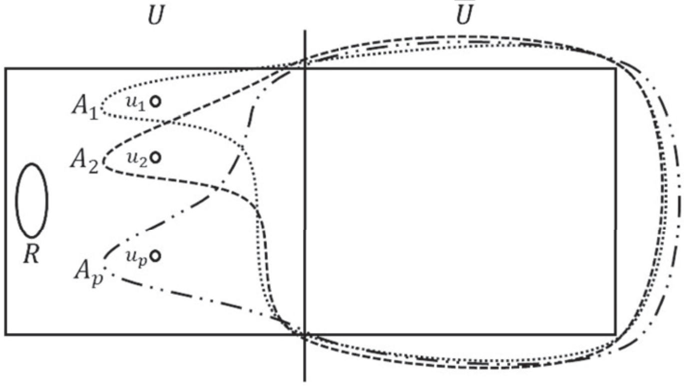
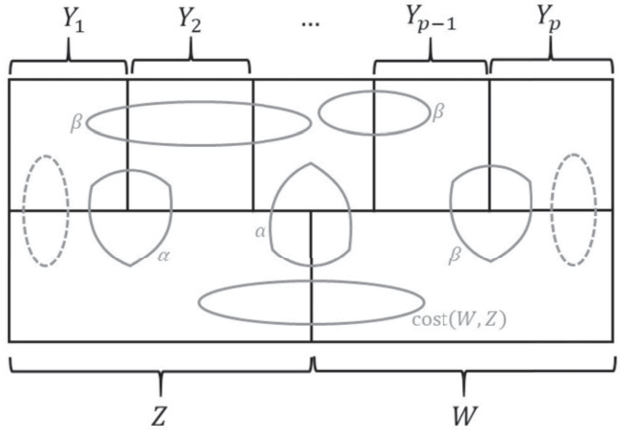


Figure 3. Hyperedges counted By $\sigma(Y_1, \ldots, Y_p, W, Z)$.



Notes. The dashed hyperedges are counted only by cost$(Y_1, \ldots, Y_p, W, Z)$. The rest of the hyperedges are counted twice in $\sigma(Y_1, \ldots, Y_p, W, Z)$: once by the term cost$(Y_1, \ldots, Y_p, W, Z)$ and once more by the indicated term.

The motivation behind considering the function $\sigma(Y_1, \ldots, Y_p, W, Z)$ comes from Proposition 1. We emphasize that the interpretation for $\sigma(Y_1, \ldots, Y_p, W, Z)$ given in the proposition holds only for $p = 2$.

Proposition 1. *Let (Y_1, Y_2, W, Z) be a partition of V, and let $A_1 := Y_1 \cup W$ and $A_2 := Y_2 \cup W$. Then,*

$$d(A_1) + d(A_2) = \sigma(Y_1, Y_2, W, Z).$$

Proof. We show the equality by a counting argument. We prove that each hyperedge is counted the same number of times in LHS and RHS. We note that both LHS and RHS count only hyperedges that cross the partition (Y_1, Y_2, W, Z). Let e be a hyperedge that crosses the partition (Y_1, Y_2, W, Z). Figure 4 can be used to verify the equality. Formally we have the following cases:

1. Suppose e intersects Z and exactly one of the sets in $\{Y_1, Y_2, W\}$.

(a) Suppose e intersects W. Then, e is counted twice in the LHS: by both $d(A_1)$ and $d(A_2)$. Moreover, e is counted twice in the RHS: by cost(Y_1, Y_2, W, Z) and by cost(W, Z).

(b) Suppose e intersects exactly one of the sets in $\{Y_1, Y_2\}$. Then, e is counted once in the LHS: by exactly one of $d(A_1)$ and $d(A_2)$. Moreover, e is counted exactly once in the RHS: by cost(Y_1, Y_2, W, Z).

2. Suppose e intersects Z and at least two of the sets in $\{Y_1, Y_2, W\}$. Then, e is counted twice in the LHS: by both $d(A_1)$ and $d(A_2)$. Moreover, e is counted twice in the RHS: by cost(Y_1, Y_2, W, Z) and by $\alpha(Y_1, Y_2, W, Z)$.

3. Suppose e is disjointed from Z and intersects both Y_1 and Y_2. Then, e is counted twice in the LHS: by both $d(A_1)$ and $d(A_2)$. Moreover, e is counted twice in the RHS: by cost(Y_1, Y_2, W, Z) and by $\beta(Y_1, Y_2, Z)$.

4. Suppose e is disjointed from Z and intersects exactly one of the sets in $\{Y_1, Y_2\}$. Because e is crossing the partition (Y_1, Y_2, W, Z), it has to intersect W. Then, e is counted once in the LHS: by exactly one of $d(A_1)$ and $d(A_2)$. Moreover, e is counted exactly once in the RHS: by cost(Y_1, Y_2, W, Z). □

The next lemma will help in obtaining a $(p+3)$-partition from a $(p+2)$-partition while controlling the increase in σ-value. This will be used in a subsequent inductive argument. See Figure 5 for an illustration of the sets appearing in the statement of the lemma. Our proof of Lemma 1 is through case analysis. Currently, we do not know how to prove this lemma without a somewhat laborious case analysis. We remark that this is partly because of the fact that hyperedges can have different cardinalities as well as because of the fact that we cannot rely only on submodularity of the hypergraph cut function.

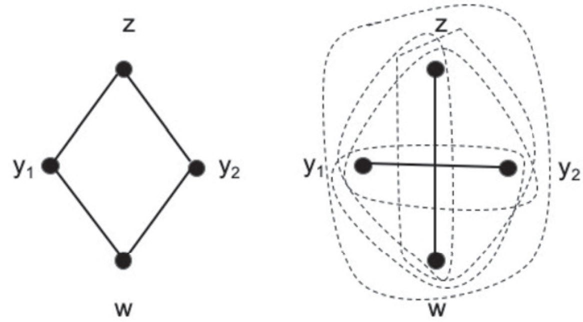
Lemma 1. *Let $G = (V, E)$ be a hypergraph, and let $(X_1, \ldots, X_p, W_0, Z_0)$ be a partition for some integer $p \geq 1$. Let $Q \subset V$ be a set such that*

$$Y_i := X_i - Q \neq \emptyset \;\; \forall i \in [p],\; Y_{p+1} := Q \cap Z_0 \neq \emptyset,\; Z := Z_0 - Q \neq \emptyset, \text{ and } W := W_0 \cup (Q \backslash Z_0) \neq \emptyset.$$

Then, $(Y_1, \ldots, Y_p, Y_{p+1}, W, Z)$ is a partition of V such that

$$\sigma(Y_1, \ldots, Y_p, Y_{p+1}, W, Z) \leq \sigma(X_1, \ldots, X_p, W_0, Z_0) + d(Q) - d(W_0 \cap Q).$$

Figure 4. Pictorial representation of hyperedges counted by $\sigma(Y_1, Y_2, W, Z)$.



Notes. Contract each part to a single vertex. *Left*: hyperedges that are counted once; *right*: all the rest that are counted twice. Edges are shown as lines, and hyperedges of size ≥ 3 are shown in dashed lines. One can verify that only hyperedges that are counted once in $d(A_1) + d(A_2)$ correspond to precisely those on the left.

Proof. By definition, $(Y_1, \ldots, Y_p, Y_{p+1}, W, Z)$ is a partition of V.

We rewrite the required inequality in the following form as it becomes convenient to prove:

$$\sigma(X_1, \ldots, X_p, W_0, Z_0) - \sigma(Y_1, \ldots, Y_p, Y_{p+1}, W, Z) \geq d(W_0 \cap Q) - d(Q). \tag{1}$$

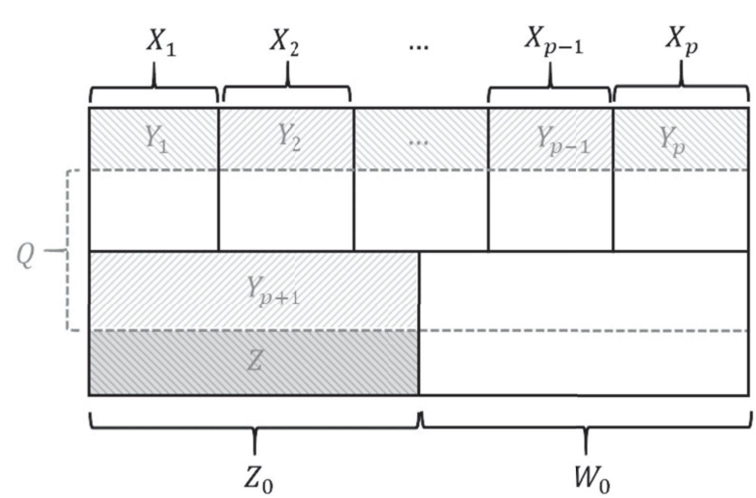
For a hyperedge $e \in E$, let $\lambda_e^0 \in \{0, 1, 2\}$ and $\lambda_e^1 \in \{0, 1, 2\}$ be the number of times that e is counted by $\sigma(X_1, \ldots, X_p, W_0, Z_0)$ and $\sigma(Y_1, \ldots, Y_p, Y_{p+1}, W, Z)$, respectively, and let $\lambda_e^Q \in \{0, 1\}$ and $\lambda_e^{W_0 \cap Q} \in \{0, 1\}$ be the number of times that e is counted by $d(Q)$ and $d(W_0 \cap Q)$ respectively.

Let $\ell_e := \lambda_e^0 - \lambda_e^1$ and $r_e := \lambda_e^{W_0 \cap Q} - \lambda_e^Q$. Thus, ℓ_e and r_e denote the number of times the hyperedge e is counted in the LHS and RHS of inequality (1), respectively, and moreover, $\ell_e \in \{0, \pm 1, \pm 2\}$ and $r_e \in \{0, \pm 1\}$ for every hyperedge $e \in E$. Let

$$\begin{aligned}
\text{Positives}(\ell) &:= \sum_{e \in E: \ell_e \geq 1} \ell_e, \\
\text{Negatives}(\ell) &:= \sum_{e \in E: \ell_e \leq -1} \ell_e, \\
\text{Positives}(r) &:= \sum_{e \in E: r_e = 1} r_e, \text{ and} \\
\text{Negatives}(r) &:= \sum_{e \in E: r_e = -1} r_e.
\end{aligned}$$

Claims 1 and 2 complete the proof of the lemma. □

Figure 5. Sets appearing in Lemma 1; unshaded portion corresponds to W.



Claim 1.

$$Positives(\ell) \geq Positives(r).$$

Proof. Let e be a hyperedge such that $r_e = 1$. Then, e is counted by $d(W_0 \cap Q)$ but not $d(Q)$. This means that $e \subseteq Q$, $e \cap (W_0 \cap Q) \neq \emptyset$, and $e \cap (Q \backslash W_0) \neq \emptyset$. Thus, e intersects $W_0 \cap Q$ and at least one of the sets in $\{X_1 \cap Q, \ldots, X_p \cap Q, Z_0 \cap Q\}$. It suffices to show that $\ell_e \geq 1$. We consider different cases for e below and show that $\ell_e \geq 1$ in all cases.

1. Suppose e intersects $Z_0 \cap Q$.
   (a) Suppose e is disjointed from $X_1 \cap Q, \ldots, X_p \cap Q$. Then, $\lambda_e^0 = 2$ because e is counted by both $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$ and by $\text{cost}(W_0, Z_0)$. However, $\lambda_e^1 = 1$ because e is counted only by $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$. Hence, $\ell_e = \lambda_e^0 - \lambda_e^1 \geq 1$.
   (b) Suppose e intersects at least one of the sets in $\{X_1 \cap Q, \ldots, X_p \cap Q\}$. Then, $\lambda_e^0 = 2$, because e is counted by both $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$ and by $\alpha(X_1, \ldots, X_p, W_0, Z_0)$. However, $\lambda_e^1 = 1$ because e is counted only by $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$. Hence, $\ell_e = \lambda_e^0 - \lambda_e^1 \geq 1$.
2. Suppose e is disjointed from $Z_0 \cap Q$. Then, e has to intersect at least one of the sets in $\{X_1 \cap Q, \ldots, X_p \cap Q\}$.
   (a) Suppose e intersects exactly one of the sets in $\{X_1 \cap Q, \ldots, X_p \cap Q\}$. Then, $\lambda_e^0 = 1$ because e is counted only by $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$. However, $\lambda_e^1 = 0$ because e does not cross the partition $(Y_1, \ldots, Y_{p+1}, W, Z)$. Hence, $\ell_e = \lambda_e^0 - \lambda_e^1 \geq 1$.
   (b) Suppose e intersects at least two of the sets in $\{X_1 \cap Q, \ldots, X_p \cap Q\}$. Then, $\lambda_e^0 = 2$ because e is counted by both $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$ and by $\beta(X_1, \ldots, X_p, Z_0)$. However, $\lambda_e^1 = 0$ because e does not cross the partition $(Y_1, \ldots, Y_{p+1}, W, Z)$. Hence, $\ell_e = \lambda_e^0 - \lambda_e^1 = 2 \geq 1$. □

Claim 2.

$$Negatives(\ell) \geq Negatives(r).$$

Proof. Let e be a hyperedge such that $\ell_e \leq -1$, that is, $\lambda_e^1 \geq \lambda_e^0 + 1$. Then, $\lambda_e^1 \geq 1$, and hence, e crosses the partition $(Y_1, \ldots, Y_{p+1}, W, Z)$. It suffices to show that $r_e \leq \ell_e$, that is, $\lambda_e^Q \geq \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$. We consider different cases for e below, and for each case, we show that either $\lambda_e^Q \geq \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$ or the case is impossible.

1. Suppose e is disjointed from Z. Then, e intersects at least one of the sets in $\{Y_1, \ldots, Y_{p+1}\}$ because e crosses the partition $(Y_1, \ldots, Y_{p+1}, W, Z)$.
   (a) Suppose e intersects exactly one of the sets in $\{Y_1, \ldots, Y_{p+1}\}$, say Y_i for some $i \in [p+1]$. Then, e intersects W and consequently, $\lambda_e^1 = 1$ because e is counted only by $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$. Because $1 = \lambda_e^1 \geq \lambda_e^0 + 1$, it follows that $\lambda_e^0 = 0$. This implies that e does not cross the partition $(X_1, \ldots, X_p, W_0, Z_0)$. Therefore, $i \in [p]$ and $e \subseteq X_i$ with e intersecting $X_i \cap Q$ and $Y_i = X_i \backslash Q$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \geq \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.
   (b) Suppose e intersects at least two of the sets in $\{Y_1, \ldots, Y_{p+1}\}$. Then, $\lambda_e^1 = 2$ because e is counted by both $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$ as well as $\beta(Y_1, \ldots, Y_{p+1}, Z)$.
      i. Suppose e intersects at least two of the sets in $\{Y_1, \ldots, Y_p\}$. If e intersects Z_0, then $\lambda_e^0 = 2$ because e is counted by both $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$ and $\alpha(X_1, \ldots, X_p, W_0, Z_0)$. If e is disjointed from Z_0, then again $\lambda_e^0 = 2$ because e is counted by both $\text{cost}(X_1, \ldots, X_p, W_0, Z_0)$ and $\beta(X_1, \ldots, X_p, W_0, Z_0)$. In both cases, we have $2 = \lambda_e^1 \geq \lambda_e^0 + 1 = 3$, a contradiction.
      ii. Suppose e intersects Y_{p+1} and exactly one of the sets in $\{Y_1, \ldots, Y_p\}$, say Y_i for some $i \in [p]$. Then, $\lambda_e^0 \geq 1$ because e crosses the partition $(X_1, \ldots, X_p, W_0, Z_0)$. Because $2 = \lambda_e^1 \geq \lambda_e^0 + 1$, it follows that $\lambda_e^0 = 1$. This implies that none of $\text{cost}(W_0, Z_0)$, $\alpha(X_1, \ldots, X_p, W_0, Z_0)$, and $\beta(X_1, \ldots, X_p, Z_0)$ count e. Therefore, e is disjointed from W_0, and e intersects $Y_{p+1} = Z_0 \cap Q$ and $Y_i = X_i \backslash Q$. Thus, e is counted by $d(Q)$ but not $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \geq \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.
2. Suppose e intersects Z. Then, e intersects at least one of the sets in $\{Y_1, \ldots, Y_{p+1}, W\}$ because e crosses the partition $(Y_1, \ldots, Y_{p+1}, W, Z)$.
   (a) Suppose e intersects exactly one of the sets in $\{Y_1, \ldots, Y_{p+1}, W\}$.
      i. Suppose e is disjointed from W. Then, e intersects exactly one of the sets in $\{Y_1, \ldots, Y_{p+1}\}$. Hence, e is counted only by $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$, and consequently, $\lambda_e^1 = 1$. Because $1 = \lambda_e^1 \geq \lambda_e^0 + 1$, we have that $\lambda_e^0 = 0$. This implies that e does not cross the partition $(X_1, \ldots, X_p, W_0, Z_0)$. Hence, e can only intersect Y_{p+1}. Thus, $e \subseteq Z_0 = Z \cup Y_{p+1}$ with e intersecting $Z = Z_0 \backslash Q$ and $Y_{p+1} = Z_0 \cap Q$. Thus, e is counted by $d(Q)$ but not $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \geq \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.
      ii. Suppose e intersects W. Then, $e \subseteq W \cup Z$, and e intersects W and e intersects Z. In particular, e is counted by $\text{cost}(Y_1, \ldots, Y_{p+1}, W, Z)$ and by $\text{cost}(W, Z)$ and hence, $\lambda_e^1 = 2$. Because $\lambda_e^1 \geq \lambda_e^0 + 1$, we have that

$\lambda_e^0 \le 1$. We also note that e crosses the partition $(X_1, \dots, X_p, W_0, Z_0)$, and therefore, $\lambda_e^0 \ge 1$. Thus, $\lambda_e^0 = 1$. This implies that none of cost(W_0, Z_0), $\alpha(X_1, \dots, X_p, W_0, Z_0)$, and $\beta(X_1, \dots, X_p, Z_0)$ count e and moreover, e intersects $X_i \backslash Y_i$ for some $i \in [p]$, with e being contained in $(X_i \backslash Y_i) \cup Z$. Consequently, e intersects $X_i \backslash Y_i = X_i \cap Q$, and e intersects Z, which implies that e is counted by $d(Q)$. Because e is contained in $(X_i \backslash Y_i) \cup Z$, it follows that e is disjointed from W_0, and hence, e is not counted by $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \ge \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.

(b) Suppose e intersects at least two of the sets in $\{Y_1, \dots, Y_{p+1}, W\}$. Then, $\lambda_e^1 = 2$ because e is counted by both cost$(Y_1, \dots, Y_{p+1}, W, Z)$ and $\alpha(Y_1, \dots, Y_{p+1}, W, Z)$.

i. Suppose e intersects at least two of the sets in $\{Y_1, \dots, Y_p\}$. Then $\lambda_e^0 = 2$ because e is counted by cost$(X_1, \dots, X_p, W_0, Z_0)$ as well as $\alpha(X_1, \dots, X_p, W_0, Z_0)$. Thus, $2 = \lambda_e^1 \ge \lambda_e^0 + 1 = 3$, a contradiction.

ii. Suppose e intersects exactly one of the sets in $\{Y_1, \dots, Y_p\}$, say Y_i for some $i \in [p]$, and e intersects Y_{p+1} but is disjointed from W. Then, $\lambda_e^0 \ge 1$ because e crosses the partition $(X_1, \dots, X_p, W_0, Z_0)$. Because $2 = \lambda_e^1 \ge \lambda_e^0 + 1$, it follows that $\lambda_e^0 = 1$. This implies that none of cost(W_0, Z_0), $\alpha(X_1, \dots, X_p, W_0, Z_0)$, and $\beta(X_1, \dots, X_p, Z_0)$ counts e and hence, e is contained in $Y_i \cup Z_0 \subseteq X_i \cup Z_0$, with e intersecting $Y_{p+1} = Z_0 \cap Q$ and $Y_i = X_i \backslash Q$. Thus, e is counted by $d(Q)$ but not $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \ge \lambda^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.

iii. Suppose e intersects exactly one of the sets in $\{Y_1, \dots, Y_p\}$, say Y_i for some $i \in [p]$, and e intersects W but is disjointed from Y_{p+1}. Then, $\lambda_e^0 \ge 1$ because e crosses the partition $(X_1, \dots, X_p, W_0, Z_0)$. Because $2 = \lambda_e^1 \ge \lambda_e^0 + 1$, it follows that $\lambda_e^0 = 1$. This implies that none of cost(W_0, Z_0), $\alpha(X_1, \dots, X_p, W_0, Z_0)$, and $\beta(X_1, \dots, X_p, Z_0)$ counts e. Therefore, e is contained in $X_i \cup Z$, and e intersects $X_i \cap Q$ because e has to intersect W. Moreoever, e intersects $Y_i = X_i \backslash Q$. Thus, e is counted by $d(Q)$ but not $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \ge \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$.

iv. Suppose e is disjointed from $Y_1, \dots, Y_p$ and intersects both Y_{p+1} and W.

A. Suppose e intersects at least two of the sets in $\{X_1 \cap Q, \dots, X_p \cap Q\}$. Then, $\lambda_e^0 = 2$ because e is counted by cost$(X_1, \dots, X_p, W_0, Z_0)$ as well as $\alpha(X_1, \dots, X_p, W_0, Z_0)$. Thus, $2 = \lambda_e^1 \ge \lambda_e^0 + 1 = 3$, a contradiction.

B. Suppose e does not intersect $X_1 \cap Q, \dots, X_p \cap Q$. Then, e intersects W_0 because e is counted by both cost$(Y_1, \dots, Y_{p+1}, W, Z)$ and $\alpha(Y_1, \dots, Y_{p+1}, W, Z)$ (recall that we are in case (b)). Moreoever, $e \subseteq W_0 \cup Z_0$. Therefore, $\lambda_e^0 = 2$ because e is counted by cost$(X_1, \dots, X_p, W_0, Z_0)$ as well as cost(W_0, Z_0). Thus, $2 = \lambda_e^1 \ge \lambda_e^0 + 1 = 3$, a contradiction.

C. Suppose e intersects exactly one of the sets in $\{X_1 \cap Q, \dots, X_p \cap Q\}$, say $X_i \cap Q$ for some $i \in [p]$, and e intersects $W_0 \cap Q$. Then, $\lambda_e^0 = 2$ because e is counted by both cost$(X_1, \dots, X_p, W_0, Z_0)$ and $\alpha(X_1, \dots, X_p, W_0, Z_0)$. Thus, $2 = \lambda_e^1 \ge \lambda_e^0 + 1 = 3$, a contradiction.

D. Suppose e intersects exactly one of the sets in $\{X_1 \cap Q, \dots, X_p \cap Q\}$, say $X_i \cap Q$ for some $i \in [p]$, and e is disjointed from $W_0 \cap Q$. Then, $\lambda_e^0 \ge 1$ because e crosses the partition $(X_1, \dots, X_p, W_0, Z_0)$. Because $2 = \lambda_e^1 \ge \lambda_e^0 + 1$, it follows that $\lambda_e^0 = 1$. This implies that none of cost(W_0, Z_0), $\alpha(X_1, \dots, X_p, W_0, Z_0)$, and $\beta(X_1, \dots, X_p, Z_0)$ counts e. Therefore, e is contained in $(X_i \cap Q) \cup Z_0$, and e intersects $Y_{p+1} = Z_0 \cap Q$ and $Z = Z_0 \backslash Q$. Thus, e is counted by $d(Q)$ but not $d(W_0 \cap Q)$. Consequently, $\lambda_e^Q = 1$ and $\lambda_e^{W_0 \cap Q} = 0$. Hence, $\lambda_e^Q \ge \lambda_e^{W_0 \cap Q} + \lambda_e^1 - \lambda_e^0$. □

The next lemma will help in uncrossing a collection of sets to obtain a partition with small σ-value. See Figure 2 for an illustration of the sets that appear in the statement of the lemma.

Lemma 2. *Let $G = (V, E)$ be a hypergraph and $\emptyset \ne R \subsetneq U \subsetneq V$. Let $S = \{u_1, \dots, u_p\} \subseteq U \backslash R$ for $p \ge 2$. Let $(\overline{A_i}, A_i)$ be a minimum $((S \cup R) \backslash \{u_i\}, \overline{U})$-terminal cut. Suppose that $u_i \in A_i \backslash (\cup_{j \in [p] \backslash \{i\}} A_j)$ for every $i \in [p]$. Let*

$$Z := \cap_{i=1}^p \overline{A_i},\ W := \cup_{1 \le i < j \le p} (A_i \cap A_j), \text{ and } Y_i := A_i - W \quad \forall i \in [p].$$

Then, $(Y_1, \dots, Y_p, W, Z)$ is a $(p+2)$-partition of V with

$$\sigma(Y_1, \dots, Y_p, W, Z) \le \min\{d(A_i) + d(A_j) : i, j \in [p], i \ne j\}.$$

Proof. For every $i \in [p]$, the set Y_i is nonempty because $u_i \in Y_i$. The set W is nonempty because $\overline{U} \subseteq W$. The set Z is nonempty because $R \subseteq Z$. By definition, the sets $Y_1, \dots, Y_p, W, Z$ are all disjointed, and their union contains all vertices. Hence, $(Y_1, \dots, Y_p, W, Z)$ is a partition of V. Without loss of generality, let $d(A_1) \le d(A_2) \le \dots \le d(A_p)$. We bound the σ-value of the partition by induction on p.

The base case of $p = 2$ follows from Proposition 1. We show the induction step. Suppose that the statement holds for $p = q$. We prove that it holds for $p = q + 1$. Consider $R_0 := R \cup \{u_{q+1}\}$ and $S_0 := S \backslash \{u_{q+1}\}$. Then, $(\overline{A_i}, A_i)$ is still a minimum $((S_0 \cup R_0)\backslash\{u_i\}, \overline{U})$-terminal cut for every $i \in [q]$, and moreover, $u_i \in A_i \backslash \cup_{j\in[q]\backslash\{i\}} A_j$ for every $i \in [q]$. By induction hypothesis, we get that for the sets

$$Z_0 := \cap_{i=1}^{q} \overline{A_i},\ W_0 := \cup_{1\le i<j\le q}(A_i \cap A_j), \text{ and } X_i := A_i - W \quad \forall i \in [q],$$

we have

$$\sigma(X_1, \ldots, X_q, W_0, Z_0) \le d(A_1) + d(A_2).$$

The partition $(X_1, \ldots, X_q, W_0, Z_0)$ and the set $Q := A_{q+1}$ satisfy the conditions of Lemma 1. By Lemma 1, we obtain that

$$\sigma(Y_1, \ldots, Y_q, Y_{q+1}, W, Z) \le \sigma(X_1, \ldots, X_q, W_0, Z_0) + d(A_{q+1}) - d(W_0 \cap A_{q+1}).$$

Because $(\overline{W_0 \cap A_{q+1}}, W_0 \cap A_{q+1})$ is a feasible $((S \cup R)\backslash\{u_{q+1}\}, \overline{U})$-terminal cut, we have that $d(A_{q+1}) \le d(W_0 \cap A_{q+1})$. Hence,

$$\sigma(Y_1, \ldots, Y_q, Y_{q+1}, W, Z) \le \sigma(X_1, \ldots, X_q, W_0, Z_0) \le d(A_1) + d(A_2). \quad \square$$

The next lemma will help in aggregating the parts of a $2k$-partition $\mathcal{P}$ to a k-partition $\mathcal{K}$ so that the cost of $\mathcal{K}$ is at most half the σ-value of $\mathcal{P}$.

Lemma 3. *Let $G = (V, E)$ be a hypergraph, $k \ge 2$ be an integer, and $(Y_1, \ldots, Y_p, W, Z)$ be a partition of V for some integer $p \ge 2k - 2$. Then, there exists distinct $i_1, \ldots, i_{k-1} \in [p]$ such that*

$$\begin{aligned}2\text{cost}\Big(Y_{i_1}, \ldots, Y_{i_{k-1}}, V\backslash(\cup_{j=1}^{k-1} Y_{i_j})\Big) \le\ & \text{cost}(Y_1, \ldots, Y_p, W, Z) + \alpha(Y_1, \ldots, Y_p, W, Z)\\ & + \beta(Y_1, \ldots, Y_p, Z) - \text{cost}(W, Z).\end{aligned}$$

Proof. Suppose that the lemma is false. Pick a counterexample hypergraph $G = (V, E)$ such that $|V| + |E|$ is minimum. Hence, for every distinct $i_1, \ldots, i_{k-1} \in [p]$, we have

$$\begin{aligned}2\text{cost}\Big(Y_{i_1}, \ldots, Y_{i_{k-1}}, V\backslash(\cup_{j=1}^{k-1} Y_{i_j})\Big) >\ & \text{cost}(Y_1, \ldots, Y_p, W, Z) + \alpha(Y_1, \ldots, Y_p, W, Z)\\ & + \beta(Y_1, \ldots, Y_p, Z) - \text{cost}(W, Z).\end{aligned}$$

Minimality of the counterexample implies that $|Y_i| = 1$ for every $i \in [p]$ and $|W| = 1 = |Z|$ (otherwise, we can obtain a smaller counterexample by contracting the corresponding subset). If there exists a hyperedge $e \subseteq W \cup Z$ with e intersecting both W and Z, then discarding e would still preserve the counterexample property because e is not counted in both LHS and RHS, and hence, no such hyperedge exists in G, that is, $\text{cost}(W, Z) = 0$. For similar reasons, if there exists a hyperedge e that is double counted by RHS (see Figure 3), then discarding this hyperedge would still preserve the counterexample property. Minimality of the counterexample implies that no such hyperedge can exist. Consequently, all hyperedges present in the hypergraph G are in fact edges with one end-vertex in Y_i for some $i \in [p]$ and another end-vertex in W or Z. Thus,

$$RHS = \text{cost}(Y_1, \ldots, Y_p, W, Z) = \sum_{i=1}^{p} d(Y_i).$$

Without loss of generality, let $d(Y_1) \le d(Y_2) \le \ldots \le d(Y_p)$. Then,

$$2\text{cost}\Big(Y_1, \ldots, Y_{k-1}, V\backslash(\cup_{j=1}^{k-1} Y_{i_j})\Big) = 2\sum_{i=1}^{k-1} d(Y_i) \le \sum_{i=1}^{p} d(Y_i) = RHS.$$

The inequality above is because $p \ge 2(k-1)$. Thus, G cannot be a counterexample. $\square$

Remark 1. Lemma 3 can also be proved by picking a random subset of $k - 1$ sets among $\{Y_1, \ldots, Y_p\}$. The proof that we have given above illustrates that the tight case for the lemma is in fact a graph and not necessarily a hypergraph.

We now restate and prove the main uncrossing theorem of this section.

Theorem 4. *Let $G=(V,E)$ be a hypergraph, $k\ge 2$ be an integer, and $\emptyset\ne R\subsetneq U\subsetneq V$. Let $S=\{u_1,\dots,u_p\}\subseteq U\backslash R$ for $p\ge 2k-2$. Let $(\overline{A_i},A_i)$ be a minimum $((S\cup R)\backslash\{u_i\},\overline{U})$-terminal cut. Suppose that $u_i\in A_i\backslash(\bigcup_{j\in[p]\backslash\{i\}}A_j)$ for every $i\in[p]$. Then, there exists a k-partition $(P_1,\dots,P_k)$ of V with $\overline{U}\subsetneq P_k$ such that*

$$\mathrm{cost}(P_1,\dots,P_k)\le\frac{1}{2}\min\{d(A_i)+d(A_j):i,j\in[p],i\ne j\}.$$

Proof. By applying Lemma 2, we obtain a $(p+2)$-partition $(Y_1,\dots,Y_p,W,Z)$ such that

$$\sigma(Y_1,\dots,Y_p,W,Z)\le\min\{d(A_i)+d(A_j):i,j\in[p],i\ne j\},$$

and moreover, $\overline{U}\subseteq W$. We recall that $p\ge 2k-2$. Hence, by applying Lemma 3 to the $(p+2)$-partition $(Y_1,\dots,Y_p,W,Z)$, we obtain a k-partition $(P_1,\dots,P_k)$ of V such that $W\cup Z\subseteq P_k$ and

$$\mathrm{cost}(P_1,\dots,P_k)\le\frac{1}{2}\sigma(Y_1,\dots,Y_p,W,Z)\le\frac{1}{2}\min\{d(A_i)+d(A_j):i,j\in[p],i\ne j\}.$$

We note that $\overline{U}$ is strictly contained in P_k because $\overline{U}\cup Z\subseteq W\cup Z\subseteq P_k$, and Z is nonempty. □

Remark 2. The lower-bound condition on p (i.e., $p\ge 2k-2$) in the statement of Theorem 4 is tight. In particular, the conclusion of the theorem does not hold for $p=2k-3$, as illustrated by the graph in Figure 6.

Remark 3. A natural counterpart of Theorem 4 for (symmetric) submodular functions is false. For a submodular function $f:2^V\to\mathbb{R}_+$, by defining $f_{\mathrm{sym}}(U):=f(U)+f(\overline{U})$ to be the value of the 2-partition $(U,\overline{U})$, and assuming the conditions of the theorem, it is tempting to conjecture that there exists a k-partition $(P_1,\dots,P_k)$ such that

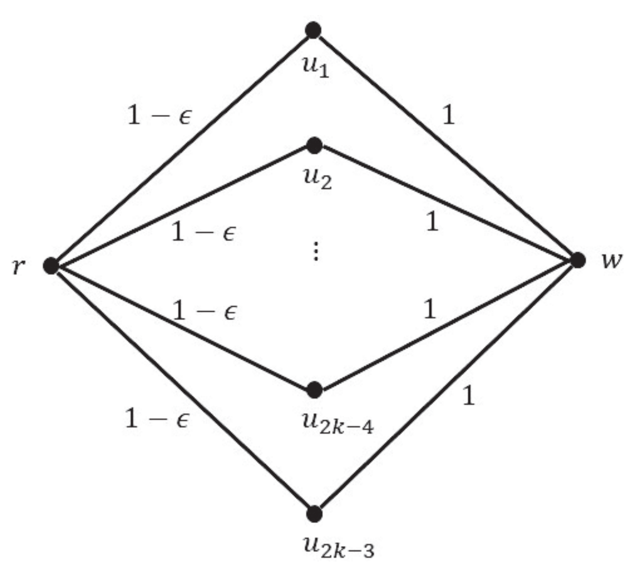
$$\sum_{i=1}^{k}f(P_i)\le\frac{1}{2}\min\{f_{\mathrm{sym}}(A_i)+f_{\mathrm{sym}}(A_j):i,j\in[p],i\ne j\}.$$

Here is a counterexample. Consider the function $f(S):=1$ if $\emptyset\ne S\subsetneq V$, $f(\emptyset):=0$, and $f(V):=0$. Then, for any k-partition $(P_1,\dots,P_k)$, we have $\sum_{i=1}^{k}f(P_i)=k$. However, the RHS in the above inequality is only 2. We note that this example is an instance of Hypergraph-k-Partition.

4. Proof of Theorem 2

In this section, we prove Theorem 2. We start with a useful containment property captured by the next lemma.

Figure 6. An edge-weighted graph showing the necessity of the condition $p\ge 2k-2$ in Theorem 4 (where ϵ is a small positive constant).



Notes. We consider $U=\{r,u_1,u_2,\dots,u_{2k-3}\}$ and $R=r$. Then, the RHS of the theorem is $2k-3-\epsilon$, whereas the cost of any k-cut is at least $2k-2-O(\epsilon)$.

Lemma 4. *Let $G = (V, E)$ be a hypergraph, $(V_1, \dots, V_k)$ be a maximal minimum k-partition in G for an integer $k \geq 2$, and $S \subseteq V_1$, $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \dots, k\}$. Suppose $(U, \overline{U})$ is a minimum (S, T)-terminal cut. Then, $U \subseteq V_1$.*

Proof. For the sake of contradiction, suppose $U \backslash V_1 \neq \emptyset$. We will obtain another minimum k-partition that will contradict the maximality of V_1 in the minimum k-partition $(V_1, \dots, V_k)$. We observe that

$$d(U) \leq d(U \cap V_1) \tag{2}$$

because $(U \cap V_1, \overline{U \cap V_1})$ is a (S, T)-terminal cut. We need the following claim:

Claim 3.

$$d(V_1) \leq d(U \cup V_1).$$

Proof. For the sake of contradiction, suppose $d(U \cup V_1) < d(V_1)$. Then, consider that $W_1 := U \cup V_1$ and $W_j := V_j \backslash U$ for every $j \in \{2, \dots, k\}$ (see Figure 7). We have $d(W_1) < d(V_1)$. Because $S \subseteq W_1$ and $T \cap W_j \neq \emptyset$ for every $j \in \{2, \dots, k\}$, we have that $(W_1, \dots, W_k)$ is a k-partition. We will show that $\text{cost}(W_1, \dots, W_k)$ is strictly smaller than $\text{cost}(V_1, \dots, V_k)$, thus contradicting the optimality of the k-partition $(V_1, \dots, V_k)$.

We recall that for a subset A of vertices, the graph $G[A]$ is obtained from G by discarding the vertices in $\overline{A}$ and by discarding the hyperedges that intersect $\overline{A}$. With this notation, we can write

$$\text{cost}_G(W_1, \dots, W_k) = d(W_1) + \text{cost}_{G[\overline{W_1}]}(W_2, \dots, W_k) \text{ and}$$
$$\text{cost}_G(V_1, \dots, V_k) = d(V_1) + \text{cost}_{G[\overline{V_1}]}(V_2, \dots, V_k).$$

Moreover, every hyperedge that is disjointed from $W_1 = U \cup V_1$ but crosses the $(k-1)$-partition $(W_2 = V_2 \backslash U, \dots, W_k = V_k \backslash U)$ is also disjointed from V_1 but crosses the $(k-1)$-partition $(V_2, \dots, V_k)$. Hence, $\text{cost}_{G[\overline{W_1}]}(W_2, \dots, W_k) \leq \text{cost}_{G[\overline{V_1}]}(V_2, \dots, V_k)$. We also have $d(W_1) < d(V_1)$. Therefore,

$$\text{cost}(W_1, \dots, W_k) < \text{cost}(V_1, \dots, V_k),$$

a contradiction to optimality of the k-partition $(V_1, \dots, V_k)$. □

By inequality (2), Claim 3, and submodularity of the hypergraph cut function, we have that

$$d(U) + d(V_1) \leq d(U \cap V_1) + d(U \cup V_1) \leq d(U) + d(V_1).$$

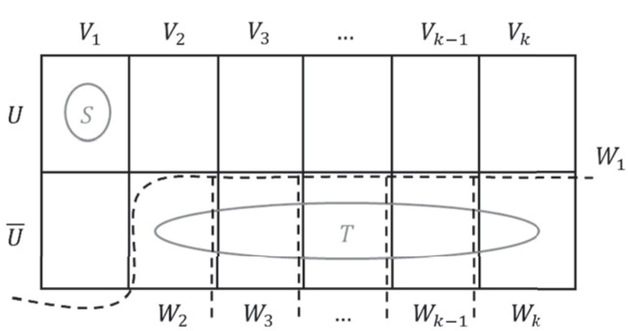
Therefore, the inequality in Claim 3 should in fact be an equation, that is,

$$d(V_1) = d(U \cup V_1).$$

Going through the proof of Claim 3 with this additional fact, we obtain that the k-partition $(U \cup V_1, V_2 \backslash U, \dots, V_k \backslash U)$ has cost at most that of $(V_1, \dots, V_k)$. Hence, the k-partition $(U \cup V_1, V_2 \backslash U, \dots, V_k \backslash U)$ is also a minimum k-partition, and it contradicts the maximality of V_1. □

Remark 4. Lemma 4 also holds for Submod-k-Part. That is, for a submodular function $f: 2^V \to \mathbb{R}_+$ with $(V_1, \dots, V_k)$ being a maximal minimum k-partition for an integer $k \geq 2$, subsets $S \subseteq V_1$ and $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \dots, k\}$, and $(U, \overline{U})$ being an S,T-separating 2-partition with minimum $f(U) + f(\overline{U})$ among

Figure 7. Uncrossing in the proof of Claim 3.



all S,T-separating 2-partitions, we have that $U \subseteq V_1$. This can be shown using the proof of Theorem 5 in Okumoto et al. [34].

Remark 5. Lemma 4 also applies to Submod-Multiway-Cut. We recall that the input here is a submodular function $f: 2^V \to \mathbb{R}$ (by value oracle) and terminals $\{s_1, \dots, s_k\} \subset V$, and the goal is to find a partition $(V_1, \dots, V_k)$ to minimize $\sum_{i=1}^{k} f(V_i)$ subject to the constraint that $s_i \in V_i$ for every $i \in [k]$. Let $(V_1, \dots, V_k)$ be an optimum solution so that there is no other optimum solution $(V_1', \dots, V_k')$, with V_1 being strictly contained in V_1'. Suppose $(U, \overline{U})$ is a minimum $(\{s_1\}, \{s_2, \dots, s_k\})$-terminal cut. Then, $U \subseteq V_1$. This was shown implicitly for graph multiway cut by Dahlhaus et al. [10].

We now restate and prove Theorem 2.

Theorem 2. *Let $G = (V, E)$ be a hypergraph, and let $(V_1, \dots, V_k)$ be a maximal minimum k-partition in G for an integer $k \ge 2$. Suppose $|V_1| \ge 2k - 2$. Then, for every subset $T \subseteq \overline{V_1}$ such that T intersects V_j for every $j \in \{2, \dots, k\}$, there exists a subset $S \subseteq V_1$ of size $2k - 2$ such that $(V_1, \overline{V_1})$ is the source maximal minimum (S, T)-terminal cut.*

Proof. For the sake of contradiction, suppose that the theorem is false for some subset $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for all $j \in \{2, \dots, k\}$. Our proof strategy is to obtain a cheaper k-partition than $(V_1, \dots, V_k)$, thereby contradicting the optimality of $(V_1, \dots, V_k)$. For a subset $X \subseteq V_1$, let $(V_X, \overline{V_X})$ be the source maximal minimum (X, T)-terminal cut.

Among all possible subsets of V_1 of size $2k - 2$, pick a subset S such that $d(V_S)$ is maximum. By Lemma 4 and assumption, we have that $V_S \subsetneq V_1$. By source maximality of the minimum (S,T)-terminal cut $(V_S, \overline{V_S})$, we have that $d(V_S) < d(V_1)$. Let $u_1, \dots, u_{2k-2}$ be the vertices in S. Because $V_S \subsetneq V_1$, there exists a vertex $u_{2k-1} \in V_1 \backslash V_S$. Let $C := \{u_1, \dots, u_{2k-1}\} = S \cup \{u_{2k-1}\}$. For $i \in [2k-1]$, let $(B_i, \overline{B_i})$ be the source maximal minimum $(C - \{u_i\}, T)$-terminal cut. We note that $(B_{2k-1}, \overline{B_{2k-1}}) = (V_S, \overline{V_S})$ and the size of $C - \{u_i\}$ is $2k - 2$ for every $i \in [2k-1]$. By Lemma 4 and assumption, we have that $B_i \subsetneq V_1$ for every $i \in [2k-1]$. Hence, we have

$$d(B_i) \le d(V_S) < d(V_1) \text{ and } B_i \subsetneq V_1 \text{ for every } i \in [2k-1]. \tag{3}$$

The next claim will set us up to apply Theorem 4.

Claim 4. *For every $i \in [2k-1]$, we have that $u_i \in \overline{B_i}$.*

Proof. The claim holds for $i = 2k - 1$ by choice of u_{2k-1}. For the sake of contradiction, suppose $u_i \in B_i$ for some $i \in [2k-2]$. Then, the 2-partition $(V_S \cap B_i, \overline{V_S \cap B_i})$ is a (S,T)-terminal cut, and hence,

$$d(V_S \cap B_i) \ge d(V_S).$$

We also have that

$$d(V_S \cup B_i) \ge d(V_S)$$

because $(V_S \cup B_i, \overline{V_S \cup B_i})$ is a (S,T)-terminal cut. Thus,

$$\begin{aligned} 2d(V_S) &\ge d(V_S) + d(B_i) && \text{(By choice of } S) \\ &\ge d(V_S \cup B_i) + d(V_S \cap B_i) && \text{(By submodularity)} \\ &\ge 2d(V_S). \end{aligned}$$

Therefore, $d(V_S) = d(V_S \cup B_i)$. Moreover, $B_i \backslash V_S$ is nonempty because the vertex $u_{2k-1} \in B_i \backslash V_S$. Hence, the 2-partition $(V_S \cup B_i, \overline{V_S \cup B_i})$ is a minimum (S,T)-terminal cut. However, this contradicts source maximality of the minimum (S,T)-terminal cut $(V_S, \overline{V_S})$ because $u_{2k-1} \in B_i$ and $u_{2k-1} \notin V_S$. □

We note that for every $i \in [2k-1]$, the 2-partition $(B_i, \overline{B_i})$ is a minimum $(C - \{u_i\}, \overline{V_1})$-terminal cut because $\overline{V_1} \subseteq \overline{B_i}$.

We will now apply Theorem 4. We consider $U := V_1$, $R := \{u_{2k-1}\} \subseteq U$, $S = \{u_1, \dots, u_{2k-2}\} \subseteq U \backslash R$. Let $p := 2k - 2$, and let $(\overline{A_i}, A_i) := (B_i, \overline{B_i})$ for every $i \in [p]$. The 2-partition $(\overline{A_i}, A_i)$ is a minimum $((S \cup R) \backslash \{u_i\}, \overline{U})$-terminal cut for every $i \in [p]$. By Claim 4, we have that $u_i \in A_i$ for every $i \in [p]$. Because $(B_j, \overline{B_j})$ is a $(C - \{u_j\}, T)$-terminal cut, we have that $u_i \notin \overline{B_j}$ for every distinct $i, j \in [p]$. Thus, $u_i \in A_i \backslash (\cup_{j \in [p] \backslash \{i\}} A_j)$ for every $i \in [p]$. Therefore, the sets U, R, S and the 2-partitions $(\overline{A_i}, A_i)$ for $i \in [p]$ satisfy the conditions of Theorem 4. By Theorem 4, symmetry of the cut function, and statement (3), we obtain a k-partition $(P_1, \dots, P_k)$ of V such that

$$\begin{aligned} \text{cost}(P_1, \dots, P_k) &\le \frac{1}{2} \min\{d(A_i) + d(A_j) : i, j \in [p], i \neq j\} \\ &= \frac{1}{2} \min\{d(B_i) + d(B_j) : i, j \in [p], i \neq j\} \\ &< d(V_1) \le OPT_k. \end{aligned}$$

Thus, we have obtained a k-partition whose cost is smaller than OPT_k, a contradiction. □

Remark 6. The proof techniques in this section relied only on the submodularity of the hypergraph cut function and the use of Theorem 4. The proof of Theorem 4 relied heavily on the properties of the hypergraph cut function. As we remarked in Section 3, there does not seem to be a counterpart of Theorem 4 for general submodular functions.

5. Structural Theorem for Divide and Conquer

We need a slightly stronger structural theorem to design a faster algorithm that is based on divide and conquer. We remark again that the proof techniques in this section will rely only on the submodularity of the hypergraph cut function and the use of Theorem 4.

We note that the source maximal minimum (S,T)-terminal cut is identical to the sink minimal minimum (S,T)-terminal cut. We define a 2-partition $(U,\overline{U})$ to be a *balanced minimum k-partition split* if there exists a minimum k-partition $(V_1,\dots,V_k)$ such that $U=\bigcup_{i=1}^{\lfloor k/2\rfloor}V_i$. Because there could be multiple balanced minimum k-partition splits, we will be interested in a minimal balanced minimum k-partition split; a balanced minimum k-partition split $(U,\overline{U})$ is *minimal* if there does not exist another balanced minimum k-partition split $(U',\overline{U'})$ such that U' is strictly contained in U.

We need the following two theorems. We defer their proofs to Sections 5.1 and 5.2, respectively.

Theorem 5. *Let $G=(V,E)$ be a hypergraph, and let OPT_k be the value of a minimum k-cut in G for some integer $k\ge 2$. Suppose $(U,\overline{U})$ is a 2-partition of V with $d(U)\le OPT_k$. Then, there exists a subset $S\subseteq U$ with $|S|\le 2k-2$ such that $(U,\overline{U})$ is the source maximal minimum $(S,\overline{U})$-terminal cut in G.*

Theorem 6. *Let $G=(V,E)$ be a hypergraph, and let $(U,\overline{U})$ be a minimal balanced minimum k-partition split in G for some integer $k\ge 2$. Then, for every vertex $u_0\in U$, there exists a subset $S\subseteq U\backslash\{u_0\}$ with $|S|\le 2k-3$ such that $(U,\overline{U})$ is the unique minimum $(S\bigcup\{u_0\},\overline{U})$-terminal cut in G.*

We now state and prove the structural theorem that facilitates the faster divide-and-conquer algorithm.

Theorem 7. *Let $G=(V,E)$ be a hypergraph, and let $(U,\overline{U})$ be a minimal balanced minimum k-partition split in G for some integer $k\ge 2$. Then, for every vertex $u_0\in U$, there exist subsets $S\subseteq U\backslash\{u_0\}$ and $T\subseteq\overline{U}$ with $|S|\le 2k-3$ and $|T|\le 2k-2$ such that $(U,\overline{U})$ is the source minimal minimum $(S\bigcup\{u_0\},T)$-terminal cut in G.*

Proof. Let $u_0\in U$. Applying Theorem 6 to $(U,\overline{U})$ with respect to vertex $u_0\in U$, we obtain a set $S\subseteq U$ with $|S|\le 2k-3$ such that $(U,\overline{U})$ is the unique minimum $(S\bigcup\{u_0\},\overline{U})$-terminal cut in G.

Applying Theorem 5 to $(\overline{U},U)$, we obtain a set $T\subseteq\overline{U}$ with $|T|\le 2k-2$ such that $(\overline{U},U)$ is a source maximal minimum (T,U) cut in G. Hence, by interchanging source and sink, $(U,\overline{U})$ is the source minimal minimum (U,T) cut in G.

We will show that $(U,\overline{U})$ is the source minimal minimum $(S\bigcup\{u_0\},T)$-terminal cut in G. We first show that $(U,\overline{U})$ is a minimum $(S\bigcup\{u_0\},T)$-terminal cut. Let $(X,\overline{X})$ be a minimum $(S\bigcup\{u_0\},T)$-terminal cut. Then,

$$d(U)\ge d(X)$$

because $(U,\overline{U})$ is a $(S\bigcup\{u_0\},T)$-terminal cut. Because $(X\cap U,\overline{X\cap U})$ is a $(S\bigcup\{u_0\},\overline{U})$-terminal cut, we have

$$d(X\cap U)\ge d(U).$$

Because $(X\bigcup U,\overline{X\bigcup U})$ is a (U,T)-terminal cut, we have

$$d(X\bigcup U)\ge d(U).$$

The above three inequalities in conjunction with the submodularity of the cut function imply that

$$2d(U)\ge d(X)+d(U)\ge d(X\cap U)+d(X\cup U)\ge 2d(U).$$

Hence, all of the above inequalities should be equations, and therefore, $d(U)=d(X)$.

Next, we show that $(U,\overline{U})$ is the source minimal minimum $(S\bigcup\{u_0\},T)$-terminal cut. For the sake of contradiction, suppose $(X,\overline{X})$ is the source minimal minimum $(S\bigcup\{u_0\},T)$-terminal cut with $X\ne U$. We have the following cases.

Case 1. Suppose $X\supsetneq U$. Then, $(U,\overline{U})$ contradicts source minimality of the minimum $(S\bigcup\{u_0\},T)$-terminal cut $(X,\overline{X})$.

Case 2. Suppose $X \subsetneq U$. Then, $(X, \overline{X})$ is also a minimum $(S \cup \{u_0\}, \overline{U})$-terminal cut, a contradiction since the choice of S implies that $(U, \overline{U})$ is unique minimum $(S \cup \{u_0\}, \overline{U})$-terminal cut.

Case 3. Suppose $X \backslash U \neq \emptyset$ and $X \backslash \overline{U} \neq \emptyset$. Then, we have

$$d(X \cap U) \geq d(X)$$

because $(X \cap U, \overline{X \cap U})$ is a $(S \cup \{u_0\}, T)$-cut. We also have

$$d(X \cup U) \geq d(X)$$

because $(X \cup U, \overline{X \cup U})$ is a $(S \cup \{u_0\}, T)$-cut. The above two inequalities in conjunction with the submodularity of the cut function imply that

$$2d(X) = d(X) + d(U) \geq d(X \cap U) + d(X \cup U) \geq 2d(X).$$

Therefore, $d(X \cap U) = d(X)$. Thus, the 2-partition $(X \cap U, \overline{X \cap U})$ contradicts source minimality of the minimum $(S \cup \{u_0\}, T)$-terminal cut $(X, \overline{X})$. □

5.1. Proof of Theorem 5

We restate and prove Theorem 5 in this section.

Theorem 5. *Let $G = (V, E)$ be a hypergraph, and let OPT_k be the value of a minimum k-cut in G for some integer $k \geq 2$. Suppose $(U, \overline{U})$ is a 2-partition of V with $d(U) \leq OPT_k$. Then, there exists a subset $S \subseteq U$ with $|S| \leq 2k - 2$ such that $(U, \overline{U})$ is the source maximal minimum $(S, \overline{U})$-terminal cut in G.*

Proof. For the sake of contradiction, suppose that the theorem is false. Our proof strategy is to obtain a cheaper k-partition with cost strictly less than OPT_k, thereby contradicting optimality. For a subset $X \subseteq U$, let $(V_X, \overline{V_X})$ be the source maximal minimum $(X, \overline{U})$-terminal cut.

Let X be an arbitrary subset of U with $|X| = 2k - 2$. Because we are assuming that the theorem is false, it follows that $V_X \neq U$. By definition, we have that $V_X \subsetneq U$. By source maximality of the minimum $(X, \overline{U})$-terminal cut $(V_X, \overline{V_X})$, we have that $d(V_X) < d(U)$.

Among all possible subsets of U of size $2k - 2$, pick a subset S such that $d(V_S)$ is maximum. Then, $V_S \subsetneq U$ and

$$d(V_X) \leq d(V_S) < d(U) \text{ for every } X \subseteq U \text{ with } |X| = 2k - 2.$$

The rest of the proof is identical to the proof of Theorem 2. Let $u_1, \ldots, u_{2k-2}$ be the vertices in S. Since $V_S \subsetneq U$, there exists a vertex $u_{2k-1} \in U \backslash V_S$. Let $C := \{u_1, \ldots, u_{2k-1} = S \cup \{u_{2k-1}\}$. Also, let $(B_i, \overline{B_i})$ be the source maximal minimum $(C - \{u_i\}, \overline{U})$-terminal cut for every $i \in [2k - 1]$. We note that $(B_{2k-1}, \overline{B_{2k-1}}) = (V_S, \overline{V_S})$, and the size of $C - \{u_i\}$ is $2k - 2$ for every $i \in [2k - 1]$. Hence, we have

$$d(B_i) \leq d(V_S) < d(U) \text{ and } B_i \subsetneq U \text{ for every } i \in [2k - 1]. \tag{4}$$

The next claim will set us up to apply Theorem 4.

Claim 5. For every $i \in [2k - 1]$, we have that $u_i \in \overline{B_i}$.

Proof. The claim holds for $i = 2k - 1$ by choice of u_{2k-1}. For the sake of contradiction, suppose $u_i \in B_i$ for some $i \in [2k - 2]$. Then, the 2-partition $(V_S \cap B_i, \overline{V_S \cap B_i})$ is a $(S, \overline{U})$-terminal cut, and hence,

$$d(V_S \cap B_i) \geq d(V_S).$$

We also have

$$d(V_S \cup B_i) \geq d(V_S)$$

because $(V_S \cup B_i, \overline{V_S \cup B_i})$ is a $(S, \overline{U})$-terminal cut. Thus,

$$\begin{aligned} 2d(V_S) &\geq d(V_S) + d(B_i) && \text{(By choice of } S\text{)} \\ &\geq d(V_S \cup B_i) + d(V_S \cap B_i) && \text{(By submodularity)} \\ &\geq 2d(V_S). \end{aligned}$$

Therefore, $d(V_S) = d(V_S \cup B_i)$. Moreover, $B_i \backslash V_S$ is nonempty because the vertex $u_{2k-1} \in B_i \backslash V_S$. Hence, the 2-partition $(V_S \cup B_i, \overline{V_S \cup B_i})$ is a minimum $(S, \overline{U})$-terminal cut, and it contradicts source maximality of the minimum $(S, \overline{U})$-terminal cut $(V_S, \overline{V_S})$. □

Let $p := 2k-2$. Using Claim 5, we observe that the sets U, $R := \{u_{2k-1}\}$, $S = \{u_1, \dots, u_{2k-2}\}$, and the partitions $(\overline{A_i}, A_i) := (B_i, \overline{B_i})$ for $i \in [p]$ satisfy the conditions of Theorem 4. By Theorem 4, symmetry of the cut function, and statement (4), we obtain a k-partition $(P_1, \dots, P_k)$ of V such that

$$\begin{aligned}\text{cost}(P_1, \dots, P_k) &\le \frac{1}{2}\min\{d(A_i) + d(A_j) : i, j \in [p], i \neq j\}\\ &= \frac{1}{2}\min\{d(B_i) + d(B_j) : i, j \in [p], i \neq j\}\\ &< d(U)\\ &\le OPT_k.\end{aligned}$$

Thus, we have obtained a k-partition whose cost is smaller than OPT_k, a contradiction. □

5.2. Proof of Theorem 6

We restate and prove Theorem 6 in this section.

Theorem 6. *Let $G = (V, E)$ be a hypergraph, and let $(U, \overline{U})$ be a minimal balanced minimum k-partition split in G for some integer $k \ge 2$. Then, for every vertex $u_0 \in U$, there exists a subset $S \subseteq U \backslash \{u_0\}$ with $|S| \le 2k-3$ such that $(U, \overline{U})$ is the* unique *minimum $(S \cup \{u_0\}, \overline{U})$-terminal cut in G.*

Proof. Let $u_0 \in U$, and let OPT_k be the value of a minimum k-cut in G. Consider the collection

$$\mathcal{C} := \{Q \subseteq V \backslash \{u_0\} : \overline{U} \subsetneq Q, d(Q) \le d(U)\}.$$

Let $S \subseteq U \backslash \{u_0\}$ be an inclusion-wise minimal transversal of the collection $\mathcal{C}$; that is, S is an inclusion-wise minimal subset of $U \backslash \{u_0\}$ such that $S \cap Q \neq \emptyset$ for all $Q \in \mathcal{C}$. Proposition 2 and Lemma 5 complete the proof of the theorem for this choice of S. □

Proposition 2. *The 2-partition $(U, \overline{U})$ is the unique minimum $(S \cup \{u_0\}, \overline{U})$-terminal cut in G.*

Proof. For the sake of contradiction, suppose $(X, \overline{X})$ is a minimum $(S \cup \{u_0\}, \overline{U})$-terminal cut in G such that $X \neq U$. Then, $d(\overline{X}) \le d(U)$ because $(U, \overline{U})$ is a feasible $(S \cup \{u_0\}, \overline{U})$-terminal cut. By definition, $\overline{U} \subsetneq \overline{X} \subseteq V \backslash \{u_0\}$. Hence, the set $\overline{X}$ is in the collection $\mathcal{C}$. Because S is a transversal of the collection $\mathcal{C}$, we have that $S \cap \overline{X} \neq \emptyset$. This contradicts the fact that S is contained in X. □

Lemma 5. *The size of the transversal S is at most $2k-3$.*

Proof. For the sake of contradiction, suppose $|S| \ge 2k-2$. We will construct a balanced minimum k-partition split in G that contradicts the minimality of the balanced minimum k-partition split $(U, \overline{U})$. Let $S = \{u_1, \dots, u_p\}$ for $p \ge 2k-2$. For each $i \in [p]$, let $(\overline{A_i}, A_i)$ be the source minimal minimum $((S \cup \{u_0\}) \backslash \{u_i\}, \overline{U})$-terminal cut.

Claim 6. *For every $i \in [p]$, we have that $d(A_i) \le d(U)$ and $u_i \in A_i$.*

Proof. Let $i \in [p]$. Because S is a minimal transversal for the collection $\mathcal{C}$, there exists a set $B_i \in \mathcal{C}$ such that $B_i \cap S = \{u_i\}$. Hence, $(\overline{B_i}, B_i)$ is a feasible $(S \cup \{u_0\} \backslash \{u_i\}, \overline{U})$-terminal cut. Therefore,

$$d(A_i) \le d(B_i) \le d(U).$$

We will show that A_i is in the collection $\mathcal{C}$. By definition, $A_i \subseteq V \backslash \{u_0\}$ and $\overline{U} \subseteq A_i$. If $\overline{U} = A_i$, then the above inequalities are equations implying that $(B_i, \overline{B_i})$ is a minimum $((S \cup \{u_0\}) \backslash \{u_i\}, \overline{U})$-terminal cut, and consequently, $(B_i, \overline{B_i})$ contradicts source minimality of the minimum $((S \cup \{u_0\}) \backslash \{u_i\}, \overline{U})$-terminal cut $(\overline{A_i}, A_i)$. Therefore, $\overline{U} \subsetneq A_i$. Hence, A_i is in the collection $\mathcal{C}$.

We recall that the set S is a transversal for the collection $\mathcal{C}$, and none of the elements of $S \backslash \{u_i\}$ are in A_i. Hence, the element u_i must be in A_i. □

Using Claim 6, we observe that the sets U, $R := \{u_0\}$, S, and the partitions $(\overline{A_i}, A_i)$ for $i \in [p]$ satisfy the conditions of Theorem 4. By Theorem 4 and Claim 6, we obtain k-partition $(P_1, \dots, P_k)$ of V such that $\overline{U} \subsetneq P_k$ and

$$\text{cost}(P_1, \dots, P_k) \le \frac{1}{2}\min\{d(A_i) + d(A_j) : i, j \in [p], i \neq j\} \le d(U) \le OPT_k.$$

Thus, we have obtained a minimum k-partition $(P_1, \dots, P_k)$ such that $\overline{U} \subsetneq P_k$. Now, consider $U' := \cup_{i=1}^{\lfloor k/2 \rfloor} P_i$. We observe that $(U', \overline{U'})$ is a balanced minimum k-partition split such that U' is strictly contained in U, a contradiction to minimality of the balanced minimum k-partition split $(U, \overline{U})$. □

6. Divide-and-Conquer Algorithm

In this section, we design a faster algorithm based on divide and conquer with a runtime of $n^{O(k)}$ source minimal minimum (S,T)-terminal cut computations. We describe the algorithm in Figure 8 and its runtime guarantee in Theorem 8. We recall that the source maximal minimum (s,t)-terminal cut in a hypergraph can be computed in deterministic polynomial time. To recap from the introduction, the high-level idea is to use minimum (S,T)-terminal cuts to find a balanced minimum k-partition split $(U,\overline{U})$; the balance helps in cutting the recursion depth that results in savings in the overall runtime.

Theorem 8. *Let $G=(V,E)$ be a n-vertex hypergraph of size p, and let k be an integer. Then, algorithm DIVIDE-AND-CONQUER-CUT(G, k) given in Figure 8 returns a partition corresponding to a minimum k-cut in G, and it can be implemented to run in $O(n^{8k}T(n,p))$ time, where T(n, p) denotes the time complexity for computing the source minimal minimum (S, T)-terminal cut in a n-vertex hypergraph of size p.*

Proof. We first show the correctness of the algorithm. All candidates considered by the algorithm correspond to a k-partition, so we only have to show that the algorithm returns a k-partition corresponding to a minimum k-cut. We show this by induction on k. The base case of $k=1$ is trivial. We show the induction step. Let $(P_1,\ldots,P_k)$ be a minimum k-partition in G such that for $p=\lfloor k/2\rfloor$, the 2-partition $(U_0 := \cup_{i=1}^{p} P_i, \overline{U_0}=\cup_{i=p+1}^{k} P_i)$ is a minimal balanced minimum k-partition split. Let OPT_k denote the value of a minimum k-partition in G.

We observe that $|U_0|\ge p$ and $|\overline{U_0}|\ge k-p$. By Theorem 7, the 2-partition $(U_0,\overline{U_0})$ is in $\mathcal{R}$. By induction hypothesis, the algorithm will return a p-partition $\mathcal{P}_{U_0}=(Q_1,\ldots,Q_p)$ of U_0 and a $(k-p)$-partition $\mathcal{P}_{\overline{U_0}}=(Q_{p+1},\ldots,Q_k)$ of $\overline{U_0}$ such that

$$\begin{aligned}\mathrm{cost}_{G[U_0]}(Q_1,\ldots,Q_p) &\le \mathrm{cost}_{G[U_0]}(P_1,\ldots,P_p) \text{ and}\\ \mathrm{cost}_{G[\overline{U_0}]}(Q_{p+1},\ldots,Q_k) &\le \mathrm{cost}_{G[U_0]}(P_{p+1},\ldots,P_k).\end{aligned}$$

Hence, the cost of the partition $(Q_1,\ldots,Q_k)$ returned by the algorithm is

$$\begin{aligned}&d(U_0)+\mathrm{cost}_{G[U_0]}(Q_1,\ldots,Q_p)+\mathrm{cost}_{G[\overline{U_0}]}(Q_{p+1},\ldots,Q_k)\\ &\le d(U_0)+\mathrm{cost}_{G[U_0]}(P_1,\ldots,P_p)+\mathrm{cost}_{G[\overline{U_0}]}(P_{p+1},\ldots,P_k)\\ &=\mathrm{cost}_G(P_1,\ldots,P_k)\\ &=OPT_k.\end{aligned}$$

Next, we prove the runtime bound. We will derive an upper bound $N(k,n)$ on the number of source minimal minimum (S,T)-terminal cut computations executed by the algorithm, where we assume that $N(k,n)$ is an increasing function of k and n. We know that $N(1,n)=O(1)$. We have

$$N(k,n)=O(n^{4k-4})\left(1+N\left(\left\lceil\frac{k}{2}\right\rceil,n\right)+N\left(\left\lfloor\frac{k}{2}\right\rfloor,n\right)\right).$$

By substitution, it can be verified that $N(k,n)=O(n^{8k})$. The running time is dominated by the number of terminal cut computations, and this yields the desired time bound on the algorithm. □

Figure 8. Divide-and-conquer algorithm to compute minimum k-cut hypergraphs.

Algorithm DIVIDE-AND-CONQUER-CUT(G,k)
Input: Hypergraph $G=(V,E)$ and an integer $k\ge 1$
Output: A k-partition corresponding to a minimum k-cut in G
If $k=1$
  Return V
Initialize $\mathcal{R}\leftarrow\emptyset$ and $p\leftarrow\lfloor k/2\rfloor$
For every disjoint $S,T\subset V$ with $|S|,|T|\le 2k-2$
  Compute the source minimal minimum (S,T)-terminal cut $(U,\overline{U})$
  If $|U|\ge p$ and $\overline{U}\ge k-p$
    $\mathcal{R}\leftarrow\mathcal{R}\cup\{(U,\overline{U})\}$
    $\mathcal{P}_U :=$ DIVIDE-AND-CONQUER-CUT$(G[U],p)$
    $\mathcal{P}_{\overline{U}} :=$ DIVIDE-AND-CONQUER-CUT$(G[\overline{U}],k-p)$
    $C_U :=$ Partition of V obtained by concatenating the parts in $\mathcal{P}_U$ and $\mathcal{P}_{\overline{U}}$
Among all k-partitions C_U with $(U,\overline{U})\in\mathcal{R}$, pick the one with minimum cost and return it

7. Concluding Remarks

Our work generalizes the approach pioneered by Goldschmidt and Hochbaum [16, 17] for Graph-k-Cut to Hypergraph-k-Cut with new insights along the way. The main open problem is to resolve the complexity of Submod-k-Part and Sym-Submod-k-Part for fixed k. We note that there is a simple reduction of Submod-k-Part to the special case of monotone submodular functions. The complexity of the special case of Sym-Submod-k-Part where the input submodular function is a hypergraph cut function, termed as Hypergraph-k-Partition, is also open for any fixed $k \geq 5$. A natural approach to attack these problems is via minimum S-T terminal cuts, like we did in this work. Although this approach is able to address some special cases, such as the rank function of matroids, as well as other objectives for submodular k-partition [3], there are limitations when applying directly to Submod-k-Part. Some of these results and observations will appear in a future article. The approximability of these partitioning problems when k is part of the input is also an interesting avenue of research, and we refer the reader to Chekuri and Li [7] and Santiago [38] for pointers.

Random contraction-based algorithms for Hypergraph-k-Cut [5] also yield, as a corollary, that there are $O(n^{2k-2})$ distinct hyperedge subsets that cross some optimum k-cut, and moreover, these can be enumerated in randomized polynomial time with high probability. We note that the number of distinct k-partitions that are optimal can be exponential in n even for $k = 2$; a simple example is a hypergraph with a single spanning hyperedge. Our algorithmic approach is based on partitions, and hence, it does not yield a deterministic algorithm for enumerating all hyperedge subsets that cross some optimum k-cut. In an upcoming work, Beideman et al. [1], strengthened Theorem 4 and used it to obtain an algorithm that enumerates all hyperedge sets that cross some optimum k-cut in deterministic polynomial time.

In a companion paper [3] following this work, we have used the minimum (S,T)-terminal cut approach to also solve the min-max symmetric submodular k-partition problem. The input here is a finite ground set V, a symmetric submodular function $f : 2^V \to \mathbb{R}$ (provided by an evaluation oracle) and a fixed integer k, and the goal is to partition V into k nonempty parts $V_1, \ldots, V_k$ to minimize $\max_{i=1}^k f(V_i)$. We note that if f is not symmetric, then the problem is NP-hard even for $k = 2$.

Acknowledgments

The authors thank Kristóf Bérczi, Tamás Király, and Chao Xu for helpful discussions on alternative approaches for deterministic Hypergraph-k-Cut during preliminary stages of this work. The authors thank Sagemath (www.sagemath.org) and CoCalc (www.cocalc.com) for providing software platforms to conduct experiments that led us toward our main structural theorem. The authors also thank the reviewers for careful reading and pointing out gaps in our arguments that helped improve the presentation of this work. A preliminary version of this work appeared at the *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020.

Endnotes

[1] Sym-Submod-k-Part when the input function f is the cut function of a hypergraph is known as Hypergraph-k-Partition in the literature [34, 43]. We emphasize that the objective in Hypergraph-k-Partition is different from the objective in Hypergraph-k-Cut.

[2] In min-max graph k-cut, the input is a graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_+$ and an integer k, and the goal is to partition the vertex set V into k nonempty parts $V_1, \ldots, V_k$ to minimize $\max_{i=1}^k w(\delta(V_i))$.

References

[1] Beideman C, Chandrasekaran K, Wang W (2021) Deterministic enumeration of all minimum k-cut-sets in hypergraphs for fixed k. Preprint, submitted October 27, https://arxiv.org/abs/2110.14815v2.
[2] Buchbinder N, Schwartz R, Weizman B (2019) A simple algorithm for the multiway cut problem. *Oper. Res. Lett.* 47(6):587–593.
[3] Chandrasekaran K, Chekuri C (2021) Min-max partitioning of hypergraphs and symmetric submodular functions. *Proc. 32nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1026–1038.
[4] Chandrasekaran K, Wang W (2021) Fixed parameter approximation scheme for min-max k-cut. *Integer Programming and Combinatorial Optimization.* IPCO, 354–367.
[5] Chandrasekaran K, Xu C, Yu X (2021) Hypergraph k-cut in randomized polynomial time. *Math. Programming* 186:85–113.
[6] Chekuri C, Ene A (2011) Approximation algorithms for submodular multiway partition. *Proc. 52nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, 807–816.
[7] Chekuri C, Li S (2020) On the hardness of approximating the k-way hypergraph cut problem. *Theory Comput.* 16(14):1–8.
[8] Chekuri C, Xu C (2018) Minimum cuts and sparsification in hypergraphs. *SIAM J. Comput.* 47(6):2118–2156.
[9] Chekuri C, Quanrud K, Xu C (2020) LP relaxation and tree packing for minimum k-cut. *SIAM J. Discrete Math.* 34(2):1334–1353.
[10] Dahlhaus E, Johnson D, Papadimitriou C, Seymour P, Yannakakis M (1994) The complexity of multiterminal cuts. *SIAM J. Comput.* 23(4): 864–894.
[11] Downey R, Estivill-Castro V, Fellows M, Prieto E, Rosamund F (2003) Cutting up is hard to do: The parameterised complexity of k-cut and related problems. *Electron. Notes Theor. Comput. Sci.* 78:209–222.

[12] Ene A, Vondrák J, Wu Y (2013) Local distribution and the symmetry gap: Approximability of multiway partitioning problems. *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 306–325.
[13] Fox K, Panigrahi D, Zhang F (2019) Minimum cut and minimum *k*-cut in hypergraphs via branching contractions. *Proc. 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 881–896.
[14] Fukunaga T (2013) Computing minimum multiway cuts in hypergraphs. *Discrete Optim.* 10(4):371–382.
[15] Ghaffari M, Karger D, Panigrahi D (2017) Random contractions and sampling for hypergraph and hedge connectivity. *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1101–1114.
[16] Goldschmidt O, Hochbaum D (1988) Polynomial algorithm for the *k*-cut problem. *Proc. 29th Annual Symposium on Foundations of Computer Science*, 444–451.
[17] Goldschmidt O, Hochbaum D (1994) A polynomial algorithm for the *k*-cut problem for fixed *k*. *Math. Oper. Res.* 19(1):24–37.
[18] Guinez F, Queyranne M (2012) The size-constrained submodular k-partition problem, unpublished manuscript. https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxmbGF2aW9ndWluZXpob21lcGFnZXxneDo0NDVlMThkMDg4ZWRlOGI1. See also https://smartech.gatech.edu/bitstream/handle/1853/43309/Queyranne.pdf.
[19] Gupta A, Lee E, Li J (2018) An FPT algorithm beating 2-approximation for *k*-cut. *Proc. 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2821–2837.
[20] Gupta A, Lee E, Li J (2020) The Karger-Stein algorithm is optimal for *k*-cut. *Proc. 52nd Annual ACM Symposium on Theory of Computing (STOC)*, 473–484.
[21] Gupta A, Harris D, Lee E, Li J (2020) Optimal bounds for the *k*-cut problem. Preprint, submitted May 17, https://arxiv.org/abs/2005.08301.
[22] Kamidoi Y, Yoshida N, Nagamochi H (2007) A deterministic algorithm for finding all minimum *k*-way cuts. *SIAM J. Comput.* 36(5):1329–1341.
[23] Karger D (1993) Global min-cuts in RNC, and other ramifications of a simple min-cut algorithm. *Proc. 4th Annual ACM-SIAM Sympos. Discrete Algorithms (SODA)*, 21–30.
[24] Karger D, Stein C (1996) A new approach to the minimum cut problem. *J. ACM.* 43(4):601–640.
[25] Kawarabayashi Ki, Lin B (2020) A nearly 5/3-approximation FPT algorithm for min-*k*-cut. *Proc. 31st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 990–999.
[26] Kawarabayashi K, Thorup M (2011) The minimum k-way cut of bounded size is fixed-parameter tractable. *Proc. 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, 160–169.
[27] Lawler E (1973) Cutsets and partitions of hypergraphs. *Networks* 3:275–285.
[28] Li J (2019) Faster minimum *k*-cut of a simple graph. *Proc. 60th Annual Symposium on Foundations of Computer Science (FOCS)*, 1056–1077.
[29] Lokshtanov D, Saurabh S, Surianarayanan V (2020) A parameterized approximation scheme for min cut. *Proc. 61st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*.
[30] Manurangsi P (2017) Almost-polynomial ratio ETH-hardness of approximating densest *k*-subgraph. *Proc. 49th Annual ACM Symposium on Theory of Computing (STOC)*, 954–961.
[31] Manurangsi P (2018) Inapproximability of maximum biclique problems, minimum *k*-cut and densest at-least-*k*-subgraph from the Small Set Expansion Hypothesis. *Algorithms* 11(1):1–10, preliminary version in *Proc. ICALP 2017*.
[32] Marx D (2006) Parameterized graph separation problems. *Theoret. Comput. Sci.* 351(3):394–406.
[33] Nagamochi H, Ibaraki T (1992) A linear-time algorithm for finding a sparse *k*-connected spanning subgraph of a *k*-connected graph. *Algorithmica* 7(1–6):583–596.
[34] Okumoto K, Fukunaga T, Nagamochi H (2012) Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. *Algorithmica* 62(3):787–806.
[35] Quanrud K (2019) Fast and deterministic approximations for *k*-cut. *Proc. Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX)*, 23:1–23:20.
[36] Queyranne M (1999) On optimum size-constrained set partitions. Talk at Aussiois workshop on Combinatorial Optimization.
[37] Raghavendra P, Steurer D (2010) Graph expansion and the unique games conjecture. *Proc. 42nd Annual ACM Symposium on Theory of Computing (STOC)*, 755–764.
[38] Santiago R (2019) Multi-agent submodular optimization: variations and generalizations. Ph.D. thesis, McGill University, Montreal.
[39] Saran H, Vazirani V (1995) Finding k cuts within twice the optimal. *SIAM J. Comput.* 24(1):101–108.
[40] Stoer M, Wagner F (1997) A simple min-cut algorithm. *J. ACM.* 44(4):585–591.
[41] Thorup M (2008) Minimum *k*-way cuts via deterministic greedy tree packing. *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, 159–166.
[42] Xiao M (2010) Finding minimum 3-way cuts in hypergraphs. *Inform. Proc. Lett.* 110(14):554–558, preliminary version in TAMC 2008.
[43] Zhao L, Nagamochi H, Ibaraki T (2005) Greedy splitting algorithms for approximating multiway partition problems. *Math. Programming* 102(1):167–183.