Min-max Partitioning of Hypergraphs and Symmetric Submodular Functions*

Karthekeyan Chandrasekaran Chandra Chekuri

Abstract

We consider the complexity of *minmax* partitioning of graphs, hypergraphs and (symmetric) submodular functions. Our main result is an algorithm for the problem of partitioning the ground set of a given *symmetric* submodular function $f : 2^V \to \mathbb{R}$ into k non-empty parts $V_1, V_2, \ldots, V_k$ to minimize $\max_{i=1}^k f(V_i)$. Our algorithm runs in $n^{O(k^2)}T$ time, where $n = |V|$ and T is the time to evaluate f on a given set; hence, this yields a polynomial time algorithm for any fixed k in the evaluation oracle model. As an immediate corollary, for any fixed k, there is a polynomial-time algorithm for the problem of partitioning a given hypergraph $H = (V, E)$ into k non-empty parts to minimize the maximum capacity of the parts. The complexity of this problem, termed MINMAX-HYPERGRAPH-k-PART, was raised by Lawler in 1973 [16]. In contrast to our positive result, the reduction in [6] implies that when k is part of the input, MINMAX-HYPERGRAPH-k-PART is hard to approximate to within an almost polynomial factor under the Exponential Time Hypothesis (ETH).

1 Introduction

Partitioning problems in graphs and hypergraphs are extensively studied for their applications and theoretical value. In this work, we consider the *minmax* objective. A hypergraph $G = (V, E)$ consists of a finite vertex set V and a collection of hyperedges where each hyperedge $e \in E$ is a subset of vertices, that is, $e \subseteq V$. If $|e| = 2$ for all $e \in E$, then the hypergraph is simply an undirected graph. The input to minmax hypergraph k-partitioning is a hypergraph $G = (V, E)$ with non-negative hyperedge weights $w : E \to \mathbb{R}_+$ and an integer k. The goal is to partition V into *non-empty* sets $V_1, V_2, \ldots, V_k$ to minimize $\max_{i=1}^k w(\delta(V_i))$; here $\delta(V_i)$ is the set of hyperedges crossing[1] V_i and $w(\delta(V_i)) = \sum_{e\in\delta(V_i)} w(e)$ is the total weight of the hyperedges in $\delta(V_i)$. We refer to this problem as MINMAX-HYPERGRAPH-k-PART. We refer to the special case when G is a graph as MINMAX-GRAPH-k-PART. Closely related to these problems are GRAPH-k-CUT, HYPERGRAPH-k-CUT and HYPERGRAPH-k-PART that we will discuss later. The complexity of MINMAX-HYPERGRAPH-k-PART was raised as early as 1973 in Lawler's work on hypergraph mincut [16], and has remained open. In this work, we show that MINMAX-HYPERGRAPH-k-PART has a polynomial-time algorithm for any fixed constant k.

THEOREM 1.1. MINMAX-HYPERGRAPH-k-PART *has a polynomial-time algorithm for any fixed k. In particular, there is an algorithm that runs in time $n^{O(k^2)}m$, where n is the number of nodes and m is the number of hyperedges.*

In contrast to the preceding positive result, when k is part of the input, one can easily show that the reduction in [6] that proves conditional hardness of HYPERGRAPH-k-CUT also applies to MINMAX-HYPERGRAPH-k-PART; this was observed in [4]. Consequently, under the Exponential Time Hypothesis (ETH) there is no $n^{1/(\log\log n)^c}$-approximation for MINMAX-HYPERGRAPH-k-PART for some absolute constant c. Our algorithmic result in Theorem 1.1, of course, also applies to the special case of MINMAX-GRAPH-k-PART. We will later point out that an alternative algorithm for MINMAX-GRAPH-k-PART can be obtained from previous results on GRAPH-k-CUT while it is not the case for hypergraphs.

Several results on graphs and hypergraphs rely on *submodularity* of their cut function. We recall that a real-valued set function $f : 2^V \to \mathbb{R}$ is *submodular* if $f(A \cup B) + f(A \cap B) \le f(A) + f(B)$ for all $A, B \subseteq V$ and is *symmetric* if $f(A) = f(V \setminus A)$ for all $A \subseteq V$. The cut function of a hypergraph is symmetric and submodular when the hyperedge weights are non-negative. Our algorithm for hypergraphs is a special case of our more general result on minmax partitioning of symmetric submodular functions. In this problem, the input is a finite

*University of Illinois, Urbana-Champaign. Email: {karthe, chekuri}@illinois.edu. Supported in part by NSF grant CCF-1907937.

[1]A hyperedge e crosses $S \subseteq V$ if $e\cap S \neq \emptyset$ and $e\cap(V\setminus S) \neq \emptyset$.

ground set V, a symmetric submodular function f (provided by an evaluation oracle[2]) and an integer k. The goal is to partition V into k non-empty parts $V_1, \ldots, V_k$ to minimize $\max_{i=1}^k f(V_i)$. We refer to this problem as MINMAX-SYMSUBMOD-k-PART and observe that MINMAX-HYPERGRAPH-k-PART is a special case. MINMAX-SUBMOD-k-PART refers to the problem when f is submodular (but not necessarily symmetric). MINMAX-SUBMOD-k-PART is NP-Hard even for $k = 2$.[3] However we show that MINMAX-SYMSUBMOD-k-PART is polynomial-time solvable for any fixed k.

THEOREM 1.2. MINMAX-SYMSUBMOD-k-PART *has a polynomial-time algorithm for any fixed* k. *In particular, there is an algorithm that runs in time* $n^{O(k^2)}T$, *where* n *is the size of the ground set and* T *is the time to evaluate the input function* f *on a given set.*

We note that the preceding theorem does not require the input function to be non-negative. This is not surprising since we can add a large positive constant to the function to make it non-negative without violating submodularity and symmetry and an optimum solution to the shifted function yields an optimum solution to the original function.

When k is part of the input, MINMAX-SYMSUBMOD-k-PART inherits the hardness of approximation of MINMAX-HYPERGRAPH-k-PART that we already mentioned. One can also easily obtain a $2k$-approximation for MINMAX-SYMSUBMOD-k-PART when f is non-negative.

1.1 Motivation and related problems

Given a real-valued set function $f : 2^V \to \mathbb{R}$ and a partition $V_1, \ldots, V_k$ of V, one can measure the quality of the partition in various natural ways. Two natural measures are $\max_{i=1}^k f(V_i)$ and $\sum_{i=1}^k f(V_i)$. Once a measure is defined, a corresponding optimization problem arises where one seeks to find a partition that minimizes the measure (we can also consider maximizing the measure but the focus of this paper is on minimizing the measure).

Minmax objective is particularly useful in load-balancing scenarios. Consider the classical MULTIPROCESSOR SCHEDULING problem of assigning n jobs with given real-valued processing times $p_1, \ldots, p_n$ to k machines to minimize the maximum load. This can be easily cast as a special case of MINMAX-SUBMOD-k-PART where the function f is the modular function p defined by $p(S) = \sum_{i \in S} p_i$; this special case is NP-Hard even for $k = 2$ via a reduction from 2-PARTITION. Motivated by such load balancing problems and considerations, several problems have been considered in algorithms literature. Svitkina and Tardos [23] introduced the minmax version of the multiway cut problem (MINMAX-MULTIWAY-CUT) motivated by applications in networking: the input is an edge-weighted graph $G = (V, E)$ and k terminals $\{s_1, s_2, \ldots, s_k\} \subseteq V$, and the goal is to partition V into k parts $V_1, \ldots, V_k$ such that $s_i \in V_i$ for all $i \in [k]$ so as to minimize $\max_{i=1}^k w(\delta(V_i))$. They showed that it is NP-hard for $k = 4$ and gave a poly-logarithmic approximation for arbitrary k. This was subsequently improved by Bansal, Feige, Krauthgamer, Makarychev, Nagarajan, Naor, and Schwartz [1] who obtained an $O(\sqrt{\log n \log k})$ approximation. Bansal et al. also considered related problems where there are additional balance constraints on the number of vertices in each part; we refer the reader to their paper for more details. In addition, they showed a negative result that suggests that an approximation better than $k^{1-\epsilon}$ (without a dependence on n) is unlikely. We note that MINMAX-GRAPH-k-PART differs from MINMAX-MULTIWAY-CUT in that no terminals are specified in the former problem.

Svitkina and Fleischer [22] considered SUBMOD-LOAD-BALANCING which is the restriction of MINMAX-SUBMOD-k-PART to *monotone*[4] submodular functions—monotonicity is natural in some applications. They showed that, when k is part of the input, SUBMOD-LOAD-BALANCING is hard to approximate to within an $o(\sqrt{n/\log n})$-factor unless the algorithm makes exponential number of queries to the function evaluation oracle. They also describe an $O(\sqrt{n \log n})$ approximation. The approximability of the problem when k is a fixed constant appears to be open.

The minmax objective for submodular functions has also been investigated among other objectives in machine learning applications from an empirical perspective [25].

[2] An evaluation oracle for a set function f over a ground set V returns the value of $f(S)$ given $S \subseteq V$.

[3] MINMAX-SUBMOD-k-PART for $k = 2$ is NP-hard by reduction from 2-PARTITION. However, it is an interesting exercise to the reader to see that MINMAX-SYMSUBMOD-k-PART for $k = 2$ reduces to submodular function minimization and is hence, solvable in polynomial time.

[4] A set function is *monotone* if $f(A) \leq f(B)$ for all $A \subseteq B$.

Minsum objective: The minmax objective has several important connections to the minsum objective in terms of motivation, problems, and techniques. The minsum objective in partition problems captures several well-known problems that we discuss now. The Graph-k-Cut problem is the following: given an undirected graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_+$, remove a minimum weight subset of edges so that the resulting graph has at least k connected components. One can also view this equivalently as a minsum partition problem where the goal is to partition V into k non-empty parts $V_1, \ldots, V_k$ to minimize $\sum_{i=1}^{k} w(\delta(V_i))$. There are two natural generalizations of Graph-k-Cut to hypergraphs based on these two viewpoints: (1) In the Hypergraph-k-Cut problem, one seeks to find a minimum weight subset of hyperedges of a given hyperedge-weighted hypergraph whose deletion leads to at least k non-empty connected components. (2) In the Hypergraph-k-Part problem, one seeks to find a k-partition $V_1, \ldots, V_k$ of the vertex set of a given hypergraph $G = (V, E)$ with hyperedge-weights $w : E \to \mathbb{R}_+$ to minimize $\sum_{i=1}^{k} w(\delta(V_i))$. In contrast to graphs, Hypergraph-k-Cut and Hypergraph-k-Part are not equivalent. One can consider generalizations of minsum problems to submodular functions leading to the Submod-k-Part problem: given a submodular function $f : 2^V \to \mathbb{R}$ over a ground set V and an integer k, the goal is to partition V into k non-empty parts $V_1, \ldots, V_k$ to minimize $\sum_{i=1}^{k} f(V_i)$. Sym-Submod-k-Part is the special case of Submod-k-Part when the input function f is symmetric submodular. One can easily see that Hypergraph-k-Part is a special case of Sym-Submod-k-Part while it takes a bit more work to see that Hypergraph-k-Cut is a special case of Submod-k-Part [18].

Graph-k-Cut has been extensively studied. It generalizes the global mincut problem and is non-trivial even when $k = 3$. Graph-k-Cut was shown to be polynomial-time solvable for any fixed k by Goldschmidt and Hochbaum [10]. The same work also showed NP-Hardness when k is part of the input. There have been several other algorithms including the random contraction approach of Karger and Stein [15], and the tree packing approach of Karger [14] and Thorup [24]. We refer the reader to [7, 12] for several recent results and additional pointers on Graph-k-Cut. In contrast, the complexity of Hypergraph-k-Cut was open until fairly recently. A randomized polynomial-time algorithm for any fixed k, based on random contraction, was first described by Chandrasekaran, Xu, and Yu [3] which was subsequently improved by Fox, Panigrahi, and Zhang [8]. A deterministic algorithm based on a generalization of the Goldschmidt-Hochbaum approach was given very recently by the authors of the current paper [2]. When k is part of the input, Graph-k-Cut admits a $2(1 - 1/k)$-approximation [20], and moreover conditional hardness results show that this is the best possible [17]. Chekuri and Li [6] show that Hypergraph-k-Cut is hard to approximate to within almost polynomial-factor under ETH.

The complexity of Submod-k-Part and Sym-Submod-k-Part when k is fixed are important open problems. Polynomial time algorithms are known for Submod-k-Part for $k \leq 3$ [18] and for Sym-Submod-k-Part for $k \leq 4$ [11].

As far as we are aware, no prior results existed on the worst-case complexity of the minmax partition problems that we study in this work. The minmax objective is in general more complex to handle as shown by negative results and prior work.

Minmax from Minsum objective: There is a useful connection between the minsum and the minmax objectives that we describe now: an α-approximation for Submod-k-Part implies an αk approximation for Minmax-Submod-k-Part when the underlying function f is non-negative. We sketch this argument: Suppose there is an optimum k-partition for the minmax objective with value B. Then the sum-objective value of the same partition is at most kB. Thus, an α-approximation for minsum yields a partition whose sum-objective value is at most αkB and this partition has max-objective value at most αkB (since f is non-negative).

The above-mentioned connection also leads to an $n^{O(k^2)}$-time algorithm for Minmax-Graph-k-Part as follows: Suppose $\mathcal{P}$ is an optimum k-partition for Minmax-Graph-k-Part on the given graph with optimum minmax objective value being B. Then, the optimum minsum objective value is at least B. Moreover, $\mathcal{P}$ has sum-objective value at most kB. Thus, if we can enumerate all k-approximate solutions to the minsum objective, then one of them will have max-objective value at most B. In graphs, we can indeed enumerate all β-approximate minsum k-partitions in time $n^{O(\beta k)}$ [15]. So, we can get an optimum partition for Minmax-Graph-k-Part by choosing the best among the k-approximate optimum solutions to Graph-k-Cut, which would take $n^{O(k^2)}$-time. However, this approach does not extend to hypergraphs or general symmetric submodular functions.

Gomory-Hu tree and symmetric submodular functions. An important structural property of symmetric submodular functions is that they admit a *Gomory-Hu tree* and it can be found efficiently [21]. A *Gomory-Hu tree* for a symmetric submodular function $f : 2^V \to \mathbb{R}$ is a tree $T = (V, E)$ such that for every tree-edge $st \in E$, the partition $(A, V \setminus A)$ is a minimum (s,t)-terminal cut (with respect to f), where A is a component in $T - st$. In algorithmic applications, one often endows the tree T with edge weights $w : E \to \mathbb{R}$ given by $w(st) = f(A)$, where $st \in E$ and A is a component in $T - st$. The existence of Gomory-Hu trees provide a unified explanation for efficient solvability/approximability of certain partitioning problems for symmetric submodular functions (e.g., efficient solvability of T-odd cut, 2-approximation for SYM-SUBMOD-k-PART, etc.)—all these algorithms construct the Gomory-Hu tree with edge weights and solve the problem of interest on the resulting edge-weighted tree, which tends to be substantially simpler owing to the tree structure. However, we note that Gomory-Hu trees may carry no information for MINMAX-SYMSUBMOD-k-PART. E.g., consider MINMAX-GRAPH-k-PART for the complete graph K_n on n vertices: the optimal value is $(n-k+1)(k-1)$. The natural approach of finding a Gomory-Hu tree and solving the problem on the tree fails even for this example: the star graph on n vertices with all edge weights being $n-1$ is a Gomory-Hu tree for K_n. The optimum value of MINMAX-GRAPH-k-PART for this tree is $(n-1)(k-1)$.

1.2 Technical overview and main structural result

Our algorithm for MINMAX-SYMSUBMOD-k-PART is inspired, at a high-level, by the work of Goldschmidt and Hochbaum on GRAPH-k-CUT [10]. Their approach has been subsequently refined and applied with additional ideas to improve the running time for GRAPH-k-CUT [13,26], to obtain a deterministic polynomial-time algorithm for HYPERGRAPH-k-CUT [2], and to obtain a polynomial-time algorithm for SYM-SUBMOD-k-PART for $k = 4$ [11]. Our approach for MINMAX-SYMSUBMOD-k-PART also builds on the ideas of Goldschmidt and Hochbaum, so we briefly recall their ideas.

A key algorithmic tool in the approach of [10], as well as our approach here, is the use of terminal cuts. We need some notation. Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function over the ground set V. For subsets A and B of the ground set V, we will use $A - B$ to denote $A \setminus B$. For a subset U of the ground set V, we use $\overline{U}$ to denote $V - U$. The value of a 2-partition $(U, \overline{U})$ is $f(U)$. Let S, T be disjoint subsets of the ground set V. A 2-partition $(U, \overline{U})$ is an (S,T)-terminal cut if $S \subseteq U \subseteq V - T$. 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. Since there could be multiple minimum (S,T)-terminal cuts, we will be interested in the *source maximal* minimum (S,T)-terminal cut. There exists a unique source maximal minimum (S,T)-terminal cut and it can be found in polynomial-time if we are given evaluation access to the submodular function (by relying on submodular function minimization)—e.g., see [9].

The approach of Goldschmidt and Hochbaum [10] for GRAPH-k-CUT is the following (for unit-weights on the edges). For $S \subseteq V$, let $\delta(S)$ denote the set of edges crossing S in the input graph. Suppose $(V_1, V_2, \ldots, V_k)$ is an optimum minsum k-partition such that V_1 is the cheapest part (that is, $|\delta(V_1)| \leq |\delta(V_i)|$ for every $i \in [k]$), and V_1 is maximal subject to this condition. They show that one can identify V_1 via the following key structural theorem: either $|V_1| \leq k-1$ or there *exist* disjoint vertex subsets S, $T \subseteq V$ with $|S| \leq k-2$, $|T| = k-1$ so that the source maximal minimum (S,T)-terminal cut is $(V_1, \overline{V_1})$. Thus, one can guess/enumerate all pairs (S,T) of small sizes to find an $O(n^{2k})$-sized collection of sets containing V_1. This enables a simple recursive algorithm: For each set in the collection, we assume it is V_1 and recurse to find a cheapest $(k-1)$-partition in the graph $G[V \setminus V_1]$. This leads to an $n^{O(k^2)}$-time algorithm.

The proof of the key structural theorem in [10] is non-trivial and relies heavily on properties of the cut function of graphs. Queyranne [19] claimed that a natural generalization of the preceding structural theorem holds in the more general setting of symmetric submodular functions, namely for the problem of SYM-SUBMOD-k-PART which generalizes GRAPH-k-CUT. However, as reported in [11], the claimed proof was incorrect and it was only proved for $k = 3, 4$. A starting point for our work here is a proof of (a mild relaxation of) the claim of Queyranne for all k—see Theorem 4.1 in the conclusion section (the claim is not directly relevant to the present work, so we state it in the conclusion section). Our proof of his claim relies only on submodularity (and symmetry) and hence, it gives a conceptually clean proof of the original algorithmic approach of [10] for GRAPH-k-CUT.

Unfortunately, as noted in [11], this structural theorem does not lead to an algorithm for SYM-SUBMOD-k-PART because one cannot recurse on $V \setminus V_1$; the function f restricted to $V \setminus V_1$ may no longer be symmetric! However, the approach works for GRAPH-k-CUT fortuitously because in graphs, we can afford to work with the cut function of the subgraph $G[V \setminus V_1]$ as opposed to the original cut function restricted to $V \setminus V_1$.

Recall that we are actually interested in the minmax objective and we wish to handle the general setting of MINMAX-SYMSUBMOD-k-PART. For this objective we prove a structural theorem that is similar in spirit to that of the minsum objective but technically somewhat different. We state this structural theorem now.

We need some notation. We will denote a k-partition by an ordered tuple—it will be important to view it as an ordered tuple rather than a collection of k disjoint sets whose union is V. Given a k-partition $(V_1, V_2, \dots, V_k)$ of V, we denote

$$\text{cost}_f(V_1, V_2, \dots, V_k) := \max\{f(V_i) : i \in [k]\}.$$

A k-partition is a *minmax k-partition with respect to f* if it has the least cost among all possible k-partitions. We will drop the subscript f from the cost notation and avoid repeating the phrase "with respect to f" when the function f of interest is clear from context (the subscript and the phrasing will be needed primarily in Section 3). We will be interested in minmax k-partitions $(V_1, \dots, V_k)$ for which V_1 is maximal: formally, we define a minmax k-partition $(V_1, \dots, V_k)$ to be a *V_1-maximal minmax k-partition* if there is no other minmax k-partition $(V_1', \dots, V_k')$ such that V_1 is strictly contained in V_1'. The following is our main structural result.

THEOREM 1.3. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function and let $k \geq 2$ be an integer. Let $(V_1, \dots, V_k)$ be a V_1-maximal minmax k-partition with respect to f. Suppose $|V_1| \geq k-1$. Then, for every subset $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \dots, k\}$, there exists a subset $S \subseteq V_1$ of size $k-1$ such that $(V_1, \overline{V_1})$ is the source maximal minimum (S,T)-terminal cut.*

We emphasize a key feature of our structural result: it does not require V_1 to be a cheapest part among all parts of the optimum k-partition (in contrast to the structural result of Goldschmidt-Hochbaum for the minsum objective in graphs). Informally speaking, our structural result says that under maximality of V_1, there exist disjoint vertex sets $S, T \subseteq V$ with $|S|, |T| \leq k-1$ such that $(V_1, \overline{V_1})$ is the source-maximal minimum (S,T)-terminal cut. Thus, we can compute a collection of $n^{O(k)}$ candidate sets such that one of them is V_1.

The key feature of our structural theorem immediately implies that the problem can be solved in $(n^{O(k^2)} + n^{O(k)}T)$-time if there exists a *unique* minmax k-partition, say $V_1, \dots, V_k$ (i.e., if the only ordered tuple of minmax k-partitions are permutations of the ordered tuple $(V_1, \dots, V_k)$). For this, we note that the reordered k-partition $(A_1 := V_i, A_2 := V_1, \dots, A_i := V_{i-1}, A_{i+1} := V_{i+1}, \dots, A_k := V_k)$ is an A_1-maximal minmax k-partition due to uniqueness and hence, the theorem applies to this reordered k-partition. As a consequence, our candidate collection of $n^{O(k)}$ sets not only contains V_1, but also contains $V_2, \dots, V_k$. Hence, we can iterate over all possible k-tuples of the sets in the collection to compute their cost (if they form a k-partition of V) and return the cheapest k-partition among all. We emphasize that this approach fails if the optimum k-partition is not unique. Moreover, minmax objective tends to have multiple optimum solutions in general. We next discuss the case in which the input function has multiple optimum solutions.

It may now appear that once we have the structural theorem, we can remove V_1 (by trying all possible candidate sets) and recurse to find a minmax $(k-1)$-partition of f restricted to $V \setminus V_1$. As we remarked earlier, the function f when restricted to $V \setminus V_1$ is not symmetric, so we will not be able to apply the structural theorem in the next step already! Moreover, the minmax objective is not conducive to the removal of a part of an optimum k-partition (e.g., consider what happens when f is the graph cut function and we try to find V_2 after removing V_1 from the graph). We overcome these issues by *contracting* V_1 as opposed to removing it. Contracting V_1 to a singleton allows us to continue working with a symmetric submodular function. Now, our goal is to find a second *non-singleton* part in the optimum k-partition to make progress. We show that this is indeed possible. In order to do this we crucially rely on two aspects: (1) the fact that we are working with the minmax objective, and (2) the key feature of our main structural result that only relies on the maximality of V_1. The same approach extends inductively to enable us to find all parts of the optimum k-partition (see Section 3 for the complete algorithm and details).

As we saw, minmax submodular k-partition is

NP-Hard even for $k = 2$ for simple asymmetric functions (e.g., modular functions). Symmetry of the submodular function is a crucial ingredient in the proof of our structural result. Symmetric submodular functions are also *posi-modular*, i.e., they satisfy

$$f(A) + f(B) \geq f(A - B) + f(B - A) \ \forall\ A, B \subseteq V.$$

Posimodularity allows for certain uncrossing properties that have been exploited in past work [1, 5, 23] explicitly or implicitly. Another important ingredient in the proof of our structural result is a strengthening of an uncrossing lemma underlying a containment property from [18]—we use symmetry to strengthen their uncrossing lemma for the minmax objective (see Lemma 2.1). Our proof of Lemma 2.1 gives an alternate simpler proof of their uncrossing lemma for symmetric submodular functions.

Organization. We prove our main structural theorem in Section 2 and design the algorithm in Section 3. We conclude with some approaches for the minsum objective for symmetric submodular functions in Section 4.

2 Main Structural Theorem

In this section, we prove Theorem 1.3. The proof consists of two high-level steps:

1. In the first step, we show that for *any* $S \subseteq V_1$ and *any* $T \subseteq \overline{V_1}$ such that T intersects V_j for every $j \in \{2, \ldots, k\}$, a minimum (S, T) terminal cut $(U, \overline{U})$ satisfies the property that $U \subseteq V_1$. This containment property is captured in Lemma 2.1.

2. In the second step, we show that for any T as above, there exists a *small* set $S \subseteq V_1$ such that the source maximal minimum (S, T)-terminal cut will be $(V_1, \overline{V_1})$. We show that $|S| = k - 1$ suffices. The proof of this relies on an uncrossing property that is captured in Theorem 2.1.

2.1 Containment Property

We show the containment property in this section. The uncrossing underlying the proof of the containment property is inspired by an uncrossing due to Okumoto, Fukunaga, and Nagamochi for the minsum objective (Theorem 5 of [18]). Our proof exploits the symmetry of the input function to strengthen their uncrossing for the minmax objective.

LEMMA 2.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function, $k \geq 2$ be an integer, $(V_1, \ldots, V_k)$ be a V_1-maximal minmax k-partition with respect to f, and $S \subseteq V_1$, $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \ldots, k\}$. Suppose $(U, \overline{U})$ is a minimum (S, T)-terminal cut. Then, $U \subseteq V_1$.*

Proof. For the sake of contradiction, suppose $U \setminus V_1 \neq \emptyset$. Consider $W_1 := V_1 \cup U$ and $W_j := V_j - U$ for every $j \in \{2, \ldots, k\}$ (see Figure 1).

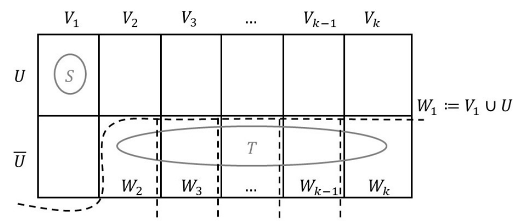

Figure 1: Uncrossing in the proof of Lemma 2.1.

Since $W_1 \supseteq S \neq \emptyset$ and $W_j \supseteq T \cap V_j \neq \emptyset$ for all $j \in \{2, \ldots, k\}$, we have that $(W_1, \ldots, W_k)$ is a k-partition. Claim 2.1 shows that the cost of this k-partition is at most that of $(V_1, \ldots, V_k)$. Hence, $(W_1, \ldots, W_k)$ is a minmax k-partition. Moreover, W_1 is a strict superset of V_1 as $U \setminus V_1 \neq \emptyset$ and hence, $(W_1, \ldots, W_k)$ contradicts V_1-maximality of the minmax k-partition $(V_1, \ldots, V_k)$. □

CLAIM 2.1. *For every $i \in [k]$, we have $f(W_i) \leq f(V_i)$.*

Proof. We distinguish two cases. Suppose $i = 1$. Then, $f(V_1 \cap U) \geq f(U)$ since $(V_1 \cap U, \overline{V_1 \cap U})$ is a (S, T)-terminal cut while $(U, \overline{U})$ is a minimum (S, T)-terminal cut. Hence, we have that

$$\begin{aligned} f(V_1) + f(U) &\geq f(V_1 \cup U) + f(V_1 \cap U) \\ &\qquad \text{(by submodularity)} \\ &\geq f(V_1 \cup U) + f(U). \end{aligned}$$

Consequently, $f(V_1) \geq f(V_1 \cup U) = f(W_1)$.

Next, suppose $i \in \{2, \ldots, k\}$. Then, $f(U - V_i) \geq f(U)$ since $(U - V_i, \overline{U - V_i})$ is a (S, T)-terminal cut while $(U, \overline{U})$ is a minimum (S, T)-terminal cut. Hence, we have that

$$\begin{aligned} f(V_i) + f(U) &\geq f(V_i - U) + f(U - V_i) \\ &\qquad \text{(by posimodularity)} \\ &\geq f(V_i - U) + f(U). \end{aligned}$$

Consequently, $f(V_i) \geq f(V_i - U) = f(W_i)$. □

2.2 Uncrossing Theorem Our next ingredient is an uncrossing theorem to obtain a cheap k-partition.

THEOREM 2.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function, $k \geq 2$ be an integer, and $\emptyset \neq U \subsetneq V$. Let $C = \{u_1, \ldots, u_k\} \subseteq U$. Let $(\overline{A_i}, A_i)$ be a minimum $(C \setminus \{u_i\}, \overline{U})$-terminal cut for every $i \in [k]$. Suppose that $u_i \in A_i \setminus (\cup_{j\in[k]\setminus\{i\}} A_j)$ for every $i \in [k]$ and $f(A_1) \leq f(A_2) \leq \ldots \leq f(A_k)$. Then, there exists a k-partition $(P_1, \ldots, P_k)$ of V such that*

$$f(P_i) \leq f(A_i) \ \forall\ i \in [k].$$

In particular $cost_f(P_1, \ldots, P_k) \leq \max\{f(A_i) : i \in [k]\}$.

Proof. See Figure 2 for an illustration of the sets that appear in the statement of the theorem.

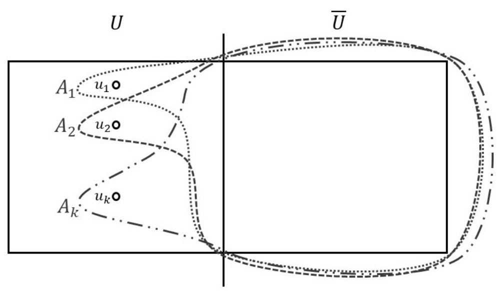

Figure 2: Illustration of the sets that appear in Theorem 2.1.

We begin with the following uncrossing claim showing that there exists a cheap $(k-1)$-partition of $\cup_{i=1}^{k-1} A_i$ (cheap in the sense that the function value of every part is small). Its proof relies on posimodularity and it has appeared implicitly (for the graph cut function) in previous works [1, 23].

CLAIM 2.2. *There exist subsets $P_1, \ldots, P_{k-1}$ of V such that*

(i) P_i and P_j are disjoint for every distinct $i, j \in [k-1]$,

(ii) $u_i \in P_i$ for every $i \in [k-1]$,

(iii) $\cup_{i=1}^{k-1} P_i = \cup_{i=1}^{k-1} A_i$,

(iv) $f(P_i) \leq f(A_i)$ for every $i \in [k-1]$.

Proof. We use the procedure given in Figure 3 to obtain subsets $P_1, \ldots, P_{k-1}$ of V with the desired properties.

The procedure terminates in finite number of iterations since the steps in the while loop make progress towards ensuring that the sets $P_1, \ldots, P_{k-1}$ are mutually disjoint. Property (i) is achieved due to the termination condition. Properties (ii) and (iii) are maintained as invariants throughout the procedure (recall that $u_i \in A_i$ but $u_i \notin A_j$ for all distinct $i, j \in [k-1]$). Property (iv) is also maintained as an invariant throughout the procedure by posimodularity: recall that by posimodularity, for any two sets X and Y we have that $f(X-Y)+f(Y-X) \leq f(X) + f(Y)$ and hence, either $f(X-Y) \leq f(X)$ or $f(Y-X) \leq f(Y)$ should hold. ∎

Let $P_k = V - \cup_{i=1}^{k-1} P_i$. By property (i) of Claim 2.2, the sets P_i and P_j are disjoint for every distinct $i, j \in [k]$. By properties (ii) and (iii) of Claim 2.2, we have that $u_i \in P_i$ for every $i \in [k]$. Hence, $(P_1, \ldots, P_k)$ form a k-partition of the ground set V. By property (iv) of Claim 2.2, we have that $f(P_i) \leq f(A_i)$ for every $i \in [k-1]$. It remains to show that $f(P_k) \leq f(A_k)$.

By property (iii) of Claim 2.2, we have that $P_k = \overline{\cup_{i=1}^{k-1} P_i} = \overline{\cup_{i=1}^{k-1} A_i} = \cap_{i=1}^{k-1} \overline{A_i}$. Let $B_i := \overline{A_i}$ for every $i \in [k]$. Claim 2.3 below shows that the function value of P_k is at most that of A_k by relying on submodularity, thus completing the proof. We note that Claim 2.3 has been used in previous works [9] in different contexts.

∎

CLAIM 2.3. *We have that*

$$f\left(\cap_{i\in[k-1]} B_i\right) \leq f(B_k).$$

Proof. Suppose not. Choose maximal $J \subseteq [k-1]$ such that $f(\cap_{j\in J} B_j) \leq f(B_k)$. We note that $J \neq \emptyset$ since $f(B_j) \leq f(B_k)$ for every $j \in [k-1]$. By assumption, $J \subsetneq [k-1]$ (otherwise we are done). Let $i \in [k-1] \setminus J$ and $R := \cap_{j\in J} B_j$. We note that $f(R \cup B_i) \geq f(B_i)$ since $(R \cup B_i, \overline{R \cup B_i})$ is a $(C \setminus \{u_i\}, \overline{U})$-terminal cut while $(B_i, \overline{B_i})$ is a minimum $(C \setminus \{u_i\}, \overline{U})$-terminal cut. Then,

$$\begin{aligned} f(B_k) + f(B_i) &\geq f(R) + f(B_i) \\ &\qquad \text{(By choice of } J\text{)} \\ &\geq f(R \cup B_i) + f(R \cap B_i) \\ &\qquad \text{(By submodularity)} \\ &\geq f(B_i) + f(R \cap B_i). \end{aligned}$$

Therefore, $f(R \cap B_i) \leq f(B_k)$ and hence, the set $J \cup \{i\}$ contradicts the choice of J. ∎

PROCEDURE:

1. Initialize $P_i \leftarrow A_i$ for every $i \in [k-1]$
2. While there exist distinct $i, j \in [k-1]$ such that $P_i \cap P_j \neq \emptyset$:
   (a) If $f(P_i - P_j) \leq f(P_i)$, then $P_i \leftarrow P_i - P_j$
   (b) Else, $P_j \leftarrow P_j - P_i$

Figure 3: Procedure for the proof of Claim 2.2.

2.3 Proof of Theorem 1.3 We now restate and prove Theorem 1.3.

THEOREM 2.2. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function and let $k \geq 2$ be an integer. Let $(V_1, \ldots, V_k)$ be a V_1-maximal minmax k-partition with respect to f. Suppose $|V_1| \geq k-1$. Then, for every subset $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \ldots, k\}$, there exists a subset $S \subseteq V_1$ of size $k-1$ 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 $j \in \{2, \ldots, k\}$. Our proof strategy is to obtain a cheaper k-partition than $(V_1, \ldots, V_k)$, thereby contradicting the optimality of $(V_1, \ldots, V_k)$. Let OPT_k denote the cost of $(V_1, \ldots, V_k)$. For a subset $X \subseteq V_1$, let $(V_X, \overline{V_X})$ be the source maximal minimum (X,T)-terminal cut. By Lemma 2.1, we have that $V_X \subseteq V_1$ for all $X \subseteq V_1$.

Among all possible subsets of V_1 of size $k-1$, pick a subset S such that $f(V_S)$ is maximum. Then, by Lemma 2.1 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 $f(V_S) < f(V_1)$. Let $u_1, \ldots, u_{k-1}$ be the vertices in S. Since $V_S \subsetneq V_1$, there exists a vertex $u_k \in V_1 \setminus V_S$. Let $C := \{u_1, \ldots, u_k\} = S \cup \{u_k\}$. For $i \in [k]$, let $(B_i, \overline{B_i})$ be the source maximal minimum $(C - \{u_i\}, T)$-terminal cut. We note that $(B_k, \overline{B_k}) = (V_S, \overline{V_S})$ and the size of $C - \{u_i\}$ is $k-1$ for every $i \in [k]$. By Lemma 2.1 and assumption, we have that $B_i \subsetneq V_1$ for every $i \in [k]$. Hence, we have

(2.1)
$$f(B_i) \leq f(V_S) < f(V_1) \text{ and } B_i \subsetneq V_1 \text{ for every } i \in [k].$$

The next claim will set us up to apply Theorem 2.1.

CLAIM 2.4. *For every $i \in [k]$, we have that $u_i \in \overline{B_i}$.*

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

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

We also have that

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

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

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

Therefore, all inequalities above should be equations and hence, $f(V_S \cup B_i) = f(V_S)$. Consequently, 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 $(V_S, \overline{V_S})$ since $u_k \in B_i$ and $u_k \notin V_S$. □

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

We will now apply Theorem 2.1. We consider $U := V_1$ and $C = \{u_1, \ldots, u_k\} \subseteq U$. Let $(\overline{A_i}, A_i) := (B_i, \overline{B_i})$ for every $i \in [k]$. The 2-partition $(\overline{A_i}, A_i)$ is a minimum $(C \setminus \{u_i\}, \overline{U})$-terminal cut for every $i \in [k]$. By Claim 2.4, we have that $u_i \in A_i$ for every $i \in [k]$. Since $(B_j, \overline{B_j})$ is a $(C - \{u_j\}, T)$-terminal cut, we have that $u_i \notin \overline{B_j}$ for every distinct $i, j \in [k]$. Thus, $u_i \in A_i \setminus (\cup_{j \in [k] \setminus \{i\}} A_j)$ for every $i \in [k]$. We may reindex the elements in C so that

$f(A_1) \le f(A_2) \le \ldots \le f(A_k)$. Therefore, the sets U, C, and the 2-partitions $(\overline{A_i}, A_i)$ for $i \in [k]$ satisfy the conditions of Theorem 2.1. By Theorem 2.1 and statement (2.1), we obtain a k-partition $(P_1, \ldots, P_k)$ of V such that

$$\begin{aligned}\text{cost}(P_1, \ldots, P_k) &\le \max\{f(A_i) : i \in [k]\} \\ &= f(V_S) < f(V_1) \le OPT_k.\end{aligned}$$

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

3 Algorithm

In this section, we design an algorithm to solve Minmax-SymSubmod-k-Part based on Theorem 1.3. Using Theorem 1.3, it is possible to efficiently enumerate n^{2k-2} candidate subsets such that one of them is V_1, where $(V_1, \ldots, V_k)$ is a V_1-maximal minmax k-partition with respect to the input symmetric submodular function $f : 2^V \to \mathbb{R}$. However, after finding V_1, we cannot recurse on the function f restricted to $\overline{V_1}$ to find a cheapest $(k-1)$-partition: the restricted function may not be symmetric. Instead, we will work with the function obtained by contracting V_1. We define the contraction operation now.

Let $f : 2^V \to R$ be a symmetric submodular function. Let U be a subset of the ground set V. We define the contracted function f/U as follows: the ground set is $V' := V - U + \{u\}$, where u denotes the contracted element. The function $f/U : 2^{V'} \to \mathbb{R}$ is defined as:

$$(f/U)(A) := \begin{cases} f(A \cup U) & \text{if } u \in A \subseteq V', \\ f(A) & \text{if } u \notin A \subseteq V'. \end{cases}$$

We note that the function f/U is symmetric and submodular. The following observation is easy but we give a proof for the sake of completeness.

Observation 3.1. *If $(V_1, \ldots, V_k)$ is a minmax k-partition with respect to f, then $(V_2, \ldots, V_k, \{v_1\})$ is a minmax k-partition with respect to f/V_1 where v_1 is the contracted element.*

Proof. For notational simplicity, let g denote the function f/V_1. Say $(P_1, \ldots, P_k)$ is a minmax k-partition with respect to g. For the sake of contradiction, suppose

$$\text{cost}_g(P_1, \ldots, P_k) < \text{cost}_g(V_2, \ldots, V_k, \{v_1\}).$$

We observe that

$$\text{cost}_g(V_2, \ldots, V_k, \{v_1\}) = \text{cost}_f(V_1, \ldots, V_k).$$

Without loss of generality, let $v_1 \in P_1$. Consider the k-partition $(P_1' := (P_1 \setminus \{v_1\}) \cup V_1, P_2, \ldots, P_k)$ of V. We have that

$$\text{cost}_f(P_1', P_2, \ldots, P_k) = \text{cost}_g(P_1, \ldots, P_k).$$

Hence, we have a k-partition $(P_1', P_2, \ldots, P_k)$ that is cheaper with respect to f than $(V_1, \ldots, V_k)$, thus contradicting optimality of $(V_1, \ldots, V_k)$. □

Next, we define a crucial tie-breaking rule which will help us find the next part V_2 by working with the contracted function f/V_1, where $(V_1, \ldots, V_k)$ is a minmax k-partition with respect to f. A minmax k-partition $(V_1, \ldots, V_k)$ is a *lexicographically maximal minmax k-partition* if there is no other minmax k-partition $(U_1, \ldots, U_k)$ with an $i \in [k-1]$ such that $U_1 = V_1, \ldots, U_{i-1} = V_{i-1}$, but $U_i \supsetneq V_i$. We observe that a lexicographically maximal minmax k-partition always exists. Furthermore, if $(V_1, \ldots, V_k)$ is a lexicographically maximal minmax k-partition, then it is also a V_1-maximal minmax k-partition. The following lemma shows that contraction of V_1 preserves lexicographic maximality in a certain sense.

Lemma 3.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function, $k \ge 2$ be a positive integer, and $(V_1, \ldots, V_k)$ be a lexicographically maximal minmax k-partition with respect to f. Let f/V_1 be the contracted function with v_1 being the contracted element. Then, $(V_2, \ldots, V_k, \{v_1\})$ is a lexicographically maximal minmax k-partition with respect to the contracted function f/V_1.*

Proof. For notational simplicity, let g denote the function f/V_1. By Observation 3.1, the k-partition $(V_2, \ldots, V_k, \{v_1\})$ is a minmax k-partition with respect to g. We now show that $(V_2, \ldots, V_k, \{v_1\})$ is a lexicographically maximal minmax k-partition with respect to g. Suppose not. Then, there exists a minmax k-partition $(U_1, \ldots, U_k)$ with respect to g and an index $i \in [k-1]$ such that $U_1 = V_2, U_2 = V_3, \ldots, U_{i-1} = V_i$, but $U_i \supsetneq V_{i+1}$. Without loss of generality, let $v_1 \in U_j$ for some $j \in \{i, \ldots, k\}$. We distinguish two cases.

Case 1. Suppose $U_j \setminus \{v_1\} \neq \emptyset$. Then, the partition

$$(V_1 \cup (U_j \setminus \{v_1\}), U_1, \ldots, U_{j-1}, U_{j+1}, \ldots, U_k)$$

is a minmax k-partition with respect to f that contradicts lexicographic maximality of $(V_1, \ldots, V_k)$ (in particular, it contradicts V_1-maximality).

Case 2. Suppose $U_j = \{v_1\}$. Since $|U_i| \ge |V_{i+1}| + 1 \ge 2$ while $|U_j| = 1$, it follows that

$j \neq i$. We recall that $j \geq i$ and consequently, $j \geq i+1$. Then, the partition $(V_1, V_2 = U_1, \ldots, V_i = U_{i-1}, U_i, U_{i+1}, \ldots, U_{j-1}, U_{j+1}, \ldots, U_k)$ is a minmax k-partition with respect to f that contradicts lexicographic maximality of $(V_1, \ldots, V_k)$ (in particular, it contradicts V_{i+1}-maximality as $U_i \supsetneq V_{i+1}$). ∎

Lemma 3.1 suggests that if we know the part V_1 of a lexicographically maximal minmax k-partition $(V_1, \ldots, V_k)$ with respect to f, then using Theorem 1.3 for the function f/V_1, we can efficiently enumerate n^{2k-2} candidate subsets such that one of them is V_2. We will now use this idea inductively to recover all parts of a lexicographically maximal minmax k-partition.

We begin with the procedure in Figure 4 that returns a collection of candidate sets such that one of them is V_1, where $(V_1, \ldots, V_k)$ is a V_1-maximal minmax k-partition with respect to f. We summarize the guarantees of this procedure in Corollary 3.1. The corollary follows immediately from Theorem 1.3.

Corollary 3.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function on a n-element ground set V and let $k \geq 2$ be an integer. Let $(V_1, \ldots, V_k)$ be a V_1-maximal minmax k-partition with respect to f. Suppose $Q \subseteq V \setminus V_1$ such that $|Q \cap V_j| = 1$ for every $j \in \{2, \ldots, k\}$ for which $Q \cap V_j \neq \emptyset$. Then,*

(i) V_1 is in the collection $\mathcal{R}$ returned by Generate-Candidates(f, k, Q), and

(ii) the size of the collection $\mathcal{R}$ that is returned by Generate-Candidates(f, k, Q) is $O(n^{2k-2})$.

Moreover, Generate-Candidates procedure can be implemented to run in time $n^{2k-2}T(n)$, where $T(n)$ is the time for computing source maximal minimum (S,T)-terminal cuts for a symmetric submodular function on n elements.

We now describe our algorithm to find a minmax k-partition in Figure 5 and summarize its guarantees in Theorem 3.1.

Theorem 3.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function on a n-element ground set V and let $k \geq 2$ be an integer. Then, Algorithm Partition(f, k) given in Figure 5 returns a minmax k-partition with respect to f and it can be implemented to run in $n^{O(k^2)}T(n)$ time, where $T(n)$ denotes the time complexity for computing the source maximal minimum (S,T)-terminal cut for a submodular function defined over a ground set of size n.*

Proof. We first prove the run-time bound. For every $i \in [k-1]$, we have that $|\mathcal{C}_i| = |\mathcal{C}_{i-1}||\mathcal{R}_i| = \prod_{j=1}^{i} |\mathcal{R}_i| = O(n^{2i(k-1)})$ and the time to compute $\mathcal{C}_i$ is $O(n^{2i(k-1)})T(n)$ using Corollary 3.1. Hence, $|\mathcal{C}_k| = |\mathcal{C}_{k-1}| = O(n^{2(k-1)^2})$ and the total run-time is $O(n^{2(k-1)^2})T(n)$.

Next, we prove correctness. Let $(V_1, \ldots, V_k)$ be a lexicographically maximal minmax k-partition. We will show that $(V_1, \ldots, V_i) \in \mathcal{C}_i$ for every $i \in [k-1]$ by induction on i. The base case of $i = 1$ follows immediately from Corollary 3.1. For the induction step, let $i \geq 2$. By induction hypothesis, we have that $(V_1, \ldots, V_{i-1}) \in \mathcal{C}_{i-1}$. Consider $g = f/V_1/V_2/\ldots/V_{i-1}$ where $v_1, \ldots, v_{i-1}$ are the contracted elements. Let A be the ground set of g. Contraction operation preserves symmetry and submodularity. So, g is a symmetric submodular function. By Lemma 3.1, the k-partition given by $(A_1 := V_i, A_2 := V_{i+1}, \ldots, A_{k-i+1} := V_k, A_{k-i+2} := \{v_1\}, \ldots, A_k := \{v_{i-1}\})$ is a lexicographically maximal minmax k-partition with respect to g. Thus, $(A_1, \ldots, A_k)$ is a A_1-maximal minmax k-partition with respect to g. Moreover, the set $Q = \{v_1, \ldots, v_{i-1}\}$ is a subset of $A \setminus A_1$ such that $|Q \cap A_j| = 1$ for every $j \in \{2, \ldots, k\}$ for which $Q \cap A_j \neq \emptyset$. Hence, by Corollary 3.1, the set A_1 is in the collection $\mathcal{R}_i$. Thus, $V_i \in \mathcal{R}_i$ and consequently, $(V_1, \ldots, V_{i-1}, V_i) \in \mathcal{C}_i$. ∎

We note that source maximal minimum (S,T)-terminal cut for a submodular function $f : 2^V \to \mathbb{R}$ can be computed in time $n^{O(1)}T$, where $n = |V|$ and T is the time per evaluation (e.g., see [9]). Thus, Theorem 1.2 follows from Theorem 3.1. Moreover, the evaluation oracle for the hypergraph cut function can be implemented in $T = m$ time, where m is the number of hyperedges in the input hypergraph. Thus, Theorem 1.1 follows from Theorem 1.2.

4 Conclusion

Given the general sense that it is harder to design algorithms/approximations for the minmax objective than the minsum objective, our result is somewhat surprising: we have given a polynomial-time algorithm for Minmax-SymSubmod-k-Part for all fixed k while such a result is not yet known for Sym-Submod-k-Part (or even for the special case of Hypergraph-k-Part). As a special case, our algorithm resolves the complexity of Minmax-Hypergraph-k-Part for fixed k which was posed by Lawler in 1973. Our key technical contribution is a structural theorem (Theorem 1.3) that enables

GENERATE-CANDIDATES($f : 2^V \to \mathbb{R}, k, Q$)

Input: A function $f : 2^V \to \mathbb{R}$, an integer $k \geq 2$, and a subset Q of V

Output: A collection $\mathcal{R} \subseteq 2^V$

1. Initialize $\mathcal{R} \leftarrow \{U \subset V : |U| \leq k-2\}$
2. For every disjoint $S, T \subset V$ with $|S| = k-1$, $T \supseteq Q$ and $|T| = k-1$:
   (a) Compute the source maximal minimum (S,T)-terminal cut $(U, \overline{U})$
   (b) $\mathcal{R} \leftarrow \mathcal{R} \cup \{U\}$
3. Return $\mathcal{R}$

Figure 4: Procedure to generate candidates for V_1.

efficient recovery of each part of an optimum min-max k-partition by solving minimum (S,T)-terminal cuts. The ideas underlying the proof of our structural theorem can also be adapted to prove the following mild relaxation of Queyranne's retracted claim for symmetric submodular functions under the minsum objective.[5]

THEOREM 4.1. *Let $f : 2^V \to \mathbb{R}$ be a symmetric submodular function and let $k \geq 2$ be an integer. Let $(V_1, V_2, \ldots, V_k)$ be an optimum* minsum *k-partition with respect to f such that V_1 is the cheapest part (that is, $f(V_1) \leq f(V_i)$ for every $i \in [k]$), and V_1 is maximal subject to this condition. Suppose $|V_1| \geq k-1$. Then, for every subset $T \subseteq \overline{V_1}$ such that $T \cap V_j \neq \emptyset$ for every $j \in \{2, \ldots, k\}$, there exists a subset $S \subseteq V_1$ of size $k-1$ such that $(V_1, \overline{V_1})$ is the source maximal minimum (S,T)-terminal cut.*

The proof of this theorem conceptually simplifies the main structural theorem of Goldschmidt and Hochbaum for GRAPH-k-CUT. Using this theorem, we can solve GRAPH-k-CUT in $n^{O(k^2)}$-time: we can enumerate a collection of n^{2k-2} candidate sets for V_1 and for each candidate set U in the collection, we recurse on $G[V \setminus U]$ to find a minsum $(k-1)$-partition, concatenate it with U to obtain a k-partition, and return the best of all k-partitions. It would be interesting to see if the above theorem can be generalized/adapted to obtain a polynomial-time algorithm for SYM-SUBMOD-k-PART for fixed k.

[5]Theorem 4.1 is a mild relaxation of Queyranne's retracted claim since his claim concluded the size of the set S to be $k-2$ whereas our theorem states that the size of the set S is $k-1$.

Acknowledgements: We thank Weihang Wang for helpful discussions during preliminary stages of this work and pointing to the failure of the Gomory-Hu tree based approach for the minmax objective.

References

[1] N. Bansal, U. Feige, R. Krauthgamer, K. Makarychev, V. Nagarajan, J. Naor, and R. Schwartz, *Min-max graph partitioning and small set expansion*, SIAM Journal on Computing **43** (2014), no. 2, 872–904.

[2] K. Chandrasekaran and C. Chekuri, *Hypergraph k-cut for fixed k in deterministic polynomial time*, Proceedings of the 61st IEEE Annual Symposium on Foundations of Computer Science (to appear), FOCS, 2020.

[3] K. Chandrasekaran, C. Xu, and X. Yu, *Hypergraph k-cut in randomized polynomial time*, Mathematical Programming (Preliminary version in SODA 2018) (2019).

[4] K. Chandrasekaran and X. Zhang, Private communication.

[5] C. Chekuri and A. Ene, *Approximation Algorithms for Submodular Multiway Partition*, Proceedings of the 52nd IEEE Annual Symposium on Foundations of Computer Science, FOCS, 2011, pp. 807–816.

[6] C. Chekuri and S. Li, *A note on the hardness of approximating the k-way Hypergraph Cut problem*, Manuscript, `http://chekuri.cs.illinois.edu/papers/hypergraph-kcut.pdf`, 2015.

[7] C. Chekuri, K. Quanrud, and C. Xu, *LP relaxation and tree packing for minimum k-cut*, SIAM Journal on Discrete Mathematics **34** (2020), no. 2, 1334–1353.

[8] K. Fox, D. Panigrahi, and F. Zhang, *Minimum cut and minimum k-cut in hypergraphs via branching*

Algorithm Partition($f : 2^V \to \mathbb{R}, k$)

Input: Symmetric submodular function $f : 2^V \to \mathbb{R}$, an integer $k \geq 2$
Output: A minmax k-partition with respect to f

1. Initialize $\mathcal{C}_1 \leftarrow$ Generate-Candidates$(f, k, Q \leftarrow \emptyset)$
2. For $i = 2, \ldots, k-1$:
   (a) $\mathcal{C}_i \leftarrow \emptyset$
   (b) For each $(i-1)$-partition $(P_1, \ldots, P_{i-1}) \in \mathcal{C}_{i-1}$:
      (i) $g \leftarrow f/P_1/P_2/\ldots/P_{i-1}$ where $p_1, \ldots, p_{i-1}$ are the contracted elements
      (ii) $Q \leftarrow \{p_1, \ldots, p_{i-1}\}$
      (iii) $\mathcal{R}_i \leftarrow$ Generate-Candidates(g, k, Q)
      (iv) $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \{(P_1, \ldots, P_{i-1}, U) : U \in \mathcal{R}_i\}$
3. $\mathcal{C}_k \leftarrow \{(P_1, \ldots, P_{k-1}, V \setminus \cup_{i=1}^{k-1} P_i) : (P_1, \ldots, P_{k-1}) \in \mathcal{C}_{k-1}\}$
4. Among all k-partitions in $\mathcal{C}_k$, pick the one with minimum cost and return it

Figure 5: Algorithm to compute minimum k-partition for a symmetric submodular function.

contractions, Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, 2019, pp. 881–896.

[9] M. X. Goemans and V. S. Ramakrishnan, *Minimizing submodular functions over families of sets*, Combinatorica **15** (1995), 499–513.

[10] O. Goldschmidt and D. Hochbaum, *A Polynomial Algorithm for the k-cut Problem for Fixed k*, Mathematics of Operations Research **19** (1994), no. 1, 24–37.

[11] F. Guinez and M. Queyranne, *The size-constrained submodular k-partition problem*, Unpublished manuscript. See also `https://smartech.gatech.edu/bitstream/handle/1853/43309/Queyranne.pdf`, 2012.

[12] A. Gupta, E. Lee, and J. Li, *The Karger-Stein Algorithm is Optimal for k-Cut*, Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC, 2020, p. 473–484.

[13] Y. Kamidoi, N. Yoshida, and H. Nagamochi, *A Deterministic Algorithm for Finding All Minimum k-Way Cuts*, SIAM Journal on Computing **36** (2007), no. 5, 1329–1341.

[14] D. Karger, *Minimum cuts in near-linear time*, Journal of the ACM **47** (2000), no. 1, 46–76.

[15] D. Karger and C. Stein, *A new approach to the minimum cut problem*, Journal of the ACM **43** (1996), no. 4, 601–640.

[16] E. Lawler, *Cutsets and Partitions of Hypergraphs*, Networks **3** (1973), 275–285.

[17] P. Manurangsi, *Inapproximability of Maximum Biclique Problems, Minimum k-Cut and Densest At-Least-k-Subgraph from the Small Set Expansion Hypothesis*, Proceedings of the 44th International Colloquium on Automata, Languages, and Programming, ICALP, 2017, pp. 79:1–79:14.

[18] K. Okumoto, T. Fukunaga, and H. Nagamochi, *Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems*, Algorithmica **62** (2012), no. 3, 787–806.

[19] M. Queyranne, *On optimum size-constrained set partitions*, 1999, Talk at Aussois workshop on Combinatorial Optimization.

[20] H. Saran and V. Vazirani, *Finding k Cuts within Twice the Optimal*, SIAM Journal on Computing **24** (1995), no. 1, 101–108.

[21] A. Schrijver, *Combinatorial optimization: Polyhedra and efficiency*, Algorithms and Combinatorics, Springer, 2003.

[22] Z. Svitkina and L. Fleischer, *Submodular approximation: Sampling-based algorithms and lower bounds*, SIAM Journal on Computing **40** (2011), no. 6, 1715–1737.

[23] Z. Svitkina and É. Tardos, *Min-max multiway cut*, Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX, 2004, pp. 207–218.

[24] M. Thorup, *Minimum k-way Cuts via Deterministic Greedy Tree Packing*, Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC, 2008, pp. 159–166.

[25] K. Wei, R. Iyer, S. Wang, W. Bai, and J. Bilmes,

Mixed robust/average submodular partitioning: Fast algorithms, guarantees, and applications, Advances in Neural Information Processing Systems, 2015, pp. 2233–2241.

[26] M. Xiao, *An Improved Divide-and-Conquer Algorithm for Finding All Minimum k-Way Cuts*, Proceedings of 19th International Symposium on Algorithms and Computation, ISAAC, 2008, pp. 208–219.