The Number of Minimum k-Cuts: Improving the Karger-Stein Bound

Anupam Gupta[*]
anupamg@cs.cmu.edu
CMU
Pittsburgh, United States

Euiwoong Lee[†]
euiwoong@cims.nyu.edu
NYU
New York City, United States

Jason Li[‡]
jmli@cs.cmu.edu
CMU
Pittsburgh, United States

ABSTRACT

Given an edge-weighted graph, how many minimum k-cuts can it have? This is a fundamental question in the intersection of algorithms, extremal combinatorics, and graph theory. It is particularly interesting in that the best known bounds are *algorithmic*: they stem from algorithms that compute the minimum k-cut.

In 1994, Karger and Stein obtained a randomized contraction algorithm that finds a minimum k-cut in $O(n^{(2-o(1))k})$ time. It can also *enumerate* all such k-cuts in the same running time, establishing a corresponding extremal bound of $O(n^{(2-o(1))k})$. Since then, the algorithmic side of the minimum k-cut problem has seen much progress, leading to a deterministic algorithm based on a tree packing result of Thorup, which enumerates all minimum k-cuts in the same asymptotic running time, and gives an alternate proof of the $O(n^{(2-o(1))k})$ bound. However, beating the Karger–Stein bound, even for computing a single minimum k-cut, has remained out of reach.

In this paper, we give an algorithm to enumerate all minimum k-cuts in $O(n^{(1.981+o(1))k})$ time, breaking the algorithmic and extremal barriers for enumerating minimum k-cuts. To obtain our result, we combine ideas from both the Karger–Stein and Thorup results, and draw a novel connection between minimum k-cut and *extremal set theory*. In particular, we give and use tighter bounds on the size of set systems with bounded dual VC-dimension, which may be of independent interest.

CCS CONCEPTS

• **Mathematics of computing → Extremal graph theory**; **Graph algorithms**; • **Theory of computation → Parameterized complexity and exact algorithms**.

KEYWORDS

graph cuts, graph algorithms, minimum k-cut, extremal graph theory

ACM Reference Format:
Anupam Gupta, Euiwoong Lee, and Jason Li. 2019. The Number of Minimum k-Cuts: Improving the Karger-Stein Bound. In *Proceedings of the 51st Annual ACM SIGACT Symposium on the Theory of Computing (STOC '19), June 23–26, 2019, Phoenix, AZ, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3313276.3316395

1 INTRODUCTION

We consider the k-Cut problem: given an edge-weighted graph $G = (V, E, w)$ and an integer k, delete a minimum-weight set of edges so that G has at least k connected components. This problem is a natural generalization of the global min-cut problem, where the goal is to break the graph into $k = 2$ pieces. This problem has been actively studied in theory of both exact and approximation algorithms, where each result brought new insights and tools on graph cuts.

Goldschmidt and Hochbaum gave the first polynomial-time algorithm for fixed k, with $O(n^{(1/2-o(1))k^2})$ runtime [10]. Since then, the exact exponent in terms of k has been actively studied. The textbook minimum cut algorithm of Karger and Stein [17], based on random edge contractions, can be adapted to solve k-Cut in $\widetilde{O}(n^{2(k-1)})$ (randomized) time. The deterministic algorithms side has seen a series of improvements since then [7, 15, 32]. The fastest algorithm for general edge weights is due to Chekuri et al. [7]. It runs in $O(mn^{2k-3})$ time and is based on a deterministic tree packing result of Thorup [32]. Hence, the leading algorithms on the randomized and deterministic fronts utilize completely different approaches, but neither is able to break the $O(n^{(2-o(1))k})$ bound for the problem.

On the lower bounds side, a simple reduction from Max-Weight $(k-1)$-Clique to k-Cut implies that the conjectured time lower bound $\tilde{\Omega}(n^{(1-o(1))k})$ for Max-Weight $(k-1)$-Clique [2] also holds for k-Cut.[1] Given the recent interest in fine-grained complexity, *for the algorithmic problem of finding the minimum k-cut, should the true exponent in n for k-Cut be k, $2k$, or somewhere in between?*

Another closely related, extremal question concerns the number of minimum k-cuts that a graph can have. The algorithms of Karger-Stein, Thorup, and Chekuri et al. can be adapted to enumerate all

[*]Supported in part by NSF awards CCF-1536002, CCF-1540541, and CCF-1617790, and the Indo-US Joint Center for Algorithms Under Uncertainty.
[†]Supported in part by the Simons Collaboration on Algorithms and Geometry.
[‡]Supported in part by NSF awards CCF-1536002, CCF-1540541, and CCF-1617790.

[1]The conjectured lower bounds are $\tilde{\Omega}(n^{(1-o(1))k})$ for Max-Weight k-Clique when weights are integers in the range $[1, \Omega(n^k)]$, and $\tilde{\Omega}(n^{(\omega/3)k})$ for Unweighted $(k-1)$-Clique, where ω is the matrix multiplication constant.

minimum k-cuts in $O(n^{(2-o(1))k})$ time, implying the same bound for the extremal number of minimum k-cuts in a graph. To this date, no proof of a better bound—algorithmic or otherwise—is known. On the other hand, there are some graphs (e.g., a cycle with n vertices) where the number of minimum k-cuts is $\Omega(n^k)$.[2] Thus, the mathematical question remains: *for the* ***extremal*** *number of minimum k-cuts of a graph, should the exponent of n be k, 2k, or somewhere in between?*

In recent work [11], we improved on the algorithmic barrier of $O(n^{(2-o(1))k})$ for the case when the input graph has edge weights that are integers polynomially bounded in n. In particular, our deterministic algorithm runs in time $O(k^{O(k)}n^{(2\omega/3+o(1))k})$, where $\omega \le 2.3738$ denotes the matrix multiplication constant [21, 33]. Aside from the unfortunate restriction to integer-weighted graphs, our algorithm suffers other disadvantages compared to the previous k-Cut algorithms.

(1) When edge weights are integers bounded by $n^{O(1)}$ (in particular, exponent independent of k), the reduction from Max-Weight $(k-1)$-Clique no longer holds. Consequently, our algorithm solves an easier variant of k-Cut whose lower bound is only $\Omega(n^{(\omega/3-o(1))k})$ from Unweighted k-Clique, and not $\Omega(n^{(1-o(1))k})$.
(2) The previous algorithms are "combinatorial", where ours uses fast matrix-multiplication as a black-box. (See [1, 34] for discussion about combinatorial and matrix multiplication-based algorithms for k-Clique and other problems.) One concrete difference is that our algorithm inherently requires $n^{\Omega(k)}$ space, whereas all of the aforementioned k-Cut algorithms require only poly(n) space.
(3) Our algorithm only finds *one* minimum k-cut and, unlike the previous algorithms, cannot be adapted to find all of them in the same asymptotic running time. (This is a common weakness with algorithms that use matrix multiplication.) Hence, it does not imply any improved bound on the extremal number of minimum k-cuts, even for integer-weights.

To summarize, beating either of the $O(n^{(2-o(1))k})$ algorithmic and extremal bounds has remained open for the general case. In this paper, we break these bounds for large enough k, and get the first true improvement over the Karger–Stein result for k-Cut, in both the algorithmic and extremal settings.

Theorem 1.1 (Enumeration Algorithm). *For any large enough constant k, there is a randomized algorithm that enumerates all minimum k-cuts in time $O(n^{1.981k})$ w.h.p.*

Corollary 1.2 (Extremal Result). *For any large enough constant k, there are at most $O(n^{1.981k})$ many minimum k-cuts.*

1.1 Our Techniques

We believe that an important component of the paper's contributions are the techniques, which draw on different areas: some of them are perhaps surprising and suggesting directions for further investigation. As mentioned previously, the two leading approaches so far to k-Cut—random contractions and tree packing—utilize completely different techniques. Our algorithm is the first to incorporate both approaches, which gives us a broader collection of tools to draw from. In addition, we establish a novel connection to *extremal set theory* in the context of graph cut algorithms, which may be of independent interest. Finally, our actual k-Cut algorithm resembles the bounded-depth branching algorithms commonly found in *fixed-parameter tractable* algorithms.

Our k-Cut algorithm is conceptually simple at a high level. Let $S_1^*, \dots, S_k^* \subseteq V$ be an arbitrary minimum k-cut. We compute a set $\mathcal{A} \subseteq 2^V$ of potential subsets of vertices $A \subseteq V$ such that one of these sets is exactly S_i^* for some i. We then branch on each computed set $A \in \mathcal{A}$ by guessing A as one component in the targeted k-cut, and then recursively calling $(k-1)$-Cut on $G \setminus A$. It is clear that this algorithm will return the correct k-cut on one of its branches. Naturally, to obtain an efficient algorithm, we want the size of $\mathcal{A}$ to be small to ensure a small branching factor, and furthermore, $\mathcal{A}$ should be computable efficiently.

A Simpler Case. The actual set $\mathcal{A}$ is complicated to describe, so to strive for the simplest exposition that highlights most of our techniques, let us assume (*with* much loss of generality) that $\mathcal{A} := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$. (Here, any constant less than 1.5 will do.) This set $\mathcal{A}$ works if there exists a component S_i^* satisfying $w(\partial_G S_i^*) \le \frac{1.49}{k} OPT$. This may not always hold, since $\sum_i w(\partial_G S_i^*) = 2\,OPT$ and hence we can only get a bound of $w(\partial_G S_i^*) \le \frac{2}{k} OPT$ in general—but this is a simple case considered for the purposes of intuition. It remains to bound the size of $\mathcal{A}$, and the time to compute it.

(1) *The Size of $\mathcal{A}$:* We can bound the size of $\mathcal{A}$ by $O(2^k n)$ using a well-known result in extremal set theory, discussed below. But assuming this bound, we pay a branching factor of $O(2^k n)$ to decrease k by 1, which is great, because if we can continue this process all the way to $k = 0$, then our running time becomes $O(2^{k^2} n^{k+O(1)})$. (This bound is much better than the one in Theorem 1.1, but recall that we obtained it with much loss in generality.)
(2) *Computing $\mathcal{A}$:* We run a modified Karger-Stein randomized contraction procedure poly(n) times, and then take the $O(2^k n)$ sets A with smallest boundary $w(\partial_G A)$. An argument similar to Karger and Stein's original analysis shows that our computed set contains $\mathcal{A}$ w.h.p., which is good enough for our purposes.

We now sketch the proof of $|\mathcal{A}| \le O(2^k n)$, highlighting its connection to extremal set theory. It will be useful to view each set $A \in \mathcal{A}$ also as the (2-)cut $(A, V \setminus A)$ in the graph G. For a contradiction, suppose that $|\mathcal{A}| > O(2^k n)$. Since $|\mathcal{A}| > 2n$, a simple result in extremal set theory says that there exist two such sets $A_1, A_2 \in \mathcal{A}$ that *cross*. Hence, if we cut out the edges in $\partial_G A_1 \cup \partial_G A_2$, we obtain a 4-cut, not a 3-cut. This means that we obtain 3 new components for the price of $2 \cdot \frac{1.49}{k} OPT = \frac{2.98}{k} OPT$. This amortizes to a cost of $\frac{2.98}{3k} OPT$ per additional component. If we can repeat this process—always finding two such crossing cuts whose removal introduces 3 additional components—then we eventually obtain roughly k components for roughly $(1-\varepsilon)OPT$, contradicting our choice of OPT. (See Figure 1.) In §3.2, we prove that this process

[2]Technically, it is $\Omega(n^k/k!)$, but we assume that k is a large but fixed constant throughout.

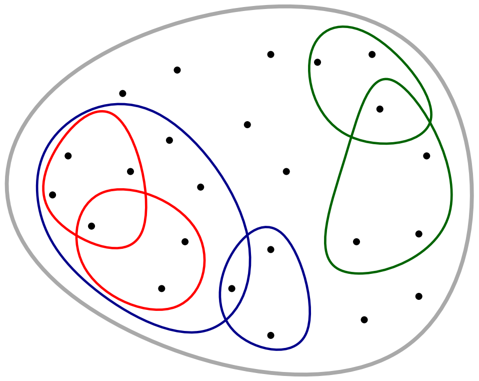

Figure 1: An example with $k = 10$; we picked 3 pairs of crossing cuts (i.e., 6 cuts) each of weight $\frac{1.49}{k} OPT$. Because each of them gives 3 new pieces, we get $k = 10$ parts. But the cut edge weight totals $\frac{6 \times 1.49}{k} OPT < OPT$, a contradiction.

is possible whenever $|\mathcal{A}| > O(2^k n)$, implying the desired extremal bound on $|\mathcal{A}|$.

The General Case: Let's try to remove the simplifying assumption we made. What if every component S_i^* satisfies $w(S_i^*) \geq \frac{1.5}{k} OPT$? We could try to take $\mathcal{A} := \{A \subseteq V : w(\partial_G A) \leq \frac{2}{k} OPT\}$, which would certainly contain some S_i^*. We do take this approach, bounding the size of $\mathcal{A}$ (which is substantially more technical), but in this case, it is possible that $|\mathcal{A}| = \Omega(n^2)$. Branching would result in an overhead of $\Omega(n^2)$, leading to an $\Omega(n^{2k-O(1)})$ time algorithm, which is no good. Our solution is to still branch on each set in $\mathcal{A}$, but not start over completely in each recursive step. Rather, we keep track of a global measure of progress that amortizes our branching cost over the k recursive calls. In particular, we maintain a nonnegative *potential function* such that every time we branch on a large set $\mathcal{A}$ (and thus pay an expensive branching factor), the potential function decreases by a lot. This ensures that we do not branch expensively too often.

Our measure of progress is obtained via the tree packing result of Thorup. For a fixed min k-cut $\mathcal{S}^* = \{S_1^*, \dots, S_k^*\} \subseteq V$, we start from a spanning tree T that crosses $\mathcal{S}^*$ (i.e., connects $S_i^* \neq S_j^*$) at most $2k - 2$ times, and run the previous branching procedure by guessing one component S_i^* at a time. Our potential function is based on the current value of k and the number of edges of the current tree crossing $\mathcal{S}^*$. Whenever we branch on an expensive set $\mathcal{A}$ (say, $|\mathcal{A}| = \Omega(n^2)$), we ensure $\mathcal{A}$ contains some S_i^* that cuts many edges in T, which decreases the potential function substantially.

Connection to VC-Theory: Consider the set system $(V, \mathcal{A})$ with universe V and subsets $\mathcal{A}$. The set theory result above (saying that any set system with more than $2n$ sets has a pair of crossing sets) implies that the *dual VC dimension* of $(V, \mathcal{A})$ is at least 2 when $|\mathcal{A}| > 2|V|$; there exist two sets $A_1, A_2 \in \mathcal{A}$ such that in their Venn diagram, all four cells are nonempty. (The dual VC dimension is the standard VC dimension of the dual set system.) In § 3.1 and §5, we prove an analogous result for the dual VC dimension 3, which has not been studied to the best of our knowledge. We also prove better bounds when we want three sets $\mathcal{A}$ *partially shattered* (i.e., at least x out of 8 cells in the Venn diagram are nonempty for some $x < 8$). This saves the branching cost and allows the global inductive analysis based on the potential function.

1.2 Other Related Work

The k-Cut problem is NP-hard when k is part of the input [10]. Karger and Stein gave a randomized Monte-Carlo algorithm with runtime $O(n^{(2-o(1))k})$, via random edge-contractions [17]. On the deterministic front, Thorup improved the $O(n^{4k+o(1)})$-time algorithm of [15] to $\tilde{O}(n^{2k})$-time, based on tree packings [32]. These approaches also can be used to enumerate all minimum k-cuts, and hence to show that there are at most $O(n^{2k})$ such cuts. Our work [11] gave a deterministic algorithm that runs in time roughly $O(k^{O(k)} n^{(2\omega/3)k})$ for bounded integer weights, where $\omega \leq 2.3738$ is the matrix-multiplication constant, but it did not bound the number of minimum k-cuts. Finally, better algorithms are known for small values of $k \in [2, 6]$ [4, 13, 16, 22, 25–27]. The Karger-Stein algorithm was recently extended to Hypergraph k-Cut [5, 9], which also gave a bound on the number of minimum k-cuts. For the minimum cut ($k = 2$), the number of and the structure of approximate min-cuts also have been studied [3, 14].

Approximation algorithms. The k-Cut problem has several $2(1 - 1/k)$-approximations due to [28, 30, 31]; the more general Steiner k-cut problem also has the same approximation ratio [6]. This approximation can be extended to a $(2 - h/k)$-approximation in time $n^{O(h)}$ [35]. Chekuri et al. [7] studied the LP relaxation of [28] and gave alternate proofs for both approximation and exact algorithms with slightly improved guarantees. A fast $(2 + \varepsilon)$-approximation algorithm was also recently given by Quanrud [29]. On the hardness front, Manurangsi [23] showed that for any $\varepsilon > 0$, it is NP-hard to achieve a $(2 - \varepsilon)$-approximation in time poly(n, k) assuming the Small Set Expansion Hypothesis.

FPT algorithms. Kawarabayashi and Thorup give the first $f(OPT) \cdot n^2$-time algorithm [19] for unweighted graphs. Chitnis et al. [8] used a randomized color-coding idea to give a better runtime, and to extend the algorithm to weighted graphs. Here, the FPT algorithm is parameterized by the cardinality of edges in the optimal k-Cut, not by the number of parts k. Our past work [11, 12] gave a 1.81-approximation for k-Cut in FPT time $f(k) \cdot \text{poly}(n)$; this has since been simplified and improved to a $\frac{5}{3}$-approximation by Kawarabayashi and Lin [18].

2 PRELIMINARIES

Notations. Consider a weighted graph $G = (V, E, w)$. For any vertex set S, let $\partial_G S$ (or ∂S if G is clear from the context) denote the edges with exactly one endpoint in S. For a partition $\mathcal{S} = \{S_1, \dots, S_r\}$ of V, let $E_G(\mathcal{S}) := \bigcup_{S_i \in \mathcal{S}} \partial S_i$. For a collection of edges $F \subseteq E$, let $w(F) := \sum_{e \in F} w(e)$ be the sum of weights of edges in F. In particular, for a k-Cut solution $\{S_1, \dots, S_k\}$, the value of the solution is $w(E_G(S_1, \dots, S_k))$. Let $\kappa(G)$ be number of connected components of G.

Thorup's Tree packing. The algorithm starts from the following result of Thorup [32].

Theorem 2.1 (Thorup's Tree Packing). *Given a graph $G = (V, E, w)$ with n vertices and m edges, we can compute a collection $\mathcal{T}$ of $\tilde{O}(k^3 m)$ trees in time $\tilde{O}(k^3 m^2)$ such that for every minimum k-cut $\mathcal{S}^* = \{S_1^*, \ldots, S_k^*\}$, $\mathbb{E}_{T \in \mathcal{T}}[|E_T(S_1^*, \ldots, S_k^*)|] \leq 2k - 2$, where the expectation is taken over a uniform random tree $T \in \mathcal{T}$. In particular, there exists a tree $T \in \mathcal{T}$ with $|E_T(S_1^*, \ldots, S_k^*)| \leq 2k - 2$.*

Given a fixed minimum k-cut $\mathcal{S}^* = \{S_1^*, \ldots, S_k^*\}$, we say a tree T is a T-tree if it crosses $\mathcal{S}^*$ at most $2k - 2$ times. If we choose a T-tree $T \in \mathcal{T}$, we get the following problem: cut some $\leq 2k - 2$ edges of T and then merge the connected components into exactly k components $S_1, \ldots, S_k$ so that $E_G(S_1, \ldots, S_k)$ is minimized. Thorup's algorithm accomplishes this task using brute force: try all possible $O(n^{2k-2})$ ways to cut and merge, and output the best one. This gives a runtime of $\widetilde{O}(n^{2k-2} m)$ [32]. The natural question is: can we do better than brute-force?

Enumerating Small Cuts via Karger-Stein. Our algorithm also uses a subroutine inspired by the Karger-Stein algorithm, which can generate all α-minimum cuts in a graph, (i.e., those with cut value at most α times the min-cut in the graph) in time $\tilde{O}(n^{2\alpha})$. We note that a slight modification of this algorithm can generate all cuts of size at most $\alpha \cdot \frac{OPT_H}{h}$, where OPT_H is the minimum h-cut value, in essentially the same run-time. The following lemma is proved in §A.1.

Lemma 2.2. *Let OPT_H be the weight of the optimum minimum h-cut in $H = (V, E, w)$, and let $M := \frac{OPT_H}{h}$. For any $\alpha \geq 0$, there are at most $2^h n^{2\alpha}$ many subsets $A \subseteq V$ with $\partial_H A \leq \alpha M$. Moreover, we can output (a superset of) all such subsets in $O(2^h n^{2\alpha + O(1)})$ time, w.h.p.*

3 A REFINED BOUND FOR SMALL CUTS

Lemma 2.2 above says that when $M = \frac{OPT}{k}$, there are at most $2^k \cdot n^{2\alpha}$ cuts of size αM. The main theorem of this section, Theorem 3.4, improves this bound by a factor of $n^{\Omega(1)}$ for four different values of $\alpha = 3/2 - \gamma, 5/3 - \gamma, 2 - \gamma, 7/3 - \gamma$ for arbitrarily small constant $\gamma > 0$. This is then useful for our algorithm in §4.

Our proof strategy is the natural one: suppose $\alpha = 2 - \gamma$. Lemma 2.2 gives a bound of $2^k \cdot n^{4-2\gamma}$ cuts, whereas Theorem 3.4 below will show a bound of $O(2^k n^{3-1/4})$ cuts, which is much better. To prove it, we show that if there are more than $O(2^k n^{3-1/4})$ cuts of size αM, there are $\approx k/2$ of these cuts such that their removal yields k components, witnessing a feasible solution of weight $k/2 \cdot \alpha M = (k(2-\gamma)/2k) \cdot OPT$, which is strictly less than OPT when $k > \Theta(1/\gamma)$, yielding the contradiction.

The formal proof of Theorem 3.4 will consist of two parts. In §3.1, we will prove that if a set system has a large enough number of sets, there exist three sets whose *Venn diagram* has many nonempty regions. (If we require all regions to be nonempty, the definition is equivalent to the *dual VC dimension*; see §3.1 for details.) This implies that starting from the whole graph, in each iteration we can remove edges corresponding to three cuts and increase the number of connected components by many more than three (as many as seven). Finally, in §3.2 we will show that as long as the number of sets is large also in terms of k, we can iterate this process until we obtain a k-cut cheaper than OPT, leading to the contradiction.

3.1 Extremal Set Bounds

In this section, we show that if we have a "large" number of sets, then there exist some collection of three sets whose Venn diagram contains a many non-empty regions. Using the contra-positive, given certain forbidden configurations, the number of sets (cuts) must be small. In this section, we merely state our bounds, deferring the proofs to §5. First, a classical bound:

Claim 3.1. *Given a set system $(X, \mathcal{R})$ on $|X| = n$ elements. If the number of sets $|\mathcal{R}| > 2n - 2$ then there exists two sets $A, B \in \mathcal{R}$ that cross: i.e., all four of $A \setminus B$, $B \setminus A$, $A \cap B$, and $X \setminus (A \cup B)$ are non-empty.*

I.e., if there are more than $2n - 2$ sets, then there exist a pair of sets A, B, such that all four regions in their Venn diagram are non-empty. We now show analogous results for intersections of three sets. Our main results show that if there are "many" sets, then there exist three sets whose Venn diagram has "many" non-empty regions.

Theorem 3.2 (7-ot-of-8 Regions). *Let $(X, \mathcal{R})$ be a set system with $|X| = n$. There exists a constant $c > 0$ such that if $|\mathcal{R}| > cn^{3-1/4}$, then there exist three sets whose Venn diagram has 7 out of 8 non-empty regions.*

Theorem 3.3 (All 8 Regions). *Let $(X, \mathcal{R})$ be a set system with $|X| = n$. There exists a constant $c > 0$ such that if $|\mathcal{R}| > cn^{4-1/4}$, then there exist three sets whose Venn diagram has all 8 regions being non-empty.*

A few comments: firstly, Theorem 3.3 can be stated in terms of the *dual VC dimension* (see Definition 5.5): any set system $\mathcal{R}$ with $|\mathcal{R}| > cn^{4-1/4}$ has dual VC dimension at least 3. Secondly, a polynomial bound of $O(n^8)$ also follows from VC dimension theory, namely, the upper bound $O(n^{2^d})$ on set systems with dual VC dimension less than d. Finally, the bound cannot be improved below $\Omega(n^3)$, since if $\mathcal{R}$ is all subsets of X of size 3, then $|\mathcal{R}| = \Omega(n^3)$ but no three sets in $\mathcal{R}$ have all eight regions non-empty. As mentioned earlier, all proofs are in §5.

3.2 Number of Small Cuts

We can now use the extremal theorems to bound the number of small cuts in a graph. Fix an optimal k-cut $\mathcal{S}^* = \{S_1^*, \ldots, S_k^*\}$ where the total weight of the cut edges is $OPT := w(E(\mathcal{S}^*))$. Let $M := \frac{OPT}{k}$, and define the *normalized weight* of the cut $\partial_G A$ to be

$$\bar{w}(A) := \frac{w(\partial_G A)}{M/2}. \tag{3.1}$$

This normalization means the $\bar{w}(E(\mathcal{S}^*)) = 2k$, and hence the normalized cut value of an average part of $\mathcal{S}^*$ is 4. We prove the main result of this section upper bounding the number of small cuts, improving Lemma 2.2 that gives $2^k \cdot n^{\bar{w}(A)}$. Note that this theorem is non-constructive, but these improved bounds the number of small sets (that we compute by a variant of Karger-Stein procedure) reduces the number of required branchings in our recursive main algorithm in §4, improving the overall running time.

Theorem 3.4 (Few Small Cuts). *Fix a small enough positive constant $\gamma > 0$ and assume that $k > \Omega(1/\gamma)$. The graph G has at most:*

1. *$O(2^k n)$ subsets A with $\bar{w}(A) \leq 3 - \gamma$.*
2. *$O(2^k n^2)$ subsets A with $\bar{w}(A) \leq 10/3 - \gamma$.*
3. *$O(2^k n^{3-1/4})$ subsets A with $\bar{w}(A) \leq 4 - \gamma$.*
4. *$O(2^k n^{4-1/4})$ subsets A with $\bar{w}(A) \leq 14/3 - \gamma$.*

Proof. The proofs for all four statements follows the same outline. For the sake of a contradiction, we assume that the statement is false. We then construct a k-cut $\mathcal{S}^\dagger$ with $\bar{w}(\mathcal{S}^\dagger) < 2k$, and hence get a contradiction. We prove statement (1) here and defer the other proofs to the full version. We construct $\mathcal{S}^\dagger$ in two stages. In the first stage, we use an iterative process to get an r_0-cut $\mathcal{S}_0$ for some $r_0 \in [k-2, k]$ such that $\bar{w}(E(\mathcal{S}_0)) \leq r_0 \cdot (2 - \frac{2}{3}\gamma)$. In a second stage we then augment this to get a k-cut. In the following, let $\mathcal{A}^1$ be the set of subsets $A \subseteq V$ with $\bar{w}(A) < 3 - \gamma$, with $|\mathcal{A}^1| > c2^k n$ subsets for a large enough c.

For the first stage, let $\mathcal{S}$ be the current cut at some point in the algorithm, r is a lower bound on the current number of components. We start with a single part $\mathcal{S} = \{V\}$, and hence $r \leftarrow 1$. In each iteration, we increase r by some $r' \in \{2, 3\}$ and increase $\bar{w}(\mathcal{S})$ by $\leq r' \cdot (2 - \frac{2}{3}\gamma)$. If we can maintain this until $r \in [k-2, k]$, we have our desired cut $\mathcal{S}_0$. Each iteration is simple: while $r < k-2$, if there exists a subset $A \in \mathcal{A}^1$ that cuts two or more components in $\mathcal{S}$, then we cut the edges of $\partial_G A$ inside $\mathcal{S}$, and increase r by 2. Note that $\bar{w}(\mathcal{S})$ increases by at most $3 - \gamma \leq 2 \cdot (2 - \frac{2}{3}\gamma)$. (Technically, the number of connected components can increase by more than 2, but that only helps us, and it is still an $(r+2)$-cut.)

Otherwise, every subset $A \in \mathcal{A}^1$ either does not cut any component of $\mathcal{S}$, or cuts exactly one of them. Since we have r components, there are $\leq 2^r$ subsets that do not cut any component. Moreover, for a given component $S \in \mathcal{S}$ and a subset $\varnothing \subsetneq X \subsetneq S$, there are $\leq 2^{r-1}$ many subsets $A \subseteq V$ such that $A \cap S = X$ and A does not cut any component in $\mathcal{S} - \{S\}$. For each such subset $A \in \mathcal{A}^1$ that cuts one component $S \in \mathcal{S}$ with intersection $A \cap S = X$, add A to a bucket labeled (S, X)—so each bucket has size $\leq 2^{r-1}$. As long as $|\mathcal{A}^1| > 2^r + 2^{r-1} \cdot 2n$, there are $> 2^{r-1} \cdot 2n$ subsets cutting exactly one component of $\mathcal{S}$, so there are $> 2n$ nonempty buckets. Consequently, there exists a component $S \in \mathcal{S}$ for which there are $> 2|S|$ non-empty buckets (S, X). Put another way, the sets in A intersect S in more then $2n$ distinct ways. Now we can use the (easy) extremal result from Claim 5.2 to infer that this set system cannot be laminar, and there exist two nonempty buckets (S, X_1) and (S, X_2) such that X_1 and X_2 cross. Taking one subset from each bucket (call them A_1, A_2) and cutting the edges $\partial_G A_1 \cup \partial_G A_2$ inside $\mathcal{S}$ increases the number of components by at least 3. Hence we can increase r by 3 at the expense of increasing $\bar{w}(\mathcal{S})$ by at most $2 \cdot (3 - \gamma) = 3 \cdot (2 - \frac{2}{3}\gamma)$. Repeating this, we obtain our desired r_0-cut $\mathcal{S}_0$, for $r_0 \in [k-2, k]$. This ends the first stage.

In the second stage of the construction, we iteratively augment $\mathcal{S}_0$ to a k-cut. We let $\mathcal{S}^\dagger$ be the current cut, initialized to $\mathcal{S}_0$: for $k - r_0$ iterations, we take a subset $A \in \mathcal{A}^1$ that cuts at least one component in $\mathcal{S}^\dagger$ and cut the edges of ∂A inside $\mathcal{S}^\dagger$. (Since $|\mathcal{A}^1| > 2^k$, such a set must exist.) Thus,

$$\begin{aligned}\bar{w}(\mathcal{S}^\dagger) &\leq (r_0 - 1) \cdot (2 - \tfrac{2}{3}\gamma) + (k - r_0) \cdot (3 - \gamma) \\ &\leq k(2 - \frac{2}{3}\gamma) < 2(k-1) < 2k\end{aligned}$$

the second inequality using that preceding expression is maximized when $r_0 = k - 2$, and that $k > \frac{3}{2\gamma}$. This k-cut $\mathcal{S}^\dagger$ gives us the desired contradiction, and hence proves statement (1). The proofs of statements (2)-(4) follow the same outline, with the difference lying in the argument about how the sets in $\mathcal{A}^i$ intersect the components in $\mathcal{S}$. Indeed, we use the more sophisticated extremal bounds given by Theorems 5.3 and 5.4 instead of using the naive bound from Claim 5.2. Due to space constraints, we leave the details to the full version. ■

4 THE `MinKCut` ALGORITHM

In this section, we introduce the `MinKCut` algorithm that achieves the bound from Theorem 1.1. The algorithm `MinKCut`(G, k, F, s) outputs all the minimum k-cuts of G that can be achieved by cutting s edges of the given forest F (resulting in $s + \kappa(F)$ connected components of G) and reassembling them back to k connected components. To distinguish from k in recursive steps, let k_0 be the initial value of k such that our final goal is to find minimum k_0-cuts. Since Thorup's tree packing theorem (Theorem 2.1) gives a collection of $n^{O(1)}$ trees such that every for minimum k_0-cut in the original graph G there exists a tree in the collection that intersects the k_0-cut at most $2k_0 - 2$ times, running `MinKCut`$(G, k_0, T, 2k_0 - 2)$ for every T in the family finds every minimum k_0-cut. We need the extra generality to handle the recursive calls: we allow `MinKCut` to take in a forest F (instead of a tree) and a parameter s that may not be $2k - 2$.

An important feature of our algorithm is that it is unchanged under scaling the weights. Hence for the sake of simplicity of the analysis, throughout this section, for each call to `MinKCut` (G, k, F, s), we will assume that the total weight of the optimal k-cut is $w(E(\mathcal{S}^*)) = 2k$. This choice ensures that $w(S) = \bar{w}(S)$ for all S, where $\bar{w}$ was defined in §3.2. Finally, let $\gamma > 0$ be a small enough constant throughout this section, whose value will be specified at the end.

4.1 The `EnumCuts` Helper Function

We use the function `EnumCuts` which, given a graph G and parameters β and N, outputs all (2-)cuts in G with weight at most β (recall that we normalized weights so that $OPT = 2k$). By Lemma 2.2, there are at most $2^k n^\beta$ such cuts, and they can be found in $O(2^k n^{\beta + O(1)})$ time. In some cases, we have better bounds on the number of such cuts using the (nonconstructive) Theorem 3.4; we pass such an improved bound to the algorithm via the parameter N, and hence the algorithm returns at most N cuts.

Algorithm 1 `EnumCuts` (G, β, N)

```
Input: weighted graph $G = (V, E)$ on $n$ vertices, $N \in \mathbb{N} \cup \{\infty\}$.
Output: W.h.p., the $N$ subsets $A \subseteq V$ satisfying $w(\partial_G A) \leq \beta$ that have
the smallest values of $w(\partial_G A)$. If there are not $N$ such subsets $A$, then
output (a superset of) all such subsets $A$.
Runtime: $O(2^k n^{\beta + O(1)})$
1: Run the algorithm of Lemma 2.2 with $\alpha := \beta/2$ and let $\mathcal{P}$ be the output
2: return the $N$ subsets $A \in \mathcal{P}$ with the smallest $w(\partial_G A)$, or the entire
   $\mathcal{P}$ if $|\mathcal{P}| \leq N$
```

Algorithm 2 MinKCut (G, k, F, s)

Input: $G = (V, E)$ is an integer-weighted graph on n vertices, F is a forest, and $s + \kappa(F) \geq k$.

Output: (A superset of) valid partitions $\mathcal{S} \in \mathcal{P}_{F,s,k}$ (Definition 4.1) with weight equal to the minimum k-cut in G.

```
1:  $z(k, s) := s - (1.75 + \Theta(\gamma))k$
2:  $\mathcal{P} \leftarrow \varnothing$                    ▷ All partitions found will be stored in $\mathcal{P}$
3:  if $k < \Theta(1/\gamma)$ then
4:      $\mathcal{P} \leftarrow$ all minimum $k$-cuts, enumerated in $O(n^{2k})$ time with Karger-Stein, etc.
5:  else if $z(k, s) < 0$ then                                          ▷ Brute force
6:      $\mathcal{P} \leftarrow$ all ways to delete $s$ edges in $F$ and merge into $k$ components.
        ▷ $O(k^{s+\kappa(F)}n^{s+O(1)})$ time
7:  else                                                     ▷ Recursive algorithm
8:      $\mathcal{A}^0 \leftarrow \{A \subseteq V : |\partial_F A| = 0\}$                   ▷ $2^{\kappa(F)}$ of them
9:      $\mathcal{A}^1 \leftarrow \{A \subseteq V : |\partial_F A| = 1\}$.               ▷ $2^{\kappa(F)+1}n$ of them
10:     $\mathcal{A}^2 \leftarrow$ EnumCuts$(G, 3-\gamma, \Theta(2^k n)) \cap \{A \subseteq V : |\partial_F A| = 2\}$   ▷
        $\Theta(2^k n)$ bound from Thm 3.4
11:     $\mathcal{A}^3 \leftarrow$ EnumCuts$(G, 4-\gamma, \Theta(2^k n^{3-1/4})) \cap \{A \subseteq V : |\partial_F A| = 3\}$
        ▷ $\Theta(2^k n^{3-1/4})$ from Thm 3.4
12:     $\mathcal{A}^4 \leftarrow$ EnumCuts$(G, 14/3-\gamma, \Theta(2^k n^{4-1/4})) \cap \{A \subseteq V : |\partial_F A| = 4\}$
        ▷ $\Theta(2^k n^{4-1/4})$ from Thm 3.4
13:     for $\ell \in [5, s]$ do
14:         $\mathcal{A}^\ell \leftarrow$ EnumCuts$(G, \beta_\ell, \infty) \cap \{A \subseteq V : |\partial_F A| = \ell\}$, where ▷
            $|\mathcal{A}^\ell| = O(2^k n^{\beta_\ell})$ by Lem 2.2
                $\beta_\ell := g_{k,s}^{-1}(\ell) \iff \ell = g_{k,s}(\beta_\ell)$  (see (4.2))
15:     for $\ell \in [0, s]$, $A \in \mathcal{A}^\ell$ do
16:         let $G' \leftarrow G[V - A]$ and $F' \leftarrow F[V - A]$
17:         let $s' \leftarrow s - |\partial_F A|$
18:         Recursively call MinKCut$(G', k-1, F', s')$ and add its output to $\mathcal{P}$
19: return $\{\mathcal{S} \in \mathcal{P} : w(E_G[\mathcal{S}]) = \min_{\mathcal{S}' \in \mathcal{P}} w(E_G[\mathcal{S}'])\}$
```

4.2 The Algorithm Description

The algorithm MinKCut is formally given as pseudocode, but let us explain the main steps. The algorithm is recursive: at some stage, we are given a current graph G (think of this as the original graph with some vertices carved away) and a forest F (think of this as T induced by the current graph). We want to delete s edges from F, then combine the resulting pieces into a k-partition of $V(G)$ to obtain an optimal k-cut. The algorithm considers the ways in which a generic part S_i^* from the optimal cut $\mathcal{S}^* = \{S_1^*, S_2^*, \ldots, S_k^*\}$ cuts the forest.

For $\ell = 0, \ldots, s$, the set $\mathcal{A}^\ell$ is supposed to contain candidates for S_i^* that cut F in ℓ edges. The algorithm tries each $A \in \mathcal{A}^\ell$ as one of the optimal parts, and recursively run MinKCut $(G[V - A], k-1, F[V-A], s-\ell)$. Therefore if we are able to show that there exists S_i^* and ℓ such that $S_i^* \in \mathcal{A}^\ell$, we will try S_i^* and call MinKCut $(G[V \setminus S_i^*], k-1, F[V \setminus S_i^*], s - \partial_F(S_i^*))$, and since $\mathcal{S}^* \setminus \{S_i^*\}$ is an optimal $(k-1)$-cut in $G[V \setminus S_i^*]$, the recursive algorithm will eventually output $\mathcal{S}^*$.

By naively setting $\mathcal{A}^\ell$ to be the set of all subsets $A \subseteq V$ that cross F exactly ℓ times, $|\mathcal{A}^\ell| \leq 2^{O(k)} n^\ell$ for all ℓ. Since branching on $A \in \mathcal{A}^\ell$ drops s by ℓ, together with the fact that $\log_n |\mathcal{A}^\ell| \leq \ell + o(1)$, we can think s as a *potential* and easily argue that the total running time is $n^{s+O(1)}$. However, since s could be $2k-2$ for a T-tree, we may not win anything.

To get faster running time, we further restrict $\mathcal{A}^\ell$ to subsets that induce cuts of small weights. For small $\ell = 2, 3, 4$, we restrict to $\mathcal{A}^2, \mathcal{A}^3, \mathcal{A}^4$ to subsets whose cut weight in G is at most $3 - \gamma, 4 - \gamma, 14/3 - \gamma$ respectively. The standard Karger-Stein bound (Lemma 2.2) only guarantees upper bounds worse than $O(2^k n^\ell)$, but our new bounds on the small cuts (Theorem 3.4) bound their numbers by $O(2^k n)$, $O(2^k n^{3-1/4})$, $O(2^k n^{4-1/4})$, so $\log_n |\mathcal{A}^\ell| < \ell$ for $\ell = 2, 3, 4$. For $\ell \geq 5$, we set $\mathcal{A}^\ell$ more conservatively so that it contains sets of cut weight at most $\beta_\ell < \ell$, so $\log_n |\mathcal{A}^\ell| < \ell$ from the Karger-Stein bound. Our analysis will show that this choice of $\mathcal{A}^\ell$'s will make sure that S_i^* is contained in one of them for correctness. Using a different potential function, it will also prove the desired running time.

Before we go further, let's give a convenient definition:

Definition 4.1 (Valid Partitions). *Given a forest $F = (V, E_F)$ with $\kappa(F)$ connected components, and two integers k, s with $s \geq k - \kappa(F)$, a partition $\mathcal{S} = \{S_1, \ldots, S_k\}$ of V is (F, s, k)-valid if it is formed by deleting exactly s edges in the forest F and merging the resulting $s + \kappa(F)$ connected components of F into k components. Let $\mathcal{P}_{F,s,k}$ be all such (F, s, k)-valid partitions.*

4.3 Overview of the Parameter Selection and Analysis

Suppose we are solving k-cut on G, where we want to delete a total of s edges in the forest F. By our assumption about scaling the total weight, observe that the target minimum k-cut $\mathcal{S}^* = \{S_1^*, \ldots, S_k^*\}$ satisfies $\sum_i w(\partial_F S_i^*) = 4$ and $\sum_i |\partial_F S_i^*| = 2s$. Suppose we have some "good" optimal component S_i^* that we'd like to guess, and then recurse on the graph $G[V \setminus S_i^*]$. What properties would we like S_i^* to have? Here are some observations.

(1) For brevity, define $w := w(\partial_G S_i^*)$, and $\ell := |\partial_F S_i^*|$. Then by running EnumCuts with parameter w, we can enumerate a set of $O(2^k n^w)$ subsets $A \subseteq V$ such that one of them equals this targeted S_i^*. Then, by branching on each of these subsets, we pay a multiplicative branching overhead of $O(2^k n^w)$. Moreover, we can use the extremal bounds of Theorem 3.4 to do even better: if $w < 14/3 - \gamma$, then we can enumerate a set much smaller than $O(2^k n^w)$. (For example, if $w \leq 3 - \gamma$, then our set has size only $O(2^k n)$, using Theorem 3.4.)

(2) Secondly, we reduce the parameter s in a recursive call by $\ell = |\partial_F S_i^*|$. Intuitively, a set S_i^* with $\ell \gg w$ is a good candidate, since we pay a comparatively small branching factor for deleting many edges in F.

(3) So let's keep track of how much we *gain relative to the brute-force algorithm.* Namely, if we pay a branching cost of n^β to delete ℓ edges (we ignore any constants or $f(k)$ in front of the n^β), then since brute-force requires n^ℓ time to guess the ℓ edges in $\partial_F S_i^*$, we are a factor $n^{\ell-\beta}$ "ahead" of brute force. In this case, we measure our *gain* as the quantity $\ell - \beta$. And we will never branch on a set with a negative gain.

(4) Finally, when might we *not* be able to make any positive gain? Suppose half the S_i^* have $(w = 3, \ell = 2)$ and the

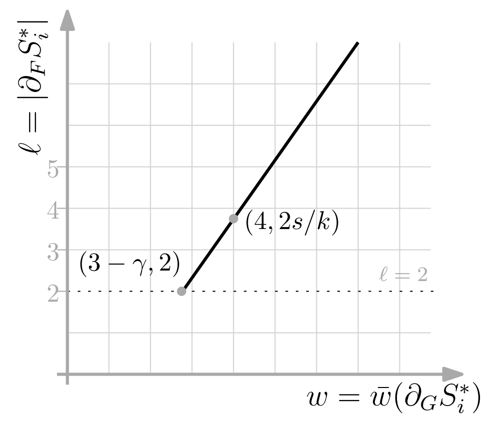

Figure 2: The line $\ell = g_{k,s}(w)$.

other half have ($w = 5, \ell = 5$), which means that $s = 1.75k$. The extremal bounds from Theorem 3.4 and Lemma 2.2 only guarantee us branching factors of $O(2^k n^2)$ and $O(2^k n^5)$ respectively—in both cases there is zero gain. Therefore, if $s \le 1.75k$, then in the worst case, there may be no choice of set that gives us a positive gain, and we might as well do brute-force. Conversely, we will show below that as long as $s \ge (1.75 + \Theta(\gamma))k$, positive gain is always possible.

Now assume $s \ge (1.75+\Theta(\gamma))k$, and imagine the two-dimensional plane $\mathbb{R}^2$ where we plot the (w, ℓ) values for the different sets $S_i^* \in \mathcal{S}^*$ as points. The above discussion suggests a natural strategy of choosing a S_i^*, which is to draw a line with positive slope and choose S_i^* corresponding to a point above the line. To guarantee the existence of a point above the line, we make sure that the lines pass the centroid of these k points $(4, 2s/k)$. Two points $(4, 2s/k)$ and $(3-\gamma, 2)$ (instead of $(3, 2)$ for technical reasons) decide the following line.

$$\ell = g_{k,s}(w) := \left(\left(\frac{2s}{k} - 2\right)/(1+\gamma)\right) w + 8/(1+\gamma) + \frac{s}{k}\left(2 - 8/(1+\gamma)\right) \qquad (4.2)$$

$$= \left(\frac{2s}{k} - 2 - O(\gamma)\right) w + \left(8 - \frac{6s}{k} + O(\gamma)\right). \qquad (4.3)$$

See Figure 2. The line passes through the centroid, so we get the following statement, whose (easy) proof we omit:

Claim 4.2. *There exists a set S_i^* such that $\ell \ge g_{k,s}(w)$.*

(We handle the case $\ell < 2$ separately, so only consider the segment from $\ell \ge 2$). Consider a S_i^* whose (w, ℓ) point is above the line guaranteed by Claim 4.2. Note that the line is above $(5, 5)$ as long as $s \ge (1.75+\Theta(g))k$, and the slope is $2s/k - 2 - O(\gamma) \ge 1.5 - O(\gamma)$ so for large values of $w, \ell > 5$, we have $w < \ell$, and the branching cost $O(n^\beta)$ with $\beta = w$ by Lemma 2.2 is good. For small values, ℓ can be less than w, but our improved bounds Theorem 3.4 make sure that the branching factor is $O(n^\beta)$ for $\beta < \ell$. In any case, our gain $\ell - \beta$ is strictly positive.

How do we analyze the performance of this algorithm? Suppose we run this process from $(k_0, 2k_0-2)$ and end up doing brute-force at the point (k, s) with total gain g over the entire process. This means that we have paid a total branching cost of $f(k)n^{(2k_0-2)-s-g}$ so far, leaving a forest F with $\sum_i \ell \le s$. We will then brute-force over the remaining $\le s$ edges for a total running time of $f(k)n^{2k_0-2-g}$. Thus, to obtain the fastest k_0-cut possible, it is clear that our objective should be to maximize our total gain over the course of the process (which we can terminate at any point). Let us define the following quantity that measures how large s is compared to $(1.75 + \Theta(\gamma))k$.

$$z(k, s) := s - (1.75 + \Theta(\gamma))k. \qquad (4.4)$$

We can view z as a *budget* that starts out as $(0.25 - \Theta(\gamma))k_0$ and eventually becomes zero. We are interested in the amount of gain per unit of budget, which may depend on the current budget.

Recall that whenever $w \le 14/3 - \gamma$, our bounds in Theorem 3.4 provide better bounds than what is guaranteed in Lemma 2.2. We now formalize this below. Define the function

$$d(w) := \begin{cases} 1 & \text{if } w \le 3-\gamma, \\ 3 - 1/4 & \text{if } 3-\gamma < w \le 4-\gamma, \\ 4 - 1/4 & \text{if } 4-\gamma < w \le 14/3 - \gamma, \text{ and} \\ w & \text{if } w > 14/3 - \gamma. \end{cases}$$

By Theorem 3.4, if we branch on (w, ℓ), then the gain is $\ell - d(w)$ and the difference in budget is $z(k, s) - z(k-1, s-\ell) = \ell - (1.75 + \Theta(\gamma))$. We now prove the lemma below that bounds the budget-gain ratio, whose proof is deferred to the full version.

Lemma 4.3. *Consider the current state (k, s) with budget $z = z(k, s) = s - (1.75+\Theta(\gamma))k$, where $s \ge (1.75+\Theta(\gamma))k$, and $k \le k_0$. Every point (w, ℓ) with $\ell \ge 2$ guaranteed by Claim 4.2 satisfies*

$$\frac{\ell - d(w)}{\ell - (1.75 + \Theta(\gamma))} \ge \min\left(\frac{1}{9}, \frac{4z}{6.5z + 4.875k}(1 - O(\gamma))\right)$$
$$\ge \min\left(\frac{1}{9}, \frac{4z}{6.5z + 4.875k_0}(1 - O(\gamma))\right).$$

Note that to handle the guessing of the coordinates (w, ℓ), the algorithm essentially *guesses all values of ℓ*. Namely, for each integer value ℓ, the algorithm enumerates all sets $A \subseteq V$ such that $(w(A), \ell)$ is on or above the line, which is good enough.

4.4 Correctness

We prove that `MinKCut`(G, k, F, s) finds every minimum k-cut that can be formed by deleting s edges in F and merging the $s + \kappa(F)$ connected components of F into k components. Fix any such minimum k-cut $\mathcal{S}^*$. If $s(k, s) < 0$, then line 6 enumerates over all possible valid partitions, and hence $\mathcal{S}^*$ will be found. Otherwise, the following lemma shows that at least one component $S_i^* \in \mathcal{S}^*$ will be in $\mathcal{A}^\ell$ for some $\ell \in [0, s]$ so that we can recurse on $G \setminus S_i^*$ and eventually find $\mathcal{S}^*$.

Lemma 4.4. *If the* **else** *branch in* `MinKCut` *is taken (i.e., $z(k, s) \ge 0 \iff s \ge (1.75 + \Theta(\gamma))k$), then there exists an $\ell \in [0, s]$ and $A \in \mathcal{A}^\ell$ such that $A \in \mathcal{S}^*$.*

Proof. If there exists S_i^* whose ℓ is 0 or 1, $S_i^* \in \mathcal{A}^\ell$. When every S_i^* has $\ell \ge 2$, by Claim 4.2, there exists S_i^* such that $\ell \ge g_{k,s}(w) \iff g_{k,s}^{-1}(\ell) \ge w$. If $\ell \ge 5$, $\beta_\ell = g_{k,s}^{-1}(\ell) \ge w$, and since

$\mathcal{A}^\ell$ contains all cuts that crosses F in ℓ edges and has weight at most β_ℓ in G, $S_i^* \in \mathcal{A}^\ell$. For $\ell = 2, 3, 4$, let $\beta_2 = 3 - \gamma$, $\beta_3 = 4 - \gamma$, $\beta_4 = 14/3 - \gamma$. Since $\ell \le g_{k,s}(\beta_\ell) \iff \beta_\ell \ge g_{k,s}^{-1}(\ell)$, so again $S_i^* \in \mathcal{A}^\ell$. ■

4.5 Running Time

We now proceed to the running time analysis. Motivated by Lemma 4.3, we define the following *potential function*:

$$\Phi(k,s) := \begin{cases} \int_{t=0}^{z(k,s)} \min\left(\frac{1}{9}, \frac{4t}{6.5t+4.875k_0}(1-\Theta(\gamma))\right) dt & \text{if } z(k,s) \ge 0 \\ 1 & \text{otherwise.} \end{cases} \tag{4.5}$$

The function has a discontinuity at $z(k,s) = 0$, but this will be convenient later on. Below, we list the technical properties that we need for Φ, whose routine proofs are deferred to the full version.

Lemma 4.5. *For values (k,s) such that $z(k,s) \ge 0$, the function $\Phi(k,s)$ satisfies:*

1. $\Phi(k,s) \le s$.
2. $\Phi(k,s) \le \Phi(k,s-1) + 1$.
3. $\Phi(k,s) \le \min\{\Phi(k-1,s), \Phi(k-1,s-1)\}$.
4. *For all (w,ℓ) satisfying the condition in Claim 4.2,*
$$\Phi(k,s) \le \Phi(k-1, s-\ell) + \ell - d(w).$$
5. $g_{k,s}^{-1}(s) \le s - \Phi(k,s) + O(1)$.

The running time of `MinKCut` is bounded by the following recursive analysis.

Lemma 4.6. *There exist $c_1 = O(1)$ and $c_2 = O(1/\gamma)$ such that Algorithm* `MinKCut`(G,k,F,s) *takes time*

$$(c_1\, 2^{k+s+\kappa(F)})^k \cdot n^{s-\Phi(k,s)+c_2}.$$

Proof. If $k < \Theta(1/\gamma)$, we take the **if** branch, and Karger-Stein takes time $O(n^{2k}) = O(n^{O(1/\gamma)})$. This meets the bound for $c_2 = O(1/\gamma)$, using that $\Phi(k,s) \le s$ from Lemma 4.5.

If $s < (1.75 + \Theta(\gamma))k$, we take the **else if** branch, and the enumeration on line 6 takes time

$$k^{s+\kappa(F)} n^{s+O(1)} \le \left(2^{k+s+\kappa(F)}\right)^k \cdot n^{s-\Phi(k,s)+O(1)}.$$

Here, we use that if $z(k,s) < 0$, then $\Phi(k,s)$ is defined to be 1.

We now focus on the case $s \ge (1.75 + \Theta(\gamma))k$, applying induction on k. Since all the recursive calls happen for each $A \in \mathcal{A}^\ell$ for some $\ell \in [0,s]$, we examine each $\mathcal{A}^\ell$ and bound the running time of the recursive calls based on $A \in \mathcal{A}^\ell$.

Recall that β_ℓ was defined to be $g_{k,s}^{-1}(\ell)$ for $\ell \ge 5$. Extend this definition so that $\beta_0 = 0, \beta_1 = \beta_2 = 1, \beta_3 = 3 - 1/4, \beta_4 = 4 - 1/4$. Observe the following:

(1) $|\mathcal{A}^\ell| \le O(2^k n^{\beta_\ell})$ by Lemma 2.2 and Theorem 3.4.
(2) For every $A \in \mathcal{A}^\ell$ where $\ell \ge 0$, we define $s' = s - |\partial_F A|$ and $F' = F[V - A]$ in lines 16–17. This ensures that $s' + \kappa(F') \le s + \kappa(F)$.

The number of recursive calls on line 18 is at most $|\mathcal{A}^\ell|$, and by induction, the recursive call corresponding to $A \in \mathcal{A}^\ell$ has runtime $(c_1 2^{(k-1)+s'+\kappa(F')})^{k-1} \cdot n^{s'-\Phi(k-1,s')+c_2}$. The total time of these $|\mathcal{A}^\ell| \le O(2^k n^{\beta_\ell})$ recursive calls for this value is ℓ is at most

$$O(2^k n^{\beta_\ell}) \cdot (c_1 2^{k+s+\kappa(F)})^{k-1} \cdot n^{s'-\Phi(k-1,s')+c_2} \tag{4.6}$$

Let us focus on the exponent of n in this expression, $\beta_\ell + s' - \Phi(k-1,s') + c_2$. Lemma 4.5 shows that $\beta_\ell + s' - \Phi(k-1,s') \le \Phi(k,s)$.

Substituting into (4.6), we get that for each $\ell \in [2,s]$, the recursive calls take total time

$$O(2^k) \cdot (c_1 2^{k+s+\kappa(F)})^{k-1} \cdot n^{s-\Phi(k,s)+c_2}. \tag{4.7}$$

Now to bound the time for the nonrecursive part of the algorithm. By the definition of `EnumCuts` and Lemma 2.2, the runtime for lines 11 to 14 is dominated by the runtime for `EnumCuts`(G, β_s, ∞), which is

$$n^{\beta_s + O(1)} \le n^{s-\Phi(k,s)+c_2}, \tag{4.8}$$

for large enough constant c_2, using item (6) of Lemma 4.5.

Using (4.7) to bound the time for recursive calls, and (4.8) for the non-recursive part, the total time of `MinKCut` is at most

$$\begin{aligned} s \cdot O(2^k) \cdot (c_1 2^{k+s+\kappa(F)})^{k-1} \cdot n^{s-\Phi(k,s)+c_2} \\ \le (c_1 2^{k+s+\kappa(F)})^k \cdot n^{s-\Phi(k,s)+c_2}, \end{aligned}$$

for large enough c_1. This completes the induction and finishes the lemma. ■

Now using Thorup's tree packing result from Theorem 2.1, we obtain an instance (G, k_0, F, s) with a single tree and hence $\kappa(F) = 1$, where the optimal solution cuts this tree in at most $2k_0 - 2$ edges. We can try all choices of s from $k_0 - 1$ to $2k_0 - 2$. By item (2) of Lemma 4.5, the value $s - \Phi(k_0, s)$ is increasing in s, so the runtime will be maximum when $s = 2k_0 - 2$. Our final running time is therefore

$$2^{O(k_0^2)} \cdot n^{s-\Phi(k_0, 2k_0-2)+O(1)}.$$

It remains to compute $\Phi(k_0, 2k_0 - 2)$. If we let $f(t) := \min\left(\frac{1}{9}, \frac{4t}{6.5t+4.875k_0}(1-\Theta(\gamma))\right)$, then the two terms inside the $\min(\cdot,\cdot)$ are equal when t equals $T := \frac{4.875}{29.5(1-\Theta(\gamma))}k_0$. That is, $f(t) = 1/9$ for $t \ge \frac{4.875}{29.5(1-\Theta(\gamma))}k_0$ and $f(t) = \frac{4t}{6.5t+4.875k_0}(1-\Theta(\gamma))$ otherwise. Integrating $\Phi(k_0, s)$ for $z = z(k_0, s) < k_0 T$, we obtain

$$\begin{aligned} \Phi(k_0, s) &= \int_{t=0}^{z} \frac{4t}{6.5t + 4.875k_0}(1-\Theta(\gamma)) \\ &= \frac{(1-\Theta(\gamma))}{1.625}(z/k_0 - 0.75\ln(4z/k_0+3) + 0.75\ln 3)k_0. \end{aligned}$$

Therefore, for a value of s satisfying $z(k_0, s) > k_0 T$, we have

$$\Phi(k_0, s) = \frac{(1-\Theta(\gamma))}{1.625}(T - 0.75\ln(4T+3) + 0.75\ln 3)k_0 + \frac{1}{9}(z - k_0 T).$$

Plugging in $s := 2k_0 - 2$, we obtain

$$\begin{aligned} &\Phi(k_0, 2k_0 - 2) \\ &\quad = \frac{1}{1.625} \cdot \left(\frac{4.875}{29.5} - 0.75\ln\left(4 \cdot \frac{4.875}{29.5} + 3\right) + 0.75\ln 3\right)k_0 \\ &\quad + \frac{1}{9}\left(0.25k_0 - \frac{4.875}{29.5}k_0\right) - O(\gamma k_0) \\ &\quad \approx (0.0192055688 - O(\gamma))k_0. \end{aligned}$$

Assuming that k_0 is large enough, set $\gamma > 0$ so that

$$\begin{aligned} s - \Phi(k_0, 2k_0-2) + O(1) &\approx (2k_0 - 2) - (0.0192055688 - O(\gamma))k_0 + O(1) \\ &\le 1.981k_0. \end{aligned}$$

Finally, plugging in this bound of $s - \Phi(k_0, 2k_0 - 2) + O(1)$ into Lemma 4.6 proves Theorem 1.1.

5 THE EXTREMAL PROBLEM

In this section we will prove the extremal theorems (Theorems 3.2 and 3.3) from §3.2 about the size of certain set systems that do not have certain intersection patterns. The results of this section may be read independently of the other sections, if desired.

Since we will be talking about two kinds of sets, one over a universe X and another over a small universe $[k]$, we use the vocabulary of range spaces commonly used in the computational and discrete geometry literature. A *range space* $(X, \mathcal{R})$ is just a set system over the ground set X, where the sets in $\mathcal{R}$ are called *ranges*. The following notion of *representation* will be useful to state our results formally and concisely: please refer to Figure 3 as well.

Definition 5.1 (Represent). *Let X be a universe of elements. Let $k \in \mathbb{N}$ be a positive integer and $\mathcal{S} \subseteq 2^{[k]}$ be a set of subsets of $[k]$. For given ranges $R_1, \ldots, R_k \subseteq X$ and subset $S \in \mathcal{S}$, we say that $R_1, \ldots, R_k$ **witnesses** (k, S) if there exists element $x \in X$ such that for all indices $i \in [k]$,*

$$i \in S \iff x \in R_i.$$

*We say that a k-tuple of ranges $(R_1, \ldots, R_k)$ α-**witnesses** $(k, \mathcal{S})$ in X if they witness at least an α-fraction of subsets $S \in \mathcal{S}$. We drop the "in X" if the universe X is clear from context.*

*Now consider a range space $(X, \mathcal{R})$. We say that $\mathcal{R}$ α-**represents** $(k, \mathcal{S})$ in X if there exists subsets $R_1, \ldots, R_k \in \mathcal{R}$ that α-witness $\mathcal{S}$.*

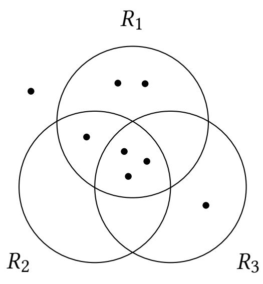

Figure 3: *A Venn diagram representation of three ranges R_1, R_2, R_3 on a universe X of 8 elements. The tuple (R_1, R_2, R_3) $5/8$-witnesses $(3, 2^{[3]})$, $4/7$-witnesses $(3, 2^{[3]} \setminus \{\varnothing\})$, and 1-witnesses $(3, \{\varnothing, \{1\}, \{3\}, \{1, 2\}, \{1, 2, 3\}\})$.*

Again, Figure 3 may be useful in understanding the notion. Readers familiar with the notion of VC-dimension will recognize that if some range space $(X, \mathcal{R})$ 1-represents the set system $(k, 2^{[k]})$, this corresponds to the *dual range space* $(\mathcal{R}, \mathcal{R}^*)$ having VC dimension at least k. The notion of witnesses and representation we define above is therefore a more refined notion than the usual notion of (dual) shattering. While we do not need familiarity with any of these connections, we refer to, e.g., [24, §5.1] for more details on VC dimension and dual shatter functions.

Claim 5.2. *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. If $|\mathcal{R}| > 2(n-2)$ then there exists two ranges in $\mathcal{R}$ that cross.*

In this section, we prove Theorems 3.2 and 3.3 on extremal set bounds. First, let us restate them in terms of the above notation.

Theorem 5.3 (7/8-representation). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{3-1/4}$, then $\mathcal{R}$ $7/8$-represents $(3, 2^{[3]})$.*

Theorem 5.4 (1-representation). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{4-1/4}$, then $\mathcal{R}$ 1-represents $(3, 2^{[3]})$.*

We can rephrase Theorem 5.4 in terms of the concept *dual VC dimension* from VC dimension theory. The following is a simple, equivalent definition of dual VC dimension in terms of our notion of representation:

Definition 5.5 (Dual VC dimension). *A range space $(X, \mathcal{R})$ has* dual VC dimension *d if d is the largest integer such that $\mathcal{R}$ 1-represents $(d, 2^{[d]})$.*

Thus, 5.4 implies the following extremal bound concerning range spaces of dual VC dimension at least 3:

Corollary 5.6. *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{4-1/4}$, then $\mathcal{R}$ has dual VC dimension at least* 3.

We first develop some basic tools in §5.1 to give weaker results; the proofs of the above theorems then appear in §5.2. Before we proceed, here are some standard notions and a basic result about the maximum number of sets in a laminar set system.

Definition 5.7 (Crossing Sets). *Let X be a universe of elements. We say that two ranges $A, B \subseteq X$ **cross** in X if (A, B) 1-witnesses $(2, 2^{[2]})$ in X. That is, there is at least one element in each of the sets $A \setminus B$, $B \setminus A$, $A \cap B$, and $\overline{A \cup B}$.*

Definition 5.8 (Cutting Sets). *Let X be a universe of elements. We say that range A **cuts** range B if there is an element in each of the sets $A \cap B$ and $B \setminus A$.*

Using these definitions, we can easily prove Claim 5.2 restated below.

Claim 5.2. *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. If $|\mathcal{R}| > 2(n-2)$ then there exists two ranges in $\mathcal{R}$ that cross.*

Proof. Fix an arbitrary element $x \in X$. Let $\mathcal{R}'$ be the set $\{R : R \in \mathcal{R}, x \notin R\} \cup \{\overline{R} : R \in \mathcal{R}, x \in R\}$, so that $(X \setminus \{x\}, \mathcal{R}')$ is a range space. We have $|\mathcal{R}'| \geq |\mathcal{R}|/2$, since for every pair of ranges $R, \overline{R}$ satisfying $\{R, \overline{R}\} \cap \mathcal{R} \neq \varnothing$, one of $R, \overline{R}$ is in $\mathcal{R}'$. Since $|\mathcal{R}'| \geq |\mathcal{R}|/2 > n - 2 = |X \setminus \{x\}| - 1$, $\mathcal{R}$ is not a laminar set of ranges, so there exists $A, B \in \mathcal{R}'$ such that $A \setminus B$, $B \setminus A$, and $A \cap B$ are nonempty. Since $x \in \overline{A \cup B}$, the sets A, B cross in X. It is easy to see that any of $(A, \overline{B})$, $(\overline{A}, B)$, and $(\overline{A}, \overline{B})$ also cross in X. Since for one of these pairs, both ranges are in R, the claim follows. ■

5.1 Warm-up

In this section, we prove simpler results while developing some basic machinery. The ideas here are similar to those used, e.g., in proofs of the Sauer-Shelah theorem about the VC dimension of set systems.

Lemma 5.9 (Extension Lemma). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. Suppose the following statement is true, for some fixed constant c, nonnegative integers d, k, r, and subset $\mathcal{S} \subseteq 2^{[k]}$:*

1. *If $|\mathcal{R}| > cn^d$, then $\mathcal{R}$ $(r/|\mathcal{S}|)$-represents $(k, \mathcal{S})$.*

Then there exists constant c' depending on c, d such that the following statement is also true:

2. *If $|\mathcal{R}| > c'n^{d+1}$, then $\mathcal{R}$ $(r+1/|\mathcal{S}|)$-represents $(k, \mathcal{S})$.*

Proof. It suffices to prove statement (2) for n large enough: by setting $c' \geq 2^{n_0}$ for some $n_0 \geq 0$, the statement is true for all $n \leq n_0$, since $|\mathcal{R}| > c'n^{d+1}$ is impossible if $|X| \leq n_0$.

We apply induction for large enough n. Pick an arbitrary $x \in X$, and let $\mathcal{R}_{\text{both}}$ be the set of all R satisfying $R \not\ni x$, $R \in \mathcal{R}$, and $R \cup \{x\} \in \mathcal{R}$.

First, suppose that $|\mathcal{R}_{\text{both}}| > cn^d$. Then, by the assumption in the lemma, there exists a $r/|\mathcal{S}|$-witness $(R_1, R_2, \ldots, R_k)$ for $(k, \mathcal{S})$. If $(R_1, R_2, \ldots, R_k)$ also $(r+1)/|\mathcal{S}|$-witnesses $(k, \mathcal{S})$, then we are done; otherwise, let $S^* \in \mathcal{S}$ be a subset not witnessed by $(R_1, R_2, \ldots, R_k)$. For each $i \in [k]$, define R_i' to be R_i if $x \notin S^*$, and $R_i \cup \{x\}$ otherwise; note that $R_i' \in \mathcal{R}$ always. Then, every subset $S \in \mathcal{S}$ witnessed by $(R_1, R_2, \ldots, R_k)$ is also witnessed by $(R_1', R_2', \ldots, R_k')$, and moreover, S^* is now witnessed by $(R_1', R_2', \ldots, R_k')$. Therefore, $(R_1', R_2', \ldots, R_k')$ $(r+1)/|\mathcal{S}|$-represents $(k, \mathcal{S})$.

Otherwise, suppose that $|\mathcal{R}_{\text{both}}| \leq cn^d$. Define $\mathcal{R}_{\text{one}}$ to be the set of all R satisfying $R \not\ni x$ and exactly one of $R \in \mathcal{R}$ and $R \cup \{x\} \in \mathcal{R}$; observe that $|\mathcal{R}_{\text{one}}| + 2 \cdot |\mathcal{R}_{\text{both}}| = |\mathcal{R}|$, and that $(X \backslash x, \mathcal{R}_{\text{one}} \uplus \mathcal{R}_{\text{both}})$ is a range space with $|X \backslash x| = n-1$. If there exists a $(r+1)/|\mathcal{S}|$-witness for $\mathcal{R}_{\text{one}} \uplus \mathcal{R}_{\text{both}}$, then clearly, this tuple also $(r+1)/|\mathcal{S}|$-witnesses $\mathcal{R}$, so assume not. Applying induction on the contrapositive of statement (2) gives $|\mathcal{R}_{\text{one}} \uplus \mathcal{R}_{\text{both}}| \leq c'(n-1)^{d+1}$. Therefore,

$$|\mathcal{R}| = |\mathcal{R}_{\text{one}} \uplus \mathcal{R}_{\text{both}}| + |\mathcal{R}_{\text{both}}| \leq c'(n-1)^{d+1} + cn^d$$
$$\leq c'n^{d+1} - c' \cdot Cn^d + cn^d,$$

for some constant $C \geq 0$ depending on d, and assuming that n is large enough. Setting $c' \geq c/C$ gives

$$|\mathcal{R}| \leq c'n^{d+1} - c' \cdot Cn^d + cn^d \leq c'n^{d+1},$$

so the assumption of statement (2) is false, completing the induction. ■

Lemma 5.10. *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists constants c_1, c_2, c_3, c_4 such that:*

1. *If $|\mathcal{R}| > c_1 n$, then $\mathcal{R}$ $5/8$-represents $(3, 2^{[3]})$.*
2. *If $|\mathcal{R}| > c_2 n^2$, then $\mathcal{R}$ $6/8$-represents $(3, 2^{[3]})$.*
3. *If $|\mathcal{R}| > c_3 n^3$, then $\mathcal{R}$ $7/8$-represents $(3, 2^{[3]})$.*
4. *If $|\mathcal{R}| > c_4 n^4$, then $\mathcal{R}$ 1-represents $(3, 2^{[3]})$.*

Proof. We first prove (1). Let $c_1 := 16$. Since $|\mathcal{R}| > 16n$, there exists A, B that cross, by Claim 5.2. If C is any set other than the union of some subset of $\{A \backslash B, B \backslash A, A \cap B, \overline{A \cup B}\}$, then (A, B, C) is a $5/8$-witness for $(3, 2^{[3]})$. There are at most $2^4 = 16$ such unions and $|\mathcal{R}| > 16n \geq 16$, so there is such a choice for C.

This serves as the base case: we can now use the Extension Lemma above: Statement (2) follows from (1) and an application of Lemma 5.9 with $d := 1$, $k := 3$, $r := 5$, and $\mathcal{S} := 2^{[3]}$. Likewise, statement (3) follows from (2) with $d := 2$ and $r := 6$, and statement (4) follows from (3) with $d := 3$ and $r := 7$. ■

Observe that statement (4) of Lemma 5.10 is weaker than Theorem 5.4 by an $n^{1/4}$ factor. If we were to stick to the proof strategy in Lemma 5.10, we would want to prove improved statements (1) to (3) in order to get an improved statement (4). However, statements (1) and (2) of Lemma 5.10 are tight, and become false if we replace $|\mathcal{R}| \geq c_d n^d$ with $|\mathcal{R}| \geq c_d n^{d-\varepsilon}$ for $d = 1, 2$ and some $\varepsilon > 0$. For example, if $\mathcal{R}$ is all ranges of size 2 in X, then $\mathcal{R}$ has size $\binom{n}{2}$ but still does not $6/8$-represent $(3, 2^{[3]})$. However, statement (3) does not suffer from a simple, matching lower bound, and indeed, as promised by Theorem 5.3, the bound can be improved by $n^{1/4}$. Our main focus will be to prove this improved upper bound on statement (3), from which the improved statement (4) from Theorem 5.4 will follow.

5.2 Proof of the Extremal Theorems

The main idea behind our improved proofs is prove statements that give a finer-grained control over the occupied regions of the Venn diagrams. Since some regions of the Venn diagram are "easy" to achieve (say the intersection of all sets, or the complement of their union), we seek results that show that with enough sets, there exist three sets such that many of the "difficult" regions are occupied.

For example, the first structure lemma shows that with linear number of sets, one can have four Venn diagram regions be occupied. Note that statement (1) of Lemma 5.10 already showed how to achieve five out of eight regions. But the lemma below ensures that the four occupied regions do not include the "easy" regions [3] or $\varnothing$; this makes the proof considerably more technical. Due to space contraints, the proof of the lemma is deferred to the full version.

Lemma 5.11 (First Structure Lemma). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists positive constant c such that if $|\mathcal{R}| > cn$, then $\mathcal{R}$ $4/6$-represents $(3, 2^{[3]} \backslash \{\varnothing, [3]\})$.*

Corollary 5.12. *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists positive constant c such that if $|\mathcal{R}| \geq cn^2$, then $\mathcal{R}$ $5/6$-represents $(3, 2^{[3]} \backslash \{\varnothing, [3]\})$.*

Proof. Start with Lemma 5.11 and apply the Extension Lemma 5.9 with parameters $d := 1$, $k := 3$, $r := 4$, and $\mathcal{S} := 2^{[3]} \backslash \{\varnothing, [3]\}$. ■

The next structure lemma is again in the same vein: the statement is similar to statement (2) of Lemma 5.10, and in fact seems quantitatively worse. (It requires a large number of ranges, and also that ranges have small size.) But again it does not require the "easy" set $\varnothing$ to be represented; we will use this flexibility soon. Again, due to space contraints, the proof of the lemma is deferred to the full version.

Lemma 5.13 (Second Structure Lemma: Small Sizes). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$, and let $\varepsilon \leq 1$ be a positive constant. Suppose that every range $R \in \mathcal{R}$ satisfies $|R| \leq n^\varepsilon$. Then, there exists positive constant c such that if $|\mathcal{R}| \geq cn^{2+3\varepsilon}$, then $\mathcal{R}$ $(6/7)$-represents $(3, 2^{[3]} \backslash \{\varnothing\})$.*

Recall that the Second Structure Lemma 5.13 required the sets to have small sizes. We now give the easy extension to handle all sizes of sets.

Lemma 5.14 (Second Structure Lemma: General Sizes). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{3-1/4}$, then $\mathcal{R}$ 6/7-represents $(3, 2^{[3]} \backslash \{\varnothing\})$.*

Proof. Set $\varepsilon := 1/4$. Call a range in $\mathcal{R}$ *small* if its size is at most n^ε, and *large* otherwise, and let $\mathcal{R}_{\text{small}}$ and $\mathcal{R}_{\text{large}}$ be the small and large ranges, respectively. If $|\mathcal{R}_{\text{small}}| > (c/2)n^{3-\varepsilon}$, then applying Lemma 5.13 on $\mathcal{R}_{\text{small}}$ proves the lemma, assuming that c is large enough. Otherwise, $|\mathcal{R}_{\text{large}}| > (c/2)n^{3-\varepsilon}$. If so, there is some element $x \in X$ that is in more than $(c/2)n^{3-\varepsilon} \cdot n^\varepsilon / n = (c/2)n^2$ many large ranges. Let $\mathcal{R}^x_{\text{large}}$ be these large ranges; by Corollary 5.12, if c is large enough, then there is tuple (R_1, R_2, R_3) of ranges in $\mathcal{R}'$ that 5/6-witnesses $(3, 2^{[3]} \backslash \{\varnothing, [3]\})$. Since $x \in R_1 \cap R_2 \cap R_3$, the tuple 6/7-witnesses $(3, 2^{[3]} \backslash \{\varnothing\})$, as desired. ■

Having proved the structure lemmas, we can turn to proving the main theorems of this section. We first prove Theorem 5.3, restated below. (This improves on statement (3) of Lemma 5.10.)

Theorem 5.3 (7/8-representation). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{3-1/4}$, then $\mathcal{R}$ 7/8-represents $(3, 2^{[3]})$.*

Proof. Set $c := 2c'$, where c' is the constant in Lemma 5.14, and suppose that $|\mathcal{R}| > cn^{3-1/4}$. Following the proof of Claim 5.2, fix an arbitrary element $x \in X$, and let $\mathcal{R}'$ be the set $\{R : R \in \mathcal{R}, x \notin R\} \cup \{\overline{R} : R \in \mathcal{R}, x \in R\}$, so that $(X \backslash \{x\}, \mathcal{R}')$ is a range space. We have $|\mathcal{R}'| \geq |\mathcal{R}|/2 > c'n^{3-1/4}$, since for every pair of ranges $R, \overline{R}$ satisfying $\{R, \overline{R}\} \cap \mathcal{R} \neq \varnothing$, one of $R, \overline{R}$ is in $\mathcal{R}'$. By the Second Structure Lemma 5.14, there exists ranges R_1, R_2, R_3 such that (R_1, R_2, R_3) 6/7-witnesses $(3, 2^{[3]} \backslash \{\varnothing\})$ in $X \backslash \{x\}$. Since $x \in \overline{R_1 \cup R_2 \cup R_3}$, (R_1, R_2, R_3) also 7/8-witnesses $(3, 2^{[3]})$ in X. Finally, one of the eight tuples obtained by taking or not taking the complement of each R_i gives a tuple of ranges in $\mathcal{R}$ that also 7/8-witnesses $(3, 2^{[3]})$, proving the theorem. ■

An application of the Extension Lemma to the above result proves Theorem 5.4, restated below.

Theorem 5.4 (1-representation). *Let $(X, \mathcal{R})$ be a range space with $|X| = n$. There exists a positive constant c such that if $|\mathcal{R}| > cn^{4-1/4}$, then $\mathcal{R}$ 1-represents $(3, 2^{[3]})$.*

Proof. Starting with Theorem 5.3, apply the Extension Lemma 5.9 with parameters $d := 3 - 1/4$, $k := 3$, $r := 7$, and $\mathcal{S} := 2^{[3]}$. ■

A DEFERRED PROOFS

A.1 Proof of Lemma 2.2

Following Karger-Stein's arguments, we prove Lemma 2.2 restated below.

Lemma 2.2. *Let OPT_H be the weight of the optimum minimum h-cut in $H = (V, E, w)$, and let $M := \frac{OPT_H}{h}$. For any $\alpha \geq 0$, there are at most $2^h n^{2\alpha}$ many subsets $A \subseteq V$ with $\partial_H A \leq \alpha M$. Moreover, we can output (a superset of) all such subsets in $O(2^h n^{2\alpha+O(1)})$ time, w.h.p.*

Proof. The algorithm is simple: start with the original graph, and while there are more than h vertices left, contract a random edge selected proportional to its weight. When there are h vertices remaining, consider all 2^h possible subsets of these h vertices. For each subset A', map it back to a subset A in the original graph (by taking all vertices in V that were contracted into a vertex of A'), and if $\partial_H A \leq \alpha M$, then add it to our collection of cuts. Repeat this process $O(n^{2\alpha} \log n)$ times.

To see correctness, consider an iteration with more than h vertices remaining, and let $H' = (V', E')$ be the remaining graph. Considering the $h-1$ vertices with smallest degree, and the rest of V', when un-contracted, gives us a h-cut of weight, whose weight must be at least OPT_H. Therefore, if $\overline{d}$ is the average degree of G' and $\overline{d_{h-1}}$ is the average degree of the $(h-1)$ smallest-degree vertices in G', then

$$\frac{OPT_H}{h} \leq \frac{OPT_H}{h-1} \leq \overline{d_{h-1}} \leq \overline{d} = \frac{2w(E')}{|V'|}. \tag{A.9}$$

Consider a cut $\partial_H A$ of size $\leq \alpha M$ that we want to preserve, and let $C \subseteq E$ be the edges in this cut. The probability that an edge in C is contracted on this iteration is

$$\frac{w(C)}{w(E')} \overset{(A.9)}{\leq} \frac{w(C) \cdot 2h}{OPT_H \cdot |V'|} \leq \frac{\alpha M \cdot 2h}{hM \cdot |V'|} \leq \frac{2\alpha}{|V'|}.$$

Thus, the probability that C survives for all iterations is at least

$$\prod_{r=h+1}^{n} \left(1 - \frac{2\alpha}{r}\right) \geq \frac{1}{n^{2\alpha}}.$$

Repeating the algorithm $O(n^{2\alpha} \log n)$ times produces all αM-cuts w.h.p., as desired. (The bound also holds for non-integer values of α, using generalized binomial coefficients [17, 20].) Observe that the algorithm does not need to know OPT_H or M. ■

REFERENCES

[1] Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. 2015. If the current clique algorithms are optimal, so is Valiant's parser. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*. IEEE, 98–117.

[2] Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. 2014. Consequences of faster alignment of sequences. In *International Colloquium on Automata, Languages, and Programming*. Springer, 39–51.

[3] András A Benczúr and Michel X. Goemans. 2008. Deformable polygon representation and near-mincuts. In *Building Bridges*. Springer, 103–135.

[4] Michel Burlet and Olivier Goldschmidt. 1997. A new and improved algorithm for the 3-cut problem. *Oper. Res. Lett.* 21, 5 (1997), 225–227. https://doi.org/10.1016/S0167-6377(97)00043-6

[5] Karthekeyan Chandrasekaran, Chao Xu, and Xilin Yu. 2018. Hypergraph k-cut in randomized polynomial time. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 1426–1438.

[6] Chandra Chekuri, Sudipto Guha, and Joseph Naor. 2006. The Steiner k-cut problem. *SIAM J. Discrete Math.* 20, 1 (2006), 261–271. https://doi.org/10.1137/S0895480104445095

[7] Chandra Chekuri, Kent Quanrud, and Chao Xu. 2018. LP Relaxation and Tree Packing for Minimum k-cuts. *arXiv preprint arXiv:1808.05765* (2018).

[8] Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michał Pilipczuk. 2016. Designing FPT algorithms for cut problems using randomized contractions. *SIAM J. Comput.* 45, 4 (2016), 1171–1229. https://doi.org/10.1137/15M1032077

[9] Mohsen Ghaffari, David R Karger, and Debmalya Panigrahi. 2017. Random contractions and sampling for hypergraph and hedge connectivity. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 1101–1114.

[10] Olivier Goldschmidt and Dorit S. Hochbaum. 1994. A polynomial algorithm for the k-cut problem for fixed k. *Math. Oper. Res.* 19, 1 (1994), 24–37. https://doi.org/10.1287/moor.19.1.24

[11] Anupam Gupta, Euiwoong Lee, and Jason Li. 2018. Faster Exact and Approximate Algorithms for k-Cut. In *Foundations of Computer Science (FOCS), 2018 IEEE 59th Annual Symposium on.*

[12] Anupam Gupta, Euiwoong Lee, and Jason Li. 2018. An FPT Algorithm Beating 2-Approximation for k-Cut. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018.* 2821–2837. https://doi.org/10.1137/1.9781611975031.179

[13] Jianxiu Hao and James B. Orlin. 1994. A faster algorithm for finding the minimum cut in a directed graph. *J. Algorithms* 17, 3 (1994), 424–446. https://doi.org/10.1006/jagm.1994.1043 Third Annual ACM-SIAM Symposium on Discrete Algorithms (Orlando, FL, 1992).

[14] Monika Henzinger and David P. Williamson. 1996. On the number of small cuts in a graph. *Inform. Process. Lett.* 59, 1 (1996), 41–44.

[15] Yoko Kamidoi, Noriyoshi Yoshida, and Hiroshi Nagamochi. 2006/07. A deterministic algorithm for finding all minimum k-way cuts. *SIAM J. Comput.* 36, 5 (2006/07), 1329–1341. https://doi.org/10.1137/050631616

[16] David R. Karger. 2000. Minimum cuts in near-linear time. *J. ACM* 47, 1 (2000), 46–76. https://doi.org/10.1145/331605.331608

[17] David R. Karger and Clifford Stein. 1996. A new approach to the minimum cut problem. *Journal of the ACM (JACM)* 43, 4 (1996), 601–640.

[18] Ken-ichi Kawarabayashi and Bingkai Lin. 2018. A nearly 5/3-approximation FPT Algorithm for Min-k-Cut. *Manuscript* (2018).

[19] Ken-ichi Kawarabayashi and Mikkel Thorup. 2011. The minimum k-way cut of bounded size is fixed-parameter tractable. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on.* IEEE, 160–169.

[20] Donald E Knuth. 1973. *The Art of Computer Programming, Volume 1: Fundamental Algorithms.* Addison-Wesley Publishing Company.

[21] François Le Gall. 2014. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation.* ACM, 296–303.

[22] Matthew S Levine. 2000. Fast randomized algorithms for computing minimum $\{3, 4, 5, 6\}$-way cuts. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms.* Society for Industrial and Applied Mathematics, 735–742.

[23] Pasin Manurangsi. 2017. Inapproximability of Maximum Edge Biclique, Maximum Balanced Biclique and Minimum k-Cut from the Small Set Expansion Hypothesis. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 80. 79:1–79:14. https://doi.org/10.4230/LIPIcs.ICALP.2017.79

[24] Jiří Matoušek. 1999. *Geometric discrepancy.* Algorithms and Combinatorics, Vol. 18. Springer-Verlag, Berlin. xii+288 pages. https://doi.org/10.1007/978-3-642-03942-3 An illustrated guide.

[25] Hiroshi Nagamochi and Toshihide Ibaraki. 1992. Computing edge-connectivity in multigraphs and capacitated graphs. *SIAM J. Discrete Math.* 5, 1 (1992), 54–66. https://doi.org/10.1137/0405004

[26] Hiroshi Nagamochi and Toshihide Ibaraki. 2000. A fast algorithm for computing minimum 3-way and 4-way cuts. *Math. Program.* 88, 3, Ser. A (2000), 507–520. https://doi.org/10.1007/PL00011383

[27] Hiroshi Nagamochi, Shigeki Katayama, and Toshihide Ibaraki. 2000. A faster algorithm for computing minimum 5-way and 6-way cuts in graphs. *J. Comb. Optim.* 4, 2 (2000), 151–169. https://doi.org/10.1023/A:1009804919645

[28] Joseph Naor and Yuval Rabani. 2001. Tree packing and approximating k-cuts. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (Washington, DC, 2001).* SIAM, Philadelphia, PA, 26–27.

[29] Kent Quanrud. 2018. Fast and Deterministic Approximations for k-Cut. *arXiv preprint arXiv:1807.07143* (2018).

[30] R. Ravi and Amitabh Sinha. 2008. Approximating k-cuts using network strength as a Lagrangean relaxation. *European J. Oper. Res.* 186, 1 (2008), 77–90. https://doi.org/10.1016/j.ejor.2007.01.040

[31] Huzur Saran and Vijay V. Vazirani. 1995. Finding k-cuts within twice the optimal. *SIAM J. Comput.* 24, 1 (1995), 101–108.

[32] Mikkel Thorup. 2008. Minimum k-way cuts via deterministic greedy tree packing. In *Proceedings of the fortieth annual ACM symposium on Theory of computing.* ACM, 159–166.

[33] Virginia Vassilevska Williams. 2012. Multiplying matrices faster than Coppersmith–Winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing.* ACM, 887–898.

[34] Virginia Vassilevska Williams and Ryan Williams. 2010. Subcubic equivalences between path, matrix and triangle problems. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on.* IEEE, 645–654.

[35] Mingyu Xiao, Leizhen Cai, and Andrew Chi-Chih Yao. 2011. Tight approximation ratio of a general greedy splitting algorithm for the minimum k-way cut problem. *Algorithmica* 59, 4 (2011), 510–520.