Optimal Bounds for the $k$-cut Problem

Anupam Gupta∗
CMU

David G. Harris†
UMD

Euiwoong Lee‡
University of Michigan

Jason Li§
CMU

Abstract

In the $k$-cut problem, we want to find the lowest-weight set of edges whose deletion breaks a given (multi)graph into $k$ connected components. Algorithms of Karger & Stein can solve this in roughly $O(n^{2k})$ time. On the other hand, lower bounds from conjectures about the $k$-clique problem imply that $\Omega(n^{(1-o(1))k})$ time is likely needed. Recent results of Gupta, Lee & Li have given new algorithms for general $k$-cut in $n^{1.98k+O(1)}$ time, as well as specialized algorithms with better performance for certain classes of graphs (e.g., for small integer edge weights).

In this work, we resolve the problem for general graphs. We show that the Contraction Algorithm of Karger outputs any fixed $k$-cut of weight $\alpha \lambda_k$ with probability $\Omega_k(n^{-\alpha k})$, where $\lambda_k$ denotes the minimum $k$-cut weight. This also gives an extremal bound of $O_k(n^k)$ on the number of minimum $k$-cuts and an algorithm to compute $\lambda_k$ with roughly $n^{k}\text{polylog}(n)$ runtime. Both are tight up to lower-order factors, with the algorithmic lower bound assuming hardness of max-weight $k$-clique.

The first main ingredient in our result is an extremal bound on the number of cuts of weight less than $2\lambda_k/k$, using the Sunflower lemma. The second ingredient is a fine-grained analysis of how the graph shrinks – and how the average degree evolves – in the Karger process.

∗anupamg@cs.cmu.edu. Supported in part by NSF award CCF-1907820, CCF-1955785, and CCF-2006953, and the Indo-US Joint Center for Algorithms Under Uncertainty.
†davidgharris29@gmail.com.
‡euiwoong@umich.edu. Supported in part by the Simons Collaboration on Algorithms and Geometry.
§jmli@cs.cmu.edu. Supported in part by NSF award CCF-1907820.
1 Introduction

We consider the $k$-Cut problem: given an edge-weighted graph $G = (V, E, w)$ and an integer $k$, we want to delete a minimum-weight set of edges so that $G$ has at least $k$ connected components. We let $\lambda_k$ denote the resulting weight of the deleted edges. This generalizes the global min-cut problem, where the goal is to break the graph into $k = 2$ pieces.

It was unclear that the problem admitted a polynomial-time algorithm for fixed $k$, until Goldschmidt and Hochbaum gave a deterministic algorithm with $n^{O(k^2)}$ runtime [GH88]. The algorithm of Karger [Kar93], based on random edge contractions, can also solve $k$-Cut in $\tilde{O}(mn^{2k-1})$ time; this was later improved to $\tilde{O}(n^{2k-2})$ runtime by Karger and Stein [KS96]. There have been a number of improved deterministic algorithms [GH94, KYN06, Tho08, CQX20]: notably, the tree-packing result of Thorup [Tho08] was sped up by Chekuri et al. [CQX20] to $O(mn^{2k-3})$ runtime. Thus, until recently, randomized and deterministic algorithms with very different approaches have achieved $n^{(2-o(1))k}$ runtime for the problem. (Here and subsequently, the $o(1)$ in the exponent indicates a quantity that goes to zero as $k$ increases.)

As for hardness, there is a reduction from MAX-WEIGHT $(k - 1)$-CLIQUE to $k$-Cut. It is conjectured that solving MAX-WEIGHT $k$-CLIQUE requires $\Omega(n^{(1-o(1))k})$ time when weights are integers in the range $[1, \Omega(n^k)]$, and $\Omega(n^{\omega/3-o(1)}k)$ time for unit weights, where $\omega$ is the matrix multiplication constant. Extending these bounds to $k$-Cut suggests that $n^{(1-o(1))k}$ may be a lower bound for general weighted $k$-cut instances.

There has been recent progress on this problem, showing the following results:

1. Gupta, Lee & Li gave an $n^{(1.98+o(1))k}$-time algorithm for general $k$-Cut [GLL19]. This was based on showing an extremal bound for the number of “small” 2-cuts in the graph. A bounded-depth search is then used to guess the small 2-cuts within a minimum $k$-cut and make progress. This proof-of-concept result showed that $n^{(2-o(1))k}$ was not the right bound, but the approach did not seem to extend to exponents considerably below 2k.

2. For polynomially-bounded edge-weights, Gupta, Lee & Li gave an algorithm with roughly $k^{O(k)} n^{(2\omega/3+o(1))k}$ runtime [GLL18]. For unweighted graphs, Li obtained $k^{O(k)} n^{(1+o(1))k}$ runtime [Li19]. These algorithms are both based on finding a spanning tree which crosses a small number of edges of a minimum $k$-cut. The former relies on matrix multiplication tree ideas, and the latter on the Kawarabayashi-Thorup graph decomposition [KT18], which are both intrinsically tied to graphs with small edge-weights.

In this paper, we show that the “right” algorithm, the original Contraction Algorithm of Karger [Kar93], achieves the “right” bound for general graphs. We recall the algorithm below; here, $\tau$ (the final desired graph size) is a parameter we will adjust in our specific constructions.

**Algorithm 1 Contraction Algorithm**

1: **while** $|V| > \tau$ **do**
2: Choose an edge $e \in E$ at random from $G$, with probability proportional to its weight.
3: Contract the two vertices in $e$ and remove self-loops.
4: **end while**
5: Return a $k$-cut of $G$ chosen uniformly at random.
Setting $\tau = k$, as in Karger’s original algorithm, would seem most natural, but we will require a larger value in our analysis. Our main result is the following.

**Theorem 1** (Main). For any integer $k \geq 2$ and real number $\alpha \geq 1$, the Contraction Algorithm outputs each $k$-cut of weight $\alpha \lambda_k$ with probability at least $n^{-\alpha k} k^{-O(\alpha k^2)}$ for appropriate choice of $\tau = \text{poly}(\alpha, k)$.

Since any minimum-weight $k$-cut (corresponding to $\alpha = 1$) is output with probability $n^{-k} k^{-O(k^2)}$, this immediately implies the following corollary.

**Corollary 2** (Number of Minimum $k$-cuts). For any $k \geq 2$, the number of minimum-weight $k$-cuts in a graph is at most $n^k k^{O(k^2)}$.

This improves on the previous best bound of $n^{(1.98+o(1))k}$ [GLL19]. It is almost tight because the cycle on $n$ vertices has $\binom{n}{k}$ minimum $k$-cuts.

Also, while the direct implementation of Algorithm 1 incurs an extra $O(n^2)$ in the runtime, the Recursive Contraction Algorithm of Karger and Stein [KS96] can be used to get an almost-matching running time to enumerate all minimum $k$-cuts.

**Theorem 3** (Faster Algorithm to Find a Minimum $k$-cut). There is an algorithm to enumerate all minimum $k$-cuts in time $n^k (\log n)^{O(k^2)}$ with probability at least $1 - 1/\text{poly}(n)$.

This improves the runtime $n^{(1.98+o(1))k}$ from [GLL19] and even beats the runtime $n^{(1+o(1))k}$ for the unweighted case [Li19]. It is almost optimal under the hypothesis that MAX-WEIGHT $k$-CLIQUE requires $n^{(1-o(1))k}$ time. Achieving an $O(n^{ck})$-time algorithm for unit-weighted graphs for any constant $c < 1$ still remains an open problem.

See Section 7 for the formal statements of the above theorems.

1.1 Our Techniques

Although we have stated the general $k$-CUT problem for a weighted graph, we will assume throughout that $G = (V, E)$ is an unweighted multigraph with $n$ vertices and $m$ edges. The viewpoint in terms of weighted graphs is equivalent via replicating edges; note that, in this case, $m$ may be exponentially large compared to $n$. Our computational and combinatorial bounds will depend on $n$ and not directly on $m$.

In the spirit of [GLL19], our proof has two main parts: (i) a bound on the extremal number of “medium” cuts in a graph, and (ii) a new algorithmic analysis for the Contraction Algorithm. To begin, let us first state a crude version of our extremal result. Define $\overline{\lambda}_k := \lambda_k / k$, which we think of as the average contribution of the $k$ components of a minimum $k$-cut, and let “medium” cuts denote 2-cuts whose weight is in $[\overline{\lambda}_k, 2\overline{\lambda}_k]$.

The graph may contain a negligible number of “small” 2-cuts of weight less than $\overline{\lambda}_k$. Loosely speaking, the extremal bound says the following:

\[(\ast)\text{ For fixed } k, \text{ the graph has at most } O(n) \text{ many medium cuts.}\]

To develop some intuition for this claim, it is instructive to consider the cycle and clique graphs. These are two opposite ends of the spectrum in the context of graph cut. In the cycle, we have

\[\lambda_k = \frac{1}{k}, \quad \overline{\lambda}_k = \frac{1}{2k}, \quad \text{and } \overline{\lambda}_k - 2\overline{\lambda}_k = 0.\]
\( \lambda_k = 1 \), and there are no 2-cuts with weight less than \( 2\lambda_k \), hence (\( \star \)) holds. However, the \( \binom{n}{2} \) minimum 2-cuts have size equal to \( 2\lambda_k = 2 \). In the clique, the minimum \( k \)-cut chops off \( k - 1 \) singleton vertices, so \( \lambda_k = (k-1) + (k-1)(n-k+1) \), which gives \( \lambda_k \approx \frac{k-1}{k}n \) for \( n \gg k \). There are \( n \) minimum 2-cuts, which have weight \( n - 1 < 2\lambda_k \) (the singletons), so again (\( \star \)) holds. And again, there are \( \binom{n}{2} \) 2-cuts of weight approximately \( 2\lambda_k \) (the doubletons).

Therefore, in both the cycle and the clique, the bound \( 2\lambda_k \) is almost the best possible. Moreover, the \( O(n) \) bound for the number of medium cuts is also optimal in the clique.

1.1.1 Analysis of Contraction Algorithm

When we begin the Contraction Algorithm in the graph \( G \), our extremal bound ensures that there are at most \( O_k(n) \) medium cuts of size between \( \lambda_k \) and \( 2\lambda_k \), plus a negligible number of small 2-cuts of size less than \( \lambda_k \). Let us next sketch how these bounds give rise to the improved bound for \( k \)-cuts. To provide intuition, let us suppose that in fact there are \( n \) medium cuts and no smaller 2-cuts (the precise factors are not important for the overall analysis).

Each vertex during the Contraction Algorithm corresponds to a 2-cut of the original graph, and we are assuming that \( G \) has no small 2-cuts, so the number of edges in each iteration \( i \) of the Contraction Algorithm is lower-bounded by \( i\lambda_k/2 \). Again, to provide intuition, let us suppose there are precisely this many edges. Then each medium cut gets an edge selected in iteration \( i \), and is thereby removed from the graph, with probability at least \( \frac{\lambda_k}{i\lambda_k/2} = \frac{2}{i} \). So after \( n/2 \) iterations, the number of surviving medium cuts is close to

\[
n \prod_{i=n/2}^{n} \left(1 - \frac{2}{i}\right) \approx \frac{n}{4}.
\]

Thus, in the resulting subgraph with \( i = n/2 \) vertices, at most \( n/4 \) of the vertices (corresponding to the surviving medium cuts) have degree \( \lambda_k \). The remainder have degree at least \( 2\lambda_k \). Continuing this process, the graph becomes more and more enriched with high-degree vertices. After \( (1 - \varepsilon)n \) iterations (for some small constant \( \varepsilon \)), almost all of the medium cuts have been eliminated, and each graph on \( i \leq \varepsilon n \) vertices has close to \( i\lambda_k \) edges.

Now consider an arbitrary minimum \( k \)-cut \( K \). It survives the first \( (1 - \varepsilon)n \) iterations with constant probability. In each iteration \( i \) of the Contraction Algorithm when the resulting subgraph has \( i \leq \varepsilon n \) vertices, \( K \) is selected with probability roughly \( \frac{\lambda_k}{i\lambda_k} = \frac{k}{i} \). Over the entire run of the Contraction Algorithm, down to the final graph with \( \tau = \text{poly}(k) \) vertices, \( K \) survives with probability roughly

\[
\text{constant} \cdot \prod_{i=\tau}^{\varepsilon n} \left(1 - \frac{k}{i}\right) \approx \Theta(n^{-k}).
\]

To show this formally, we need to track the number of medium cuts remaining in the residual graphs produced by the Contraction Algorithm. There are two main obstructions to turning the analysis we have sketched above into a rigorous proof. First, many of our bounds made unwarranted assumptions about the parameter sizes; for example, we only know lower bounds on the edge counts, and we should not assume that these hold with equality in each iteration. Second, the Contraction Algorithm is a stochastic process; we cannot assume that relevant quantities (such as the number of medium cuts) equal their expectations.
To overcome these challenges, we adopt a proof strategy of [HS18]. First, using a number of heuristic worst-case assumptions, and relaxing the discrete stochastic process to a continuous-time system of differential equations, we make a guess as to the correct dynamics of the Contraction Algorithm. This gives us a formula for the probability that \( K \) is selected, given that the process has reached some iteration \( i \) and currently has some given number of residual medium cuts. Next, we use induction to prove that this formula holds in the worst case. For this, we take advantage of the fact that our guessed formula has nice convexity and monotonicity properties.

Let us contrast our proof strategy with the analysis in a preliminary version of this paper [GLL20]. In this work, we analyze the Contraction Algorithm as edges are contracted one at a time. In contrast, [GLL20] considered an alternate viewpoint where each edge is independently contracted with some given probability, which is equivalent to executing many steps of the Contraction Algorithm. (The alternate viewpoint is only taken for the purposes of analysis; the actual algorithm remains the same.)

In some ways, the alternate viewpoint is simpler, since it preserves many independencies among edges and since a number of relevant parameters are concentrated. However, a drawback is that it lacks fine control of precisely how many edges to contract. When the number of vertices in the graph becomes small, the independent-contractions viewpoint introduces larger errors compared to our one-at-a-time approach. For example, the preliminary version showed a bound of \( n^k k^{O(k^2 \log \log n)^2} \) on the number of \( k \)-cuts; compare this to the tighter bound of \( n^k k^{O(k^2)} \) from Theorem 1.

### 1.1.2 Extremal Result

Recall our target extremal statement (⋆): there are \( O_k(n) \) many medium cuts in the graph, i.e. 2-cuts of weight less than \( 2\lambda_k \). To show this, we consider two different cases.

In the first case, suppose the medium cuts all correspond to small vertex sets. Our key observation is that the \( k \)-cut structure of the graph forbids certain types of sunflowers in the set family corresponding to the medium cuts; on the other hand, estimates from the Sunflower Lemma would ensure that if there are many medium cuts, then such a sunflower would be forced to exist.

For, consider a \( k \)-sunflower of medium cuts \( S_1, S_2, \ldots, S_k \), in which the core \( C \) is a 2-cut of weight at least \( \frac{k}{k-1} \lambda_k \). (Handling cases where the core is empty or corresponds to a smaller 2-cut are details we defer to the actual proof.) Suppose we contract \( C \) as well as each petal \( P_i = S_i \setminus C \) to single vertices \( c \) and \( p_i \) respectively. To provide intuition, let us suppose that there are the same number of edges \( r \) between the core and each petal, and let \( a \geq \frac{k}{k-1} \lambda_k \) denote the degree of \( c \) itself; clearly \( r \leq a/k \). See Figure 1 right.

Since each set \( S_i \) is a medium cut, there are less than \( 2\lambda_k \) edges from \( \{c, p_i\} \) to \( V \setminus \{c, p_i\} \). So \( \deg(p_i) < 2\lambda_k - a + 2r \) for all \( i \) and consequently, the \( k \)-cut \( \{p_1, \ldots, p_{k-1}, V \setminus \{p_1, \ldots, p_{k-1}\}\} \) has weight at most \( \sum_{i=1}^{k-1} \deg(p_i) < (k-1) \cdot (2\lambda_k - a + 2r) \leq (k-1)(2\lambda_k - a + 2a/k) \). Due to our bound on \( a \), this is at most \( k\lambda_k = \lambda_k^* \); this is a contradiction since \( \lambda_k^* \) is the minimum \( k \)-cut value.

In the second case, suppose there is a medium cut \( S \) where both halves involve many vertices. Then consider a maximal sequence of medium cuts \( S_1, \ldots, S_\ell \) starting with \( S_1 = S \), such that the Venn diagram of \( S_1, \ldots, S_\ell \) has at least \( 2\ell \) regions. See Figure 1 left. From this, we can form two subgraphs where every atom of the Venn diagram of \( S_1, \ldots, S_\ell \) in each half of \( S \) gets contracted to a single vertex. It can be shown that every medium cut of the original graph is preserved in at least one of the two graphs. Also, the fact that both halves of \( S \) have many vertices ensures that the contracted graphs are strictly smaller than the original graph. We get our desired bound by
in the cycle must be less than $2\lambda_k$. Then, we select $\ell = 4$ such 2-cuts as shown. Their Venn diagram has $2\ell = 8$ nonempty atoms and form an 8-cut with cost less than $\ell \cdot 2\lambda_k = 8\lambda_k = \lambda_k$. Right: A $k$-sunflower with core and petals consisting of single vertices. Here $c$ has degree $a \geq \frac{k}{k-1} \lambda_k$ and each bolded edge has weight $r = a/k = \frac{\lambda_k}{k-1}$. A $k$-cut generated by $k-1$ of the vertices $p_i$ then has weight less than $\lambda_k$.

induction on $n$.

1.2 Outline

In Section 2, we discuss the Sunflower Lemma. For our result, we need a slightly strengthened version of this lemma, which involves showing the existence of multiple sunflowers and ensuring their cores are nonempty.

In Section 3, we record some elementary bounds and definitions of cuts and $k$-cuts in the graph. In Section 4, we use these for our main extremal bound on the number of medium cuts.

In Section 5, we provide an overview of the Contraction Algorithm and some simple bounds on the probability that cuts survive it. In Section 6, we carry out the more involved analysis of how the number of medium cuts evolves during the Contraction Algorithm.

In Section 7, we conclude with our main results on the behavior of the Contraction Algorithm and the Recursive Contraction Algorithm.

2 Sunflower Lemma and Extensions

In a set system $\mathcal{F}$ over a universe $U$, an $r$-sunflower is a collection of $r$ sets $F_1, \ldots, F_r \in \mathcal{F}$ which all share the same pairwise intersection. That is, there is a core $C \subseteq U$ such that $F_i \cap F_j = C$ for all $i, j$, and hence $\bigcap_i F_i = C$. Let $sf(d, r)$ be the smallest number such that any set system with more than $sf(d, r)$ sets of cardinality at most $d$ must have an $r$-sunflower. The classical bound of Erdős and Rado [ER60] shows that $sf(d, r) \leq d!(r-1)^d$. A recent breakthrough by Alweiss et al. [ALWZ20] shows that

$$sf(d, r) \leq (\log d)^d (r \cdot \log \log d)^{O(d)}.$$ \hspace{1cm} (1)

While we use this improved bound, it only changes lower-order terms: the older Erdős-Rado bound would give the same asymptotics for our applications.

For our applications for cuts, we want multiple sunflowers with distinct nonempty cores. (The cores may intersect, even though they are distinct.) The bound must then depend on the universe size $N$, since the system consisting of $N$ singleton sets has no sunflowers with nonempty core. The
following results show that we can guarantee a nonempty core by multiplying the bound by $N$.

**Proposition 4.** Let $\mathcal{F}$ be a family of nonempty sets over a universe of $N$ elements, where every set has size at most $d$. If $|\mathcal{F}| > \text{sf}(d, r) \cdot N$, then $\mathcal{F}$ contains an $r$-sunflower with nonempty core.

**Proof.** For each element $v$ of the universe, consider the set system $\mathcal{F}_v := \{ F \in \mathcal{F} : F \ni v \}$. Since every set in $\mathcal{F}$ is included in some $\mathcal{F}_v$, there must be some element $v$ with $|\mathcal{F}_v| \geq |\mathcal{F}|/N > \text{sf}(d, r)$. Thus, there is an $r$-sunflower in $\mathcal{F}_v$ and hence $\mathcal{F}$. The core is nonempty since it contains $v$. $\square$

**Lemma 5.** Let $\mathcal{F}$ be a family of nonempty sets over a universe of $N$ elements, where every set has size at most $d$. If $|\mathcal{F}| > \text{sf}(d, r) \cdot sN$, then $\mathcal{F}$ contains $s$ many $r$-sunflowers, each with a distinct, nonempty core.

**Proof.** We show this by induction on $s$. The base case $s = 0$ is vacuous. For the induction step with $s \geq 1$, consider a maximal nonempty set $C$ such that $\mathcal{F}$ contains an $r$-sunflower with core $C$; this exists by Proposition 4 since $|\mathcal{F}| > \text{sf}(d, r) \cdot N$.

We claim that the set system $\mathcal{F}_C := \{ F \in \mathcal{F} : F \subseteq C \}$ has size at most $\text{sf}(d, r) \cdot N$. For, if not, then applying Proposition 4 to the set system $\{ F \setminus C : F \in \mathcal{F}_C \}$ (which has the same cardinality as $\mathcal{F}_C$) would give an $r$-sunflower $S_1, \ldots, S_r$ with nonempty core $C'$. The sets $S_1 \cup C, \ldots, S_r \cup C$ in $\mathcal{F}$ then form an $r$-sunflower with core $C \cup C'$, contradicting maximality of $C$.

Now consider the set system $\mathcal{F}' = \mathcal{F} \setminus \mathcal{F}_C$. It has size $|\mathcal{F}| - |\mathcal{F}_C| > \text{sf}(d, r) \cdot sN - \text{sf}(d, r) \cdot N = \text{sf}(d, r) \cdot (s-1)N$. By the induction hypothesis, it has $s-1$ many $r$-sunflowers with distinct nonempty cores. These cores are all distinct from $C$, since no sets containing $C$ remain in $\mathcal{F}'$. Combining them with the $r$-sunflower of core $C$ gives $s$ many $r$-sunflowers with distinct, nonempty cores. $\square$

### 3 Simple Bounds and Definitions for Cuts

We assume throughout we have a fixed value $k \geq 3$. A $k$-cut $K$ is a partition of $V$ into $k$ nonempty sets, and we let $\partial K$ denote the set of edges crossing different parts of $K$. The weight of $K$ is the cardinality of the edge set $\partial K$. We let $\lambda_k$ be the minimum weight of any $k$-cut, and $\bar{\lambda}_k := \lambda_k/k$.

A 2-cut $\{ C, V \setminus C \}$ will often simply be called a cut, and we often denote it merely by $C$. The shore of the cut is whichever of the sets $C$ or $V \setminus C$ is smaller. (If they are the same size, choose one arbitrarily), and the shore size is the cardinality of the shore.

For vertex sets $A, B$ we let $E(A, B)$ denote the set of edges crossing from $A$ to $B$. We also write $\partial S = E(S, V \setminus S)$ for a set $S \subseteq V$.

We define a small cut to be a cut $C$ with

$$|\partial C| < \frac{k}{k-1} \bar{\lambda}_k,$$

and we define a medium cut to be a cut $C$ such that

$$\frac{k}{k-1} \bar{\lambda}_k \leq |\partial C| < 2\bar{\lambda}_k.$$

Given vertex sets $F_1, \ldots, F_t$, we denote their Venn diagram by Venn$(F_1, \ldots, F_t)$. An atom denotes a nonempty region of the diagram, i.e., a nonempty set that can be expressed as $G_1 \cap \ldots \cap G_t$, where each set $G_i$ is either $F_i$, or its complement $V \setminus F_i$. See Figure 2.
Figure 2: The Venn diagram above has eight atoms.

We say that \( F_1, \ldots, F_t \) generate the \( \ell \)-cut \( K = \{ A_1, \ldots, A_\ell \} \) where \( A_1, \ldots, A_\ell \) are the atoms of \( \text{Venn}(F_1, \ldots, F_t) \). Observe that the weight of \( K \) is at most \( |\partial F_1| + \cdots + |\partial F_t| \).

We begin with a few straightforward bounds.

**Proposition 6.** If \( n \geq k \), then \( m \geq \frac{nk}{2^{(k-1)}} \lambda_k \).

**Proof.** Sort the vertices in ascending order of degree, so \( \deg(v_1) \leq \deg(v_2) \leq \cdots \leq \deg(v_n) \). The \( k \)-cut generated by the singleton sets \( \{v_1\}, \ldots, \{v_{k-1}\} \) has weight at most \( \deg(v_1) + \cdots + \deg(v_{k-1}) \); since \( \lambda_k \) is the minimum \( k \)-cut, we thus have \( \deg(v_1) + \cdots + \deg(v_{k-1}) \geq \lambda_k \). Also, because of the sorted vertex ordering, we have \( \deg(v_i) \geq \deg(v_{k-1}) \geq \frac{\deg(v_1) + \cdots + \deg(v_{k-1})}{k-1} \geq \frac{\lambda_k}{k-1} \) for all \( i \geq k \).

Summing vertex degrees, the total number of edges \( m \) is given by

\[
2m = (\deg(v_1) + \cdots + \deg(v_{k-1})) + (\deg(v_k) + \cdots + \deg(v_n)) \\
\geq \lambda_k + (n - k + 1) \cdot \lambda_k / (k-1) = n \lambda_k / (k-1) = nk \lambda_k / (k-1). 
\]

**Lemma 7.** There are fewer than \( 2^{k-2} \) small cuts.

**Proof.** Suppose not; in this case, we will construct a \( k \)-cut of weight less than \( \lambda_k \), which contradicts the definition of \( \lambda_k \).

For \( i = 1, \ldots, k-1 \), let us choose an arbitrary small cut \( S_i \) such that \( |\text{Venn}(S_1, \ldots, S_i)| \geq i + 1 \). We claim that we can always choose such an \( S_i \). For, if \( |\text{Venn}(S_1, \ldots, S_{i-1})| \geq i + 1 \), then \( S_i \) can be chosen arbitrarily. Otherwise, suppose that \( \text{Venn}(S_1, \ldots, S_{i-1}) \) has precisely \( i \) atoms \( A_1, \ldots, A_i \). The only small cut \( T \) such that \( |\text{Venn}(S_1, \ldots, S_{i-1}, T)| = i = |\text{Venn}(S_1, \ldots, S_{i-1})| \) would have the form \( T = \bigcup_{j \in I} A_j \) for some subset \( I \subseteq \{1, \ldots, i\} \). There are at most \( 2^{i-1} - 1 \) such cuts (keeping in mind that \( I \) and its complement determine the same cut). Since by assumption there are at least \( 2^{k-2} \) small cuts, there exists a small cut \( S_i \) with \( \text{Venn}(S_1, \ldots, S_{i-1}, S_i) > i \) as desired.

At the end, we have \( |\text{Venn}(S_1, \ldots, S_{k-1})| \geq k \). So the small cuts \( S_1, \ldots, S_{k-1} \) generate a \( t \)-cut for \( t \geq k \) whose weight is less than \( (k-1) \cdot \frac{k-1}{k} \lambda_k = \lambda_k \). This is our desired contradiction.

**Proposition 8.** Let \( T_1, \ldots, T_r \) be medium cuts where \( r = \lceil k/2 \rceil \). Then either \( |\text{Venn}(T_1, \ldots, T_{r-1})| < 2(r-1) \) or \( |\text{Venn}(T_1, \ldots, T_r)| < 2r \) (or both).

**Proof.** Let us first consider the case where \( k \) is even and \( r = k/2 \). Suppose for contradiction that \( |\text{Venn}(T_1, \ldots, T_r)| = t \geq k \). Then \( T_1, \ldots, T_r \) generate a \( t \)-cut \( K \). Since \( T_1, \ldots, T_r \) are medium cuts, the weight of \( K \) is less than \( r \cdot 2 \lambda_k = \lambda_k \); this contradicts that \( \lambda_k \) is the minimum \( k \)-cut value.

Next consider the case where \( k \) is odd and \( r = (k+1)/2 \). Suppose for contradiction that \( |\text{Venn}(T_1, \ldots, T_r)| = t \geq k+1 \) and \( |\text{Venn}(T_1, \ldots, T_{r-1})| = t' \geq k-1 \). The sets \( T_1, \ldots, T_{r-1} \) generate
a \( t' \)-cut \( K' \); since \( T_1, \ldots, T_{r-1} \) are medium cuts, the weight of \( K' \) is less than \((r - 1) \cdot 2\bar{\lambda}_k = \frac{k-1}{k} \lambda_k \). If \( t' \geq k \), this contradicts that \( \lambda_k \) is the minimum \( k \)-cut value. So it must be that \( t' = k - 1 \) exactly.

Let \( A_1, \ldots, A_j \) be the atoms of \( \text{Venn}(T_1, \ldots, T_{r-1}) \) cut by \( T_r \); since \( t \geq k + 1 \) and \( t' = k - 1 \) we must have \( j \geq 2 \). The edge sets \( E(T_r, A_i \setminus T_r) \) are all disjoint and \( T_r \) is a medium cut, so at least one atom \( A_i \) must satisfy \(|E(T_r, A_i \setminus T_r)| \leq |\partial T_r|/j \leq |\partial T_r|/2 \leq \bar{\lambda}_k \). The sets \( T_1, \ldots, T_{r-1}, A_i \) then generate a \( k \)-cut \( K'' \) of weight less than \((r - 1) \cdot 2\bar{\lambda}_k + \bar{\lambda}_k = \lambda_k \), contradicting that \( \lambda_k \) is the minimum \( k \)-cut value.

\[ \square \]

4 Bounding the Number of Medium Cuts

We now analyze the combinatorial structure of the medium cuts to show the following key bound:

**Theorem 9.** There are \( k^{O(k)}n \) many medium cuts.

We prove this in two stages. First, using the Sunflower Lemma, we show it for the special case when all the medium cuts of \( G \) have shoresize at most \( k \). We then extend to the general case by an induction on the graph size.

**Lemma 10.** Suppose the medium cuts all have shoresize at most \( k \). Then there are at most \( k^{O(k)}n \) medium cuts.

**Proof.** Let \( \mathcal{F} \) be the set family consisting of the shores of the medium cuts. We claim that \( \mathcal{F} \) cannot have \( 2^k \) many \( k \)-sunflowers with distinct nonempty cores. For, suppose for contradiction that it does so. Then, by Lemma 7, at least one of the sunflowers has a nonempty core \( C \) with \(|\partial C| \geq \frac{k}{k-1} \bar{\lambda}_k \). Let the sets in this sunflower be \( S_1, \ldots, S_k \in \mathcal{F} \) where \( S_i \cap S_j = C \) for \( i \neq j \).

Let \( P_i := S_i \setminus C \) be the petal for each \( S_i \), and let \( L_i = E(P_i, C) \) denote the set of edges between \( P_i \) and \( C \). By inclusion-exclusion, we have \(|\partial P_i| = |\partial S_i| - |\partial C| + 2|L_i|\). Since each \( S_i \) is a medium cut, it satisfies \(|\partial S_i| < 2\bar{\lambda}_k \), so

\[ |\partial P_i| < 2\bar{\lambda}_k - |\partial C| + 2|L_i|. \]

Suppose the petals are sorted in ascending order of \(|L_i|\), so that \(|L_i| \leq |L_{i+1}| \) for \( i = 1, \ldots, k - 1 \). Consider the \( k \)-cut \( K \) generated by the disjoint sets \( P_1, \ldots, P_{k-1} \). We can bound its weight \(|\partial K|\) by:

\[ |\partial K| \leq \sum_{i=1}^{k-1} |\partial P_i| < \sum_{i=1}^{k-1} (2\bar{\lambda}_k - |\partial C| + 2|L_i|) = 2(k-1)\bar{\lambda}_k - (k-1)|\partial C| + 2 \sum_{i=1}^{k-1} |L_i|. \]

Because the sets \( L_i \) are pairwise disjoint subsets of \( \partial C \) in sorted order of size, we have

\[ \sum_{i=1}^{k-1} |L_i| \leq \frac{k-1}{k} \sum_{i=1}^{k} |L_i| \leq \frac{k-1}{k} |\partial C|, \]

and so

\[ |\partial K| < 2(k-1)\bar{\lambda}_k - (k-1)|\partial C| + 2 \cdot \frac{k-1}{k} |\partial C| = 2(k-1)\bar{\lambda}_k - \frac{(k-1)(k-2)}{k} |\partial C|. \]

Finally, using the bound \(|\partial C| \geq \frac{k}{k-1} \bar{\lambda}_k \), we get

\[ |\partial K| < 2(k-1)\bar{\lambda}_k - \frac{(k-1)(k-2)}{k} \cdot \frac{k}{k-1} \bar{\lambda}_k = k\bar{\lambda}_k = \lambda_k. \]
This contradicts the definition of $\lambda_k$ as the minimum $k$-cut. Thus $\mathcal{F}$ cannot have $2^k$ many $k$-sunflowers with distinct, nonempty cores. By our hypothesis, the sets in $\mathcal{F}$ have size at most $k$. Thus, by Lemma 5 (with parameters $d = r = k$ and $N = n$ and $s = 2^k$) and Eq. (1), this means

$$|\mathcal{F}| \leq sf(d, r) \cdot sN \leq (\log d)^d (r \cdot \log \log d)^{O(d)} \cdot 2^k n \leq k^{O(k)} n. \quad \square$$

We will next remove the restriction on the shoresize, completing the proof.

**Proof of Theorem 9.** We will show by induction on $|H|$ that for $n > k$ there are at most $c_k(n - k)$ medium cuts in any graph $G$, for some constant $c_k = k^{O(k)}$.

If every medium cut has shoresize at most $k$, then we have already shown this in Lemma 10 for appropriate choice of $c_k$. (This covers the base case of the induction $n = k + 1$.) We thus consider a medium cut $S$ with shoresize larger than $k$, i.e. $k < |S| < n - k$.

Starting with $S_1 = S$, let us form a maximal sequence of medium cuts $S_1, S_2, \ldots, S_t$ with the property that $|\text{Venn}(S_1, \ldots, S_t)| \geq 2t$ for all $t = 1, \ldots, \ell$; here $\ell \geq 1$ since $|\text{Venn}(S)| = 2$. Let the atoms of $\text{Venn}(S_1, \ldots, S_t)$ inside $S$ (respectively, outside $S$) be $A_1, \ldots, A_i$ and $B_1, \ldots, B_j$. So $A_1 \cup \cdots \cup A_i = S$ and $B_1 \cup \cdots \cup B_j = V \setminus S$.

Now form a graph $H_1$ by contracting each of the atoms $A_1, \ldots, A_i$ and likewise form a graph $H_2$ by contracting each of the atoms $B_1, \ldots, B_j$. Since $A_1, \ldots, A_i$ partition $S$ and $B_1, \ldots, B_j$ partition $V \setminus S$, these graphs have $n_1 = (n - |S|) + i$ and $n_2 = |S| + j$ vertices respectively. See Figure 3 for an example.

Figure 3: Construction of graphs $H_1$ (right) and $H_2$ (middle) given medium cuts $S_1, S_2, S_3, S_4$ (left). Each colored set represents a medium cut surviving in either $H_1$ or $H_2$. The red and blue cuts survive in $H_2$, and the green cut survives in $H_1$. The purple cut survives in both $H_1$ and $H_2$.

We claim that $i + j < 2(\ell + 1)$ and $\ell < k/2$. For, if $i + j \geq 2(\ell + 1)$, then consider choosing $S_{\ell+1}$ to be an arbitrary medium cut; we would have $|\text{Venn}(S_1, \ldots, S_{\ell+1})| \geq i + j \geq 2(\ell + 1)$, contradicting maximality of $\ell$. Likewise, if $\ell \geq k/2$, then we would have $|\text{Venn}(S_1, \ldots, S_r)| \geq 2(r - 1)$ and $|\text{Venn}(S_1, \ldots, S_r)| \geq 2r$ where $r = [k/2]$; this would contradict Proposition 8.

From these two bounds, we conclude that $i + j \leq 2\ell + 1 \leq k$. Since $k < |S| < n - k$, both $n_1$ and $n_2$ are strictly larger than $k$ and strictly smaller than $n$. Hence, from the induction hypothesis, the number of medium cuts in $H_1$ and $H_2$ is at most $c_k(n_1 - k)$ and $c_k(n_2 - k)$ respectively.

We now claim that every medium cut of the original graph $G$ survives in either $H_1$ or $H_2$ (or both). For, suppose there is some medium cut $T$ where an edge $e' \in \partial T$ lies inside an atom $A_\ell$ and an edge $e'' \in \partial T$ lies inside an atom $B_\ell$. Then the atoms $A_\ell$ and $B_\ell$ would both split into two new atoms in $\text{Venn}(S_1, \ldots, S_\ell, T)$, giving $|\text{Venn}(S_1, \ldots, S_\ell, T)| \geq i + j + 2$. This contradicts maximality of $\ell$. 

9
Consequently, the number of medium cuts in $G$ is at most

$$c_k(n_1 - k) + c_k(n_2 - k) = c_k((n - |S| + i) + (|S| + j) - 2k) = c_k(n + i + j - 2k).$$

Now, $i + j \leq k$ so this is at most $c_k(n - k)$, completing the induction.

## 5 The Contraction Process

Our next goal will be to lower-bound the probability that a given $k$-cut $K$ is preserved during the Contraction Algorithm. More generally, for an edge set $J \subseteq E(G)$, we say that $J$ survives the Contraction Algorithm if no edge of $J$ ever gets selected during any iteration. Following [HS18], we define the Contraction Process up to stage $i$ for $J$ as follows.

Starting with the graph $G_n = G$, in stage $j$ we select an edge $e_j$ from the resulting (random) subgraph $G_j$ uniformly at random excluding the edges in $J$ itself, and contract $e_j$ to get the graph $G_{j-1}$. We stop when we reach $G_i$.

It is possible, and allowed, for some edges of $J$ to become self-loops and be removed from the graph. When considering a subgraph $G_j$ during the Contraction Process for $J$, bear in mind that we may have $J \not\subseteq E(G_j)$.

For the Contraction Process for $J$, we define the key statistic

$$R_i = \sum_{j=i+1}^{n} \frac{\bar{\lambda}_k}{|E(G_j)|}. \quad (2)$$

Here, $R_i$ serves as a linearized approximation to the probability of avoiding $J$ in the Contraction Algorithm. Specifically, we show the following result which is a slight reformulation of [HS18]:

**Proposition 11.** Let $J$ be an edge set and let $\alpha = |J|/\lambda_k$. Suppose we run the Contraction Algorithm up to stage $i \geq \max\{4\alpha k, k\}$. The probability that $J$ survives is at least $e^{-\alpha k E[R_i]} - \alpha k$, where the expectation is taken over the Contraction Process for $J$ up to stage $i$.

**Proof.** For $i \leq j \leq n$ let us define

$$x_j = \frac{|J|}{|E(G_j)|} = \frac{\alpha k \bar{\lambda}_k}{|E(G_j)|}$$

where $G_j$ is the subgraph obtained at stage $j$ of the Contraction Process for $J$ starting at $G$. We also define the corresponding random variable

$$L_G = \prod_{j=i+1}^{n} (1 - x_j).$$

Note that, by the property of iterated expectations, we calculate the expected value of $L_G$ as:

$$E[L_G] = \frac{1}{|E(G) \setminus J|} \sum_{e \in E(G) \setminus J} E[L_G | e_n = e] = \frac{1}{|E(G) \setminus J|} \sum_{e \in E(G) \setminus J} E[(1 - x_n) L_G/e].$$
\[ (1 - x_n) \sum_{e \in E(G) \setminus J} \frac{E[L_{G/e}]}{|E(G) \setminus J|}, \]

where \( G/e \) denotes the graph obtained by contracting edge \( e \) in \( G \); note that \( G_{n-1} = G/e_n \) during the Contraction Process.

We first claim that if we run the Contraction Algorithm on \( G \), then \( J \) survives to stage \( i \) with probability at least \( E[L_G] \). We show this by induction on \( n \). The case \( n = i \) holds vacuously since then \( L_G = 1 \) with probability one and \( J \) survives with probability one.

For the induction step, let \( n > i \). The Contraction Algorithm chooses edge \( e_n \) uniformly at random from \( E(G) \) and then continues on \( G/e_n \). Edge set \( J \) survives to stage \( i \) if and only if the following events occur: (i) \( e_n \in E(G) \setminus J \) and (ii) conditional on fixed choice of \( e_n = e \), the edge set \( J \) survives the Contraction Algorithm in \( G/e \) to stage \( i \). By the induction hypothesis, the latter event has probability at least \( E[L_{G/e}] \), and so:

\[
\Pr(J \text{ survives starting from } G) \geq \frac{1}{|E(G)|} \sum_{e \in E(G) \setminus J} E[L_{G/e}] = \frac{|E(G) \setminus J|}{|E(G)|} \sum_{e \in E(G) \setminus J} E[L_{G/e}] = E[L_G],
\]

Note now that
\[
\frac{|E(G) \setminus J|}{|E(G)|} = \frac{|E(G)| - |J|}{|E(G)|} = 1 - x_n,
\]
so this is at least
\[
(1 - x_n) \sum_{e \in E(G) \setminus J} \frac{E[L_{G/e}]}{|E(G) \setminus J|} = E[L_G],
\]
which concludes the induction.

So \( J \) survives the Contraction Algorithm with probability at least \( E[L_G] \). We need to bound \( L_G \). Consider some stage \( j \geq i \) of the Contraction Process for \( J \). By Proposition 6, since \( i \geq k \), we have \( |E(G_j)| \geq \frac{j}{2(k-1)}\lambda_k \) so \( x_j \leq 2\alpha(k-1)/j \). Since \( j \geq i \geq 4\alpha k \) this implies \( x_j \leq 1/2 \). We use the elementary identity \( 1 - x \geq e^{-x - x^2} \) for \( x \in [0, 1/2] \) to get:

\[
L_G = \prod_{j=i+1}^{n} (1 - x_j) \geq \prod_{j=i+1}^{n} e^{-x_j - x_j^2} \geq \prod_{j=i+1}^{n} e^{-\alpha k x_j / |E(G_j)| - (2\alpha(k-1)/j)^2} = e^{-\alpha k R_i - 4\alpha^2(k-1)^2} \sum_{j=i+1}^{n} 1/j^2 \geq e^{-\alpha k R_i - 4\alpha^2(k-1)^2}/i.
\]

Since \( i \geq 4\alpha k \), we thus have \( L_G \geq e^{-\alpha k R_i - \alpha k} \). Taking expectations and using Jensen’s inequality, we have \( E[L_G] \geq E[e^{-\alpha k R_i - \alpha k}] \geq e^{-\alpha k - \alpha k E[R_i]} \).

Using this, we can recover Karger and Stein’s original success probability of \( n^{-2\alpha(k-1)} \). Although it is much weaker than the bound of \( n^{-\alpha k} \) we want, this is useful for a few edge cases in the analysis.

**Corollary 12.** For any parameter \( \alpha \geq 1 \) and any \( k \)-cut \( K \) with \( |\partial K| \leq \alpha \lambda_k \), the Contraction Algorithm with parameter \( \tau = \lceil 4\alpha k \rceil \) selects \( K \) with probability at least \( n^{-2\alpha(k-1)} k^{-O(\alpha k)} \).

**Proof.** Consider the Contraction Process for edge set \( J = \partial K \). In each stage \( j \), Proposition 6 shows
that graph $G_j$ has at least \( \frac{j k}{2(k-1)} \lambda_k \) edges. Hence, with probability one, there holds

\[
R_\tau \leq \sum_{j=\tau+1}^{n} \frac{2(k-1)}{kj} \leq \frac{2(k-1)}{k} \log(n/\tau).
\]

By Proposition 11, the probability $K$ survives to stage $\tau$ is at least $e^{-\alpha k E[R_\tau] - \alpha k}$, which is at least $n^{-2\alpha(k-1)}$ with our bound on $R_\tau$. Next, suppose that $K$ does survive to stage $\tau$ (this includes the case where $n \leq \tau$). The resulting graph has at most $\tau$ vertices and hence at most $k^\tau$ different $k$-cuts. Thus, $K$ is selected from this graph with probability at least $k^{-\tau} \geq k^{-O(\alpha k)}$. Overall $K$ is selected with probability at least $n^{-2\alpha(k-1)} k^{-O(\alpha k)}$. \qed

6 Analyzing the Dynamics of the Contraction Process

Our goal now is to analyze the Contraction Process for a given edge set $J$. Let $\alpha = |J|/\lambda_k$. We fix some parameter $\varepsilon \in [0, 1/k)$, and define a good cut to be a medium cut $C$ with

\[
|\partial C \setminus J| \geq (1 - \varepsilon) \frac{k}{k-1} \lambda_k;
\]

the role of $\varepsilon$ will be explained later. We also define two related parameters

\[
\delta := \frac{1 - \varepsilon k}{k-1}, \quad \text{and} \quad \beta := k + 2\alpha k/\varepsilon.
\]

Note that $\delta > 0$ and $1 + \delta = (1 - \varepsilon)k/(k-1)$. We begin with a lower bound on edge count in a single iteration of the Contraction Algorithm.

**Proposition 13.** Let $s$ be the number of good cuts in $G$. If $n \geq \beta$, then

\[
m \geq s \cdot \frac{k}{k-1} \lambda_k / 2 + (n - s - \beta) \cdot \lambda_k.
\]

**Proof.** Each vertex $v$ of $G$ corresponds to a cut $C_v$. At most $k - 2$ of these vertex cuts may be small cuts. For, if there are $k - 1$ such vertices $v_1, v_2, \ldots, v_{k-1}$, then the $k$-cut generated by the \{v_1\}, \ldots, \{v_{k-1}\} would have weight below $(k-1) \cdot \frac{k}{k-1} \lambda_k = k \lambda_k = \lambda_k$, a contradiction.

Let $U$ denote the set of vertices $v$ for which cut $C_v$ is medium but not good. For $v \in U$, we have $|\partial C_v| \geq \frac{k}{k-1} \lambda_k$ and hence $|\partial C_v \cap J| \geq \varepsilon \lambda_k$. Each edge appears in at most two vertex cuts, so

\[
|J| \geq \frac{1}{2} \sum_{v \in U} |\partial C_v \cap J| \geq |U| \varepsilon \lambda_k / 2;
\]

since $|J| = \alpha k \lambda_k$, this implies $|U| \leq 2\alpha k/\varepsilon$.

Summarizing, at most $k - 2$ vertices correspond to small cuts, and at most $s + 2\alpha k/\varepsilon$ vertices correspond to medium cuts. The remaining vertices (at least $n - s - 2\alpha k/\varepsilon - k + 2$ of them) correspond to large cuts so their degree is at least $2\lambda_k$. We thus have

\[
2m \geq (s + 2\alpha k/\varepsilon) \cdot \frac{k}{k-1} \lambda_k + (n - s - 2\alpha k/\varepsilon - k + 2) \cdot 2\lambda_k \geq s \cdot \frac{k}{k-1} \lambda_k + (n - s - \beta) \cdot 2\lambda_k. \quad \square
\]

For our purposes, we can combine Proposition 6 and Proposition 13 to get the following (some-
what crude) estimate:

**Corollary 14.** If $G$ has $s$ good cuts, then $m \geq (n - \beta)\overline{\lambda}_k - \min\{s, (n - \beta)\}\overline{\lambda}_k/2$.

We are now ready derive the key bound on the random variable $R_i$ as defined in Eq. (2). For $p \geq j$ and $s \geq 0$, define the function

$$f(j, s, p) = \log(p/j) + \frac{\log(1 + (s/p)(1 + 1/\delta)(1 - (j/p)\delta))}{1 + \delta}.$$  \hspace{1cm} (3)

We will prove a bound on $\mathbf{E}[R_i]$ in terms of the function $f$ by induction. The derivation of the function $f$ is itself rather opaque; we describe the (non-rigorous) analysis which leads to it in Appendix A. We first observe a few analytical properties of function $f$.

**Proposition 15.** For $p \geq j$ and $s \geq 0$, we have the following:

1. Function $f(j, s, p)$ is a well-defined, nonnegative, nondecreasing, concave-down function of $s$.
2. The function $y \rightarrow y + f(j, se^{-(1+\delta)y}, p)$ is an increasing function of $y$.

**Proof.**
1. The argument of the logarithm in function $f$ is an affine function of $s$, with constant term 1 and coefficient $\frac{1}{p}(1 + 1/\delta)(1 - (j/p)\delta) \geq 0$.
2. The derivative as a function of $y$ is

$$\frac{\delta e^{(1+\delta)y}}{\delta e^{(1+\delta)y} + (s/p)(1 + \delta)(1 - (j/p)\delta)}$$

which is positive. \hspace{1cm} \Box

**Lemma 16.** Suppose that $G$ has $s$ good cuts and $n$ vertices. Then, for the Contraction Process for $J$ up to some stage $i$ with $\beta \leq i \leq n$, we have $\mathbf{E}[R_i] \leq f(i - \beta, s, n - \beta)$.

**Proof.** We show this by induction on $n$. We will write $p = n - \beta$, $j = i - \beta$ and $m = |E(G)|$. The case $n = i$ is clear, since $R_i = 0 = f(i - \beta, s, i - \beta)$.

For the induction step with $n > i$, the Contraction Process first selects an edge of $E(G) \setminus J$, arriving at a new graph $G'$ with $n - 1$ vertices. So

$$\mathbf{E}[R_i] = \frac{\overline{\lambda}_k}{m} + \mathbf{E}[R_i^{G'}],$$

where $R_i^{G'}$ denotes the random variables defined in Eq. (2) for graph $G'$.

Let random variable $S'$ denote the number of good cuts in $G'$. By the induction hypothesis applied to $G'$, we have

$$\mathbf{E}[R_i] \leq (\overline{\lambda}_k/m) + \mathbf{E}[f(j, S', p - 1)].$$  \hspace{1cm} (4)

Each good cut $C$ is selected with probability at least $\frac{|\partial C \setminus J|}{|E(G) \setminus J|} \geq \frac{k}{k-1} (1 - \varepsilon)\overline{\lambda}_k/m = (1 + \delta)\overline{\lambda}_k/m$, so

$$\mathbf{E}[S'] \leq s(1 - (1 + \delta)\overline{\lambda}_k/m) \leq se^{-1} \overline{\lambda}_k/m.$$
By Proposition 15, Jensen’s inequality applies for the random variable $S'$ in Eq. (4), giving:

$$E[R_i] \leq (\bar{X}_k/m) + f(j, E[S'], p - 1) \leq (\bar{X}_k/m) + f(j, se^{-(1+\delta)\bar{X}_k/m}, p - 1).$$

Next, by Corollary 14, we have $m \geq p\bar{X}_k - \min\{s, p\}\bar{X}_k/2$. So $\bar{X}_k/m \leq z$ where we define

$$z = \frac{2}{2p - \min\{s, p\}}.$$

Since $y + f(j, se^{-(1+\delta)y}, p - 1)$ is an increasing function of $y$, we therefore have

$$E[R_i] \leq z + f(j, se^{-(1+\delta)z}, p - 1).$$

To finish the proof and complete the induction, it suffices to show $y + f(j, se^{-(1+\delta)y}, p - 1) \leq f(j, s, p)$, or equivalently,

$$e^{(1+\delta)(z+f(j, se^{-(1+\delta)z}, p-1))} - e^{(1+\delta)f(j, s, p)} \leq 0. \quad (5)$$

After substituting in the formula for $f$, this expands to:

$$(\frac{p-1}{\frac{s}{p-1}})^{1+\delta}\left(e^{(1+\delta)z} + (\frac{s}{p-1})(1 + 1/\delta)(1 - (\frac{j}{p-1})^\delta)\right) - (\frac{p}{\frac{q}{p}})^{1+\delta}\left(1 + (\frac{p}{q})(1 + 1/\delta)(1 - (\frac{j}{p})^\delta)\right) \leq 0. \quad (6)$$

To simplify further, let us define a number of terms:

$$r = s/p, \quad q = j/p, \quad \theta = 1 - 1/p, \quad t = 2 - \min\{r, 1\}.$$

We thus have $\frac{p-1}{\frac{s}{p-1}} = \theta/q, \frac{s}{p-1} = r/\theta$, and $z = 2(1 - \theta)/t$. The inequality in Eq. (6) becomes:

$$(\theta/q)^{1+\delta}\left(e^{2(1-\theta)(1+\delta)/t} + (r/\theta)(1 + 1/\delta)(1 - (q/\theta)^\delta)\right) - (1/q)^{1+\delta}\left(1 + r(1 + 1/\delta)(1 - q^\delta)\right) \leq 0.$$

Clearing out common factor $q^{1+\delta}$ and multiplying the left term through by $\theta^{1+\delta}$, it is equivalent to:

$$(\theta^{1+\delta}e^{2(1-\theta)(1+\delta)/t} + r(1 + 1/\delta)(\theta^\delta - q^\delta)) - (1 + r(1 + 1/\delta)(1 - q^\delta)) \leq 0$$

Collecting terms, multiplying through by $\delta$, and changing signs for convenience, Eq. (5) is thus equivalent to showing:

$$\delta + r(1 + \delta)(1 - \theta^\delta) - \delta\theta^{1+\delta}e^{2(1-\theta)(1+\delta)/t} \geq 0. \quad (7)$$

Note that parameter $q$ no longer plays a role in Eq. (7). Since $r \geq 2 - t$, it suffices to show that

$$\delta + (2 - t)(1 + \delta)(1 - \theta^\delta) - \delta\theta^{1+\delta}e^{2(1-\theta)(1+\delta)/t} \geq 0. \quad (8)$$

To show Eq. (8), let us define a function

$$F(\theta, t) = \delta + (2 - t)(1 + \delta)(1 - \theta^\delta) - \delta\theta^{1+\delta}e^{2(1-\theta)(1+\delta)/t}$$

for independent variables $\theta, t$. We need to show that $F(\theta, t) \geq 0$ for all $\theta \in [0, 1]$ and $t \in [1, 2]$. 


The second partial derivative of \( F \) with respect to \( t \) is given by

\[
\frac{\partial^2 F(\theta, t)}{\partial t^2} = -4\delta(1 + \delta)(1 - \theta)\theta^{1+\delta}e^{2(1+\delta)(1-\theta)/t((1+\delta)(1-\theta) + t)}
\]

which is clearly negative for \( \delta, \theta, t \) in the given range. Thus, the minimum value of \( F(\theta, t) \) in the region occurs at either \( t = 1 \) or \( t = 2 \). So, in order to show that \( F(\theta, t) \geq 0 \), it suffices to show that \( F(\theta, 1) \geq 0 \) and \( F(\theta, 2) \geq 0 \).

At \( t = 2 \) we have \( F(\theta, 2) = \delta(1 - e^{(1+\delta)(1-\theta)\theta^{1+\delta}}) \). To show that \( F(\theta, 2) \geq 0 \), we thus need to show that \( e^{(1+\delta)(1-\theta)\theta^{1+\delta}} \leq 1 \), or equivalently \( e^{1-\theta} \leq 1 \); this can be verified by routine calculus.

At \( t = 1 \), we have \( F(\theta, 1) = \delta + (1 + \delta)(1 - \theta^\delta) - \delta\theta^{1+\delta}e^{2(1-\theta)(1+\delta)} \). Note that \( F(1, 1) = 0 \). So, in order to show that \( F(\theta, 1) \geq 0 \) for all \( \theta \in [0, 1] \), it suffices to show that the derivative of \( F(\theta, 1) \) with respect to \( \theta \) is negative for \( \theta \in (0, 1) \). This derivative is given by:

\[
\frac{\partial F(\theta, 1)}{\partial \theta} = -\delta(1 + \delta)e^{2(1+\delta)(1-\theta)\theta^{1+\delta}}(e^{-2(1+\delta)(1-\theta) - 2\theta^2 + \theta}).
\]

To show this is negative, it suffices to show that \( e^{-2(1+\delta)(1-\theta) - 2\theta^2 + \theta} > 0 \). Since \( \delta \leq 1/2 \), it suffices to show that \( e^{-3(1-\theta) - 2\theta^2 + \theta} > 0 \) which can be verified by routine calculus for \( \theta \in (0, 1) \). This shows that \( \frac{\partial F(\theta, 1)}{\partial \theta} \leq 0 \), and so \( F(\theta, t) \geq F(1, 1) = 0 \).

Thus \( F(\theta, t) \geq 0 \) for all \( \theta \in [0, 1] \) and \( t \in [1, 2] \) and hence the inequality of Eq. (8) holds. \( \square \)

### 7 Putting it together: Bounds on the Contraction Algorithm

We now finish by getting our main bound for the Contraction Algorithm.

**Lemma 17.** Suppose that \( J \) is an edge set with \( \alpha = |J|/\lambda_k \) and \( n \geq i \geq 8\alpha k^2 + 2k \). Then \( J \) survives the Contraction Algorithm to stage \( i \) with probability at least \((n/i)^{-\alpha k}k^{-O(\alpha k^2)}\).

**Proof.** Let us set \( \varepsilon = \frac{k+1}{2k} \), and also define \( \beta = k + 2\alpha k/\varepsilon \), and \( j = i - \beta, p = n - \beta \) and \( \delta = \frac{1}{2k} \). By Theorem 9, the number of medium cuts in \( G \) is at most \( an \) for \( a = k^{O(k)} \), and so Lemma 16 gives:

\[
E[R_i] \leq f(j, an, p) = \log(p/j) + \frac{\log(1 + \frac{an}{p}(1 + \delta)(1 - (j/p)\delta))}{1 + \delta}.
\]

Our condition on \( i \) ensures \( i \geq 2\beta \). So \( p \geq n/2 \) and \( j \geq i/2 \), and thus \( \log(p/j) \leq \log(n/i) + O(1) \) and \( an/p \leq 2a \). We therefore have

\[
E[R_i] \leq \log(n/i) + \frac{\log(1 + 2a(1 + \delta))}{1 + \delta} + O(1) \leq \log(n/i) + \log a + O(1)
\]

Note that \( i \geq \max\{4\alpha k, k\} \) as required in Proposition 11. Thus, \( J \) survives with probability at least \((n/i)^{-\alpha k}k^{-\alpha k}k^{-O(\alpha k^2)}\). Since \( a = k^{O(k)} \), this is at least \((n/i)^{-\alpha k}k^{-O(\alpha k^2)} \). \( \square \)

**Theorem 18.** Running the Contraction Algorithm with parameter \( \tau = \lceil 20\alpha k^2 \rceil \) produces any given \( k \)-cut \( K \) of weight at most \( \alpha \lambda_k \) with probability at least \( n^{-\alpha k}k^{-O(\alpha k^2)} \).

**Proof.** If \( n \geq \tau \), then Lemma 17 applied to \( J = \partial K \) (noting that necessarily \( \alpha \geq 1 \)) shows that \( K \) survives to \( G_\tau \) with probability at least \((n/\tau)^{-\alpha k}k^{-O(\alpha k^2)} \). Then \( K \) is selected from \( G_\tau \) with
probability at least \( k^{-\tau} \geq k^{-O(\alpha k^2)} \). Combining these probability bounds gives the stated result. If \( n < \tau \), then the Contraction Algorithm simply selects a random \( k \)-cut, and so \( K \) is chosen with probability at least \( k^{nT} \geq k^{-O(\alpha k^2)} \).

\[ \textbf{Corollary 19.} \text{ There are at most } n^{\alpha k}k^{O(\alpha k^2)} \text{ many } k \text{-cuts in } G \text{ with weight at most } \alpha \lambda_k. \]

We could enumerate these \( k \)-cuts by repeatedly running the Contraction Algorithm, but each iteration would cost \( O(n^2) \) time giving an overall runtime of roughly \( O(n^{\alpha k+2}) \). The next result shows how to remove this extraneous \( n^2 \) factor using a recursive version of the Contraction Algorithm from [KS96]. Note that directly printing out the \( k \)-cuts could take \( \Omega(n^{\alpha k+1}) \) time, since each \( k \)-cut defines a partition of \( V \). Hence, the algorithm necessarily produces the collection of \( k \)-cuts in a compressed data structure, which supports basic operations such as counting, sampling, etc. See [KS96] or [HS18] for a more in-depth discussion.

\[ \textbf{Theorem 20.} \text{ For each } k \geq 3, \text{ there is an algorithm to enumerate all } k \text{-cuts of weight at most } \alpha \lambda_k \text{ in time } n^{\alpha k}(\log n)^{O(\alpha k^2)} \text{ with probability at least } 1 - 1/\text{poly}(n). \]

\[ \text{Proof.} \text{ First, if } n \leq 2^k, \text{ then we directly use the Contraction Algorithm to stage } \tau = \lceil 4\alpha k \rceil. \text{ By Corollary 12, this enumerates any given } k \text{-cut with probability at least } n^{-2\alpha(k-1)}k^{-O(\alpha k)}, \text{ so we must run it for } n^{2\alpha(k-1)}k^{O(\alpha k)} \cdot \text{poly}(\alpha, k, \log n) \text{ trials to get them all. This gives overall runtime of } n^{2\alpha(k-1)}k^{O(\alpha k)} \cdot \text{poly}(\alpha, k, \log n) \cdot O(n^{2}), \text{ which is at most } e^{O(\alpha k^2)} \text{ by our assumption on } n. \text{ We thus assume for the remainder of the proof that } n \geq 2^k. \]

We use a recursive algorithm, whose state is represented as a pair \((H, \ell)\) where \( H \) is the current graph and \( \ell = 0, \ldots, T \) is the current level in the recursion. The algorithm begins with the input graph \((G, 0)\) at level \( \ell = 0 \). Given input \((G_\ell, \ell)\) at level \( \ell \), there are two cases. If \( \ell < T \), the algorithm runs \( t_\ell = \lceil (n_\ell/n_{\ell+1})^{\alpha k} \rceil \) independent trials of the Contraction Algorithm to \( n_{\ell+1} \) vertices and recursively calls \((H, \ell + 1)\) for each resulting contracted graph \( H \). Otherwise, if \( \ell = T \), the algorithm outputs a randomly chosen \( k \)-cut. Here, the parameters \( n_\ell \) are given by

\[ n_\ell = \lceil \max\{n \left( \frac{2}{\alpha k} \right)_i, 20\alpha k^2 \} \rceil \]

and the recursion depth \( T \) is the first value with \( n_T = [20\alpha k^2] \). Since \( \log(20\alpha k^2) \leq O(\alpha k) \) and \( \alpha k/2 \geq 3/2 \), we have \( T \leq O\left( \frac{\log n}{\log(\alpha k)} \right) \).

To calculate the algorithm’s success probability, fix some \( k \)-cut \( K \) of \( G \) with \( |\partial K| \leq \alpha \lambda_k \), and define a state \((G_\ell, \ell)\) to be \textit{successful} if no edge in \( K \) has been contracted so far from \( G_0 = G \) to \( G_\ell \). Clearly, \((G_0, 0)\) is successful. For each successful input \((G_\ell, \ell)\) with \( \ell < T \), by Lemma 17 with \( i = n_{\ell+1} \), the probability that \( K \) survives on each trial is at least \( (n_\ell/n_{\ell+1})^{-\alpha k \psi} \) where \( \psi = k^{-O(\alpha k^2)} \). Over all \( t_\ell \) trials, \( K \) survives at least once with probability

\[ 1 - \left( 1 - (n_\ell/n_{\ell+1})^{-\alpha k \psi} \right)^{t_\ell} \geq 1 - e^{-\psi}. \]

Thus, given that some instance \((G_\ell, \ell)\) in the recursion tree is successful, the probability that at least one instance \((G_{\ell+1}, \ell + 1)\) is successful is at least \( 1 - e^{-\psi} \geq \psi/2 \). Over all the \( T = O\left( \frac{\log n}{\log(\alpha k)} \right) \) levels of the recursion, the probability that there is some successful instance \((G_T, T)\) is at least \( \left( \psi/2 \right)^T \geq (\log n)^{-O(\alpha k^2)}. \) Finally the probability of selecting \( K \) from a successful instance \((G_T, T)\) is at least \( k^{-n_T} \geq k^{-O(\alpha k^2)} \). So, overall, \( K \) is selected with probability at least \( (\log n)^{-O(\alpha k^2)} \cdot k^{-O(\alpha k^2)} \).

Due to our assumption that \( n \geq 2^k \), this is at least \( (\log n)^{-O(\alpha k^2)} \).
We now bound the runtime. For each level \( \ell < T \), there are \( \prod_{j=0}^{\ell-1} t_j = \prod_{j=1}^{\ell-1} 2(n_j/n_{j+1})^{\alpha k} = 2^\ell (n_0/n_{\ell})^{\alpha k} \) instances \((G_\ell, \ell)\). In each such instance, the algorithm runs \( t_\ell \) trials of the Contraction Algorithm, each taking \( O(n_\ell^2) \) time. The running time over all instances \((G_\ell, \ell)\) is therefore at most

\[
2^\ell (n_0/n_{\ell})^{\alpha k} \cdot t_\ell \cdot O(n_\ell^2) \leq O(2^T \cdot (n_0/n_{T+1})^{\alpha k} \cdot n_T^2),
\]

which is at most \( O(2^T (2n_0)^{\alpha k}) \) since \( n_{\ell+1} \geq n_\ell^{2/(\alpha k)} / 2 \). Summed over all \( T \) recursion levels, the total runtime is at most \( T \cdot O(2^T (2n_0)^{\alpha k}) \leq (2n)^{\alpha k} \cdot (\log n)^{O(1)}. \)

If we repeat the entire recursive algorithm from \((G_0, 0)\) a total of \((\log n)^{O(\alpha k^2)}\) times, then \( K \) is selected with probability at least \( 1/2 \). There are \( K^{O(\alpha k^2)} n_0^{\alpha k} \) many such \( k \)-cuts, so we run a further \( O(\alpha k^2 \log k \log n) \) many trials to enumerate them all with probability \( 1 - 1/\poly(n) \).

As one concrete application, we get the main result:

**Theorem 21.** There is an algorithm to compute \( \lambda_k \) in time \( n^k (\log n)^{O(k^2)} \) for any value \( k \).

**Proof.** For \( k = 2 \), this is the standard Recursive Contraction Algorithm of [KS96]. Otherwise, apply Theorem 20 with \( \alpha = 1 \). This gives a large collection of \( k \)-cuts, which includes all the minimum \( k \)-cuts with high probability. We output the minimum weight of all \( k \)-cuts found. (The operation of taking minimum weight can be performed on the corresponding data structure.)

## A Heuristic Bound on \( R_i \)

Given a graph \( G \) with \( s \) medium cuts, consider running the Contraction Process for some edge set \( J \) up to stage \( i \). We will focus on the case where \( s \ll n \); as it will later turn out, the resulting formulas are also correct (although not optimized) when \( s \) is larger than \( n \).

In each stage \( j \geq i \), where the intermediate graph \( G_j \) has \( j \) vertices and \( m_j \) edges, each good cut \( C \) gets selected with probability \( \frac{|\delta C \setminus J|}{m_j - |J|} \geq (1 - \varepsilon) \frac{k}{k-1} \overline{\lambda}_k/m_j = (1 + \delta) \overline{\lambda}_k/m_j \). Letting \( S_i \) denote the number of surviving good cuts at stage \( i \), we thus have:

\[
E[S_i] \leq s \prod_{j=i+1}^{n} \left(1 - \frac{(1 + \delta) \overline{\lambda}_k}{m_j}\right) \leq se^{-\sum_{j=i+1}^{n} (1+\delta) \overline{\lambda}_k/m_i} = se^{-(1+\delta) R_i}.
\]

Since this is just a heuristic derivation, we blur the distinction between \( E[S_i] \) and \( S_i \), and we suppose that \( S_i \) itself also satisfies this bound, i.e. \( S_i \leq se^{-(1+\delta) R_i} \).

We have \( R_{i-1} = \frac{\overline{\lambda}_k}{m_i} + R_i \). By Proposition 13, we have \( m_i \geq S_i \overline{\lambda}_k / 2 + (i - S_i - \beta) \overline{\lambda}_k \), so

\[
\frac{\overline{\lambda}_k}{m_i} \leq \frac{1}{i - \beta - S_i / 2}.
\]

(9)

In order to carry out the induction proof later, we will need our bound on \( R_i \) to have a simple closed form with nice concavity properties. In order to achieve this, we will need to use an upper bound on the quantity \( \frac{\overline{\lambda}_k}{m_i} \) which is a linear function of \( S_i \). As we have mentioned, in the relevant case, we have \( s \leq n \), and in this case we will also have \( S_i \leq i \). We can then upper-bound the RHS of Eq. (9) by its secant line from \( S_i = 0 \) to \( S_i = i - \beta \), yielding

\[
\frac{\overline{\lambda}_k}{m_i} \leq \frac{1}{i - \beta} \left(1 + \frac{S_i}{i - \beta}\right);
\]
note that by Proposition 6, this upper bound will also be valid in the case where $S_i \geq i - \beta$.

Again ignoring any distinctions between random variables and their expectations, this implies

$$R_{i-1} \leq R_i + \frac{1}{i-\beta} \left( 1 + \frac{se^{-(1+\delta)R_i}}{i-\beta} \right).$$

If we define $g(x) = R_{x+\beta}$ and $p = n - \beta$, then this can be relaxed to a differential equation with $g'(i) \approx R_{i+\beta} - R_{i-1+\beta}$ defined as follows:

$$g'(x) = -\frac{1}{x} \left( 1 + \frac{se^{-(1+\delta)g(x)}}{x} \right), \quad g(p) = 0.$$

The differential equation has a closed-form solution:

$$g(x) = \log(p/x) + \frac{\log(1 + (s/p)(1 + 1/\delta)(1 - (x/p)^\delta))}{1 + \delta}.$$

Note the similarity of function $g$ to the function $f$ from Eq. (3) defined in Section 6.

References

[ALWZ20] Ryan Alweiss, Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Improved bounds for the sunflower lemma. In Proc. 52nd annual ACM Symposium on Theory of Computing (STOC), pages 624–630, 2020.

[CQX20] Chandra Chekuri, Kent Quanrud, and Chao Xu. LP relaxation and tree packing for minimum $k$-cut. SIAM Journal on Discrete Mathematics, 34(2):1334–1353, 2020.

[ER60] P. Erdös and R. Rado. Intersection theorems for systems of sets. The Journal of the London Mathematical Society, 35:85–90, 1960.

[GH88] Olivier Goldschmidt and Dorit S Hochbaum. Polynomial algorithm for the $k$-cut problem. In Proc. 29th annual IEEE Symposium on Foundations of Computer Science (FOCS), pages 444–451, 1988.

[GH94] Olivier Goldschmidt and Dorit S. Hochbaum. A polynomial algorithm for the $k$-cut problem for fixed $k$. Mathematics of Operations Research, 19(1):24–37, 1994.

[GLL18] Anupam Gupta, Euiwoong Lee, and Jason Li. Faster exact and approximate algorithms for $k$-cut. In Proc. 59th annual IEEE Symposium on Foundations of Computer Science (FOCS), pages 113–123, 2018.

[GLL19] Anupam Gupta, Euiwoong Lee, and Jason Li. The number of minimum $k$-cuts: improving the Karger-Stein bound. In Proc. 51st annual ACM Symposium on Theory of Computing (STOC), pages 229–240, 2019.

[GLL20] Anupam Gupta, Euiwoong Lee, and Jason Li. The Karger-Stein algorithm is optimal for $k$-cut. In Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, pages 473–484, 2020.

[HS18] David G. Harris and Aravind Srinivasan. Improved bounds and algorithms for graph cuts and network reliability. Random Structures & Algorithms, 52(1):74–135, 2018.

[Kar93] David R. Karger. Global min-cuts in RNC, and other ramifications of a simple min-cut algorithm. In Proc. 40th annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 21–30, 1993.

[KS96] David R. Karger and Clifford Stein. A new approach to the minimum cut problem. Journal of the ACM (JACM), 43(4):601–640, 1996.
[KT18] Ken-ichi Kawarabayashi and Mikkel Thorup. Deterministic edge connectivity in near-linear time. *Journal of the ACM (JACM)*, 66(1):1–50, 2018.

[KYN06] Yoko Kamidoi, Noriyoshi Yoshida, and Hiroshi Nagamochi. A deterministic algorithm for finding all minimum $k$-way cuts. *SIAM Journal on Computing*, 36(5):1329–1341, 2006.

[Li19] Jason Li. Faster minimum $k$-cut of a simple graph. In *Proc. 60th annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1056–1077, 2019.

[Tho08] Mikkel Thorup. Minimum $k$-way cuts via deterministic greedy tree packing. In *Proc. 40th annual ACM Symposium on Theory of Computing (STOC)*, pages 159–166, 2008.