Tight Bounds for Gomory-Hu-like Cut Counting*

Rajesh Chitnis, Lior Kamma, and Robert Krauthgamer

Weizmann Institute of Science, Rehovot, Israel.
{rajesh.chitnis,lior.kamma,robert.krauthgamer}@weizmann.ac.il

Abstract. By a classical result of Gomory and Hu (1961), in every edge-weighted graph $G = (V, E, w)$, the minimum st-cut values, when ranging over all $s, t \in V$, take at most $|V| - 1$ distinct values. That is, these $\binom{|V|}{2}$ instances exhibit *redundancy factor* $\Omega(|V|)$. They further showed how to construct from G a tree (V, E', w') that stores all minimum st-cut values. Motivated by this result, we obtain *tight* bounds for the redundancy factor of several generalizations of the minimum st-cut problem.

1. Group-Cut: Consider the minimum (A, B)-cut, ranging over all subsets $A, B \subseteq V$ of given sizes $|A| = \alpha$ and $|B| = \beta$. The redundancy factor is $\Omega_{\alpha,\beta}(|V|)$.
2. Multiway-Cut: Consider the minimum cut separating every two vertices of $S \subseteq V$, ranging over all subsets of a given size $|S| = k$. The redundancy factor is $\Omega_k(|V|)$.
3. Multicut: Consider the minimum cut separating every demand-pair in $D \subseteq V \times V$, ranging over collections of $|D| = k$ demand pairs. The redundancy factor is $\Omega_k(|V|^k)$. This result is a bit surprising, as the redundancy factor is much larger than in the first two problems.

A natural application of these bounds is to construct small data structures that stores all relevant cut values, à la the Gomory-Hu tree. We initiate this direction by giving some upper and lower bounds.

1 Introduction

One of the most fundamental combinatorial optimization problems is *minimum st-cut*, where given an edge-weighted graph $G = (V, E, w)$ and two vertices $s, t \in V$, the goal is to find a set of edges of minimum total weight that separates s, t (meaning that removing these edges from G ensures there is no s - t path). This problem was studied extensively, see e.g. the famous minimum-cut/maximum-flow duality [8], and can be solved in polynomial time. It has numerous theoretical applications, such as bipartite matching and edge-disjoint paths, in addition to being extremely useful in many practical settings, including network connectivity, network reliability, and image segmentation, see e.g. [1]

* Supported in part by the Israel Science Foundation (grant #897/13). A full version [6] of this extended abstract is available at arXiv:1511.08647.

for details. Several generalizations of the problem, such as multiway cut, multicut, and k-cut, have been well-studied in operations research and theoretical computer science.

In every graph $G = (V, E, w)$, there are in total $\binom{|V|}{2}$ instances of the minimum st-cut problem, given by all pairs $s, t \in V$. Potentially, each of these instances could have a different value for the minimum cut. However, the seminal work of Gomory and Hu [9] discovered that *undirected* graphs admit a significantly stronger bound (see also [1, Lemma 8.15] or [7, Section 3.5.2]).

Theorem 1 ([9]). *Let $G = (V, E, w)$ be an edge-weighted undirected graph. Then the number of distinct values over all possible $\binom{|V|}{2}$ instances of the minimum st-cut problem is at most $|V| - 1$.*

The beautiful argument of Gomory and Hu shows the existence of a tree $\mathcal{T} = (V, E', w')$, usually called a *flow-equivalent tree*, such that for every $s, t \in V$ the minimum st-cut value in $\mathcal{T}$ is exactly the same as in G. (They further show how to construct a so-called *cut-equivalent tree*, which has the stronger property that every vertex-partitioning that attains a minimum st-cut in $\mathcal{T}$, also attains a minimum st-cut in G; see Section 1.3 for more details on this and related work.) Every G which is a tree (e.g., a path) with distinct edge weights has exactly $|V| - 1$ distinct values, and hence the Gomory-Hu bound is existentially tight.

Another way to view Theorem 1 is that there is always a huge redundancy between the $\binom{|V|}{2}$ minimum st-cut instances in a graph. More precisely, the "redundancy factor", measured as the ratio between the number of instances and the number of distinct optimal values attained by them, is always $\Omega(|V|)$. We study this question of redundancy factor for the following generalizations of minimum st-cut. Let $G = (V, E, w)$ be an undirected edge-weighted graph.

- Group-Cut: Given two disjoint sets $A, B \subseteq V$ find a minimum (A, B)-cut, i.e., a set of edges of minimum weight that separates every vertex in A from every vertex in B.
- Multiway-Cut: Given $S \subseteq V$ find a minimum-weight set of edges, whose removal ensures that for every $s \neq s' \in S$ there is no s-s' path.
- Multicut: Given $Q \subseteq V \times V$ find a minimum-weight set of edges, whose removal ensures that for every $(q, q') \in Q$ there is no q-q' path.

In order to present our results about the redundancy in these cut problems in a streamlined way, we introduce next the terminology of vertex partitions and demand graphs.

Cut Problems via Demand Graphs. Denote by $\texttt{Par}(V)$ the set of all partitions of V, where a *partition* of V is, as usual, a collection of pairwise disjoint subsets of V whose union is V. Given a partition $\Pi \in \texttt{Par}(V)$ and a vertex $v \in V$, denote by $\Pi(v)$ the unique $S \in \Pi$ satisfying $v \in S$. Given a graph $G = (V, E, w)$, define the function $\texttt{Cut}_G : \texttt{Par}(V) \to \mathbb{R}^{\geq 0}$ to be $\texttt{Cut}_G(\Pi) = \sum_{uv \in E : \Pi(u) \neq \Pi(v)} w(uv)$. We shall usually omit the subscript G, since the graph will be fixed and clear from the context.

Cut problems as above can be defined by specifying the graph G and a collection D of *demands*, which are the vertex pairs that need to be separated. We can view (V, D) as an (undirected and unweighted) *demand graph*, and by slight abuse of notation, D will denote both this graph and its edges. For example, an instance of Group-Cut is defined by G and demands that form a complete bipartite graph $K_{A,B}$ (to formally view it as a graph on V, let us add that vertices outside of $A \cup B$ are isolated). We say that partition $\Pi \in \mathtt{Par}(V)$ *agrees* with D if every $uv \in D$ satisfies $\Pi(u) \neq \Pi(v)$. The optimal cut-value for the instance defined by G and D is given by

$$\mathtt{mincut}_G(D) := \min\{\mathtt{Cut}_G(\Pi) : \Pi \in \mathtt{Par}(V) \text{ agrees with } D\}.$$

Redundancy among Multiple Instances. We study multiple instances on the same graph $G = (V, E, w)$ by considering a family $\mathcal{D}$ of demand graphs. For example, all minimum *st*-cut instances in a single G corresponds to the family $\mathcal{D}$ of all demands of the form $D = \{(s,t)\}$ (i.e., demand graph with one edge). The collection of optimal cut-values over the entire family $\mathcal{D}$ of instances in a single graph G, is simply $\{\mathtt{mincut}(D) : D \in \mathcal{D}\}$. We are interested in the ratio between the size of this collection as a multiset and its size as a set, i.e., with and without counting multiplicities. Equivalently, we define the *redundancy factor* of a family $\mathcal{D}$ of demand graphs to be

$$\text{redundancy}(\mathcal{D}) := \frac{|\mathcal{D}|}{|\{\mathtt{mincut}(D) : D \in \mathcal{D}\}|} ,$$

where throughout, $|A|$ denotes the size of A as a *set*, i.e., ignoring multiplicities.

Motivation and Potential Applications. A natural application of the redundancy factor is to construct small data structures that stores all relevant cut values. For the minimum *st*-cut problem, Gomory and Hu were able to collect all the cut values into a tree on the same vertex set V. This tree can easily support fast query time, or a distributed implementation (labeling scheme) [12].

In addition, large redundancy implies that there is a small collection of cuts that contains a minimum cut for each demand graph. Indeed, first make sure all cut values in G are distinct (e.g., break ties consistently by perturbing edge weights), and then pick for each cut-value in $\{\mathtt{mincut}(D) : D \in \mathcal{D}\}$ just one cut that realizes it. This yields a data structure that reports, given demands $D \in \mathcal{D}$, a vertex partition that forms a minimum cut (see more in Section 1.2).

1.1 Main Results

Throughout, we denote $n = |V|$. We use the notation $O_\gamma(\cdot)$ to suppress factors that depend only on γ, and similarly for Ω and Θ.

The Group-Cut *problem.* In this problem, the demand graph is a complete bipartite graph $K_{A,B}$ for some subsets $A, B \subset V$. We give a tight bound on the redundancy factor of the family of all instances where A and B are of given sizes α and β, respectively. The special case $\alpha = \beta = 1$ is just all minimum st-cuts in G, and thus recovers the Gomory-Hu bound (Theorem 1).

Theorem 2. *For every graph $G = (V, E, w)$ and for every $\alpha, \beta \in \mathbb{N}$, we have $|\{\texttt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}| = O_{\alpha,\beta}(n^{\alpha+\beta-1})$, hence the family of (α, β)-group-cuts has redundancy factor $\Omega_{\alpha,\beta}(n)$. Furthermore, this bound is existentially tight (attained by some graph G) for all α, β and n.*

The Multiway-Cut *problem.* In this problem, the demand graph is a complete graph K_S for some subset $S \subseteq V$. We give a tight bound on the redundancy factor of the family of all instances where S is of a given size $k \geq 2$. Again, the Gomory-Hu bound is recovered by the special case $k = 2$.

Theorem 3. *For every graph $G = (V, E, w)$ and for every integer $k \in \mathbb{N}$, we have $|\{\texttt{mincut}(K_S) : |S| = k\}| = O_k(n^{k-1})$, hence the family of k-multiway-cuts has redundancy factor $\Omega_k(n)$. Furthermore, this bound is existentially tight for all n and k.*

The Multicut *problem.* In this problem, the demand graph is a collection D of demand pairs. We give a tight bound on the redundancy factor of the family of all instances where D is of a given size $k \in \mathbb{N}$. Again, the Gomory-Hu bound is recovered by the special case $k = 1$.

Theorem 4. *For every graph $G = (V, E, w)$ and $k \in \mathbb{N}$, we have $|\{\texttt{mincut}(D) : D \subseteq V \times V,\ |D| = k\}| = O_k(n^k)$, and hence the family of k-multicuts has redundancy factor $\Omega_k(n^k)$. Furthermore, this bound is existentially tight for all n and k.*

Theorem 4 is a bit surprising, since it shows a redundancy factor that is polynomial, rather than linear, in n (for fixed α, β and k), so in general Multicut has significantly larger redundancy than Group-Cut and Multiway-Cut.

1.2 Extensions and Applications

Our main results above actually apply more generally and have algorithmic consequences, as discussed below briefly.

Terminals Version. In this version, the vertices to be separated are limited to a subset $T \subseteq V$ called *terminals*, i.e., we consider only demands inside $T \times T$. All our results above (Theorems 2, 3, and 4) immediately extend to this version of the problem — we simply need to replace $|V|$ by $|T|$ in all the bounds. As an illustration, the terminals version of Theorem 1 states that the $\binom{|T|}{2}$ minimum st-cuts (taken over all $s, t \in T$) attain at most $|T| - 1$ distinct values. (See also [7, Section 3.5.2] for this same version.) Extending our proofs to the terminals version is straightforward; for example, in Section 2.1 we need to consider polynomials in $|T|$ variables instead of $|V|$ variables.

Data Structures. Flow-equivalent or cut-equivalent trees, such as those constructed by Gomory and Hu [9], may be viewed more generally as succinct data structures that support certain queries, either for the value of an optimal cut, or for its vertex-partition, respectively. Motivated by this view, we define data structures, which we call as evaluation schemes, that preprocess an input graph G, a set of terminals T, and a collection of demand graphs $\mathcal{D}$, so as to answer a cut query given by a demand graph $D \in D$. The scheme has two flavors, one reports the minimum cut-value, the second reports a corresponding vertex-partition. In Section 5 we initiate the study of such schemes, and provide constructions and lower bounds for some special cases.

Functions Different From Cuts. Recall that the value of the minimum *st*-cut is $\min\{\texttt{Cut}_G(X, V \setminus X) : X \subseteq V,\ s \in X,\ t \notin X\}$. Cheng and Hu [5] extended the Gomory-Hu bound (Theorem 1) to a wider class of problems as follows. Instead of a graph G, fix a ground set V and a function $f : 2^V \to \mathbb{R}$. Now for every $s, t \in V$, consider the optimal value $\min\{f(X) : X \subseteq V,\ |X \cap \{s,t\}| = 1\}$. They showed that ranging over all $s, t \in V$, the number of distinct optimal values is also at most $|V|-1$. All our results above (Theorems 2, 3, and 4) actually extend to every function $f : \texttt{Par}(V) \to \mathbb{R}$. However, to keep the notation simple, we opted to present all our results only for the function $\texttt{Cut}$.

Directed Graphs. What happens if we ask the same questions for the directed variants of the three problems considered previously? Here, an $s \to t$ cut means a set of edges whose removal ensures that no $s \to t$ path exists. Under this definition, we can construct explicit examples for the directed variants of our three problems above where there is no *non-trivial redundancy*, i.e., the number of distinct cut values is asymptotically equal to the total number of instances.

1.3 Related Work

Gomory and Hu [9] showed how to compute a cut-equivalent tree, and in particular a flow-equivalent tree, using $|V| - 1$ minimum *st*-cut computations on graphs no larger than G. Gusfield [10] has shown a version where all the cut computations are performed on G itself (avoiding contractions). For unweighted graphs, a faster (randomized) algorithm for computing a Gomory-Hu tree which runs in $\tilde{O}(|E| \cdot |V|)$ time was recently given by Bhalgat et al. [3].

We have mentioned that Cheng and Hu [5] extended Theorem 1 from cuts to an arbitrary function $f : 2^V \to \mathbb{R}$. They further showed how to construct a flow-equivalent tree for this case (but not a cut-equivalent tree). Benczúr [2] showed a function f for which there is no cut-equivalent tree. In addition, he showed that for directed graphs, even flow-equivalent trees do not exist in general.

Another relevant notion here is that of mimicking networks, introduced by Hagerup et al. [11]. A mimicking network for $G = (V, E, w)$ and a terminals set $T \subseteq V$ is a graph $G' = (V', E', w')$ where $T \subset V'$ and for every $X, Y \in T$, the minimum (X, Y)-cut in G and in G' have the exact same value. They showed that every graph has a mimicking network with at most $2^{2^{|T|}}$ vertices. Some improved bounds are known, e.g., for graphs that are planar or have bounded treewidth, as well as some lower bounds [4, 14, 13]. Mimicking networks deal with the Group-Cut problem for all $A, B \subset V$; we consider A, B of bounded size, and thus typically achieve much smaller bounds.

2 Group-Cut: The Case of Complete Bipartite Demands

This section is devoted to proving Theorem 2. First we give two proofs, one in Section 2.1 via polynomials and the second in Section 2.2 via matrices, for the bound $|\{\texttt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}| = O_{\alpha,\beta}(n^{\alpha+\beta-1})$. Then in Section 2.3 we construct examples of graphs for which this bound is tight. Since $|\{K_{A,B} : |A| = \alpha, |B| = \beta\}| = \binom{n}{\alpha} \cdot \binom{n-\alpha}{\beta} = \Theta_{\alpha,\beta}(n^{\alpha+\beta})$, it follows that the redundancy factor is $\Omega_{\alpha,\beta}(n)$.

2.1 Proof via Polynomials

Let $r = \binom{n}{\alpha}\binom{n-\alpha}{\beta}$ and the set of demand graphs for (α, β)-Group-Cut be $\{K_{A_1,B_1}, K_{A_2,B_2}, \ldots, K_{A_r,B_r}\}$. For every vertex $v \in V$ we assign a boolean variable denoted by ϕ_v. Given an instance A, B we can assume that the optimal partition only contains two parts, one which contains A and other which contains B, since we can merge other parts into either of these parts.

Fix some $j \in [r]$. Recall that $\Pi = \{U, V \setminus U\} \in \texttt{Par}(V)$ agrees with, i.e., is a feasible solution for, the demand graph K_{A_j,B_j} if and only if the following holds: $\Pi(u) \neq \Pi(v)$ whenever $u \in A_j$ and $v \in B_j$ or vice versa. Fix arbitrary $a_j \in A_j$ and $b_j \in B_j$. We associate with the demand graph K_{A_j,B_j} the formal polynomial P_j over the variables $\{\phi_v : v \in V\}$

$$P_j = \prod_{b \in B_j} \left(\phi_{a_j} - \phi_b\right) \cdot \prod_{a \in A_j \setminus \{a_j\}} \left(\phi_a - \phi_{b_j}\right).$$

Note that P_j is a polynomial of degree $\alpha + \beta - 1$. Given $U \subseteq V$, we may think of $\Pi = \{U, V \setminus U\}$ as a vector in $\{0,1\}^n$. We denote by $P_j(\Pi)$ the value of the polynomial P_j (over $\mathbb{F}_2$) when instantiated on Π.

Lemma 1. *A partition Π is feasible for the demand graph K_{A_j,B_j} if and only if $P_j(\Pi) \neq 0$*

Proof. Suppose Π is feasible for the demand graph K_{A_j,B_j}. So $\Pi(u) \neq \Pi(v)$ if $u \in A_j, v \in B_j$ or vice versa. Since every term of P_j contains one variable from each of A_j and B_j, it follows that $P_j(\Pi) \neq 0$.

Conversely, assume $P_j(\Pi) \neq 0$. Let $u \in A_j$. Since $\Pi(u) \neq \Pi(b_j)$ and $\Pi(b_j) \neq \Pi(a_j)$ it follows that $\Pi(u) = \Pi(a_j)$. Similarly for every $v \in B_j$, $\Pi(v) = \Pi(b_j)$. Therefore, it follows that $\Pi(u) \neq \Pi(v)$ whenever $u \in A_j$ and $v \in B_j$ or vice versa, i.e., Π is feasible for K_{A_j,B_j}. $\square$

Next we show that the polynomials corresponding to demand graphs with distinct values under `mincut` are linearly independent.

Lemma 2. *Reorder the demand graphs such that $\texttt{mincut}(K_{A_1,B_1}) < \ldots < \texttt{mincut}(K_{A_q,B_q})$. Then the polynomials $P_1, \ldots, P_q$ are linearly independent.*

Proof. Let $\Pi_1, \ldots, \Pi_q$ be the optimal partitions for the instances corresponding to the demand graphs $K_{A_1,B_1}, \ldots, K_{A_q,B_q}$ respectively, i.e., for each $i \in [q]$ we have that $\texttt{mincut}(K_{A_i,B_i}) = \texttt{Cut}(\Pi_i)$. Since $\texttt{mincut}(K_{A_i,B_i}) < \texttt{mincut}(K_{A_j,B_j})$ whenever $i < j$, it follows that Π_i is not feasible for the demand graph K_{A_j,B_j} for all $i < j$.

Suppose that the polynomials $P_1, P_2, \ldots, P_q$ are not linearly independent. Then there exist constants $\lambda_1, \ldots, \lambda_q \in \mathbb{R}$ which are not all zero such that $P = \sum_{j\in[q]} \lambda_j P_j$ is the zero polynomial. We will now show that each of the constants $\lambda_1, \lambda_2, \ldots, \lambda_q$ is zero, leading to a contradiction. Instantiate P on Π_1. Recall that Π_1 is not feasible for any K_{A_i,B_i} with $i \geq 2$. Therefore, by Lemma 1, we have that $P_i(\Pi_1) = 0$ for all $i \geq 2$. Therefore $\lambda_1 P_1(\Pi_1) = 0$. Since Π_1 is an (optimal) feasible partition for instance corresponding to K_{A_1,B_1}, applying Lemma 1 we get that $P_1(\Pi_1) \neq 0$. This implies $\lambda_1 = 0$. Hence, we have $P = \sum_{2\leq j\leq q} \lambda_j P_j$ is the zero polynomial. Now instantiate P on Π_2 to obtain $\lambda_2 = 0$ via a similar argument as above. In the last step, we will get that $\lambda_{q-1} = 0$ and hence $P = \lambda_q P_q$ is the zero polynomial. Instantiating on Π_q gives $0 = P(\Pi_q) = \lambda_q P_q(\Pi_q)$. Since Π_q is (optimal) feasible partition for the demand graph K_{A_q,B_q} it follows that $P_q(\Pi_q) \neq 0$, and hence $\lambda_q = 0$. □

Note that each of the polynomials $P_1, P_2, \ldots, P_q$ is contained in the vector space of polynomials with n variables and degree $\leq \alpha + \beta - 1$. This vector space is spanned by $\{\prod_{v\in V} \phi_v^{r_v} : \sum_{v\in V} r_v \leq \alpha + \beta - 1\}$ and therefore is of dimension $\binom{n+(\alpha+\beta-1)}{\alpha+\beta-1} = O_{\alpha,\beta}(n^{\alpha+\beta-1})$. From Lemma 2 and the fact that size of any set of linearly independent elements is at most the size of a basis, it follows that $\Big|\{\texttt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}\Big| = O_{\alpha,\beta}(n^{\alpha+\beta-1})$.

2.2 Proof via Matrices

We shall prove the (slightly stronger) bound that $|\{\texttt{mincut}(K_{A,B}) : |A| \leq \alpha, |B| \leq \beta\}| = O_{\alpha,\beta}(n^{\alpha+\beta-1})$. Let $\texttt{Par}_2(V) \subseteq \texttt{Par}(V)$ be the set of partitions of V into exactly two parts. Let $\mathcal{Q} := \{(A, B) : |A| \leq \alpha, \quad |B| \leq \beta\}$. Consider the matrix $\mathcal{M}$ over $\mathbb{F}_2$ with $|\mathcal{Q}|$ rows (one for each element from $\mathcal{Q}$) and $|\texttt{Par}_2(V)| = 2^n$ columns (one for each partition Π of V into two parts). We now define the entries of $\mathcal{M}$. Given $(A, B) \in \mathcal{Q}$ and $\Pi \in \texttt{Par}_2(V)$, we set $\mathcal{M}_{(A,B),\Pi} = 1$ if and only if the partition $\Pi \in \texttt{Par}_2(V)$ agrees with the demand graph $K_{A,B}$, which is equivalent to saying that $\Pi(u) \neq \Pi(v)$ whenever $u \in A$ and $v \in B$ or vice versa. Fix a vertex $v_0 \in V$, and consider the set $\mathcal{R} := \{(A, B) \in \mathcal{Q} : v_0 \in A \cup B\}$.

Proposition 1. *Over $\mathbb{F}_2$, the row space of $\mathcal{M}$ is spanned by the rows corresponding to elements from $\mathcal{R}$.*

Proof. Consider $(A, B) \in \mathcal{Q}$ and $\Pi \in \mathtt{Par}_2(V)$. If $v_0 \in A \cup B$ then $(A, B) \in \mathcal{R}$. Henceforth we assume that $v_0 \notin A \cup B$. Let

$$L(\Pi) := \mathcal{M}_{(A,B),\Pi} + \sum_{A' \subset A} \mathcal{M}_{(v_0 \cup A',B),\Pi} + \sum_{B' \subset B} \mathcal{M}_{(A,B' \cup v_0),\Pi} \ ,$$

where addition is over $\mathbb{F}_2$. Note that $(v_0 \cup A', B), (A, B' \cup v_0) \in \mathcal{R}$ for every $A' \subset A$ and $B' \subset B$, and therefore it is enough to show that $L(\Pi) \equiv 0 \pmod 2$.

Assume first that $\mathcal{M}_{(A,B),\Pi} = 1$, i.e. Π agrees with the demand graph $K_{A,B}$. Without loss of generality assume that $\Pi(v_0) = \Pi(a)$ for some $a \in A$. Then we have $\mathcal{M}_{(v_0 \cup A',B),\Pi} = 1$ for all $A' \subset A$, and $\mathcal{M}_{(A,v_0 \cup B'),\Pi} = 0$ for all $B' \subset B$. So, $L(\Pi) = 1 + (2^{|A|} - 1) \equiv 0 \pmod 2$.

Otherwise, we have $\mathcal{M}_{(A,B),\Pi} = 0$. If for every $v \in A \cup B$ it holds that $\Pi(v) \neq \Pi(v_0)$ then

$$L(\Pi) = 0 + \mathcal{M}_{(v_0,B),\Pi} + \mathcal{M}_{(A,v_0),\Pi} = 1 + 1 \equiv 0 \pmod 2 \ .$$

Hence suppose that there exists $v \in A \cup B$ such that $\Pi(v) = \Pi(v_0)$. Without loss of generality, assume $v \in A$. Then $\mathcal{M}_{(A,B'+v_0),\Pi} = 0$ for all $B' \subset B$. Note that if $A_1, A_2 \subset A$ satisfy $\mathcal{M}_{(v_0 \cup A_1,B),\Pi} = 1 = \mathcal{M}_{(v_0 \cup A_2,B),\Pi}$, then $\mathcal{M}_{(v_0 \cup A_1 \cup A_2,B),\Pi} = 1$. Hence there is an inclusion-wise maximal set $A^* \subseteq A$ such that $\mathcal{M}_{(v_0 \cup A^*,B),\Pi} = 1$. Since $\mathcal{M}_{(A,B),\Pi} = 0$, we conclude that $A^* \subset A$. If $A^* = \emptyset$, then since $\Pi(v) = \Pi(v_0)$, we conclude that there exists $v' \in B$ such that $\Pi(v') = \Pi(v_0)$. Then $\mathcal{M}_{(A'+v_0,B),\Pi} = 0$ for all $A' \subset A$, and $L(\Pi) = 0$. Otherwise, $|A^*| \geq 1$, and therefore

$$L(\Pi) = \mathcal{M}_{(A,B),\Pi} + \sum_{A' \subset A} \mathcal{M}_{(v_0 \cup A',B),\Pi} = \sum_{A' \subseteq A^*} \mathcal{M}_{(v_0 \cup A',B),\Pi} = 2^{|A^*|} \equiv 0 \bmod (2)$$

□

An argument similar to Lemma 2 shows that rows corresponding to demand graphs with distinct values under `mincut` are linearly independent. Hence, we have $\left| \{\mathtt{mincut}(K_{A,B}) : |A| \leq \alpha, |B| \leq \beta\} \right| \leq \mathtt{rank}(\mathcal{M}) \leq |\mathcal{R}|$, where the last inequality follows since $\mathcal{R}$ spans the row space of $\mathcal{M}$. We now obtain the final bound

$$\begin{aligned} |\mathcal{R}| &= \sum_{i \leq \alpha-1, j \leq \beta} \binom{n-1}{i} \cdot \binom{n-i-1}{j} + \sum_{j \leq \beta-1, i \leq \alpha} \binom{n-1}{j} \cdot \binom{n-j-1}{i} \\ &= \sum_{i \leq \alpha-1, j \leq \beta} O_{i,j}(n^{i+j}) + \sum_{j \leq \beta-1, i \leq \alpha} O_{i,j}(n^{i+j}) \\ &= O_{\alpha,\beta}(n^{\alpha+\beta-1}) \end{aligned}$$

2.3 Lower Bound on Number of Distinct Cuts for (α, β)-Group-Cut

We now turn to prove that the bound given in Theorem 2 is existentially tight. To this end, we construct an infinite family $G_n^{\alpha,\beta}$ of graphs satisfying $|\{\mathtt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}| \geq \Omega_{\alpha,\beta}(n^{\alpha+\beta-1})$.

Let $n, \alpha, \beta \in \mathbb{N}$ be such that n is odd, and both α and $\beta - 1$ divide $(n-3)/2$. We define a graph $G_n^{\alpha,\beta}$ on n vertices as follows. $G_n^{\alpha,\beta}$ is composed of two graphs that share a common vertex H_n^α and J_n^β defined below.

– H_n^α has $(n+1)/2$ vertices, and is given by α parallel paths $P_1, \ldots, P_\alpha$ between two designated vertices s, t, each path having $(n-3)/2\alpha$ internal vertices. The edge weights are given by distinct powers of 2, monotonically decreasing from s to t. All edges in H_n^α incident on t have ∞ weight (see Figure 1).
– J_n^β has $(n+1)/2$ vertices, and is given by $(\beta-1)$ parallel paths $Q_1, \ldots, Q_{\beta-1}$, between t and a designated vertex u, each having $(n-3)/2(\beta-1)$ internal vertices. As in H_n^α, edge weights are given by distinct powers of 2, monotonically decreasing from t to u, and all of which are strictly smaller than the weights of H_n^α. All edges in J_n^β incident on u have ∞ weight.

The following proposition implies the desired lower bound.

Proposition 2. *$|\{\mathtt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}| \geq \Omega_{\alpha,\beta}(n^{\alpha+\beta-1})$.*

Proof. Pick one internal vertex from each P_i for $i \in [\alpha]$ to form A. Similarly for $\beta - 1$ elements in B, we pick one internal vertex from each Q_j for $j \in [\beta]$. In addition, $s \in B$ (as demonstrated in Figure 1). We claim that every such choice of A, B gives a distinct value for the minimum (A, B)-cut.

Indeed, for $i \in [\alpha]$ let a_i be the unique element in $A \cap P_i$. In order to separate A from B, we need to separate a_i from s. This implies that at least one edge on the segment of P_i between s and a_i has to be in the cut. By monotonicity of weights and minimality of the cut, this must be the edge incident to a_i. Similarly, for every $b \in B \setminus \{s\}$, the left edge incident to b must be cut. It is easy to see (as demonstrated in Figure 1) that this set of edges also separates A and B.

By the choice of weights, each such cut has a unique value, and therefore $|\{\mathtt{mincut}(K_{A,B}) : |A| = \alpha, |B| = \beta\}| \geq ((n-3)/2\alpha)^\alpha((n-3)/2(\beta-1))^{\beta-1} = \Omega_{\alpha,\beta}(n^{\alpha+\beta-1})$. □

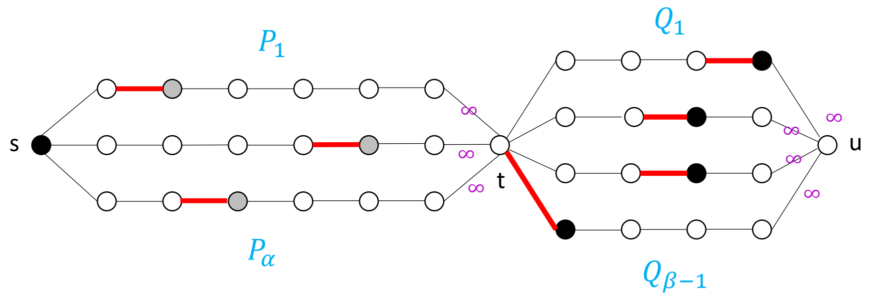

Fig. 1. The graph $G_n^{\alpha,\beta}$ used in the lower bound of Section 2.3. The left part of the graph is H_n^α, consisting of α parallel s-t paths. The right part of the graph is J_n^β, consisting of $(\beta-1)$ parallel t-u paths. The gray vertices are in A, and the black ones are in B. The red edges represent the minimum cut for this choice of A and B.

3 Multiway-Cut: The Case of Clique Demands

Our tight bounds for Multiway-Cut are described in Theorem 3. First we show that for every graph $G = (V, E, w)$ we have $|\{\texttt{mincut}(K_S) : |S| = k\}| = O_k(n^{k-1})$. The proof technique for this upper bound is quite similar to that from Section 2.2. We also show that this bound is tight for paths (with specially chosen edge-weights). Hence, the redundancy factor is $\Omega_k(n)$, since $|\{K_S : |S| = k\}| = \binom{n}{k} = \Theta_k(n^k)$. We refer to the full version [6] for the technical details.

4 Multicut: The Case of Demands with Fixed Number of Edges

Our tight bounds for Multicut are described in Theorem 4. First we show that $|\{\texttt{mincut}(D) : D \subseteq V \times V, |D| = k\}| = O_k(n^k)$. The proof technique for this upper bound is quite similar to that from Section 2.1. We also show that this bound is tight for graphs which are perfect matchings (with specially chosen edge-weights). Since $|\{D : D \subseteq V \times V, |D| = k\}| = \binom{\binom{n}{2}}{k} = \Theta_k(n^{2k})$, it follows that the redundancy factor is $\Omega_k(n^k)$. We refer to the full version [6] for the technical details.

5 Evaluation Schemes: Constructing Succinct Data Structures

Gomory and Hu [9] showed that for every undirected edge-weighted graph $G = (V, E, w)$ there is a tree $\mathcal{T} = (V, E', w')$ that represents the minimum st-cuts exactly both in terms of the *cut-values* and in terms of their *vertex-partitions*. The common terminology, probably due to Benczúr [2], is to say that $\mathcal{T}$ is cut-equivalent to G. A tree $\mathcal{T}$ is flow-equivalent to G if it satisfies only the first property.[1]

Flow-equivalent and cut-equivalent trees can be viewed more generally as succinct data structures that support certain queries, either just for the value of an optimal cut, or also for its vertex-partition. Motivated by this view, we define two data structures, which we call a *flow-evaluation scheme* and a *cut-evaluation scheme* (analogously to the common terminology in the literature). These schemes are arbitrary data structures (e.g., need not form a tree), and address the terminals version of a certain cut problem. Both of these schemes, first preprocess an input that consists of a graph $G = (V, E, w)$, a terminals set $T \subset V$, and a collection of demand graphs $\mathcal{D}$. The preprocessed data can then be used (without further access to G) to answer a cut query given by a demand

[1] A tree $\mathcal{T}$ is flow-equivalent to G if for every $s, t \in V$ the minimum st-cut value in $\mathcal{T}$ is exactly the same as in G. We say $\mathcal{T}$ is cut-equivalent to G if, in addition, every vertex partition that attains a minimum st-cut in $\mathcal{T}$, also attains a minimum st-cut in G.

graph $D \in \mathcal{D}$. The answer of a *flow-evaluation scheme* is the corresponding minimum cut-value $\mathtt{mincut}(D)$. A *cut-evaluation scheme* will also give a vertex-partition that attains this cut-value $\mathtt{mincut}(D)$.

A natural goal is to provide succinct constructions and lower bounds for the storage and query time of flow-evaluation schemes and cut-equivalent schemes, for the three cut problems studied in this paper, viz. Group-Cut, Multiway-Cut and Multicut. In order to analyze the storage size (in terms of bits) of such data structures, we henceforth assume that all edge weights are integers. Our bounds on the number of distinct cut values naturally lead to the following construction of cut-evaluation schemes for the aforementioned three problems. We state below the result for Group-Cut (proof deferred to the full version [6]); similar results also hold for the Multiway-Cut and Multicut problems.

Theorem 5. *For every $\alpha, \beta \in \mathbb{N}$ there is a cut-evaluation scheme such that for every graph $G = (V, E, w)$, and a set of terminals $T \subseteq V$, the scheme uses a storage of $O_{\alpha,\beta}(|T|^{\alpha+\beta-1} \cdot (|T| + \log W))$ bits, where $W = \sum_{e \in E} w(e)$, to answer every (α, β)-Group-Cut query in time $O_{\alpha,\beta}(|T|^{\alpha+\beta-1})$.*

The result of Theorem 5 is especially meaningful for the case where $|T|$ is much smaller than n. For large $|T|$, say $T = V$, the graph G itself serves as a cut-evaluation scheme of size $O(n^2 \log W)$.

We do not know whether the upper bound in Theorem 5 is tight, and proving lower bounds for the storage size of such schemes is left as an interesting open question. However, for $(2, 1)$-Group-Cut with $V = T$ and edge weights bounded by $n^{O(1)}$, we can prove that simply storing the graph G using $O(n^2 \log n)$ bits is essentially optimal, even for the weaker notion of flow-evaluation schemes.

5.1 Lower Bound on Flow-Evaluation Schemes for $(2, 1)$-Group-Cut

Using an information-theoretic argument, we can show the following lower bound (proof deferred to [6]) on the storage required by any flow-evaluation scheme for $(2, 1)$-Group-Cut. We remark that similar arguments give a lower bound of $\Omega(n^3 \log n)$ by allowing weights which are exponential in n^3.

Theorem 6. *For every $n \geq 3$, a flow-evaluation scheme for $(2, 1)$-Group-Cut on graphs with n terminals (in which $T = V$) and with edge-weights bounded by a polynomial in n requires storage of $\Omega(n^2 \log n)$ bits.*

6 Future Directions

A natural direction for future work is to construct better data structures for the problems discussed in this paper. Our tight bounds on the number of distinct cut values (redundancy factor) yield straightforward schemes with improved storage requirement, as described in Section 5. But one may potentially improve these schemes in several respects. First, our storage requirement is a factor $|T|$ larger than the number of distinct cut values. The latter number could possibly be the "right bound", and it is important to prove it is a lower bound for required storage; we only proved this for $(2,1)$-Group-Cut. Second, it is desirable to achieve fast query time, say sublinear in $|T|$ or perhaps even constant. Third, one may ask for a distributed version of the data structure (i.e., a labeling scheme) that can report the same cut values; this would extend the known results [12] for minimum *st*-cuts. All these improvements require better understanding of the structure of the optimal partitions (those that attain minimum cut values). Such structure is known for minimum *st*-cuts, where the Gomory-Hu tree essentially shows the existence of a family of minimum *st*-cuts, one for each $s, t \in V$, which is laminar.

Another very interesting question is to explore approximation to the minimum cut, i.e., versions of the above problems where we only seek for each instance a cut within a small factor of the optimal. For instance, the cut values of (α, β)-Group-Cut can be easily approximated within factor $\alpha \cdot \beta$ using Gomory-Hu trees, which requires storage that is linear in $|T|$, much below the aforementioned "right bound" of $|T|^{\alpha+\beta-1}$.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows - theory, algorithms and applications.* Prentice Hall, 1993.

[2] A. A. Benczúr. Counterexamples for directed and node capacitated cut-trees. *SIAM J. Comput.*, 24(3):505–510, 1995.

[3] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi. An $\tilde{O}(mn)$ Gomory-Hu tree construction algorithm for unweighted graphs. In *STOC 2007*, pages 605–614.

[4] S. Chaudhuri, K. V. Subrahmanyam, F. Wagner, and C. D. Zaroliagis. Computing mimicking networks. *Algorithmica*, 26:31–49, 2000.

[5] C. Cheng and T. Hu. Ancestor tree for arbitrary multi-terminal cut functions. *Annals of Operations Research*, 33(3):199–213, 1991.

[6] R. Chitnis, L. Kamma, and R. Krauthgamer. Tight bounds for Gomory-Hu-like cut counting. *CoRR*, abs/1511.08647, 2015.

[7] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial optimization.* John Wiley & Sons Inc., New York, 1998.

[8] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8(3):399–404, 1956.

[9] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9:551–570, 1961.

[10] D. Gusfield. Very simple methods for all pairs network flow analysis. *SIAM J. Comput.*, 19(1):143–155, 1990.

[11] T. Hagerup, J. Katajainen, N. Nishimura, and P. Ragde. Characterizing multiterminal flow networks and computing flows in networks of small treewidth. *J. Comput. Syst. Sci.*, 57(3):366–375, 1998.

[12] M. Katz, N. A. Katz, A. Korman, and D. Peleg. Labeling schemes for flow and connectivity. *SIAM J. Comput.*, 34(1):23–40, 2005.

[13] A. Khan and P. Raghavendra. On mimicking networks representing minimum terminal cuts. *Inf. Process. Lett.*, 114(7):365–371, 2014.

[14] R. Krauthgamer and I. Rika. Mimicking networks and succinct representations of terminal cuts. In *SODA 2013*, pages 1789–1799. SIAM, 2013.