Approximating Requirement Cut via a Configuration LP

Roy Schwartz
Department of Computer Science, Technion, Haifa, Israel
schwartz@cs.technion.ac.il

Yotam Sharoni
Department of Computer Science, Technion, Haifa, Israel
yotamsh@cs.technion.ac.il

Abstract

We consider the Requirement Cut problem, where given an undirected graph $G = (V, E)$ equipped with non-negative edge weights $c : E \to \mathbb{R}_+$, and g groups of vertices $X_1, \ldots, X_g \subseteq V$ each equipped with a requirement r_i, the goal is to find a collection of edges $F \subseteq E$, with total minimum weight, such that once F is removed from G in the resulting graph every X_i is broken into at least r_i connected components. Requirement Cut captures multiple classic cut problems in graphs, e.g., Multicut, Multiway Cut, Min k-Cut, Steiner k-Cut, Steiner Multicut, and Multi-Multiway Cut. Nagarajan and Ravi [Algoritmica'10] presented an approximation of $O(\log n \log R)$ for the problem, which was subsequently improved to $O(\log g \log k)$ by Gupta, Nagarajan and Ravi [Operations Research Letters'10] (here $R = \sum_{i=1}^{g} r_i$ and $k = |\cup_{i=1}^{g} X_i|$). We present an approximation of $O(X \log R \sqrt{\log k} \log \log k)$ for Requirement Cut (here $X = \max_{i=1,\ldots,g}\{|X_i|\}$). Our approximation in general is incomparable to the above mentioned previous results, however when all groups are not too large, i.e., $X = o\left((\sqrt{\log k} \log g)/(\log R \log \log k)\right)$, it is better. Our algorithm is based on a new configuration linear programming relaxation for the problem, which is accompanied by a remarkably simple randomized rounding procedure.

2012 ACM Subject Classification Theory of computation → Facility location and clustering; Mathematics of computing → Combinatorial optimization

Keywords and phrases Approximation, Requirement Cut, Sparsest Cut, Metric Embedding

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2020.53

Category APPROX

Funding *Roy Schwartz*: Research is supported by ISF grant 1336/16.

1 Introduction

We consider the Requirement Cut problem (RC), where we are given an undirected graph $G = (V, E)$ equipped with non-negative edge weights $c : E \to \mathbb{R}_+$ and g groups of vertices $X_1, \ldots, X_g \subseteq V$. Every group X_i is associated with a requirement r_i, where $2 \leq r_i \leq |X_i|$. The goal is to find a collection of edges $F \subseteq E$, with total minimum weight, such that once F is removed from G in the resulting graph $G_F = (V, E \backslash F)$ every X_i is broken into at least r_i connected components. To simplify presentation we use the notation $\text{Cut}(G, X_i)$ to denote the number of connected components of G that contain at least one vertex of X. The above requirement implies that $\text{Cut}(G_F, X_i) \geq r_i$, $\forall i = 1, \ldots, g$.

Requirement Cut captures multiple classic cut problems in graphs, e.g., Multicut [11, 15], Multiway Cut [4, 5, 6, 9, 13, 22], Min k-Cut [19, 20, 21], Steiner k-Cut [8], Steiner Multicut [14], and Multi-Multiway Cut [3]. To simplify presentation of known results for (RC), we denote by R the sum of requirements, i.e., $R = \sum_{i=1}^{g} r_i$, by X the largest group, i.e., $X = \max_{i=1,\ldots,g}\{|X_i|\}$, and by k the number of vertices in groups, i.e., $k = |\cup_{i=1}^{g} X_i|$. Nagarajan and Ravi [18] were the first to consider (RC), presenting

approximations of $O(\log n \log R)$ and $O(\log R)$ for general graphs and trees, respectively.[1] They also proved that there is a hardness of approximation of $\Omega(\log g)$ when the graph is a tree. Subsequently, Gupta et al. [12] presented improved approximations of $O(\log k \log g)$ and $O(\log g)$ for general graphs and trees, respectively. Thus, providing a tight approximation when the graph is a tree. In both these works, the main approach to solving (RC) in general graphs is first to formulate a linear programming relaxation which finds a suitable spreading metric over the vertices of V, then transform the metric into a tree metric, and finally round the solution on the tree. In [18], an additional greedy combinatorial approach is presented. In this approach the algorithm repeatedly finds cuts that (approximately) minimize the ratio between their cost and the number of groups they separate. This algorithm also yields an approximation of $O(\log n \log R)$ for (RC).

1.1 Our Result

The following theorem summarizes our main result for the (RC) problem.

▶ **Theorem 1.** *There is a randomized polynomial time algorithm for* Requirement Cut *that achieves an approximation of* $O\left(X \log R \sqrt{\log k} \log\log k\right)$.

We note that our approximation is incomparable to the current state of the art [12]. However, when all groups are not too large, i.e., $X = o\left((\sqrt{\log k} \log g)/(\log R \log\log k)\right)$, our approximation is better. Our algorithm in fact provides an approximation guarantee of $O\left(DX \log R\right)$, where D is the distortion of embedding a metric of negative type into ℓ_1 (refer to Section 2 for an exact definition). The guarantee of Theorem 1 follows by employing the embedding of Arora et al. [2].

1.2 Our Techniques

Our approach for obtaining the result is based on three ingredients, on which we currently elaborate.

The first is a new configuration linear programming relaxation of exponential size, in which each cut S is associated with a variable λ_S. In this relaxation the goal is to assign a fractional value λ_S to each cut S while satisfying two requirements: (1) for each group X_i the total fractional value of cuts that separate it is high enough, i.e., X_i is broken into enough pieces; and (2) for each vertex u the total fractional value of cuts containing u is at most one, i.e., the connected components in the output are disjoint. This configuration LP differs from the relaxation used in [12, 18], which imposes a metric over the vertices of the graph while ensuring that for every X_i the total length of every tree spanning X_i is high enough.

The second ingredient is a remarkably simple randomized rounding procedure of the configuration LP. When considering classic randomized rounding applied to our setting, each cut S, independently of other cuts, chooses all edges that cross it with a probability of λ_S. This straightforward use of randomized rounding is problematic, as it might separate a group into very few pieces, even though the total fraction of cuts that separate the group is quite high. To intuitively exemplify this, it is enough to note that choosing ℓ cuts that separate some group X_i might result in breaking X_i into only $O(\log \ell)$ pieces due to non-trivial

[1] In [18], in contrast to what is cited above, the claimed approximation does not depend on R but rather on the maximum requirement multiplied by g. However, the algorithms of [18] provide the above claimed approximation guarantee.

overlaps between the cuts. To overcome this difficulty, we note that cuts S with a high fractional value λ_S are disjoint. Hence, for these cuts the above difficulty does not occur. Unfortunately, cuts S with a high fractional value λ_S might be few and we are not guaranteed that choosing all of them breaks each X_i into enough pieces. To remedy this, we introduce a remarkably simple analysis for the classic randomized rounding procedure that assumes all cuts S have a small fractional value λ_S. In our analysis we show that for each cut S, if all other cuts have a small fractional value, then with a high enough probability S is able to break at least one additional piece from X_i on its own. This enables us to lower bound the number of pieces each X_i is broken into.

The third and last ingredient relates to solving the configuration LP. Since the configuration LP has an exponential number of variables, we need to present an (approximate) dual separation oracle for it. It turns out that the dual separation oracle for the configuration LP is a node weighted variant of Sparsest Cut with general demands where demands are given over groups, as opposed to pairs. Our approximation algorithm for the dual separation oracle, similarly to classic algorithms for Sparsest Cut, finds a suitable metric of negative type, embeds it into ℓ_1, and chooses the best cut S among all cuts in the decomposition of the ℓ_1 metric into a non-negative combination of cut metrics. The only caveat in this approach is that, unlike Sparsest Cut, our problem is not symmetric, i.e., S and $\overline{S}$ have different objective value. To overcome this, we break symmetry and introduce an artificial point to the metric space, while ensuring that all cuts in the decomposition of the ℓ_1 metric do not contain this artificial point.

We note that our dual separation oracle captures the Steiner Ratio problem (SR) [14], used in the greedy step of the combinatorial approach of [18] for (RC). Thus, it can be proved that our approximation guarantee of $O(X\sqrt{\log k}\log\log k)$ for the dual separation oracle (Theorem 8) also extends to (SR). Incorporating this into the analysis of [18] yields an overall approximation of $O\left(X\log R\sqrt{\log k}\log\log k\right)$, matching the result of Theorem 1 in yet another way.

1.3 Related Work

The literature on approximating cut problems and metric embedding is vast, we mention here only some of the most relevant work. Let us start by surveying problems captured by (RC). Multicut, a special case of (RC) where each group X_i is of size two and its requirement r_i also equals two, admits an approximation of $O(\log C\log^2 k)$ (where C is the total weight of edges in the graph) [15] and $O(\log k)$ [11]. Multiway Cut, another special case of (RC) where there is a single group X of size k and its requirement also equals k, exhibits a long sequence of works [4, 5, 6, 9, 13, 22] which culminates in an approximation of 1.2965 [22]. Min k-Cut, a special case of (RC) where there is a single group X that spans all of V and its requirement equals k, admits a simple combinatorial approximation of $2(1-1/k)$ [19, 20, 21]. Steiner k-Cut is similar to Min k-Cut, with the difference that X does not necessarily equal all of V, also admits a similar approximation of $2(1-1/k)$ [8]. Steiner Multicut is similar to Multicut, with the difference that each group X_i might be larger than two, though its requirement r_i still remains two. This problem admits an approximation of $O(\log^3(gX))$ [14], which was subsequently improved to $O(\log g\log n)$ [18] and to $O(\log g\log k)$ [12]. Multi-Multiway Cut is similar to Multiway Cut, with the difference that there are multiple groups (though the requirement of each group equals its size). This problem admits an approximation of $O(\log g)$ [3].

Metric embedding also play a central role in our work, specifically embedding metrics of negative type into ℓ_1. Chawla et al. [7] presented an embedding with a distortion of $O(\log^{3/4} n)$, which was later improved to $O(\sqrt{\log n}\log\log n)$ by Arora et al. [1, 2]. The above

works are based on the seminal result of Arora et al. for Sparsest Cut, which obtained an approximation of $O(\sqrt{\log n})$, improving upon the $O(\log n)$ guarantee of Leighton and Rao [16].

Paper Organization

Section 2 contains needed preliminaries regarding metric spaces. Section 3 presents our new configuration LP for (RC). The rounding algorithm appears in Section 4, and Section 5 is dedicated to approximately solving the configuration LP. All missing proofs appear in the appendix.

2 Preliminaries

A metric space (V, d) embeds into ℓ_1 with distortion D if there exists an embedding f satisfying: $d(a,b) \leq ||f(a) - f(b)||_1 \leq D \cdot d(a,b)$, $\forall a, b \in V$. Moreover, a metric space (V, d) is of negative-type if $(V, \sqrt{d})$ is isometric to a subset of Euclidean space. Arora et al. [1, 2] proved that $D = O(\sqrt{\log n} \log\log n)$ for metrics of negative type, when $|V| = n$. We can exploit a slightly improved guarantee (see Corollary 5.1 in [2]), which we rephrase here for simplicity of presentation.[2]

▶ **Theorem 2** (Corollary 5.1 in [2]). *Given a metric space (V, d) of negative type and $U \subseteq V$ of size k, there exists an embedding f into ℓ_1 that satisfies: (1) $||f(a) - f(b)||_1 \leq D \cdot d(a,b)$, $\forall a, b \in V$; and (2) $||f(a) - f(b)||_1 \geq d(a,b)$, $\forall a, b \in U$. In the above $D = O(\sqrt{\log k} \log\log k)$. Furthermore, f can be computed in polynomial time.*

We note that in the above theorem, expansion is upper bounded for all pairs of points in the metric space, whereas contraction is lower bounded only for pairs of points in the given subset. This enables the improvement in the value of D from $O(\sqrt{\log n} \log\log n)$ to $O(\sqrt{\log k} \log\log k)$.

3 Configuration LP

We consider the following new linear relaxation for the (RC) problem. In this relaxation each possible cut $S \subseteq V$ is associated with a variable λ_S. We denote by $\delta_G(S)$ the total weight of edges crossing the cut S defines in the graph G, and let $\mathcal{C}(T) = \{S : 0 < |S \cap T| < |T|\}$ be the collection of cuts S that separate T, i.e., S contains at least one vertex of T but does not contain T as a whole.

$$
\begin{aligned}
\text{(LP)} \quad \min \quad & \sum_{S \in V} \frac{1}{2} \delta_G(S) \cdot \lambda_S & & \\
\text{s.t.} \quad & \sum_{S \in \mathcal{C}(X_i)} \lambda_S \geq r_i & \forall i = 1, \ldots g & \quad (1) \\
& \sum_{S : u \in S} \lambda_S \leq 1 & \forall u \in V & \quad (2) \\
& \lambda_S \geq 0 & \forall S \subseteq V &
\end{aligned}
$$

In the above relaxation, Constraint (1) ensures that X_i is broken into at least r_i pieces, and Constraint (2) ensures that each vertex belongs to at most a single cut.

[2] The cited theorem of [2] is originally given for ℓ_2 and not ℓ_1. However, standard arguments imply it also applies to ℓ_1, see, e.g., [2], for further details.

For our rounding algorithm to work we actually need to iteratively resolve (LP) for subgraphs of G. To this end, let us formally define the relaxation when considering $G_F = (V, E \setminus F)$, for some $F \subseteq E$, which we denote by (LP$_F$). Moreover, we denote by $Y_{i,1}, \ldots, Y_{i,\ell_i}$ the ℓ_i pieces X_i is broken into in G_F. Formally, if G_F is broken into ℓ connected components $C_1, \ldots, C_\ell$, which form a partition of V, then every $Y_{i,j}$ is obtained by $X_i \cap C_j$ and discarding the result if X_i and C_j are disjoint (hence $\ell_i \leq \ell$). Additionally, let us denote by r'_i the residual requirement of X_i in G_F, i.e., $r'_i = \max\{0, r_i - \ell_i\}$ (we can assume that groups X_i for which the residual requirement reached zero are removed from the instance).

$$
\begin{aligned}
(\mathrm{LP}_F) \quad \min \quad & \sum_{S \in V} \frac{1}{2} \delta_{G_F}(S) \cdot \lambda_S & & \\
\text{s.t.} \quad & \sum_{j=1}^{\ell_i} \sum_{S \in C(Y_{i,j})} \lambda_S \geq r'_i & \forall i = 1, \ldots g & \quad (3) \\
& \sum_{S : u \in S} \lambda_S \leq 1 & \forall u \in V & \quad (4) \\
& \lambda_S \geq 0 & \forall S \subseteq V &
\end{aligned}
$$

Similarly to (LP), Constraint (3) ensures that X_i is broken to at least r'_i pieces, and Constraint (4) ensures that each vertex belongs to at most a single cut.

The following lemma proves that (LP$_F$), for every $F \subseteq E$, is a relaxation when considering G_F. We denote by $\mathrm{OPT}_{\mathrm{LP}_F}$ the value of an optimal solution to (LP$_F$) and by OPT the value of an optimal solution to the original instance.

▶ **Lemma 3.** *$OPT_{LP_F} \leq OPT$, for every $F \subseteq E$.*

Proof. Let F^* be an optimal solution to the problem with respect to the original instance, and denote by $\tilde{F} = F^* \setminus F$ the edges of F^* still remaining in G_F. Thus, $G_{\tilde{F} \cup F} = (V, E \setminus (\tilde{F} \cup F))$ denotes the graph resulting in the removal of F^* from G_F. Let $S^*_1, \ldots, S^*_\ell$ be the connected components in $G_{\tilde{F} \cup F}$. Define the following solution to (LP$_F$): $\lambda_{S^*_i} = 1$ for every $i = 1, \ldots, \ell$ (and $\lambda_S = 0$ for all other cuts S). We notice that $S^*_1, \ldots, S^*_\ell$ are disjoint cuts, hence Constraint (4) is satisfied for every $u \in V$. Since F^* is a feasible solution with respect to the original instance, i.e., X_i is broken to at least r_i pieces in G_{F^*} (or equivalently $\mathrm{Cut}(G_{F^*}, X_i) \geq r_i$), we can conclude that removing $\tilde{F}$ from G_F breaks X_i to at least the same number of pieces. Thus, since X_i is already broken into ℓ_i pieces in G_F: $Y_{i,1}, \ldots, Y_{i,\ell_i}$, we can conclude that $\sum_{j=1}^{\ell_i} \mathrm{Cut}(G_{\tilde{F} \cup F}, Y_{i,j}) \geq r'_i$. Therefore, Constraint (3) is also satisfied and the defined solution is feasible for (LP$_F$). Since $\delta_{G_F}(S) \leq \delta_G(S)$, for every $S \subseteq V$, we can conclude that the value of the solution is at most OPT. This finishes the proof. ◀

A main challenge is solving (LP$_F$) since it has an exponential number of variables. The following theorem summarizes our guarantee for solving (LP$_F$), and Section 5 is dedicated to its proof. The solution found is bicriteria as it violates the constraints and incurs some loss in the objective.

▶ **Theorem 4.** *For every $F \subseteq E$, there exists an efficient algorithm that finds a bicriteria solution $\{\lambda_S\}_{S \subseteq V} \subseteq \mathbb{R}_+$ to (LP$_F$) satisfying:*

1. $\sum_{S : u \in S} \lambda_S \leq \alpha$, $\forall u \in V$.
2. $\sum_{j=1}^{\ell_i} \sum_{S \in C(Y_{i,j})} \lambda_S \geq \beta \cdot r'_i$, $\forall i = 1, \ldots, g$.
3. $\sum_{S \subseteq V} \frac{1}{2} \delta_{G_F}(S) \lambda_S \leq \gamma \cdot OPT$.

In the above $\alpha = \gamma = O(XD)$ (where D is as in Theorem 2 when considering $U = \cup_{i=1}^{g} X_i \subseteq V$) and $\beta = 1$.

4 Rounding the Configuration LP

In this section we present our rounding algorithm and prove the main result, Theorem 1, given Theorem 4. Our rounding algorithm progresses in iterations, similar in spirit to the classic randomized rounding for the Set Cover problem (a similar method was employed in the context of (RC) by [18]). However, unlike the latter rounding method, we require a different handling of cuts S with a large λ_S fraction and cuts S with a small λ_S fraction.

We start by focusing on the algorithm for a single iteration, which given G_F, for some $F \subseteq E$, outputs a random collection $L \subseteq E \setminus F$ of edges such that: (1) if L is removed from G_F then the number of pieces every X_i is broken into increases additively in expectation, up to some scaling, by X_i's residual requirement r'_i; and (2) the expected cost of edges in L is not too large. We denote by $\{\tilde{\lambda}_S\}_{S\subseteq V}$ the solution $\{\lambda_S\}_{S\subseteq V}$ scaled by a factor of $1/\alpha$, where α is as in guarantee (1) of Theorem 4, *i.e.*, $\tilde{\lambda}_S = \lambda_S/\alpha$. The single iteration algorithm removes all edges crossing a collection of cuts chosen according to two different criterions to ensure that the number of pieces X_i is broken into increases as required. The first criterion consists of all cuts whose $\tilde{\lambda}_S$ value is sufficiently large. The second criterion consists of executing randomized rounding on all cuts whose $\tilde{\lambda}_S$ is sufficiently small. Algorithm 1 summarizes the single iteration rounding procedure (in the algorithm's description we denote by $\Gamma_G(S)$ the collection of edges in the graph G crossing the cut S), and Lemma 5 summarizes the guarantee of Algorithm 1.

Algorithm 1 Single Iteration.

Input: $G_F = (V, E\setminus F), \{\tilde{\lambda}_S\}_{S\subseteq V}$
Output: $L \subseteq E\setminus F$
$F_1 \leftarrow \{e \in E \setminus F : \exists S, \tilde{\lambda}_S \geq 2/3 \wedge e \in \Gamma_{G_F}(S)\}$
let $\{I_S\}_{S\subseteq V:\tilde{\lambda}_S<2/3}$ be independent random indicators where $\Pr[I_S = 1] = \tilde{\lambda}_S$
$F_2 \leftarrow \{e \in E \setminus F : \exists S \text{ s.t. } \tilde{\lambda}_S < 2/3 \wedge I_S = 1 \wedge e \in \Gamma_{G_F}(S)\}$
$L \leftarrow F_1 \cup F_2$
output L

▶ **Lemma 5.** *For every $F \subseteq E$ and $\{\lambda_S\}_{S\subseteq V}$ guaranteed in Theorem 4, executing Algorithm 1 on G_F with $\{\tilde{\lambda}_S\}_{S\subseteq V}$ results in $L \subseteq E \setminus F$ satisfying:*
1. $\mathbb{E}\left[Cut(G_{F\cup L}, X_i)\right] \geq Cut(G_F, X_i) + \Omega\left(\beta/\alpha\right) \cdot r'_i$, $\forall i = 1, ...g$.
2. $\mathbb{E}\left[\sum_{e\in L} c_e\right] \leq O\left(\gamma/\alpha\right) \cdot OPT$.

Proof. Let us start with requirement (1) above. Note that for every vertex $u \in V$, since $\{\lambda_S\}_{S\subseteq V}$ satisfies guarantee (1) of Theorem 4, we can conclude that: $\sum_{S:u\in S} \tilde{\lambda}_S \leq 1$. Thus, we can conclude that all cuts S whose $\tilde{\lambda}_S$ is large, i.e., $\tilde{\lambda}_S \geq 2/3$, are disjoint. Therefore, removing F_1 from G_F additively increases the number of pieces every X_i is broken into by:

$$\sum_{j=1}^{\ell_i} \left|\left\{S : S \in C(Y_{i,j}) \wedge \tilde{\lambda}_s \geq 2/3\right\}\right| \geq \sum_{j=1}^{\ell_i} \sum_{S:S\in C(Y_{i,j})\wedge\tilde{\lambda}_S\geq 2/3} \tilde{\lambda}_S,$$

where the above inequality follows from the fact that $\tilde{\lambda}_S \leq 1, \forall S \subseteq V$.

Let us now consider cuts $S \subseteq V$ whose $\tilde{\lambda}_S$ is small, i.e., $\tilde{\lambda}_S < 2/3$. We analyze the expected additive increase in the number of pieces X_i is broken into when removing F_2 from G_F. A cut S satisfying: $\tilde{\lambda}_S < 2/3$ and $S \in C(Y_{i,j})$, increases the number of pieces $Y_{i,j}$ is broken into by one if there exists a vertex $u \in Y_{i,j} \cap S$ such that S is the only cut having $I_S = 1$, among all cuts T with $\tilde{\lambda}_T < 2/3$ containing u. Let us denote this event by $A_{Y_{i,j},S,u}$. Thus,

$$\Pr\left[A_{Y_{i,j},S,u}\right] = \tilde{\lambda}_S \cdot \prod_{T:\tilde{\lambda}_T<2/3\wedge u\in T} \left(1 - \tilde{\lambda}_T\right).$$

We note that since $\sum_{T:u\in T}\tilde{\lambda}_T \leq 1$ (as previously mentioned in the proof), and the fact that we consider only cuts T satisfying $\tilde{\lambda}_T < 2/3$, the following holds: $\prod_{T:\tilde{\lambda}_T<2/3\wedge u\in T}(1-\tilde{\lambda}_T) = \Omega(1)$. Hence, $\Pr[A_{Y_{i,j},S,u}] \geq \Omega(\tilde{\lambda}_S)$. Therefore, removing F_2 from G_F additively increases the expected number of pieces every X_i is broken into by at least:

$$\sum_{j=1}^{\ell_i}\sum_{S:S\in C(Y_{i,j})\wedge\tilde{\lambda}_S<2/3}\Pr\left[\exists u\in Y_{i,j}\cap S \text{ s.t. } A_{Y_{i,j},S,u}\right] \geq \sum_{j=1}^{\ell_i}\sum_{S:S\in C(Y_{i,j})\wedge\tilde{\lambda}_S<2/3}\Omega(\tilde{\lambda}_S).$$

The above inequality follows since for every $Y_{i,j}$ and every $S\in C(Y_{i,j})$, satisfying $\tilde{\lambda}_S < 2/3$, one can choose an arbitrary vertex $u\in Y_{i,j}\cap S$ and apply the lower bound on the probability of the event $A_{Y_{i,j},S,u}$.

Recall that $\{\lambda_S\}_{S\subseteq V}$ satisfies guarantee (2) of Theorem 4, therefore we can conclude that $\forall i = 1,\ldots,g$:

$$\sum_{j=1}^{\ell_i}\sum_{S:S\in C(Y_{i,j})\wedge\tilde{\lambda}_S<2/3}\tilde{\lambda}_S \geq \frac{\beta}{2\alpha}r'_i \qquad \text{or} \qquad \sum_{j=1}^{\ell_i}\sum_{S:S\in C(Y_{i,j})\wedge\tilde{\lambda}_S\geq 2/3}\tilde{\lambda}_S \geq \frac{\beta}{2\alpha}r'_i.$$

Thus, removing all edges in $F_1\cup F_2$ from G_F additively increases the number of pieces every X_i is broken into, in expectation, by at least $\Omega\left(\beta/\alpha\right)r'_i$. This concludes the proof of requirement (1).

Let us focus on requirement (2). It is easy to note that the definition of F_1 implies that $\sum_{e\in F_1}c_e \leq 3/2\sum_{S:\tilde{\lambda}_S\geq 2/3}\delta_{G_F}(S)\tilde{\lambda}_S$. Additionally, it is easy to note from the definition of F_2 that $\mathbb{E}[\sum_{e\in F_2}c_e] \leq \sum_{S:\tilde{\lambda}_S<2/3}\delta_{G_F}(S)\tilde{\lambda}_S$. Summing the former and the latter and recalling that $\tilde{\lambda}_S = \lambda_S/\alpha\ \forall S\subseteq V$, along with the fact that $\{\lambda_S\}_{S\subseteq V}$ satisfies guarantee (3) of Theorem 4, we can conclude that: $\mathbb{E}[\sum_{e\in L}c_e] \leq O(\gamma/\alpha)\mathrm{OPT}$. This concludes the proof of requirement (2). ◀

We are now ready to describe our rounding algorithm, which is depicted in Algorithm 2. The algorithm is straightforward, as it iteratively applies the single iteration algorithm, Algorithm 1, as long as there is a group X_i that is not broken to its required r_i number of pieces. We note that unlike previous algorithms for the (RC) problem, e.g., [18], we need to resolve (LP_F) in each iteration as it is not clear whether the original fractional solution remains feasible for G_F.

Algorithm 2 Rounding.

Input: $G(V,E), \{X_i\}_{i=1}^g, \{r_i\}_{i=1}^g, c: E\to\mathbb{R}_+$
Output: $F\subseteq E$

```
F ← ∅
while ∃i = 1,...,g s.t. X_i is not broken into at least r_i pieces in G_F do
    solve (LP_F) to obtain {λ_S}_{S⊆V} using Theorem 4
    λ̃_S ← λ_S/α, ∀S ⊆ V
    execute Algorithm 1 with G_F and {λ̃_S}_{S⊆V} to obtain L
    F ← F ∪ L
output F
```

The following lemma bounds the number of iterations Algorithm 2 performs, and its proof follows similar lines to the proof of [18]. We present it here for completeness.

▶ **Lemma 6.** *With a probability of at least* $1/2$ *Algorithm 2 performs at most* $O(\alpha/\beta \cdot \log R)$ *iterations.*

Proof. Let R_i^s be the random variable that equals the residual requirement of X_i at the beginning of iteration s of Algorithm 2, and let $R^s = \sum_{i=1}^{g} R_i^s$ be the random variable that equals the total residual requirement at the beginning of iteration s. Requirement (1) of Lemma 5 implies that there exists an absolute constant c such that for every iteration s: $\mathbb{E}[R_i^s | R_i^{s-1}] \leq (1 - c \cdot \beta/\alpha) R_i^{s-1}$, where $(1 - c \cdot \beta/\alpha) < 1$. From linearity of expectation we get that for every iteration s: $\mathbb{E}[R^s] \leq (1 - c \cdot \beta/\alpha)^s R$. Recall that R denotes the total initial requirement, i.e., $R = R^0 = \sum_{i=1}^{g} r_i$. Thus, if we choose $s^* = \log(4R) / \log((1 - c \cdot \beta/\alpha)^{-1})$ then $\mathbb{E}[R^{s^*}] \leq 1/4$. Using Markov's Inequality we get that $\Pr[R^{s^*} < 1/2] \geq 1/2$. However, since R^{s^*} is integral we can conclude that with a probability of at least $1/2$ Algorithm 2 terminates after s^* iterations because all requirements are satisfied, i.e., R^{s^*} reached a value of 0. Noting that $s^* = O(\alpha/\beta \cdot \log R)$ concludes the proof. ◀

Proof of Theorem 1. Lemma 6 proves that with a probability of at least $1/2$ Algorithm 2 performs $O(\alpha/\beta \cdot \log R)$ iterations. Requirement (2) of Lemma 5 implies that the expected cost of every iteration is at most $O(\gamma/\alpha) \cdot \mathrm{OPT}$. Hence, the expected value of the output of the algorithm is $O(\gamma/\beta \cdot \log R) \cdot \mathrm{OPT}$. Plugging the values of γ and β as guaranteed in Theorem 4 concludes the proof. ◀

5 Solving the Configuration LP

In this section we address the problem of solving the relaxation (LP_F). This task requires us to provide an (approximate) separation oracle for the dual of (LP_F), thus proving Theorem 4. Intuitively, the dual separation oracle is a node weighted variant of sparsest cut where demands are given over subsets, as opposed to pairs, of vertices. We denote this problem by Sparsest Requirement Cut (SRC). There are multiple methods of proving that an approximate dual separation oracle provides a bicriteria solution to the primal formulation, e.g., via the Ellipsoid algorithm. In this version of the paper we use Young's iterative method [23]. To simplify presentation, let us denote by $\mathrm{Cross}(S, i) = |\{Y_{i,j} : S \in C(Y_{i,j})\}|$ the number of sets among $Y_{i,1}, \ldots, Y_{i,\ell_i}$ that S separates.

5.1 The Sparsest Requirement Cut Problem

Let us start by formally introducing the (SRC) problem, which is essential to proving Theorem 4.

▶ **Definition 7.** *An instance of the Sparsest Requirement Cut problem consists of the following tuple* $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c, y, z)$*, where:* $G = (V, E)$ *is an undirected graph,* $F \subseteq E$ *is a collection of edges removed from* G*, for every* $i = 1, \ldots, g$*:* $\{Y_{i,j}\}_{j=1}^{\ell_i}$ *is the partition of* X_i *according to the connected components of* G_F*, non-negative edge weights* $c : E \to \mathbb{R}_+$*, non-negative group weights* $z : \{1, \ldots, g\} \to \mathbb{R}_+$*, and non-negative vertex weights* $y : V \to \mathbb{R}_+$*. The goal is to find a cut* $S \subseteq V$ *minimizing:*

$$\frac{\delta_{G_F}(S) + \sum_{u \in S} y_u}{\sum_{i=1}^{g} \mathit{Cross}(S, i) z_i}.$$

The following theorem summarizes our algorithm for (approximately) solving (SRC), and it has a key role in proving Theorem 4. Its proof follows the lines of the classic algorithm for Sparsest Cut (SC) with general demands (see, e.g., [17]): first a metric is found that forms a lower bound on the value of an optimal solution, then it is rounded by embedding it

into ℓ_1. Unfortunately, (SRC) is inherently different from (SC) since it is not symmetric, i.e., S and $\overline{S}$ have (possibly) different objective values. To overcome this we introduce a new artificial point o to the metric space which represents $\overline{S}$. Furthermore, when embedding the metric into ℓ_1 we make sure to consider only cuts that do not contain o.

▶ **Theorem 8.** *Given an instance $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c, y, z)$ of the Sparsest Requirement Cut problem, there exists a polynomial time algorithm that returns a cut S satisfying:*

$$\frac{\delta_{G_F}(S) + \sum_{u\in S} y_u}{\sum_{i=1}^{g} Cross(S,i)z_i} \leq (X-1)\cdot D \cdot OPT_{SRC},$$

where OPT_{SRC} is the value of an optimal solution to the given instance, and D is as in Theorem 2 when considering $U = \cup_{i=1}^{g} X_i \subseteq V$.

Proof. Our proof is comprised of two steps: (1) introducing a semi-definite relaxation and proving it lower bounds $\mathrm{OPT}_{\mathrm{SRC}}$; and (2) rounding the fractional solution and proving it yields the desired approximation factor.

Step 1. We start by presenting a semi-definite relaxation for (SRC) which we denote by (SDP). In (SDP), a squared Euclidean metric space is imposed over $V \cup \{o\}$, where o denotes a special artificial point which is the origin. Thus, every vertex $a \in V \cup \{o\}$ is associated with a vector $\mathbf{v}_a$ and $\mathbf{v}_o$ is constrained to be the origin, i.e., the zero vector. Moreover, we denote by $\mathcal{T}_{i,j}$ the collection of all spanning trees over the complete graph whose vertices are $Y_{i,j}$.

$$\begin{aligned}
\min \quad & \sum_{e=(a,b)\in E\setminus F} c_e ||\mathbf{v}_a - \mathbf{v}_b||_2^2 + \sum_{a\in V} y_a ||\mathbf{v}_a||_2^2 & & \\
\text{s.t.} \quad & ||\mathbf{v}_a - \mathbf{v}_b||_2^2 + ||\mathbf{v}_b - \mathbf{v}_c||_2^2 \geq ||\mathbf{v}_a - \mathbf{v}_c||_2^2 & \forall a,b,c \in V\cup\{o\} & \quad (5)\\
& ||\mathbf{v}_o||_2^2 = 0 & & \quad (6)\\
& \sum_{(a,b)\in T} ||\mathbf{v}_a - \mathbf{v}_b||_2^2 \geq s_{i,j} & \forall i = 1,\ldots,g, \forall j = 1,\ldots,\ell_i, \forall T \in \mathcal{T}_{i,j} & \quad (7)\\
& \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} s_{i,j} \geq 1 & & \quad (8)
\end{aligned}$$

Constraint (5) is the triangle inequality, whereas Constraint (6) enforces that $\mathbf{v}_o$ is the origin. Constraint (7) states that each $s_{i,j}$ is upper bounded by the length of the minimum spanning tree in $\mathcal{T}_{i,j}$. Constraint (8) is a scaling constraint, similar to the standard relaxation for (SC). Clearly, (SDP) is solvable in polynomial time since the separation oracle is just the computation of a minimum spanning tree.

Let us now prove that the optimal value of (SDP) lower bounds $\mathrm{OPT}_{\mathrm{SRC}}$. Let S^* be an optimal solution to the problem, let us define the following solution to (SDP):

$$\mathbf{v}_a = \begin{cases} 0 & a \notin S^* \\ \frac{1}{\sqrt{\sum_{i=1}^{g} \mathrm{Cross}(S^*,i)z_i}}\mathbf{e} & a \in S^* \end{cases} \quad \text{and} \quad s_{i,j} = \begin{cases} \frac{1}{\sum_{i=1}^{g} \mathrm{Cross}(S^*,i)z_i} & S^* \in C(Y_{i,j}) \\ 0 & S^* \notin C(Y_{i,j}) \end{cases},$$

where $\mathbf{e}$ is an arbitrary unit vector. Additionally, we set $\mathbf{v}_0 = \mathbf{0}$. The crucial observation is that $||\mathbf{v}_a - \mathbf{v}_b||_2^2 = (\sum_{i=1}^{g} \mathrm{Cross}(S^*,i)z_i)^{-1}$ if a and b are on different sides of S^* and 0 otherwise. Clearly, the above solution satisfies Constraints (5) and (6). Focusing on Constraint (7), if $s_{i,j} \neq 0$ then $S^* \in C(Y_{i,j})$ and the minimum spanning tree of $Y_{i,j}$ crosses S^* exactly once and its length equals $(\sum_{i=1}^{g} \mathrm{Cross}(S^*,i)z_i)^{-1}\mathbf{e}\cdot\mathbf{e} = s_{i,j}$. Considering Constraint (8), we note that:

$$\sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} s_{i,j} = \sum_{i=1}^{g} z_i \frac{\mathrm{Cross}(S^*,i)}{\sum_{i=1}^{g} \mathrm{Cross}(S^*,i)z_i} = 1.$$

Hence, we can conclude that the above solution is feasible. Moreover, we note that the objective value of the above defined solution equals:

$$\sum_{e=(a,b)\in E\setminus F} c_e||\mathbf{v}_a-\mathbf{v}_b||_2^2+\sum_{a\in V} y_a||\mathbf{v}_a||_2^2=\frac{\delta_{G_F}(S^*)+\sum_{u\in S^*}y_u}{\sum_{i=1}^{g}\text{Cross}(S^*,i)z_i}.$$

Therefore, the value of an optimal solution to (SDP) lower bounds OPT_{SRC}.

Step 2. We start by presenting the rounding algorithm, Algorithm 3. As previously mentioned, Algorithm 3 follows the lines of classis algorithms for (SC) with non-uniform demands [17]. The main difference is that we make sure to consider only cuts that do not contain o when decomposing the ℓ_1 metric into a non-negative combination of cut metrics (such a decomposition can be easily found, see, e.g., [10, 17]). In what follows we use the notation δ_S to denote the cut metric S defines, i.e., $\delta_S(a,b)=1$ if a and b are on different sides of S and 0 otherwise.

Algorithm 3 Rounding (SDP).

Input: $\{\mathbf{v}_a\}_{a\in V\cup\{o\}}$
Output: $S\subseteq V$
let f be the embedding into ℓ_1 of Theorem 2 when considering $U=\cup_{i=1}^{g}X_i\subseteq V\cup\{o\}$
find $\{\mu_r\}_{r=1}^{L}\subseteq\mathbb{R}_+$ and $\{S_r\}_{r=1}^{L}\subseteq 2^V$ s.t. $||f(a)-f(b)||_1=\sum_{r=1}^{L}\mu_r\delta_{S_r}(a,b)$, $\forall a,b\in V\cup\{o\}$
for $r=1$ *to* L **do**
    if $o\in S_r$ **then**
        swap S_r with $V\cup\{o\}\setminus S_r$
output $\text{argmin}_{r=1,\ldots,L}\{(\delta_{G_F}(S_r)+\sum_{a\in S_r}y_a)/(\sum_{i=1}^{g}\text{Cross}(S_r,i)z_i)\}$

Let us now analyze Algorithm 3. Note that:

$$\min_{r=1,\ldots,L}\left\{\frac{\delta_{G_F}(S_r)+\sum_{a\in S_r}y_a}{\sum_{i=1}^{g}\text{Cross}(S_r,i)z_i}\right\}\leq\frac{\sum_{r=1}^{L}\mu_r\left(\sum_{e=(a,b)\in\Gamma(S_r)}c_e\delta_{S_r}(a,b)+\sum_{a\in S_r}y_a\right)}{\sum_{r=1}^{L}\mu_r\sum_{i=1}^{g}\text{Cross}(S_r,i)z_i}.$$

Focusing on the numerator we get that:

$$\begin{aligned}
&\sum_{r=1}^{L}\mu_r\left(\sum_{e=(a,b)\in\Gamma(S_r)}c_e\delta_{S_r}(a,b)+\sum_{a\in S_r}y_a\right)\\
&=\sum_{e=(a,b)\in E\setminus F}c_e\sum_{r=1}^{L}\mu_r\delta_{S_r}(a,b)+\sum_{a\in V}y_a\sum_{r=1}^{L}\mu_r\delta_{S_r}(o,a)\\
&=\sum_{e=(a,b)\in E\setminus F}c_e||f(a)-f(b)||_1+\sum_{a\in V}y_a||f(a)-f(o)||_1\\
&\leq D\left(\sum_{e=(a,b)\in E\setminus F}c_e||\mathbf{v}_a-\mathbf{v}_b||_2^2+\sum_{a\in V}y_a||\mathbf{v}_a||_2^2\right).
\end{aligned}$$

The first equality follows from changing the order of summation and the fact that $o\notin S_r$, $\forall r=1,\ldots,L$, i.e., $\mathbf{1}_{\{a\in S_r\}}=\delta_{S_r}(o,a)$. The inequality follows from Theorem 2 and Constraint (6).

Focusing on the denominator and choosing an arbitrary spanning tree $T_{i,j} \in \mathcal{T}_{i,j}$, we get that:

$$\begin{aligned}
\sum_{r=1}^{L} \mu_r \sum_{i=1}^{g} \text{Cross}(S_r, i) z_i &= \sum_{r=1}^{L} \mu_r \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} \mathbf{1}_{\{S_r \in C(Y_{i,j})\}} \\
&\geq \sum_{r=1}^{L} \mu_r \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} \frac{1}{|Y_{i,j}| - 1} \sum_{(a,b) \in T_{i,j}} \delta_{S_r}(a,b) \\
&\geq \frac{1}{X-1} \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} \sum_{(a,b) \in T_{i,j}} \sum_{r=1}^{L} \mu_r \delta_{S_r}(a,b) \\
&= \frac{1}{X-1} \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} \sum_{(a,b) \in T_{i,j}} ||f(a) - f(b)||_1 \\
&\geq \frac{1}{X-1} \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} \sum_{(a,b) \in T_{i,j}} ||\mathbf{v}_a - \mathbf{v}_b||_2^2 \\
&\geq \frac{1}{X-1} \sum_{i=1}^{g} z_i \sum_{j=1}^{\ell_i} s_{i,j} \geq \frac{1}{X-1}.
\end{aligned}$$

The first equality follows from the definition of $\text{Cross}(S_r, i)$. The first inequality follows from the fact that any spanning tree in $\mathcal{T}_{i,j}$ can cross the cut $S_r \in C(Y_{i,j})$ at most $|Y_{i,j}|-1$ times. The second inequality follows from changing the order of summation and from $|Y_{i,j}| \leq X$. The third inequality follows from Theorem 2 and the fact that if $(a,b) \in T_{i,j}$ then $a, b \in Y_{ß,j} \subseteq U$. The before last inequality follows from Constraint (7), whereas the last inequality follows from Constraint (8). The second step of the proof is concluded by combining the upper bound on the nominator with the lower bound on the denominator, along with the first step of the proof. ◀

5.2 Proving Theorem 4

As previously mentioned we follow the footsteps of Young's iterative method [23]. Please refer to Appendix A for the details.

References

1. Sanjeev Arora, James R. Lee, and Assaf Naor. Fréchet embeddings of negative type metrics. *Discret. Comput. Geom.*, 38(4):726–739, 2007.
2. Sanjeev Arora, James R. Lee, and Assaf Naor. Euclidean distortion and the sparsest cut. *J. Amer. Math. Soc.*, 21(1):1–21, 2008.
3. Adi Avidor and Michael Langberg. The multi-multiway cut problem. *Theor. Comput. Sci.*, 377(1-3):35–42, 2007.
4. Niv Buchbinder, Joseph (Seffi) Naor, and Roy Schwartz. Simplex partitioning via exponential clocks and the multiway-cut problem. *SIAM J. Comput.*, 47(4):1463–1482, 2018.
5. Niv Buchbinder, Roy Schwartz, and Baruch Weizman. Simplex transformations and the multiway cut problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017*, pages 2400–2410. SIAM, 2017.
6. Gruia Călinescu, Howard J. Karloff, and Yuval Rabani. An improved approximation algorithm for MULTIWAY CUT. *J. Comput. Syst. Sci.*, 60(3):564–574, 2000.

7 Shuchi Chawla, Anupam Gupta, and Harald Räcke. Embeddings of negative-type metrics and an improved approximation to generalized sparsest cut. *ACM Trans. Algorithms*, 4(2):22:1–22:18, 2008.

8 Chandra Chekuri, Sudipto Guha, and Joseph (Seffi) Naor. The steiner k-cut problem. *SIAM J. Discrete Math.*, 20(1):261–271, 2006.

9 Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM J. Comput.*, 23(4):864–894, 1994.

10 Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and combinatorics*. Springer, 1997.

11 Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. Comput.*, 25(2):235–251, 1996.

12 Anupam Gupta, Viswanath Nagarajan, and R. Ravi. An improved approximation algorithm for requirement cut. *Operations Research Letters*, 38(4):322–325, 2010.

13 David R. Karger, Philip N. Klein, Clifford Stein, Mikkel Thorup, and Neal E. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *Math. Oper. Res.*, 29(3):436–461, 2004.

14 Philip N. Klein, Serge A. Plotkin, Satish Rao, and Éva Tardos. Approximation algorithms for steiner and directed multicuts. *J. Algorithms*, 22(2):241–269, 1997.

15 Philip N. Klein, Satish Rao, Ajit Agrawal, and R. Ravi. An approximate max-flow min-cut relation for unidirected multicommodity flow, with applications. *Combinatorica*, 15(2):187–202, 1995.

16 Frank Thomson Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999.

17 Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

18 Viswanath Nagarajan and R. Ravi. Approximation algorithms for requirement cut on graphs. *Algorithmica*, 56(2):198–213, 2010.

19 Joseph (Seffi) Naor and Yuval Rabani. Tree packing and approximating k-cuts. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA*, pages 26–27, 2001.

20 R. Ravi and Amitabh Sinha. Approximating k-cuts via network strength. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA*, pages 621–622, 2002.

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

22 Ankit Sharma and Jan Vondrák. Multiway cut, pairwise realizable distributions, and descending thresholds. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014*, pages 724–733, 2014.

23 Neal E. Young. Sequential and parallel algorithms for mixed packing and covering. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001*, pages 538–546, 2001.

A Proving Theorem 4

For simplicity of presentation, for the remainder of this section, we assume that $F \subseteq E$ is given and fixed. Let M be a guess of the value of OPT, and given M define the following polytope capturing (LP_F):

$$\mathcal{Q}(M) = \Big\{\lambda \in \mathbb{R}^{2^V} : \frac{1}{M}\sum_{S\subseteq V}\frac{1}{2}\delta_{G_F}(S)\lambda_S \leq 1 \qquad , \tag{9}$$

$$\sum_{S:u\in S}\lambda_S \leq 1 \qquad \forall u \in V, \tag{10}$$

$$\frac{1}{r'_i}\sum_{j=1}^{\ell_i}\sum_{S\in C(Y_{i,j})}\lambda_S \geq 1 \qquad \forall i = 1,\ldots,g, \tag{11}$$

$$\lambda_S \geq 0 \qquad \forall S \subseteq V\Big\}.$$

We refer to Constraints (9) and (10) as the packing constraints, and Constraint (11) as the covering constraint. For a sufficiently large parameter N, to be determined later, we use the smoothed definitions of the max and min functions for the packing and covering constraints of $\mathcal{Q}(M)$, respectively:

$$\ell max(\lambda) \triangleq \frac{1}{N}\ln\left(R(\lambda) + \sum_{u\in V} y(\lambda,u)\right) \quad \text{and} \quad \ell min(\lambda) \triangleq -\frac{1}{N}\ln\left(\sum_{i=1}^{g} z(\lambda,i)\right),$$

where:

$$R(\lambda) \triangleq \exp\left(\frac{N}{M}\sum_{S\subseteq V}\frac{1}{2}\delta_{G_F}(S)\lambda_S\right)$$

$$y(\lambda,u) \triangleq \exp\left(N\sum_{S:u\in S}\lambda_S\right)$$

$$z(\lambda,i) \triangleq \exp\left(-\frac{N}{r'_i}\sum_{j=1}^{\ell_i}\sum_{S\in C(Y_{i,j})}\lambda_S\right).$$

To simplify presentation, we denote the following value of a cut $S \subseteq V$ by $\phi_\lambda(S)$:

$$\phi_\lambda(S) \triangleq \frac{\sum_{i=1}^{g} z(\lambda,i)}{R(\lambda)+\sum_{u\in V} y(\lambda,u)} \cdot \frac{\frac{1}{M}\frac{1}{2}\delta_{G_F}(S)R(\lambda) + \sum_{u\in S} y(\lambda,u)}{\sum_{i=1}^{g}\frac{1}{r'_i}\mathrm{Cross}(S,i)z(\lambda,i)}.$$

There are two important things to note regarding ϕ_λ. First, for every $S \subseteq V$:

$$\phi_\lambda(S) = \frac{\frac{\partial \ell max(\lambda)}{\partial \lambda_S}}{\frac{\partial \ell min(\lambda)}{\partial \lambda_S}}. \tag{12}$$

Observation (12) plays a crucial role when analyzing Young's iterative method applied to our setting. Second, $\phi_\lambda(S)$ equals the value of S when considering the following instance $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c_\lambda, y_\lambda, z_\lambda)$ of (SRC):

$$c_\lambda(e) = \frac{\frac{1}{2M}R(\lambda)c_e}{R(\lambda)+\sum_{u\in V} y(\lambda,u)} \qquad \forall e \in E\setminus F$$

$$y_\lambda(u) = \frac{y(\lambda,u)}{R(\lambda)+\sum_{u\in V} y(\lambda,u)} \qquad \forall u \in V$$

$$z_\lambda(i) = \frac{\frac{1}{r'_i}z(\lambda,i)}{\sum_{i=1}^{g} z(\lambda,i)} \qquad \forall i = 1,\ldots,g.$$

The following lemma proves that there exists a solution to the above instance of (SRC) whose value is at most 1, assuming our guess for M is not smaller than OPT.

▶ **Lemma 9.** *For every $\lambda \in \mathbb{R}^{2^V}$, if $M \geq OPT$ then there exists a cut $S \subseteq V$ such that $\phi_\lambda(S) \leq 1$.*

Proof. We define two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{2^V}$, indexed by S, as follows:

$$a_S \triangleq \frac{\frac{1}{M}\frac{1}{2}\delta_{G_F}(S)R(\lambda) + \sum_{u\in S} y(\lambda,u)}{R(\lambda) + \sum_{u\in V} y(\lambda,u)} \quad \text{and} \quad b_S \triangleq \frac{\sum_{i=1}^{g} \frac{1}{r'_i}\text{Cross}(S,i)z(\lambda,i)}{\sum_{i=1}^{g} z(\lambda,i)},$$

and prove that $\langle \mathbf{a}, \lambda^* \rangle \leq 1$ and $\langle \mathbf{b}, \lambda^* \rangle \geq 1$, for some non-negative vector $\lambda^* \in \mathbb{R}_+^{2^V}$. Since $\mathbf{a}$, $\mathbf{b}$, and λ^* are all non-negative, we can conclude that there exists $S \subseteq V$ such that $a_S \leq b_S$. This will conclude the proof.

We note that since $M \geq \text{OPT}$, $\mathcal{Q}(M)$ is non-empty. The reason for the latter is that $\lambda^* \in \mathcal{Q}(M)$, where λ^* is an optimal solution to the (RC) problem applied to G_F (as defined in the proof of Lemma 3). Hence, we know that:

$$\sum_{S\subseteq V} \frac{1}{2}\delta_{G_F}(S)\lambda^*_S \leq \text{OPT} \tag{13}$$

$$\sum_{S:u\in S} \lambda^*_S \leq 1 \qquad \forall u \in V \tag{14}$$

$$\sum_{j=1}^{\ell_i} \sum_{S\in C(Y_{i,j})} \lambda^*_S \geq r'_i \qquad \forall i = 1,\ldots,g. \tag{15}$$

We note that (13) above follows from Lemma 3. First, let us prove that $\langle \mathbf{a}, \lambda^* \rangle \leq 1$:

$$\begin{aligned}
\langle \mathbf{a}, \lambda^* \rangle &= \sum_{S\subseteq V} \frac{\frac{1}{M}\frac{1}{2}\delta_{G_F}(S)R(\lambda) + \sum_{u\in S} y(\lambda,u)}{R(\lambda) + \sum_{u\in V} y(\lambda,u)}\lambda^*_S \\
&= \frac{\frac{R(\lambda)}{M}\sum_{S\subseteq V}\frac{1}{2}\delta_{G_F}(S)\lambda^*_S + \sum_{u\in V} y(\lambda,u)\sum_{S:u\in S}\lambda^*_S}{R(\lambda) + \sum_{u\in V} y(\lambda,u)} \\
&\leq \frac{\frac{R(\lambda)}{M}\text{OPT} + \sum_{u\in V} y(\lambda,u)}{R(\lambda) + \sum_{u\in V} y(\lambda,u)} \leq 1.
\end{aligned}$$

The second equality follows from changing the order of summation. The first inequality follows from (13) and (14), whereas the last inequality follows since $M \geq \text{OPT}$. Second, let us prove that $\langle \mathbf{b}, \lambda^* \rangle \geq 1$:

$$\begin{aligned}
\langle \mathbf{b}, \lambda^* \rangle &= \sum_{S\subseteq V} \frac{\sum_{i=1}^{g}\frac{1}{r'_i}\text{Cross}(S,i)z(\lambda,i)}{\sum_{i=1}^{g} z(\lambda,i)}\lambda^*_S \\
&= \frac{\sum_{i=1}^{g}\frac{1}{r'_i} z(\lambda,i)\sum_{j=1}^{\ell_i}\sum_{S\in C(Y_{i,j})}\lambda^*_S}{\sum_{i=1}^{g} z(\lambda,i)} \geq 1.
\end{aligned}$$

The second equality follows from changing the order of summation, whereas the inequality follows from (15). ◀

The following lemma establishes the connection between the above and our algorithm for (approximately) solving (SRC), i.e., Theorem 8.

▶ **Lemma 10.** *Given $\lambda \in \mathbb{R}^{2^V}$, if $M \geq OPT$ then executing the algorithm of Theorem 8 on the instance $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c_\lambda, y_\lambda, z_\lambda)$ of (SRC) yields a cut $\tilde{S} \subseteq V$ satisfying: $\phi_\lambda(\tilde{S}) \leq (X-1)D$.*

Proof. Let S^* be an optimal solution to the instance $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c_\lambda, y_\lambda, z_\lambda)$ of (SRC), or equivalently, a cut that minimizes $\phi_\lambda(S^*)$. Since $M \geq \mathrm{OPT}$, we can apply Lemma 9 and obtain that $\mathrm{OPT}_{\mathrm{SRC}} \leq 1$ for the given instance. Theorem 8 concludes the proof. ◀

We are now ready to present Young's iterative approach adapted to (LP_F), Algorithm 4. The parameter $\zeta > 0$ in the input determines the step size of the algorithm and will be determined later.

Algorithm 4 Solving (LP_F).

Input: $G = (V, E), F \subseteq E, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c : E \to \mathbb{R}_+, \zeta > 0$
Output: $\lambda \in \mathbb{R}_+^{2^V}$
$\lambda \leftarrow \mathbf{0}$
while $\exists i = 1, \ldots, g$ *s.t.* $\frac{1}{r'_i} \sum_{j=1}^{\ell_i} \sum_{S \in C(Y_{i,j})} \lambda_S < 1$ **do**
  apply Theorem 8 on instance $(G, F, \{\{Y_{i,j}\}_{j=1}^{\ell_i}\}_{i=1}^{g}, c_\lambda, y_\lambda, z_\lambda)$ of (SRC) to obtain $\tilde{S}$
  $\lambda_{\tilde{S}} \leftarrow \lambda_{\tilde{S}} + \zeta$
output λ

The following lemma is used to upper bound the step size ζ in Algorithm 4 and it follows directly by adapting Lemma 1 of [23] to our setting.

▶ **Lemma 11.** *For every $0 < \varepsilon \leq 1$ and $\zeta > 0$ such that $\zeta \leq \varepsilon \min\{1, 1/X, (2M)/(\sum_{e \in E \setminus F} c_e)\}$, the following two hold for every $S \subseteq V$:*

$$\frac{\ell max(\lambda + \zeta \mathbf{1}_S) - \ell max(\lambda)}{\zeta(1+\varepsilon)} \leq \frac{\partial \ell max(\lambda)}{\partial \lambda_S} = \frac{\frac{1}{M} \cdot \frac{1}{2}\delta_{G_F}(S) \cdot R(\lambda) + \sum_{u \in S} y(\lambda, u)}{R(\lambda) + \sum_{u \in V} y(\lambda, u)} \tag{16}$$

$$\frac{\ell min(\lambda + \zeta \mathbf{1}_S) - \ell min(\lambda)}{\zeta(1-\varepsilon/2)} \geq \frac{\partial \ell min(\lambda)}{\partial \lambda_S} = \frac{\sum_{i=1}^{g} \frac{1}{r'_i} \cdot Cross(S, i) z(\lambda, i)}{\sum_{i=1}^{g} z(\lambda, i)}. \tag{17}$$

The following lemma states that when Algorithm 4 terminates, assuming $M \geq \mathrm{OPT}$, it produces an approximate solution to $\mathcal{Q}(M)$.

▶ **Lemma 12.** *For every $0 < \varepsilon \leq 1$, $\zeta > 0$ satisfying the conditions of Lemma 11, and $N = \varepsilon^{-1} \ln(g(n+1))$, when Algorithm 4 terminates, assuming $M \geq OPT$, it outputs $\lambda \in \mathbb{R}_+^{2^V}$ satisfying:*

$$\sum_{S: u \in S} \lambda_S \leq \alpha \qquad \forall u \in V \tag{18}$$

$$\frac{1}{r'_i} \sum_{j=1}^{\ell_i} \sum_{S \in C(Y_{i,j})} \lambda_S \geq \beta \qquad \forall i = 1, \ldots, g \tag{19}$$

$$\frac{1}{M} \sum_{S \subseteq V} \frac{1}{2} \delta_{G_F}(S) \lambda_S \leq \gamma. \tag{20}$$

In the above $\alpha = \gamma = \varepsilon + \frac{(1+\varepsilon)(1+2\varepsilon)}{1-\varepsilon/2}(X-1)D$ and $\beta = 1$.

Proof. From the stopping condition of Algorithm 4, it is clear that (19) holds with $\beta = 1$, i.e., the covering constraints are satisfied. Let us now focus on the remaining two covering constraints. Denote the sequence of cuts Algorithm 4 increased by: $S_1, \ldots, S_L$ (note that some cuts might appear several times in the sequence), and by λ the output, i.e., $\lambda = \zeta \sum_{r=1}^{L} \mathbf{1}_{S_r}$. For simplicity of presentation, we denote by λ^r the solution Algorithm 4 maintains after the r^{th} iteration, i.e., $\lambda^r = \zeta \sum_{s=1}^{r} \mathbf{1}_{S_s}$. Note that $\lambda^0 = \mathbf{0}$ and $\lambda^L = \lambda$. Let us now upper bound the worst packing constraint upon termination of the algorithm:

$$\begin{aligned}
&\max\left\{\frac{1}{M}\sum_{S\subseteq V}\frac{1}{2}\delta_{G_F}(S)\lambda_S, \max_{u\in V}\left\{\sum_{S:u\in S}\lambda_S\right\}\right\} \leq \ell max(\lambda)\\
&= \frac{\ln(n+1)}{N} + \sum_{r=1}^{L}\left(\ell max\left(\lambda^r\right) - \ell max\left(\lambda^{r-1}\right)\right)\\
&\leq \frac{\ln(n+1)}{N} + \frac{1+\varepsilon}{1-\varepsilon/2}\sum_{r=1}^{L}\phi_{\lambda^{r-1}}(S_r)\left(\ell min\left(\lambda^r\right) - \ell min\left(\lambda^{r-1}\right)\right)\\
&\leq \frac{\ln(n+1)}{N} + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\sum_{r=1}^{L}\left(\ell min\left(\lambda^r\right) - \ell min\left(\lambda^{r-1}\right)\right)\\
&= \frac{\ln(n+1)}{N} + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\frac{\ln g}{N} + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\ell min(\lambda)\\
&\leq \varepsilon + \frac{\varepsilon(1+\varepsilon)}{1-\varepsilon/2}(X-1)D + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\min_{i=1,\ldots,g}\left\{\frac{1}{r'_i}\sum_{j=1}^{\ell_i}\sum_{S\in C(Y_{i,j})}\lambda_S\right\}\\
&\leq \varepsilon + \frac{\varepsilon(1+\varepsilon)}{1-\varepsilon/2}(X-1)D + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\cdot(1+\zeta X)\\
&\leq \varepsilon + \frac{\varepsilon(1+\varepsilon)}{1-\varepsilon/2}(X-1)D + \frac{1+\varepsilon}{1-\varepsilon/2}(X-1)D\cdot(1+\varepsilon)\\
&= \varepsilon + \frac{(1+\varepsilon)(1+2\varepsilon)}{1-\varepsilon/2}(X-1)D.
\end{aligned}$$

The first inequality follows from the definition of ℓmax. Since $\ell max(\lambda^0) = \ell max(\mathbf{0}) = \ln(n+1)/N$, the first equality follows. The second inequality follows from Lemma 11 and the definition of ϕ. We note that the third inequality follows from Lemma 10 and how S_r is chosen by Algorithm 4. Since $\ell min(\lambda^0) = \ell min(\mathbf{0}) = -\ln g/N$, the second equality follows. The fourth inequality follows from the choice of N and the definition of ℓmin. Note that the fifth inequality follows from the stopping condition of Algorithm 4, i.e., the algorithm stops once all coverings constraints are satisfied. The last inequality follows from the restrictions on ζ. ◀

Proof of Theorem 4. All that remains is to choose ζ, such that for every guess of M Algorithm 4 performs a polynomial number of iterations. We do this by tracking the following potential: $\Phi(\lambda) \triangleq \sum_{u\in V}\sum_{S:u\in S}\lambda_S$. Note that initially $\Phi(\lambda) = \Phi(\mathbf{0}) = 0$. From guarantee (18) of Lemma 12, we know that once Algorithm 4 terminates the value of $\Phi(\lambda)$ cannot exceed $n(\varepsilon + (1+\varepsilon)(1+2\varepsilon)/(1-\varepsilon/2)\cdot(X-1)D)$. Setting the step size $\zeta = \varepsilon\min\{1, 1/X, (2M)/(\sum_{e\in E\setminus F} c_e)\}$ and choosing $\varepsilon = 1$ implies that Algorithm 4 performs at most $O(nXD/\zeta)$ iterations. Applying standard weight rescaling techniques, we can assume without loss of generality, that $1 \leq c_e \leq \text{poly(n)}$, $\forall e \in E$. Thus, given a guess M, Algorithm 4 performs at most a polynomial number of iterations (recall that $M \geq 1$ due to the rescaling). Since all edge weights are rescaled as above, one can find in polynomial time a value M such that $\text{OPT} \leq M \leq 2\text{OPT}$. This concludes the proof. ◀