Matroids and integrality gaps for hypergraphic steiner tree relaxations

Citation: Goemans, Michel X., Neil Olver, Thomas Rothvoss, and Rico Zenklusen. “Matroids and integrality gaps for hypergraphic steiner tree relaxations.” In Proceedings of the 44th symposium on Theory of Computing - STOC 12, 1161. Association for Computing Machinery, 2012.

As Published: http://dx.doi.org/10.1145/2213977.2214081

Persistent URL: http://hdl.handle.net/1721.1/80862

Version: Original manuscript: author's manuscript prior to formal peer review

Matroids and Integrality Gaps for Hypergraphic Steiner Tree Relaxations

Michel X. Goemans[*] Neil Olver[†] Thomas Rothvoß[‡] Rico Zenklusen[§]

M.I.T.

December 14, 2011

Abstract

Until recently, LP relaxations have only played a very limited role in the design of approximation algorithms for the Steiner tree problem. In particular, no (efficiently solvable) Steiner tree relaxation was known to have an integrality gap bounded away from 2, before Byrka et al. [3] showed an upper bound of ≈ 1.55 of a hypergraphic LP relaxation and presented a $\ln(4) + \epsilon \approx 1.39$ approximation based on this relaxation. Interestingly, even though their approach is LP based, they do not compare the solution produced against the LP value.

We take a fresh look at hypergraphic LP relaxations for the Steiner tree problem—one that heavily exploits methods and results from the theory of matroids and submodular functions—which leads to stronger integrality gaps, faster algorithms, and a variety of structural insights of independent interest. More precisely, along the lines of the algorithm of Byrka et al. [3], we present a deterministic $\ln(4) + \epsilon$ approximation that compares against the LP value and therefore proves a matching $\ln(4)$ upper bound on the integrality gap of hypergraphic relaxations.

Similarly to [3], we iteratively fix one component and update the LP solution. However, whereas in [3] the LP is solved at every iteration after contracting a component, we show how feasibility can be maintained by a greedy procedure on a well-chosen matroid. Apart from avoiding the expensive step of solving a hypergraphic LP at each iteration, our algorithm can be analyzed using a simple potential function. This potential function gives an easy means to determine stronger approximation guarantees and integrality gaps when considering restricted graph topologies. In particular, this readily leads to a $\frac{73}{60} \approx 1.217$ upper bound on the integrality gap of hypergraphic relaxations for quasi-bipartite graphs.

Additionally, for the case of quasi-bipartite graphs, we present a simple algorithm to transform an optimal solution to the bidirected cut relaxation to an optimal solution of the hypergraphic relaxation, leading to a fast $\frac{73}{60}$ approximation for quasi-bipartite graphs. Furthermore, we show how the separation problem of the hypergraphic relaxation can be solved by computing maximum flows, which provides a way to obtain a fast independence oracle for the matroids that we use in our approach.

[*] E-mail: goemans@math.mit.edu. Supported by NSF grants CCF-1115849 and CCF-0829878, and by ONR grant N00014-11-1-0053.

[†] E-mail: olver@math.mit.edu. Supported by NSF grant CCF-1115849.

[‡] E-mail: rothvoss@math.mit.edu. Supported by the Alexander von Humboldt Foundation within the Feodor Lynen program, by ONR grant N00014-11-1-0053 and by NSF contract CCF-0829878.

[§] E-mail: ricoz@math.mit.edu. Supported by NSF grants CCF-1115849 and CCF-0829878, and by ONR grants N00014-11-1-0053 and N00014-09-1-0326.

1 Introduction

The Steiner tree problem is one of the most fundamental and important problems in Computer Science and Operations Research. Whereas a 2-approximation is easily obtained by computing a minimum spanning tree over the terminals, obtaining algorithms with an approximation guarantee bounded away from 2 has proven to be a non-trivial task. The problem is known to be inapproximable to within $\frac{96}{95}$, unless $\mathbf{NP} = \mathbf{P}$ [1, 8]). There has been a long sequence of combinatorial approximation algorithms [9, 24, 12, 19, 21], based on different greedy approaches, culminating in the famous $1 + \frac{\ln(3)}{2} + \epsilon < 1.55$ approximation of Robins and Zelikovsky [21]. No further progress was achieved until Byrka, Grandoni, Rothvoß and Sanità [3] presented the first LP-based approach leading to a $\ln(4) + \epsilon \approx 1.39$ approximation. A major hindrance in the design of LP-based Steiner tree algorithms is a rather poor understanding of potential LP relaxations. In particular, until the result of [3], for no (efficiently solvable) LP relaxation of the Steiner tree problem was it known whether the integrality gap was bounded away from 2. Intriguingly, even though their $\ln(4) + \epsilon$ approximation algorithm is based on a particular LP relaxation, its approximation guarantee is not with respect to the LP solution and does not imply a $\ln(4)$ integrality gap for the relaxation. In [3], the authors show a weaker ≈ 1.55 integrality gap using a technique not directly linked to their algorithm. Chakrabarty et al. [7] provide a simpler alternative proof of the same bound.

The linear relaxation used by Byrka et al., the *directed component-based relaxation*, was introduced by Polzin and Vahdati-Daneshmand [18], based in turn on an equivalent *undirected* component-based LP introduced by Warme [23]. It is the undirected version that we will use in this paper. Another notable relaxation is the partition-based LP introduced by Könemann et al. [13]. In [6], Chakrabarty et al. showed that this relaxation is equivalent to the others mentioned above, and introduced the term "hypergraphic" for this family of relaxations. They also proved that basic solutions are sparse, having support size less than the number of terminals.

The limited understanding of LP relaxations of the Steiner tree problem is arguably a major barrier in the design of stronger approximation algorithms. The goal of this work is to fill this gap by providing a fresh view on the component-based LP relaxation—one that heavily exploits methods and results from the theory of matroids and submodular functions. More precisely, based on the approach of Byrka et al. [3], we present a deterministic $\ln(4) + \epsilon$ algorithm that starts with a solution to the component-based LP relaxation, iteratively contracts a component and updates the LP solution. The algorithm of Byrka et al. solves the component-based LP (through a very large extended formulation) in each iteration after contracting, in order to again obtain a feasible solution. By contrast, we show how the LP can be modified by a simple greedy algorithm over a well-chosen matroid to achieve the same goal. This leads to a considerably faster way to update the LP, but more importantly, we show how the approximation quality of our approach can be analyzed with respect to the initial LP solution. This implies a bound of the integrality gap of the component-based LP relaxation of $\ln(4)$. By comparison, the best known lower bound is $8/7 \approx 1.142$ (e.g., by the example of [13]). Furthermore, we show how the separation problem of the component-based relaxation can be reduced to computing maximum flows. Whereas this result is likely to be of independent interest, it also provides a way to obtain a fast independence oracle for the matroids that we use in our approach.

Additionally, we further investigate the special case of quasi-bipartite graphs, which has played a central role in the design of approximation algorithms for the Steiner tree problem, as well as to find $\mathbf{APX}$-hard problem classes. Rajagopalan and Vazirani [20] showed that the integrality gap of the bidirected cut relaxation for such graphs can be bounded by $3/2$. This was later improved to $4/3$ [5] and to 1.28 [7]. We obtain a $\frac{73}{60}$ bound for the integrality gap, again matching the approximation factor of [3]. Such a bound was previously known only for the case when all edge costs are equal [7]. Chakrabarty et al. [6] showed that on quasi-bipartite graphs, the bidirected cut and hypergraphic relaxations are actually equivalent. However their proof is based on a duality argument, and they leave as an open problem the question of converting a solution from the bidirected cut relaxation to the hypergraphic relaxation efficiently (more quickly than simply optimizing the hypergraphic LP). We present a simple algorithm to perform this transformation; since the bidirected cut relaxation can be solved much more efficiently via a compact extended formulation, this gives a much faster method of solving the hypergraphic LP in the quasi-bipartite case. Combining this result with the suggested approximation algorithm, we obtain a significantly faster $\frac{73}{60}$ approximation than the one of Byrka et al. [3], since we do not need to (repeatedly) optimize the component-based relaxation by using either the ellipsoid method or a very large extended formulation.

2 Discussion of results and techniques

2.1 The component-based LP

Let $G = (V, E)$ be an undirected graph with terminals $R \subseteq V$ and edge costs $c : E \to \mathbb{R}_+$. A *component* C is simply a subgraph of G with the property that it is a tree spanning $V(C)$, all leaves of C are terminals, and all internal nodes are non-terminals. Write $\mathrm{cost}(C) := \sum_{e \in E(C)} c(e)$ for the cost of a component C. We will frequently need the terminal set of a component $C \in \mathcal{K}$, and so by abuse of notation, when we refer to C as a vertex set, we mean the set $V(C) \cap R$ of terminals in C. In particular, $|C|$ refers to the number of terminals in C.

Now let $\mathcal{K}$ be the set of all components of G; we assume that all components contain at least two terminals, else they can be safely removed. We use the notation $(Z)^+ := \max\{Z, 0\}$. Then the component-based LP relaxation is as follows [23]:

$$\begin{aligned} \min \quad & \sum_{C \in \mathcal{K}} x_C \,\mathrm{cost}(C) & & \\ & \sum_{C \in \mathcal{K}} x_C (|S \cap C| - 1)^+ \;\leq\; |S| - 1 & & \forall S \subseteq R, S \neq \emptyset \\ & \sum_{C \in \mathcal{K}} x_C (|C| - 1) \;=\; |R| - 1 & & \\ & x_C \;\geq\; 0 & & \forall C \in \mathcal{K}. \end{aligned} \tag{LP}$$

Borchers and Du [2] showed that the optimal k*-restricted* Steiner tree, meaning only components with at most k terminals can be used, has cost at most $1 + 1/\lfloor \log_2 k \rfloor$ times the cost of an optimal Steiner tree. Furthermore, when restricting the variables in (LP) to components with at most k terminals, the resulting linear program can be solved efficiently, e.g., by solving a polynomial-size extended formulation [3]. It follows that for any fixed $\epsilon > 0$, a $(1 + \epsilon)$-approximate solution to (LP) can be obtained efficiently. We also point out in Appendix E that optimizing (LP) exactly is strongly $\mathbf{NP}$-hard (this does not seem to have been previously observed).

The framework of our algorithm is similar to Byrka et al. [4], and in particular, it is iterative in nature. They begin by computing a near-optimal fractional solution x to (LP). They then sample a component C at random, proportional to its entry x_C, and *contract* this component. The solution x is no longer feasible to (LP) on this new contracted instance, so they re-solve the LP and iterate this procedure until all terminals are connected.

In their analysis, they show that a single random contraction reduces the cost of the *optimum* Steiner tree by a certain factor in each iteration. The crucial ingredient is a lower bound on the expected cost of edges that could be removed from an optimum solution after a contraction, while still obtaining a Steiner tree. To obtain a bound on the integrality gap, we need a stronger result that says even a *fractional* solution becomes significantly cheaper after a random contraction. Even for a fixed set of terminals Q, it was unclear how to modify a fractional solution in order to preserve feasibility after contraction—a question that had a simple answer in the integral case. Our first goal will be to obtain an understanding of the structure of these modifications.

While it can be avoided, it significantly simplifies the discussion to consider "blown up" versions of solutions to (LP). Consider any $x \in \mathbb{Q}_+^{\mathcal{K}}$, and let $N \in \mathbb{N}$ be such that $x_C \cdot N \in \mathbb{N}$ for all $C \in \mathcal{K}$. The minimal *blowup graph* corresponding to x is the unweighted multigraph defined as follows. First take the disjoint union of $x_C \cdot N$ disjoint copies of C for each component C; then identify, for each $v \in R$, all the copies of v. The edge costs of $\mathcal{X}$ are inherited from G in the obvious way. See Figure 1 for an example of an LP solution and its associated minimal blowup graph. Observe then that $\mathrm{cost}(\mathcal{X}) = N \cdot \mathrm{cost}(x)$. Note that $\mathcal{X}$ (along with N, but this will remain fixed throughout) encodes all the information in x. In particular, given $\mathcal{X}$ we can determine all of its components: these are simply the maximal connected subgraphs that are trees whose leaves are precisely the terminals spanned. Thus we can define $\Gamma(\mathcal{X})$ as the set of components of a blowup graph $\mathcal{X}$. Each component $C \in \Gamma(\mathcal{X})$ is a subgraph of $\mathcal{X}$, but again, we will abuse notation when the context is clear and sometimes use C to refer to just the terminals of C. Thus, e.g., for some $S \subseteq R$, $S \cap C$ refers to the terminals in C that are also in S.

We will need slightly more generality in our definition of a blowup graph. For any $t \in \mathbb{N}$, let G_t be the multigraph obtained by first taking t disjoint copies of G, and then for each $v \in R$, identifying all copies of v. For a solution y with corresponding minimal blowup graph $\mathcal{Y}$, we call a multigraph $\mathcal{Y}'$ a (not necessarily minimal) blowup graph corresponding to y if (i) $\mathcal{Y}' \subseteq G_t$ for some $t \in \mathbb{N}$, (ii) $\mathcal{Y}' \supseteq \mathcal{Y}$, and (iii) for any distinct terminals $u, v \in R$, there is no u-v-path in $\mathcal{Y}'$ that is not already present in $\mathcal{Y}$. Any edges in $\mathcal{Y}'$ that were not in $\mathcal{Y}$ we call *pendant* edges. We will say that a blowup graph $\mathcal{Y}$ is *feasible* if it corresponds to a feasible solution to (LP); otherwise we call it infeasible. Note that pendant edges have no effect on feasibility; they will always be removed in what we will later call a "cleanup" step.

2.2 Edge removal after contraction

Let $\mathcal{X}$ be the blowup graph corresponding to some solution x. We are interested in the situation after contracting some full component of G. In order to avoid some annoying technicalities, for now instead of contracting Q we will think of increasing the value of x_Q by 1. In other words, in terms of the blowup graph, we take N fresh copies of component Q and add it to $\mathcal{X}$. We denote the new blowup graph obtained by $\mathcal{X} \circledast Q$. Formally, $\mathcal{X} \circledast Q$ is obtained by taking the disjoint union of $\mathcal{X}$ and N copies of Q, and then identifying all copies of v for each $v \in R$.

It is clear that $\mathcal{X} \circledast Q$ is not feasible. We are interested in describing the set of edges $F \subseteq E(\mathcal{X})$ that can be *removed* so that $(\mathcal{X} \circledast Q) - F$ is feasible.

This is the primary reason that it is simpler to work with the blowup graph $\mathcal{X}$ rather than x; this modification operation is much simpler than an equivalent operation defined on x. For example, removing a single edge from $\mathcal{X}$ can have the effect of splitting up some component C into subcomponents C_1 and C_2; the corresponding effect on x is to reduce x_C by $1/N$ and increase x_{C_1} and x_{C_2} by the same amount.

Unfortunately, the set of all possible edge removals is not so well behaved. In order to expose the structure we need, we must consider *minimal* removals. Let

$$\mathcal{B}_Q = \{B \subseteq E(\mathcal{X}) \mid (\mathcal{X} \circledast Q) - B \text{ is feasible, and } B \text{ is minimal with this property}\}.$$

Figure 2 shows an example; after a set $B \in \mathcal{B}_Q$ is removed, an edge of the blowup graph becomes pendant, and so can also be removed without affecting feasibility.

One of the most crucial elements of our analysis is the following:

Theorem 2.1. *For every component Q, $\mathcal{B}_Q$ forms the set of bases of a matroid M_Q.*

In particular, it follows that any minimal removal set has the same number of edges; this number turns out to be $N(|Q|-1)$. We are able to give a precise description of the matroid M_Q by giving its rank function; more details of this will be given in Section 3. We can also show that the matroid is a gammoid (a special type of matroid related to flows); see Appendix A. As an aside, we note that M_Q depends only on the terminals of Q, and not its structure; we could actually define a matroid M_S for *any* subset S of terminals, but this will not be important for our purposes.

We will now study which edge sets can be removed after the random contraction of a component. Even though we will finally present a deterministic algorithm, this analysis will be helpful in guaranteeing the existence of removal sets with certain properties by an averaging argument.

As before, let $\mathcal{X}$ be the blowup graph corresponding to a feasible LP solution x. Upon contracting component Q, we may remove some edges in order to again obtain a feasible solution. In particular, by Theorem 2.1, we can remove any basis of M_Q. For added flexibility, we allow choosing a basis $B_Q \in \mathcal{B}_Q$ randomly, according to any distribution we like. In this case, each edge e will be removed with some probability q_e. The probability vectors that are attainable are simply the convex combinations of incidence vectors of the bases; in other words, precisely the vectors in $B(M_Q)$, the base polytope of M_Q.

Now consider, as in [4], randomly contracting a single component, with component $Q \in \Gamma(\mathcal{X})$ contracted with probability $1/|\Gamma(\mathcal{X})|$. Note that since each original component $\tilde{Q} \in \mathcal{K}$ has $Nx_{\tilde{Q}}$ copies in $\Gamma(\mathcal{X})$, this is the same as contracting a component in $\mathcal{K}$ with probability proportional to $x_{\tilde{Q}}$. Again, we allow ourselves to choose an arbitrary distribution over $\mathcal{B}_Q$ for removals on contracting Q, and ask what probability vectors p describing edge removal probabilities are attainable. But any such probability vector is given by some convex combination $\frac{1}{|\Gamma(\mathcal{X})|}\sum_{Q\in\Gamma(\mathcal{X})} q^Q$, where $q^Q \in B(M_Q)$. In other words, the attainable probability vectors form precisely the polytope B_{rem} given by the Minkowski sum

$$B_{\mathit{rem}} = \frac{1}{|\Gamma(\mathcal{X})|}\sum_{Q\in\Gamma(\mathcal{X})} B(M_Q).$$

This implies that B_{rem} is a polymatroid [14]; from our knowledge of the rank functions of the M_Q's, we can also describe the rank function of B_{rem}, as will be described in detail in Section 3.

In the following, we use *scaled cost* to refer to costs reduced by a factor of N, compensating for the blowup factor. The goal is to show that the expected scaled cost of removed edges is large, compared to the expected cost of the component that is contracted. Perfection would be if we could always remove edges of total scaled cost as large as the cost of the contracted component, but of course this is not possible (it would imply an integrality gap of 1). Thus we lower our goals slightly. It *is* possible to show that there is a point $p \in B_{\mathit{rem}}$ with $p_e \geq \frac{N}{2|\Gamma(\mathcal{X})|}$ for all $e \in E(\mathcal{X})$. This gives an expected decrease of $\mathrm{cost}(\mathcal{X})/(2|\Gamma(\mathcal{X})|)$ in the LP solution after scaling down, and the expected cost of the contracted component is $\mathrm{cost}(\mathcal{X})/|\Gamma(\mathcal{X})|$; so this implies only an uninteresting bound of 2 on the integrality gap. Instead, we must choose the distribution more carefully.

More precisely, we will choose a well-structured subset $K \subseteq E(\mathcal{X})$ and only consider removal probabilities $p \in B_{\mathit{rem}}$ whose support is contained in K. The set K will be chosen to be a minimal subset of $E(\mathcal{X})$ whose removal from $E(\mathcal{X})$ disconnects all terminals in the blowup graph. We call such a set a *splitting set*[1]. Interestingly, the family of all splitting sets form the bases of a cographic matroid, since K is a splitting set precisely when $E(\mathcal{X}) \setminus K$ is a spanning tree in the graph obtained from $\mathcal{X}$ by contracting together all its terminals. As we will see more formally in the proof of Theorem 2.2, when choosing K to be a splitting set, the set $\mathcal{B}_Q^K = \{B \in \mathcal{B}_Q \mid B \subseteq K\}$ is nonempty for every Q, and so form the bases of the matroid M_Q^K obtained by restricting M_Q to K. This implies that the polytope $B_{\mathit{rem}}^K = \{p \in B_{\mathit{rem}} \mid \mathrm{supp}(p) \subseteq K\}$ of

[1] The complements of splitting sets are sometimes called *losses*.

removal probabilities we consider is nonempty, and thus forms the base polytope of the polymatroid obtained by restricting the polymatroid corresponding to B_{rem} to K.

Once we have chosen some splitting set K, we will call edges in K *core edges*, and all other edges *cleanup edges*. To see the reason for this name, recall that the matroid M_Q describes only the *minimal* edge removals upon contracting Q. However, there may be other removals that are possible; for $B \in \mathcal{B}_Q$, there may be pendant edges in $(\mathcal{X} \circledast Q) - B$ which can be removed without having any effect on feasibility. Our choice of K ensures that for any edge $e \in E(\mathcal{X}) \setminus K$, e can be deleted ("cleaned up") once enough edges of $K \cap C$ have been removed. But just as importantly, we can prove

Theorem 2.2. *If K is any splitting set, then there is a distribution over $\mathcal{B}_Q^K$ for each $Q \in \Gamma(\mathcal{X})$ such that if Q is chosen uniformly at random from $\Gamma(\mathcal{X})$, and then B is chosen from $\mathcal{B}_Q^K$ according to the chosen distribution, then*

$$\mathbb{P}\{e \in B\} \geq N/|\Gamma(\mathcal{X})| \qquad \textit{for each } e \in K.$$

This is discussed further in Section 3.

2.3 The algorithm

For the accounting in our analysis, we will need to keep track of precisely which edges in $E(C) \cap K$ must be removed before an edge $e \in E(C) \setminus K$ can be deleted (cleaned up). Define $W(e) \subseteq K$, the *witness set* of edge e, as the unique *minimal* set of edges such that after removing $W(e)$, e becomes a pendant edge and can be cleaned up. The fact that there exists such a unique set is shown in Lemma B.1 in the appendix. We also define $W(e) = \{e\}$ if $e \in K$. Figure 3 shows an example of a witness set.

We define a *weight* (distinct from the cost) on all core edges in such a way that the total weight of core edges equals the total cost of $\mathcal{X}$, by charging the cost of a cleanup edge to the core edges in its witness set. More precisely, let

$$w(e) = c(e) \quad + \sum_{f \notin K : e \in W(f)} \frac{c(f)}{|W(f)|} \qquad \text{for all } e \in K.$$

The following is an easy consequence of Theorem 2.2 and the fact that $\sum_{e \in K} w(e) = \mathrm{cost}(\mathcal{X})$:

Lemma 2.3. *Let K be any splitting set. There exists some component Q such that $\mathrm{cost}(Q) \leq w(B^Q)/N$, where B^Q is a maximum weight basis of M_Q^K.*

For a given Q, a maximum weight basis of M_Q^K can be found via a greedy approach; all that is needed is an independence oracle. This we can obtain immediately from our understanding of the rank function of M_Q^K; it can be computed using submodular function minimization (see (4) in the next section). However, while polynomial time, this is quite slow. We can instead exploit the result that M_Q^K is a gammoid, giving a much faster independence oracle based on solving a maximum flow problem; this is discussed in Appendix A.

We are now ready to describe precisely our deterministic algorithm, given in Algorithm 1. In the algorithm, at each stage we choose a component Q and contract it (in the usual sense, yielding an instance with a smaller vertex set). Thus at intermediate stages of the algorithm, $\mathcal{X}$ will be a feasible blowup graph of some contraction of the original graph G. We also emphasize that the witness sets $W(e)$, and hence also the weights $w(e)$, depend on the blowup graph in the particular iteration.

We now define, for any blowup graph $\mathcal{X}$ and splitting set K, a potential function $\Phi_K(\mathcal{X})$ by

$$\Phi_K(\mathcal{X}) := \sum_{e \in E(\mathcal{X})} c(e) H(|W(e)|),$$

where $H(\ell) := 1 + 1/2 + \cdots + 1/\ell$ is the harmonic function.

Algorithm 1: A deterministic algorithm for Steiner tree demonstrating a $\ln(4)$ integrality gap.

```
Input: Graph $G$ with edge costs $c$ and terminal set $R$, feasible blowup graph $\mathcal{X}$, and splitting set $K$.
Result: A Steiner tree $T$.
$T \leftarrow \emptyset$.
while $T$ is not a Steiner tree do
    Find a component $Q \in \Gamma(\mathcal{X})$ and maximum weight basis $B \in \mathcal{B}_Q^K$ with $\mathrm{cost}(Q) \leq w(B)/N$.
    Cleanup: Let $F = \{e \notin K \mid W(e) \subseteq B\}$.
    Update: $T \leftarrow T \cup Q, \quad \mathcal{X} \leftarrow (\mathcal{X} - B - F)/Q, \quad K \leftarrow K \setminus B$.
end
```

Theorem 2.4. *For any minimal splitting set K and feasible blowup graph $\mathcal{X}$, Algorithm 1 yields a solution of cost at most $\Phi_K(\mathcal{X})/N$.*

The proof of this theorem (given in Appendix B) essentially boils down to showing that in a single step of the algorithm, the expected cost of the contracted component is no larger than the decrease in the potential function scaled down by $1/N$. Let $\mathcal{X}_t$ and K_t be the blowup graph and splitting set at iteration t of the algorithm, with B_t the selected removal set. We are able to show that $\Phi_{K_t}(\mathcal{X}_t) - \Phi_{K_{t+1}}(\mathcal{X}_{t+1}) \geq w(B_t)$, from which the theorem immediately follows.

From this, we can use an averaging argument to show the $\ln(4)$ integrality gap bound. Essentially, if K is chosen randomly from the matroid of possible minimal splitting sets according to an appropriate distribution, it can be shown that

$$\mathbb{E}\{\Phi_K(\mathcal{X})\} \leq \ln(4) \cdot \mathrm{cost}(\mathcal{X}).$$

It is also possible to minimize $\Phi_K(\mathcal{X})$ as a function of K, via a dynamic program. The full proof can be found in the appendix: altogether we obtain, recalling $\mathrm{cost}(\mathcal{X}) = N \cdot \mathrm{cost}(x)$,

Theorem 2.5. *For any solution x of* (LP)*, and choosing K to minimize $\Phi_K(\mathcal{X})$, Algorithm 1 returns a solution of cost at most* $\ln(4) \cdot \mathrm{cost}(x)$.

We emphasize again that while we have described everything in terms of the blowup graph, it is possible to implement Algorithm 1 directly in terms of the LP solution, yielding a polynomial time algorithm. Details will be provided in the full version.

2.4 Quasi-bipartite graphs

The situation is much simplified in the case of quasi-bipartite graphs. In this case, we may choose K to consist of all edges except for the cheapest in each component. This clearly minimizes $\Phi_K(\mathcal{X})$, and it can be shown that

Lemma 2.6. *Let $K = E(\mathcal{X}) \setminus E_{min}$, where E_{min} consists of a cheapest edge from every component. Then*

$$\Phi_K(\mathcal{X}) \leq \tfrac{73}{60} \cdot \mathrm{cost}(\mathcal{X}).$$

A $73/60 < 1.217$ bound on the integrality gap immediately follows from Theorem 2.4. One of the major drawbacks of relying on (LP), or any of the hypergraphic LPs, is that solving them is computational intensive; in general, to obtain a $1+\epsilon$ approximation, nothing better than $n^{2^{\Omega(1/\epsilon)}}$ time is known. This can be improved somewhat to $n^{\Omega(1/\epsilon)}$ in quasi-bipartite graphs, but this is still very slow. We show how to sidestep this issue and obtain a reasonable running time for quasi-bipartite graphs by instead solving the much more tractable bidirected cut relaxation, which has only $O(n^2)$ variables. Combined with the fact that we do not need to re-solve the LP in each iteration, we obtain a markedly faster algorithm than the one of Byrka et al. [3].

More precisely, we show how a solution to the bidirected cut relaxation can be transformed into a solution to (LP) with the same cost, via a natural greedy procedure. One step of the transformation consists of taking, from a star centered around a Steiner vertex, all arcs with incoming flow and one arc with outgoing flow. This yields one component for (LP); the capacities are then uniformly reduced on these edges and the process is continued. The details of this are given in Appendix D. Previously, [6] showed that the bidirected cut relaxation always has the same objective value as the hypergraphic relaxations, suggesting that such a transformation should exist, but the question remained open.

3 Deeper into the matroid structure

In this section, we discuss in more detail the heart of our arguments; uncovering the matroid structure of edge removals, and showing appropriate uniform removal probabilities after the random contraction of a component.

In what follows, we will often need to refer to the terminal set of a component C, so we will again abuse notation and write, e.g., $|C|$ for the number of terminals in C. Define $h_{\mathcal{X}} : 2^R \to \mathbb{N}$ by

$$h_{\mathcal{X}}(S) = N(|S|-1) - \sum_{C \in \Gamma(\mathcal{X})} (|S \cap C| - 1)^+. \tag{1}$$

It is immediate from (LP) that $\mathcal{X}$ is feasible if and only if

$$h_{\mathcal{X}}(S) \geq 0 \quad \forall S \subseteq R, S \neq \emptyset \qquad \text{and} \qquad h_{\mathcal{X}}(R) = 0. \tag{2}$$

Indeed, $h_{\mathcal{X}}(S)$ is, up to scaling, simply the *slack* (or if negative, violation) of the corresponding constraint in (LP). Two important properties of $h_{\mathcal{X}}$ are the following:

Lemma 3.1. *For any blowup graph $\mathcal{X}$,*

i) $h_{\mathcal{X}}$ is intersecting submodular, i.e., for any two sets $S_1, S_2 \subseteq E(\mathcal{X})$ with $S_1 \cap S_2 \neq \emptyset$,

$$h_{\mathcal{X}}(S_1 \cup S_2) + h_{\mathcal{X}}(S_1 \cap S_2) \leq h_{\mathcal{X}}(S_1) + h_{\mathcal{X}}(S_2), \quad \textit{and}$$

ii) for any $F \subseteq E(\mathcal{X})$ and $\emptyset \neq S \subseteq R$, $h_{\mathcal{X}}(S) \leq h_{\mathcal{X}-F}(S) \leq h_{\mathcal{X}}(S) + |F|$.

Proof. i) This follows immediately from the fact that for any $C \subseteq R$, the function $S \to (|S \cap C| - 1)^+$ is intersecting supermodular.

ii) The removal of any additional edge $e \in E(\mathcal{X})$ from $\mathcal{X}$ leads to a split of some component C of $\mathcal{X}$ into subcomponents C_1, C_2 with $C_1 \cap C_2 = \emptyset$, $C_1 \cup C_2 = C$. Hence,

$$h_{\mathcal{X}-e}(S) - h_{\mathcal{X}}(S) = (|S \cap C| - 1)^+ - (|S \cap C_1| - 1)^+ - (|S \cap C_2| - 1)^+ \in \{0, 1\},$$

which leads to $h_{\mathcal{X}}(S) \leq h_{\mathcal{X}-e}(S) \leq h_{\mathcal{X}}(S) + 1$. Applying this repeatedly yields the claim. □

An interesting consequence, that essentially follows by intersecting submodularity of $h_{\mathcal{X}}$ and standard uncrossing techniques, is that any basic feasible solution to (LP) has a support of size bounded by $|R| - 1$ (see, e.g., [10] for an example of this reasoning). For an equivalent version of (LP), this result was already obtained through a rather involved technique by Chakrabarty et al. [6].

For convenience, define

$$h_{\bar{F}}(S) := h_{\mathcal{X}-F}(S) = N(|S|-1) - \sum_{C \in \Gamma(\mathcal{X}-F)} (|S \cap C| - 1)^+.$$

The following lemma describes feasibility of $(\mathcal{X} \circledast Q) - F$ in a convenient form, and also shows that we need only consider constraints corresponding to subsets containing Q.

Lemma 3.2. *The blowup graph $(\mathcal{X} \circledast Q) - F$ is feasible if and only if $h_{\bar{F}}(R) = N(|Q|-1)$ and $h_{\bar{F}}(S) \geq N(|Q|-1)$ for all $S \supseteq Q$.*

Proof. Let $\mathcal{X}' = (\mathcal{X} \circledast Q) - F$. Then $\mathcal{X}'$ is feasible iff $h_{\mathcal{X}'}(S) \geq 0$ for all $S \subseteq R$, $S \neq \emptyset$, with equality for $S = R$. But

$$h_{\mathcal{X}'}(S) = h_{\bar{F}}(S) - N(|S \cap Q| - 1)^+, \tag{3}$$

and so this can be equivalently stated as $h_{\bar{F}}(S) \geq N(|S \cap Q|-1)^+$ for all $S \neq \emptyset$, and $h_{\bar{F}}(R) = N(|Q|-1)$.

All that needs to be proved then is that only the constraints for $S \supseteq Q$ need to be considered. So suppose S is a violated set: $h_{\mathcal{X}'}(S) < 0$. Then $S \cap Q \neq \emptyset$, otherwise $h_{\mathcal{X}'}(S) = h_{\bar{F}}(S) \geq h_{\mathcal{X}}(S) \geq 0$ by feasibility of $\mathcal{X}$. But for any such S,

$$N(|S|-1) - N(|S \cap Q|-1)^+ \;=\; N(|S \cup Q|-1) - N(|Q|-1)^+$$

and clearly
$$\sum_{C \in \Gamma(\mathcal{X}-F)} (|S \cap C| - 1)^+ \;\leq\; \sum_{C \in \Gamma(\mathcal{X}-F)} (|(S \cup Q) \cap C| - 1)^+.$$

Subtracting and using (3), we obtain that $h_{\mathcal{X}'}(S \cup Q) \leq h_{\mathcal{X}'}(S)$. Since S was a violating set, so is $S \cup Q$. □

Let $E(\mathcal{X})$ be any subset of $E(\mathcal{X})$, and define $r_Q : E(\mathcal{X}) \to \mathbb{N}$ by

$$r_Q(F) = \min_{S \supseteq Q} h_{\bar{F}}(S). \tag{4}$$

We will show:

Proposition 3.3. *The function r_Q is the rank function of a matroid of rank $N(|Q|-1)$.*

Once we have this, it is straightforward to show that this matroid precisely describes the minimal edge removals:

Theorem 3.4. *The set of bases of the matroid defined by r_Q is precisely $\mathcal{B}_Q$.*

Proof. Let $\mathcal{B}'_Q$ be the set of bases of the matroid defined by r_Q, and consider any $B \in \mathcal{B}'_Q$. By the definition of r_Q, we have that

$$h_{\bar{B}}(S) \geq r_Q(B) = N(|Q|-1) \qquad \text{for any } S \supseteq Q.$$

Moreover, by Lemma 3.1 (ii),

$$h_{\bar{B}}(R) \leq h_{\mathcal{X}}(R) + |B| = N(|Q|-1);$$

the final equality follows since $|B| = r_Q(B) = N(|Q|-1)$ and $h_{\mathcal{X}}(R) = 0$ by feasibility of $\mathcal{X}$. Thus by Lemma 3.2, $(\mathcal{X} \circledast Q) - B$ is feasible.

Conversely, consider any $B \in \mathcal{B}_Q$. By feasibility and Lemma 3.2 again, $h_{\bar{B}}(S) \geq N(|Q|-1)$ for all $S \supseteq Q$, with equality for $S = R$. Thus $r_Q(B) = N(|Q|-1)$, and so there is some $B' \subseteq B$ with $B' \in \mathcal{B}'_Q$. But then B' is also a feasible removal set by the above, and so by minimality $B' = B$. □

Proof of Proposition 3.3. First, observe from (1) applied to the empty blowup graph that

$$r_Q(E(\mathcal{X})) = \min_{S \supseteq Q} N(|S|-1) = N(|Q|-1).$$

We must show that r_Q is increasing, submodular, and satisfies $r_Q(F) \leq |F|$ for all $F \subseteq E(\mathcal{X})$. The fact that r_Q is increasing follows immediately from the definitions of r_Q and $h_{\bar{F}}$; removing a larger set can only increase the slack. Considering some fixed $S \supseteq Q$, we have by Lemma 3.1 (ii) that $h_{\mathcal{X}-F}(S) \leq h_{\mathcal{X}}(S) + |F|$. Thus $r_Q(F) \leq r_Q(\emptyset) + |F| = |F|$ since $r_Q(\emptyset) = 0$ by feasibility of $\mathcal{X}$.

Now we come to the main part of the proof, showing that r_Q is submodular. We must show that for any $F_1 \subseteq F_2 \subseteq E(\mathcal{X})$ and $e \notin F_2$,

$$r_Q(F_1 + e) - r_Q(F_1) \geq r_Q(F_2 + e) - r_Q(F_2). \tag{5}$$

It is clearly sufficient to show this for F_1 and F_2 differing by a single edge. Consider any $S \supseteq Q$ and $i \in \{1, 2\}$. The difference

$$h_{\overline{F_i + e}}(S) - h_{\bar{F_i}}(S) = \sum_{C \in \Gamma(\mathcal{X} - F_i)} (|C \cap S| - 1)^+ \quad - \sum_{C \in \Gamma(\mathcal{X} - (F_i + e))} (|C \cap S| - 1)^+$$

is one or zero, and it is one precisely if e splits up some component $C \in \Gamma(\mathcal{X} - F_i)$ into two components C_1, C_2 that both intersect S. If this is the case for some component in $\Gamma(\mathcal{X} - F_2)$, then e will also split up some component in $\Gamma(\mathcal{X} - F_1)$ into two pieces both intersecting S, since $\mathcal{X} - F_2$ is a subgraph of $\mathcal{X} - F_1$. Thus for any $S \supseteq Q$,

$$h_{\overline{F_2 + e}}(S) - h_{\bar{F_2}}(S) \leq h_{\overline{F_1 + e}}(S) - h_{\bar{F_1}}(S). \tag{6}$$

It also follows that for any $S, S' \subseteq R$ with $Q \subseteq S \subseteq S'$,

$$h_{\overline{F_1 + e}}(S) - h_{\bar{F_1}}(S) \leq h_{\overline{F_1 + e}}(S') - h_{\bar{F_1}}(S'). \tag{7}$$

Let $\mathcal{S}_i$ be the set of terminal subsets containing Q that minimize $h_{\bar{F_i}}(S)$, over all $S \supseteq Q$. Since $h_{\bar{F_1}}$ is intersecting submodular by Lemma 3.1, there is a unique *maximal* set $S_1^* \in \mathcal{S}_1$, meaning $S_1^* \supseteq S$ for all $S \in \mathcal{S}_1$. Similarly, there is a unique *minimal* set $S_2^* \in \mathcal{S}_2$; so $S_2^* \subseteq S$ for all $S \in \mathcal{S}_2$. We first show $S_2^* \subseteq S_1^*$. Notice that for $S \supseteq Q$ with $S \notin \mathcal{S}_1$ we have $h_{\bar{F_2}}(S) \geq h_{\bar{F_1}}(S) \geq h_{\bar{F_1}}(S_1^*) + 1$, where the first inequality follows by Lemma 3.1 (ii). Furthermore, $h_{\bar{F_2}}(S_1^*) \leq h_{\bar{F_1}}(S_1^*) + 1$, again by Lemma 3.1 (ii). Hence $h_{\bar{F_2}}(S_1^*) \leq h_{\bar{F_2}}(S)\ \forall S \supseteq Q, S \notin \mathcal{S}_1$, and thus $\mathcal{S}_1$ must contain some minimizers of $h_{\bar{F_2}}$, i.e., $\mathcal{S}_1 \cap \mathcal{S}_2 \neq \emptyset$. Since S_2^* is the minimal set in $\mathcal{S}_2$ and S_1^* is the maximal set in $\mathcal{S}_1$ we obtain $S_2^* \subseteq S_1^*$.

We finally have

$$\begin{aligned} r_Q(F_2 + e) - r_Q(F_2) &= h_{\overline{F_2 + e}}(S_2^*) - h_{\bar{F_2}}(S_2^*) \\ &\leq h_{\overline{F_1 + e}}(S_2^*) - h_{\bar{F_1}}(S_2^*) \qquad \text{by (6)} \\ &\leq h_{\overline{F_1 + e}}(S_1^*) - h_{\bar{F_1}}(S_1^*) \qquad \text{by (7), since } S_2^* \subseteq S_1^* \\ &= r_Q(F_1 + e) - r_Q(F_1). \end{aligned}$$

□

Proof of Theorem 2.2

We first show that for any component Q,

$$r_Q(K) = N(|Q| - 1).$$

This in turn implies that $\mathcal{B}_Q = \{B \in \mathcal{B}_Q \mid B \subseteq K\}$ is nonempty, since the rank of M_Q is $N(|Q| - 1)$ by Proposition 3.3, and so $B_{rem} \neq \emptyset$. Notice that for $S \subseteq R, S \neq \emptyset$, we have $h_K(S) = N(|S| - 1)$, since in $\mathcal{X} - K$ all components contain precisely one terminal. Hence,

$$r_Q(K) = \min_{S \supseteq Q} h_K(S) = N(|Q| - 1).$$

As already discussed in Section 2, the polytope B_{rem} is simply a weighted Minkowski sum of the base polytopes $B(M_Q)$ for $Q \in \Gamma(\mathcal{X})$. It is well known that the Minkowski sum of matroid polytopes is a polymatroid, and moreover, the rank function of the sum is simply the sum of the rank functions of the summands [14]. Thus, B_{rem} is the base polytope of a polymatroid with rank function

$$r = \frac{1}{|\Gamma(\mathcal{X})|} \sum_{Q \in \Gamma(\mathcal{X})} r_Q. \tag{8}$$

To show that the point p given by $p_e = N/|\Gamma(\mathcal{X})|$ for all $e \in K$ is in B_{rem}, we need to show that $r(F) \geq |F| \cdot N/|\Gamma(\mathcal{X})|$ for every $F \subseteq K$. Expanding out (8) and the definition of r_Q, and writing S_Q for the subset $S \supseteq Q$ that attains the minimum in (4), we obtain

$$r(F) = \frac{1}{|\Gamma(\mathcal{X})|} \sum_{Q \in \Gamma(\mathcal{X})} h_{\bar{F}}(S_Q).$$

We now observe that because K is a splitting set, $h_{\bar{F}}(R) = |F|$. For imagine removing the edges of F from $\mathcal{X}$ one by one; $h_{\mathcal{X}-F}(R) - h_{\mathcal{X}}(R)$ just counts the number of times where a component is split by the deleted edge in this process. But by the nature of minimal splitting sets, this must happen at *every* step—no pendant edges are formed at any stage. Hence $h_{\bar{F}}(R) - h_{\mathcal{X}}(R) = |F|$; moreover, $h_{\mathcal{X}}(R) = 0$ by feasibility, so indeed $h_{\bar{F}}(R) = |F|$. Thus to finish the proof, it suffices to show

Claim 3.5. $\sum_{Q \in \Gamma(\mathcal{X})} h_{\bar{F}}(S_Q) \geq N \cdot h_{\bar{F}}(R)$.

To prove Claim 3.5, we replace the function $h_{\bar{F}}$ on the left-hand side of the inequality by a function f that lower bounds $h_{\bar{F}}$ and is well structured. More precisely, f is chosen to be a conic combination of a special type of intersecting submodular functions which we call *partition functions*: for any partition $\mathcal{P} = \{P_1, \ldots, P_n\}$ of R, the corresponding partition function $f_{\mathcal{P}}$ is given by

$$f_{\mathcal{P}}(S) = (|\{j \in [n] \mid P_j \cap S \neq \emptyset\}| - 1)^+ \quad \forall S \subseteq R.$$

The following theorem (whose proof can be found in Appendix C) guarantees the existence of the function f that we need to prove Claim 3.5.

Theorem 3.6. *Let $h : 2^U \to \mathbb{R}_+$ any nonnegative intersecting submodular function with $h(\{v\}) = 0$ for all $v \in U$. Then there is a monotone intersecting submodular function f of the form*

$$f = \sum_{i=1}^{k} \lambda_i f_{\mathcal{P}^i},$$

for some $k \in \mathbb{N}$, where $\lambda_i > 0$ and $\mathcal{P}^i$ is a partition of U for each $1 \leq i \leq k$, satisfying:
i) $f(S) \leq h(S)$ for all $S \subseteq U$, and
ii) $f(U) = h(U)$.

Consider the function $h^+_{\bar{F}}$ defined by $h^+_{\bar{F}}(S) = \max\{h_{\bar{F}}(S), 0\}$. Then $h^+_{\bar{F}}$ differs from $h_{\bar{F}}$ only on the empty set, since $h_{\bar{F}}(S) \geq 0$ for all $S \neq \emptyset$. Thus $h^+_{\bar{F}}$ is still intersecting submodular, and also nonnegative. Let $f = \sum_{i=1}^{k} \lambda_i f_{\mathcal{P}^i}$ be the function obtained by applying Theorem 3.6 to $h^+_{\bar{F}}$. We then have

$$\begin{aligned} \sum_{Q \in \Gamma(\mathcal{X})} h_{\bar{F}}(S_Q) &\geq \sum_{Q \in \Gamma(\mathcal{X})} f(S_Q) \geq \sum_{Q \in \Gamma(\mathcal{X})} f(Q) \\ &= \sum_{Q \in \Gamma(\mathcal{X})} \sum_{i=1}^{k} \lambda_i f_{\mathcal{P}_i}(Q) = \sum_{i=1}^{k} \lambda_i \sum_{Q \in \Gamma(\mathcal{X})} f_{\mathcal{P}_i}(Q), \end{aligned} \tag{9}$$

where the first inequality holds since $h_{\bar{F}}(S) = h^+_{\bar{F}}(S) \geq f(S)$ for all $S \neq \emptyset$, and the second inequality holds since f is monotone and $Q \subseteq S_Q$.

As observed by Chakrabarty et al. [6], any solution x to (LP) satisfies the following partition constraints for any partition $\mathcal{P}$ of R:

$$\sum_{C \in \mathcal{K}} x_C f_{\mathcal{P}}(C) \geq |\mathcal{P}| - 1,$$

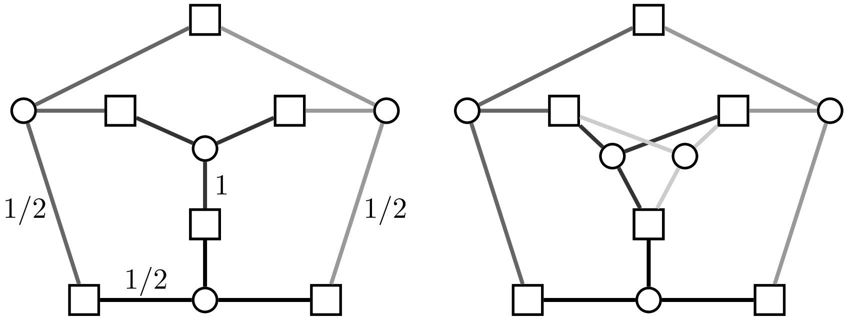
where $|\mathcal{P}|$ is the number of sets in partition $\mathcal{P}$. In our blown-up setting this translates into

$$\sum_{Q \in \Gamma(\mathcal{X})} f_{\mathcal{P}}(Q) \geq N(|\mathcal{P}| - 1).$$

Combining this observation with (9) and using $|\mathcal{P}_i| - 1 = f_{\mathcal{P}_i}(R)$, Claim 3.5 follows since

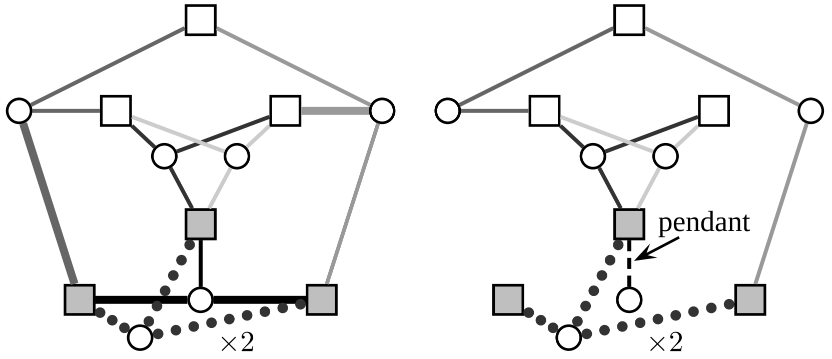
$$\begin{aligned} \sum_{Q \in \Gamma(\mathcal{X})} h_{\bar{F}}(S_Q) &\geq \sum_{i=1}^{k} \lambda_i \sum_{Q \in \Gamma(\mathcal{X})} f_{\mathcal{P}_i}(Q) \geq \sum_{i=1}^{k} \lambda_i N(|\mathcal{P}_i| - 1) \\ &= N \cdot \sum_{i=1}^{k} \lambda_i f_{\mathcal{P}_i}(R) = N \cdot f(R) = N \cdot h_{\bar{F}}(R), \end{aligned}$$

where the last equality follows from property (ii) of Theorem 3.6.



(a) fractional solution x (b) blowup graph $\mathcal{X}$

Figure 1: In (a): fractional solution x (components drawn in different gray scales and labelled with their capacity x_C). In (b): blowup graph $\mathcal{X}$ for $N = 2$.



(a) blowup graph $\mathcal{X} \circledast Q$ (b) blowup graph $\mathcal{X} \circledast Q - B$ after removal

Figure 2: In (a): $\mathcal{X} \circledast Q$, edges in $B \subseteq E(\mathcal{X})$ in bold, copies of Q are dotted, terminals in Q are filled gray. In (b): feasible blowup graph $\mathcal{X} \circledast Q - B$ (which is not minimal due to the pendant edge that may also be removed).

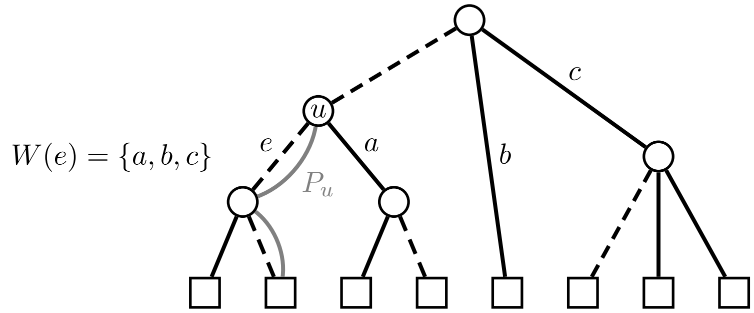


Figure 3: Illustration of the definition of $W(e)$. Depicted is some component C with core edges K (solid) and cleanup edges (dashed).

References

[1] M. Bern and P. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

[2] A. Borchers and D.-Z. Du. The k-Steiner ratio in graphs. *SIAM Journal on Computing*, 26(3):857–869, June 1997.

[3] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*. To appear.

[4] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. An improved LP-based approximation for Steiner Tree. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 583–592, 2010.

[5] D. Chakrabarty, N. R. Devanur, and V. V. Vazirani. New geometry-inspired relaxations and algorithms for the metric Steiner tree problem. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 344–358, 2008.

[6] D. Chakrabarty, J. Könemann, and D. Pritchard. Hypergraphic LP relaxations for Steiner trees. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*. 2010.

[7] D. Chakrabarty, J. Könemann, and D. Pritchard. Integrality gap of the hypergraphic relaxation of Steiner trees: A short proof of a 1.55 upper bound. *Operations Research Letters*, 38(6):567 – 570, 2010.

[8] M. Chlebík and J. Chlebíková. The Steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214, 2008.

[9] E. N. Gilbert and H. O. Pollak. Steiner minimal trees. *SIAM Journal on Applied Mathematics*, 16(1):1–29, 1968.

[10] M. X. Goemans. Minimum bounded degree spanning trees. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 273–282, 2006.

[11] M. X. Goemans and Y. Myung. A catalog of Steiner tree formulations. *Networks*, 23(1):19–28, 1993.

[12] M. Karpinski and A. Zelikovsky. New approximation algorithms for the Steiner tree problem. *Journal of Combinatorial Optimization*, 1(1):47–65, 1997.

[13] J. Könemann, D. Pritchard, and K. Tan. A partition-based relaxation for Steiner trees. *Math. Program.*, 127(2):345–370, 2011.

[14] C.J.H. McDiarmid. Rado's theorem for polymatroids. *Mathematical Proceedings of the Cambridge Philosophical Society*, 78:263–281, 1975.

[15] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983.

[16] M. Padberg and L.A. Wolsey. Trees and cuts. In *Combinatorial Mathamtics (Proceedings International Colloquium on Graph Theory and Combinatorics)*, pages 511–517, 1983.

[17] J.-C. Picard and M. Queyranne. Selected applications of minimum cuts in networks. *INFOR Canadian Journal of Operational Research and Information Processing*, 20:294–370, 1982.

[18] T. Polzin and S. Vahdati-Daneshmand. On Steiner trees and minimum spanning trees in hypergraphs. *Operations Research Letters*, 31(1):12–20, 2003.

[19] H. J. Prömel and A. Steger. A new approximation algorithm for the Steiner tree problem with performance ratio 5/3. *Journal of Algorithms*, 36:89–101, 2000.

[20] S. Rajagopalan and V. V. Vazirani. On the bidirected cut relaxation for the metric Steiner tree problem. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 742–751, 1999.

[21] G. Robins and A. Zelikovsky. Tighter bounds for graph steiner tree approximation. *SIAM Journal on Discrete Mathematics*, 19(1):122–134, 2005.

[22] A. Schrijver. *Combinatorial Optimization, Polyhedra and Efficiency*. Springer, 2003.

[23] D. Warme. *Spanning Trees in Hypergraphs with Applications to Steiner Trees*. PhD thesis, 1998.

[24] A. Zelikovsky. An $11/6$-approximation algorithm for the network Steiner problem. *Algorithmica*, 9:463–470, 1993.

A Separation and gammoid structure

In this section, we investigate the separation problem for (LP). Although it is not necessary, for convenience we will work in the blown up formulation; thus, for a given $\mathcal{X}$, our goal is to determine whether (2) is satisfied (see Section 3 for details of this and the definition of $h_{\mathcal{X}}$). In fact, we will do more; for any $Q \subseteq R$, $Q \neq \emptyset$, we will find the most violated set over all $S \supseteq Q$. Given this, we can answer the separation question by checking that $\min_{S \supseteq \{v\}} h_{\mathcal{X}}(S)$ is zero for each choice of v (note that $h_{\mathcal{X}}(\emptyset) = -N$, and so we must exclude this trivial set from consideration). For each choice of v, one max-flow calculation will be required.

The construction is inspired by one for the forest polytope [17, 16] (see also [22, §51.4]). While what follows is not precisely a generalization (in the case where all components have size 2, the resulting construction is slightly different), it is similar in spirit. In the *directed* component-based relaxation, separation via an equivalent flow-based formulation is completely straightforward. However this does not imply such a formulation for the undirected version.

Since $h_{\mathcal{X}}$ is an intersecting submodular function, it follows already that $\min_{S \supseteq Q} h_{\mathcal{X}}(S)$ can be computed in polynomial time [22], using submodular function minimization as a black box. However, the combinatorial algorithm we demonstrate here, which reduces the separation problem to a max-flow calculation, gives some additional insights (as well as being more efficient).

Let $\mathcal{X}$ be the blowup graph of some solution x. First, let

$$y_v = |\{C \in \Gamma(\mathcal{X}) : v \in C\}| - N \quad \text{for } v \in R.$$

If y_v is negative for any v, it is easily seen that $\mathcal{X}$ is not feasible (it corresponds to $x(\delta(v)) < 1$). So from now on, we assume $y_v \geq 0$ for all $v \in R$. Construct a directed multigraph $D = (W, A)$ (we will write $D_{\mathcal{X}}$ if we wish to be explicit on the choice of $\mathcal{X}$) as follows. Begin with the multigraph $\mathcal{X}$, and for each component $C \in \Gamma(\mathcal{X})$, pick an arbitrary node r_C as the root. Adjoin a source node s and sink node t. Now orient all edges of $E(C)$ away from r_C for each component, and adjoin the arcs sr_C for each $C \in \Gamma(\mathcal{X})$, and vt for each $v \in R$. We assign capacities z to the arcs; $z(vt) = y_v$ for all $v \in R$, and $z(a) = 1$ for all other arcs.

Theorem A.1. *For any nonempty $Q \subseteq R$, the value of the maximum s-$(Q \cup \{t\})$-flow in D is equal to $y(R) + N + \min_{S \supseteq Q} h_{\mathcal{X}}(S)$. More specifically, if U^* is a minimum $(Q \cup \{t\})$-s cut in D with $s \notin U^*$, then $S^* = U^* \cap R$ minimizes $h_{\mathcal{X}}(S)$ over $S \supseteq Q$, and*

$$z(\delta^-(U^*)) = y(R) + N + h_{\mathcal{X}}(S^*).$$

Proof. Use $\vec{E}(C)$ to denote the arcs in D corresponding to $E(C)$ in $\mathcal{X}$, and let $A(C) = \vec{E}(C) \cup \{sr_C\}$. For any $S \subseteq R$, let $\nu(S) \subseteq W$ be defined by

$$\nu(S) = S \cup \{t\} \ \cup \bigcup_{\substack{C \in \Gamma(\mathcal{X}):\\ C \cap S \neq \emptyset}} (V(C) \setminus R).$$

Claim A.2. *For any nonempty $S \subseteq R$, $z(\delta^-(\nu(S))) = h_{\mathcal{X}}(S) + y(R) + N$. Moreover, for any $U \subset W$ with $s \notin U$, $t \in U$ and $U \cap R = S$, we have $z(\delta^-(U)) \geq h_{\mathcal{X}}(S) + y(R) + N$.*

Proof. Consider some component $C \in \Gamma(\mathcal{X})$. If $S \cap C = \emptyset$, then clearly $\delta^-(\nu(S)) \cap A(C) = \emptyset$. So suppose $S \cap C \neq \emptyset$. If $r_C \in \nu(S)$, then clearly $\delta^-(\nu(S)) \cap \vec{E}(C) = \emptyset$, and $sr_C \in \delta^-(\nu(S))$. On the other hand, if $r_C \notin \nu(S)$ (implying in particular that r_C is a terminal), then $sr_C \notin \delta^-(\nu(S))$ and $|\delta^-(\nu(S)) \cap \vec{E}(C)| = 1$, since all terminals are leaves of the components they belong to. In either case, $z(\delta^-(\nu(S)) \cap A(C)) = 1$.

For any $v \in R$, $sv \in \delta^-(\nu(S))$ if and only if $v \notin S$. Putting this all together,

$$z(\delta^-(\nu(S))) = |\{C \in \Gamma(\mathcal{X}) : C \cap S \neq \emptyset\}| + \sum_{v \notin S} y_v.$$

Now taking (1) and adding and subtracting $y(S)$, we have

$$\begin{aligned} h_{\mathcal{X}}(S) &= N(|S|-1) - \sum_{C\in\Gamma(\mathcal{X})} (|C\cap S|-1)^+ + \left(\sum_{C\in\Gamma(\mathcal{X})} |C\cap S| \ - \ N|S|\right) - y(S) \\ &= |\{C \in \Gamma(\mathcal{X}) : C\cap S \neq \emptyset\}| - N - y(S) \\ &= z(\delta^-(\nu(S)) - N - y(R). \end{aligned}$$

Now consider any U with $t \in U$, $s \notin U$ and $U \cap R = S$. We again clearly have $sv \in \delta^-(U)$ for all $v \in R \setminus U$, and again $\delta^-(U) \cap A(C) \neq \emptyset$ if $S \cap C \neq \emptyset$. So $z(\delta^-(U)) = h_{\mathcal{X}}(S) + y(R) + N$. □

By this claim, $z(\delta^-(\nu(S^*)) \leq z(\delta^-(U^*))$; since U^* is a minimum cut, we must have equality. Then again by the claim,

$$z(\delta^-(\nu(S^*))) = h_{\mathcal{X}}(S^*) + y(R) + N.$$ □

We now show how this leads to a description of the matroid M_Q as a gammoid. Recall the definition of a gammoid: a directed graph H is given, along with two subsets $X, Y \subseteq V(H)$. The groundset of the gammoid is X, and a set $I \subseteq X$ is independent if there are vertex-disjoint paths from I to some subset of Y. We say in this case that this defines the gammoid *from X to Y* in H. It is convenient to observe that by transforming the digraph H appropriately, we can replace vertex-disjoint in the above definition with arc-disjoint, and still characterize gammoids.

We need to slightly tweak the digraph D defined above. For each $f \in E(\mathcal{X})$, there is a corresponding arc a in D. Split the arc by adding an additional node v_f, producing a "front" arc a_f^{f} with tail v_f and a "back" arc a_f^{b} with head v_f. We may also remove the node s and all its adjacent arcs. Call the resulting modified digraph D'.

Define the sets

$$X = \{v_f \mid f \in E(\mathcal{X})\}; \qquad X' = \bigcup_{C\in\Gamma(\mathcal{X})} r_C \qquad \text{and} \qquad Y = Q \cup \{t\}.$$

Let $\mathfrak{G}'_Q$ be the gammoid defined on D from $X' \cup X$ to Y, requiring arc-disjointness rather than vertex-disjointness. Then define $\mathfrak{G}_Q = \mathfrak{G}'_Q / X'$; this contraction is also a gammoid. By the one-to-one correspondence between X and $E(\mathcal{X})$, we may consider this is a matroid over $E(\mathcal{X})$.

Theorem A.3. *For any component Q, $\mathfrak{G}_Q = M_Q$.*

Proof. The rank of a set $U \subseteq X$ in $\mathfrak{G}_Q$ is $\rho(U) = \rho'(U \cup X') - \rho'(X')$, where ρ' is the rank function of $\mathfrak{G}'_Q$. Notice that the maximum number of arc-disjoint paths from X' to Y is precisely the max-flow from s to $Q \cup \{t\}$ in D. Thus by Theorem A.1, and the definition of r_Q, $\rho'(X') = r_Q(\emptyset) + y(R) + N$.

Now $\rho'(U \cup X')$ is the maximum number of arc-disjoint paths from $U \cup X'$ to Y. But imagine what would happen to $\rho'(U \cup X')$ if the arcs $A_U = \{a_f^{\mathsf{b}} \mid v_f \in U\}$ were removed from D'. Take $P_1, \ldots, P_\ell$ to be any maximum collection of arc-disjoint paths from $U \cup X'$ to Y in D'. For some $v_f \in U$, if some path P_i uses arc a_f^{b}, then certainly no other path emanates from v_f, and so we can simply remove the initial segment of P_i before v_f to obtain another maximum collection of disjoint paths that do not use a_f^{b}. Repeating this process, we obtain paths $P'_1, \ldots, P'_\ell$ that do not use any arcs in A_U. But taking $D' - A_U$, and contracting all of U to form the source, yields precisely $D_{\mathcal{X}-F}$, the digraph for the separation construction corresponding to $\mathcal{X} - F$. Thus again by Theorem A.1, $\rho'(U \cup X') = r_Q(U) + y(R) + N$. Thus $\rho(U) = r_Q(U)$, and so $\mathfrak{G}_Q = M_Q$. □

B Proofs for Section 2.3

Let us fix a component $C \in \Gamma(\mathcal{X})$ and a splitting set K. By the definition of K (as the complement of a spanning tree in the graph $\tilde{C}$ obtained by contracting terminals), every Steiner node $u \in V(C) \setminus C$ has a unique path $P_u \subseteq E(C) \setminus K$ of cleanup edges to a terminal that we term $r_u \in C$ (see again Figure 3).

Lemma B.1. *For any splitting set $K \subseteq E(\mathcal{X})$, component $C \in \Gamma(\mathcal{X})$ and edge $e \in E(C) \setminus K$, let*

$$W(e) = \{uv \in K \cap E(C) \mid e \in P_u\}.$$

Then $W(e)$ is the unique minimal subset of $E(C) \cap K$ whose removal makes e a pendant edge.

Proof. Let $\bar{W} \subseteq E(C) \cap K$ be any subset of splitting edges. If $uv \in W(e) \setminus \bar{W}$ then e remains on a path, namely $P_u \cup uv \cup P_v \subseteq E(C) \setminus \bar{W}$ between the terminals, implying that e is not pendant. Thus, any subset $\bar{W}$ which makes e pendant must contain $W(e)$.

On the other hand, we claim that e is pendant in $E(C) \setminus W(e)$. To see this, let $e = ab$ with $e \in P_a$. For e to be pendant, there would need to be a path P that does not contain e from a to a terminal. But the first edge in K on P must be in $W(e)$, contradicting the fact that $P \subseteq E(C) \setminus W(e)$. □

Theorem 2.4. For any splitting set K and feasible blowup graph $\mathcal{X}$, Algorithm 1 yields a solution of cost at most $\Phi_K(\mathcal{X})/N$.

Proof. We prove the theorem by showing that the decrease in the potential at any iteration is lower bounded by the weight of the edges we remove. More formally, consider a given iteration t with current blowup graph $\mathcal{X}_t$, splitting set K_t, and weights w_t. let Q_t be the component to contract and $B_t \in \mathcal{B}_Q^{K_t}$ the edges to be removed from $\mathcal{X}_t$ in this iteration. At the end of iteration t a new blowup graph $\mathcal{X}_{t+1}$ is obtained with splitting set $K_{t+1} = K_t \setminus B_t$. We will show

$$\Phi_{K_t}(\mathcal{X}_t) - \Phi_{K_{t+1}}(\mathcal{X}_{t+1}) \geq w_t(B_t). \tag{10}$$

This in turn implies the theorem since the potential function at any iteration, and in particular at the end of the algorithm, is nonnegative. Therefore, the total weight of all core edges being removed throughout the algorithm is upper bounded by the potential value of the initial blowup graph, i.e., $\sum_t w_t(B_t) \leq \Phi_K(\mathcal{X})$. Furthermore, since at every iteration, Q_t and B_t are chosen such that $\text{cost}(Q_t) \leq w_t(B_t)/N$, we obtain that the cost of all contracted components—which is the cost of the Steiner tree our algorithm returns—can be upper bounded by $\sum_t \text{cost}(Q_t) \leq \frac{1}{N} \sum_t w_t(B_t) \leq \Phi_K(\mathcal{X})/N$, as desired. Hence, it remains to prove (10).

For any edge $e \in \mathcal{X}_t$, we denote by $W_t(e)$ its witness set at the beginning of iteration t. For simplicity, we define W_{t+1} on all of $E(\mathcal{X}_t)$, defining $W_{t+1}(e) = \emptyset$ for $e \in E(\mathcal{X}_t) \setminus E(\mathcal{X}_{t+1})$. By definition of the witness sets, we have

$$W_{t+1}(e) = W_t(e) \setminus B_t \qquad \text{for any } e \in E(\mathcal{X}_t). \tag{11}$$

Expanding the left-hand side of (10), we obtain

$$\begin{aligned}
\Phi_{K_t}(\mathcal{X}_t) - \Phi_{K_{t+1}}(\mathcal{X}_{t+1}) &= \sum_{e \in E(\mathcal{X}_t)} c(e)\big(H(|W_t(e)|) - H(|W_{t+1}(e)|)\big) \\
&= \sum_{e \in E(\mathcal{X}_t)} c(e) \sum_{k=|W_{t+1}(e)|+1}^{|W_t(e)|} \frac{1}{k} \\
&\geq \sum_{e \in E(\mathcal{X}_t)} c(e) \cdot \frac{|W_t(e)| - |W_{t+1}(e)|}{|W_t(e)|} \\
&= \sum_{e \in E(\mathcal{X}_t)} c(e) \cdot \frac{|W_t(e) \cap B_t|}{|W_t(e)|} \qquad \text{by (11).}
\end{aligned} \tag{12}$$

Furthermore, by expanding the right-hand side of (10) using the definition of the weights w_t, we obtain

$$\begin{aligned} w_t(B_t) &= \sum_{f \in B_t} \sum_{\substack{e \in E(\mathcal{X}_t) \\ e \in W_t(f)}} \frac{c(e)}{|W_t(e)|} \\ &= \sum_{e \in E(\mathcal{X}_t)} \sum_{f \in B_t \cap W_t(e)} \frac{c(e)}{|W_t(e)|} \\ &= \sum_{e \in E(\mathcal{X}_t)} c(e) \frac{|W_t(e) \cap B_t|}{|W_t(e)|}. \end{aligned} \tag{13}$$

Inequality (10) finally follows by combining (12) with (13). □

In the following, we show that K can always be chosen s.t. $\Phi_K(\mathcal{X}) \leq \ln(4) \cdot \operatorname{cost}(\mathcal{X})$, following the proof of [4]. For the sake of a simpler exposition, we replace every Steiner node in $\mathcal{X}$ of degree higher than 3, with a binary tree consisting of cost zero edges in order to obtain nodes that have degree exactly 3. Suppose we find a suitable pair (K, F) of splitting and cleanup edges in this auxiliary graph. Then every Steiner node u in the original graph has potentially several paths $P_1, \ldots, P_q \subseteq F$ of cleanup edges to terminals. We keep the one path minimizing $c(P_i)$ and discard the first edge of all other paths. This does not increase $\Phi_K(\mathcal{X})$. Applying this iteratively, we end up with a feasible pair of cleanup edges and splitting edges.

From now on, we assume that every component $C \in \Gamma(\mathcal{X})$ is a binary tree. We pick an arbitrary edge $e_C \in E(C)$ as *root edge*. From any interior node $u \in V(C) \setminus C$, there are two outgoing edges (these are the edges that do not lie on the path from u to the root edge). We randomly pick one of these edges as cleanup edge and the other one as splitting edge. In other words, every interior node u has a unique path of cleanup edges to some terminal and hence, K is a legal splitting set. Moreover, for every non-root edge e one has $\mathbb{P}\{e \in K\} = \frac{1}{2}$.

Lemma B.2. *If $E(\mathcal{X})$ is chosen randomly according to the above distribution,*

$$\mathbb{E}\{\Phi_K(\mathcal{X})\} \leq \ln(4) \cdot \operatorname{cost}(\mathcal{X}).$$

Proof. Fix a component C and an edge $e \in E(C)$. It suffices to show that $\mathbb{E}\{H(|W(e)|)\} \leq \ln(4)$. The root edge is always a splitting edge, thus $|W(e_C)| = 1$. So, let e be a non-root edge and let $v_0, v_1, \ldots, v_{k+1}$ be the path from e to the root edge, i.e. $v_0 v_1 = e$ and $v_k v_{k+1} = e_C$. Let

$$X := \max\{i \mid v_0 v_1, v_1 v_2, \ldots, v_{i-1} v_i \in E(C) \setminus K\}$$

be the number of consecutive cleanup edges on this path, starting from e (and $X = 0$ if already $v_0 v_1 \in K$). Then $\mathbb{P}\{X = i\} = (\frac{1}{2})^{i+1}$ for $i < k$ and $\mathbb{P}\{X = k\} = (\frac{1}{2})^k$. Furthermore $|W(e)| = X + 1$ if $X < k$ and $|W(e)| = k$ otherwise. We calculate

$$\begin{aligned} \mathbb{E}\{H(|W(e)|)\} &\leq \sum_{i=0}^{k-1} \mathbb{P}\{X = i\} \cdot H(i+1) + \mathbb{P}\{X = k\} \cdot H(k) \\ &\leq \sum_{i=0}^{\infty} H(i+1) \cdot \left(\frac{1}{2}\right)^{i+1} \\ &= \ln(4). \end{aligned}$$

□

The above argument can be derandomized by the method of conditional expectations, and this leads to a proof of Theorem 2.5. Another option is to observe that the best choice of K can be found in polynomial time, via a dynamic program as is indicated below. Combined with the above lemma, this implies Theorem 2.5.

Lemma B.3. *A splitting set K minimizing $\Phi_K(\mathcal{X})$ can be found in polynomial time.*

Proof. Since the potential function can be decomposed into terms corresponding to each component, and a splitting set K consists of the union of splitting sets in each component, it suffices to consider each component separately. Hence, let C be any fixed component with vertices $V(C)$ and edges $E(C)$; our goal is to find a splitting set K for C that minimizes $\sum_{e\in E(C)} c(e)H(|W(e)|)$.

As usual when applying dynamic programming to problems on trees, we start by computing tables (to be specified soon) for subtrees consisting of a single terminal, and successively combine those tables until a table for the full tree is obtained, revealing the optimal splitting set. To specify the order in which we create tables for larger subtrees from smaller ones, we direct the edge of the tree C away from an arbitrarily chosen node in $V(C)$. We consider the following type of subtrees that we call *partial trees*. For any vertex $r \in V(C)$ and subset $U \subseteq \delta^+(r)$ of arcs leaving r, the *partial tree T_U with root r* is the induced subgraph of C consisting of r and all vertices that can be reached from r with paths starting with one of the arcs in U. To simplify notation we also use T_U to refer to the edge set of the partial tree. Furthermore, let $\overline{T}_U = E(C) \setminus T_U$, and let $R_{T_U} \subseteq R$ denote the terminals contained in the partial tree T_U.

To better understand what information should be stored for a partial tree T, we first briefly discuss how the choice of splitting set K within T impacts the witness sets in $\overline{T}$, and vice versa. We will refer to the choice of core and cleanup edges (i.e., the choice of K) within some subset of edges as a *configuration* for that subset. We distinguish two ways that the root r of T can be connected to a terminal through cleanup edges: *case (A)* through a path within the partial tree T, and *case (B)* through a path outside of T. Correspondingly, we call a configuration for T a *type (A) configuration* if case (A) holds for the root of T, and a *type (B) configuration* otherwise. Notice that in a type (A) configuration, *every* node within T is connected to a terminal in R_T by cleanup edges. For a partial tree T we will store two tables, one corresponding to case (A) and one to case (B).

Consider case (A) and let P be the path of cleanup edges connecting a terminal in R_T to r. Notice that in this case $W(e) \subseteq \overline{T}\ \forall e \in \overline{T}$. Hence, the configuration for T does not have any impact on the contribution of the edges of $\overline{T}$ to the function $\sum_{e\in E(C)} c(e)H(|W(e)|)$. However, the witness sets of the edges in P depend on the configuration for $\overline{T}$, namely every core edge that can be reached within $\overline{T}$ from r by following cleanup edges is part of the witness set of any $e \in P$. Hence, the only information about the configuration for $\overline{T}$ that matters in finding an optimal configuration within T is the number α of core edges in $\overline{T}$ that can be reached from r through cleanup edges. Thus for case (A) we want to store a table for T which contains, for each value of $\alpha \in \{0,\dots,|\overline{T}|\}$, a corresponding type (A) configuration that minimizes $\sum_{e\in T} c_e H(|W(e)|)$. Here, $|W(e)|$ can be computed without knowing the precise configuration in $\overline{T}$ (apart from α) since

$$|W(e)| = \begin{cases} |W(e)\cap T| & \text{if } e \in T \setminus P, \\ |W(e)\cap T| + \alpha & \text{if } e \in P. \end{cases}$$

Now consider case (B) and let P be the path in $\overline{T}$ connecting a terminal to r. In this case the situation is reversed and $W(e) \subseteq T$ for any $e \in T$. Hence, the configuration for $\overline{T}$ does not have any impact on the contribution of the edges of T to the function $\sum_{e\in E(C)} c(e)H(|W(e)|)$. However this time, the witness sets of edges on P depends on the configuration for T, namely every core edges that can be reached within T from r by following cleanup edges is part of the witness set of any $e \in P$. Hence, the only information that has to be stored for T in case (B), in order to describe how the configuration within T impacts the configuration outside of T, is the number β of core edges in T that can be reached from r through cleanup edges. Hence, for case (B) we want to store a table for T which contains, for each value of $\beta \in \{0, \dots, |T|\}$, a corresponding type (B) configuration that minimizes $\sum_{e \in T} c(e) H(|W(e)|)$.

Clearly, if we can compute the (A) table for the full component C, then we are done, since the globally best configuration is the one minimizing the potential function over all values of α. Computing type (A) and (B) tables for partial trees corresponding to single terminals is trivial: table (A) contains one entry corresponding to $\alpha = 0$ of value zero, and table (B) is empty. There are two constellation we exploit to compute tables for larger partial trees based on the tables of smaller ones.

The first constellation is the following. Assume that we have tables (A) and (B) for two partial trees T_{U_1} and T_{U_2} with $U_1 \cap U_2 = \emptyset$, and both having root r. Then we can compute the two tables for $T_{U_1 \cup U_2}$ from the tables of T_{U_1} and T_{U_2}. This can be done by considering all legal combinations (meaning pairs of configurations that can be completed to a splitting set) of one table entry corresponding to T_{U_1} and one corresponding to T_{U_2}, keeping the best ones. Since the size of each table is polynomially bounded in the input, this can be done efficiently. We skip the somewhat tedious details for combining those tables which are based on standard arguments.

In the second constellation, we consider a vertex r and one of its out-neighbors v, i.e., there is an arc directed from r to v, such that both tables for $T_{\delta^+(v)}$ have already been computed. We can then compute the two tables for $T_{\{rv\}}$ by considering all legal combinations of an entry of one of the tables of $T_{\delta^+(v)}$ and the two possibilities of rv being a core edge or a cleanup edge.

It is easy to observe that starting from the terminals and leveraging the above two update rules, one can construct both tables for the full component C efficiently.

□

For the following Lemma, we assume that the graph G is quasi-bipartite.

Lemma 2.6. Let $K = E(\mathcal{X}) \setminus E_{min}$, where E_{min} consists of a cheapest edge from every component. Then

$$\Phi_K(\mathcal{X}) \leq \tfrac{73}{60} \cdot \operatorname{cost}(\mathcal{X}).$$

Proof. Consider a component C, which now is a star with edges $e_1, \dots, e_k$. Assume e_k minimizes the cost, then the splitting edges in C are $K \cap C = \{e_1, \dots, e_{k-1}\}$. First of all, K is obviously a legal splitting set. Secondly $|W(e_i)| = 1$ for $i \in \{1, \dots, k-1\}$ and $|W(e_k)| = k - 1$. Thus

$$\sum_{i=1}^{k} c(e) \cdot H(|W(e_i)|) \leq (k - 1 + H(k-1)) \cdot \frac{\operatorname{cost}(C)}{k} \leq \frac{73}{60} \cdot \operatorname{cost}(C),$$

using that $1 + \frac{H(k-1)-1}{k}$ is maximized for $k = 5$. The claim follows by summing over all components $C \in \Gamma(\mathcal{X})$. □

C A lower-bound property of nonnegative intersecting submodular functions 3.6

The main goal of this section is to prove Theorem 3.6. Before presenting the core part of the proof we discuss some basic properties of partition functions, and make some general observations concerning the statement of Theorem 3.6 which are useful to understanding its proof.

Let U be a finite set. We recall that $\mathcal{F} \subseteq 2^U$ is called a *lattice family* if it is closed under unions and intersections. A function $\mathcal{F} \to \mathbb{R}$ is *submodular on* $\mathcal{F}$ if $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for all $A, B \in \mathcal{F}$; *supermodular on* $\mathcal{F}$, *intersecting supermodular on* $\mathcal{F}$ etc., are defined similarly in the obvious way.

Any partition $\mathcal{P} = \{P_1, \dots, P_n\}$ of U induces naturally a lattice family $\mathcal{F}_{\mathcal{P}} \subseteq 2^U$ which consists of all possible unions of sets in $\mathcal{P}$. Consider the coverage function $\alpha(S) = |\{j \in [n] \mid P_j \cap S \neq \emptyset\}|$, which is clearly submodular. Notice that we can write $f_{\mathcal{P}}(S) = (\alpha(S) - 1)^+$, and in particular $f_{\mathcal{P}}(S) = \alpha(S) - 1$ for all $S \neq \emptyset$. Thus $f_{\mathcal{P}}$ is intersecting submodular: for any $A, B \subseteq U$ with $A \cap B \neq \emptyset$,

$$\begin{aligned} f_{\mathcal{P}}(A) + f_{\mathcal{P}}(B) &= (\alpha(A) - 1) + (\alpha(B) - 1) \\ &\geq \alpha(A \cup B) - 1 + \alpha(A \cap B) - 1 \\ &= f_{\mathcal{P}}(A \cup B) + f_{\mathcal{P}}(A \cap B). \end{aligned}$$

Furthermore, it is easy to see that $f_{\mathcal{P}}$ is intersecting supermodular on $\mathcal{F}_{\mathcal{P}}$. Hence $f_{\mathcal{P}}$ is *intersecting modular* on $\mathcal{P}$, i.e., $f_{\mathcal{P}}(A) + f_{\mathcal{P}}(B) = f_{\mathcal{P}}(A \cup B) + f_{\mathcal{P}}(A \cap B)$ for any intersecting sets $A, B \in \mathcal{F}_{\mathcal{P}}$.

By the above observation, the function f claimed by Theorem 3.6 is by construction intersecting submodular since all $f_{\mathcal{P}^i}$ are intersecting submodular. Similarly, f is monotone due to the monotonicity of $f_{\mathcal{P}^i}$. We prove the following slightly stronger version of Theorem 3.6.

Theorem C.1. *Let $h : 2^U \to \mathbb{R}_+$ any nonnegative intersecting submodular function, such that all maximal sets $S \subseteq U$ with $h(S) = 0$ form a partition $\mathcal{P}^1$ of U. Then there is an intersecting submodular function f of the form*

$$f = \sum_{i=1}^{k} \lambda_i f_{\mathcal{P}^i},$$

where $k \leq |U| - 1$, $\lambda_i > 0\ \forall i \in [k]$, $\mathcal{P}^1, \dots \mathcal{P}^k$ are partitions of U that become coarser with increasing index, and f satisfies:
i) $f(S) \leq h(S) \quad \forall S \subseteq U$,
ii) $f(U) = h(U)$.
Furthermore, the partitions $\mathcal{P}^i$ together with the coefficients λ_i, and hence f, can be constructed efficiently.

Notice that the condition in Theorem C.1 stating that the maximal tight sets of h form a partition of U is equivalent to the property that the family of all tight sets of h covers U, due to the following uncrossing argument. If the tight sets of h cover U then so do the maximal tight sets; furthermore, for any two intersecting tight sets $A, B \subseteq U$,

$$0 = h(A) + h(B) \geq h(A \cup B) + h(A \cap B) \geq 0,$$

by submodularity and nonnegativity of h; hence $A \cup B$ is also tight. Hence, this condition is indeed weaker than the one used in Theorem 3.6, which states that all singletons must be tight.

Proof of Theorem C.1. The partitions $\mathcal{P}^1, \dots, \mathcal{P}^k$ and coefficients $\lambda_1, \dots, \lambda_k$ defining f are obtained as follows.

```
1. Let $i = 1$, $h^1 = h$, and $\mathcal{P}^1$ be the maximal tight sets with respect to $h$.
2. While $h^i(U) > 0$:
    (a) Let $\lambda_i \in \mathbb{R}_+$ be the maximum value such that
            $h^i(S) - \lambda_i f_{\mathcal{P}^i}(S) \geq 0 \quad \forall S \in \mathcal{F}_{\mathcal{P}^i}$.
    (b) $h^{i+1} \leftarrow h^i - \lambda_i f_{\mathcal{P}^i}$; let $\mathcal{P}^{i+1} \subseteq \mathcal{F}_{\mathcal{P}^i}$ be the maximal tight sets with respect to $h^{i+1}$.
    (c) $i \leftarrow i + 1$.
```

We start by observing that each function h^i encountered during the algorithm is intersecting submodular over $\mathcal{F}_{\mathcal{P}^{i-1}}$ (by convention we set $\mathcal{P}^0 = 2^U$), and that $\mathcal{P}^i$ indeed forms a partition of U. This can easily be verified through an inductive argument. By assumption h^1 is intersecting submodular over U, and $\mathcal{P}^0$ is a partition of U. The intersecting submodularity of $h^{i+1} = h^i - \lambda_i f_{\mathcal{P}^i}$ over $\mathcal{F}_{\mathcal{P}^i}$ follows by the intersecting submodularity of h^i over $\mathcal{F}_{\mathcal{P}^i}$ and the intersecting supermodularity of $f_{\mathcal{P}^i}$ over $\mathcal{F}_{\mathcal{P}^i}$. Since h^{i+1} is intersecting submodular over $\mathcal{F}_{\mathcal{P}^i}$, the maximal tight sets $\mathcal{P}^{i+1}$ of h^{i+1} in $\mathcal{F}_{\mathcal{P}^i}$ thus again form a partition of U.

The suggested procedure can indeed be implemented efficiently. At any iteration i and for any fixed $\lambda > 0$, finding the set $S \in \mathcal{F}_{\mathcal{P}^i}$ minimizing $h^i(S) - \lambda f_{\mathcal{P}^i}$ is a submodular function minimization problem. Hence, in step (2a), λ_i can be found by using e.g. binary search, or by applying the parametric search technique of Megiddo [15].

Furthermore, since $f_{\mathcal{P}_i}(S) = 0$ for all the sets $S \in \mathcal{F}_{\mathcal{P}^i}$ that are tight with respect to h^i—which are precisely the sets in $\mathcal{P}^i$—we have $\lambda_i > 0$ in each iteration. By choosing λ_i to be maximum in step (2a), there is at least one set $S \in \mathcal{F}_{\mathcal{P}^i}$ that is tight with respect to h^{i+1} but not h^i. Hence, $|\mathcal{P}^1| > |\mathcal{P}^2| > \ldots$, and the procedures will terminate. Let k be the index of the last λ that was set in step (2a). Hence, $h^{k+1}(U) = 0$, and $\mathcal{P}^{k+1} = \{U\}$. Since we start with $|\mathcal{P}^0| \leq |U|$ and the partitions coarsen at each step, this implies $k \leq |U| - 1$. Additionally, point (i) of Theorem 3.6 clearly holds by the termination criterion of the while-loop.

Hence, it remains to prove point (ii), which we prove by showing the following claim through induction from $j = k+1$ to $j = 1$, where $j = 1$ corresponds to the statement (ii):

$$h^j(S) - \sum_{i=j}^{k} \lambda_i f_{\mathcal{P}^i}(S) \geq 0 \quad \forall S \in \mathcal{F}_{\mathcal{P}^{j-1}}. \tag{14}$$

For $j = k+1$, (14) clearly holds, since $h^{k+1}(S) = h^k(S) - \lambda_k f_{\mathcal{P}^k}(S) \geq 0 \; \forall S \in \mathcal{F}_{\mathcal{P}^k}$, by choice of λ_k. Now let $j \in \{1, \ldots, k\}$ and assume that (14) holds for all values above j. Let $S \in \mathcal{F}_{\mathcal{P}^{j-1}}$, and we define $S' \in \mathcal{F}^j_{\mathcal{P}}$ to be the minimal set in $\mathcal{F}^j_{\mathcal{P}}$ that contains S, i.e.,

$$S' := \bigcup_{\substack{P \in \mathcal{P}^j, \\ P \cap S \neq \emptyset}} P.$$

Notice that

$$h^j(S) = h^j(S) + \sum_{\substack{P \in \mathcal{P}^j, \\ P \cap S \neq \emptyset}} h^j(P) \geq h^j(S') \tag{15}$$

where the equality holds since all sets in $\mathcal{P}^j$ are tight with respect to h^j by construction, and the inequality follows by standard uncrossing arguments: for any set $P \in \mathcal{P}^j, P \cap S \neq \emptyset$, we have $h^j(S) + h^j(P) \geq h^i(S \cup P)$ by intersecting submodularity and nonnegativity of h^i, and thus the two terms $h^i(S_i)$ and $h^i(P)$ can be replaced by $h^i(S_{i-1} \cup P)$ and this procedure can be repeated. In other words, we simply exploit that any nonnegative intersecting submodular function has the subadditivity property for any family of sets that are connected when seen as hyperedges on the given ground set.

The inductive step of the proof of (14) finally follows by

$$h^j(S) - \sum_{i=j}^{k} \lambda_i f_{\mathcal{P}^i}(S) \geq h^j(S') - \sum_{i=j}^{k} \lambda_i f_{\mathcal{P}^i}(S') = h^{j+1}(S') - \sum_{i=j+1}^{k} \lambda_i f_{\mathcal{P}^i}(S') \geq 0,$$

where the first inequality follows from (15) and the monotonicity of $\sum_{i=1}^{k} \lambda_i f_{\mathcal{P}^i}(S)$, and the last one by the inductive hypothesis. □

D Equivalence of the hypergraphic and bidirected cut relaxations in quasi-bipartite graphs

Let $G = (V, E)$ be a *quasi-bipartite* graph, where Steiner vertices are not connected by edges (i.e., we have edges only between terminals and Steiner vertices or between terminals and terminals). Let $\vec{E}$ be the bidirection of E, i.e., for any $\{u, v\} \in E$, $\vec{E}$ contains arcs (u, v) and (v, u).

The *bidirected cut relaxation* with *root* terminal $r \in R$ is

$$\begin{aligned} \min \sum_{e \in \vec{E}} c_e x_e & \\ x(\delta^+(S)) \geq 1 \quad & \forall S \subseteq V \backslash \{r\} : S \cap R \neq \emptyset \\ x_e \geq 0 \quad & \forall e \in \vec{E} \end{aligned} \tag{BCR(r)}$$

In words: we need to reserve enough capacity in order to support a unit flow from every terminal to the current root r. It was proven in [6] that in quasi-bipartite graphs, the value of (BCR(r)) coincides with the optimum value of (LP). This was done by lifting an optimum dual solution for the partition-based relaxation (which is equivalent to (LP) even in general graphs [6]) to a dual solution of (BCR(r)). However, the authors of [6] posed as an open question: for a given optimum bidirected cut solution, can a corresponding primal solution to (LP) be directly extracted without solving (LP)? We answer this question affirmatively.

To avoid an unnecessary case analysis, we split direct edges between terminals by inserting a dummy Steiner vertex (we split the edge cost arbitrarily among the two parts). Let x be an optimum solution to (BCR(r)); then the *natural decomposition* is as follows. For a star with center $u \in V \backslash R$, take an arc (u, s) with positive outgoing flow and all arcs $H = \{(t, u) \mid x(t, u) > 0;\ t \neq s\}$ carrying incoming flow. Let ϵ be the minimum capacity on any of these arcs. Then transfer this capacity into a component $\{s\} \cup \{t \mid (t, u) \in H\}$. Iterate this process until all capacity has been transferred. The main result of this section is:

Theorem D.1. *Let x be an optimum solution for* (BCR(r))*. Then the natural decomposition yields a feasible optimum solution for* (LP) *with the same objective value.*

Imagine that we want to "relocate" the root from r to another terminal r'. We can do this by considering the unit flow from r' to r, and reversing all capacity corresponding to this flow. This provides a feasible solution for BCR(r') that we term $x^{(r')}$, which again has the same cost (see [11] for a proof). Note that for any $\{u, v\} \in E$, the sum $x^{(r)}(u, v) + x^{(r)}(v, u)$ is independent of r. For a Steiner vertex $u \in V \backslash R$, let $N(u) := \{v \mid \{u, v\} \in E\}$ be the set of neighbours in the star with center u. It suffices to show Theorem D.1 for basic solutions, since the decomposition of a convex combination of capacity vectors equals the convex combination of natural decompositions. By standard arguments, we may assume that the edge costs are distinct for all edges in the same star.

Lemma D.2. *In a star with center u and $r \in N(u)$ one has $x^{(r)}(r, u) = 0$ and $x^{(r)}(u, s) = 0$ for each $s \in N(u)$ with $c(u, s) > c(u, r)$.*

Proof. The flow on arc (r, u) can be removed and the flow on (u, s) arc can be redirected to (u, r). Both operations would leave the solution feasible and decrease the cost, contradicting optimality. □

See Figure 4 for an illustration of the claim.

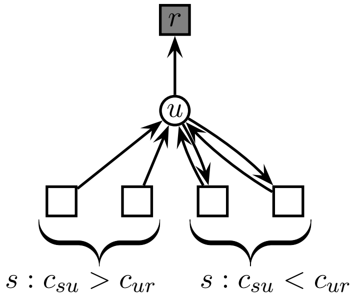
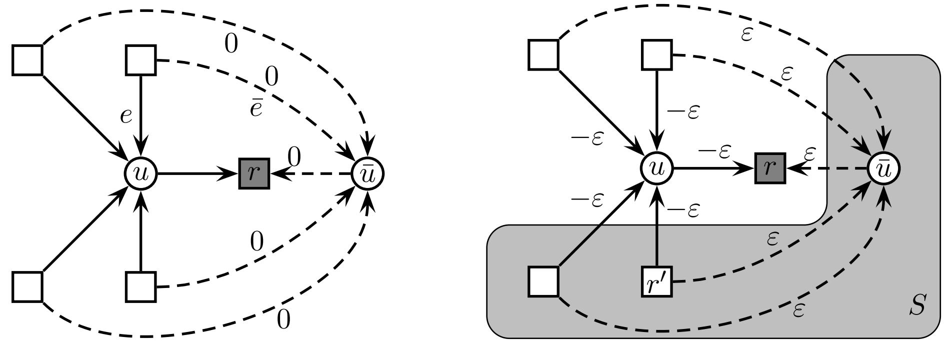


Figure 4: Arcs in the optimum solution $x^{(r)}$ that may carry positive flow.

Next, we consider one iteration of the natural decomposition for a star with center u. For this reason, insert an extra Steiner vertex $\bar{u}$ into the graph, which has an edge $\{\bar{u}, s\}$ with $s \in R$ iff there is an edge $\{u, s\} \in E$ with $x^{(r)}(s, u) > 0$. For $e = (u, s)$, we abbreviate $\bar{e} = (\bar{u}, s)$ (see Figure 5). We define $c(\bar{e}) := c(e)$ and $x^{(r)}(e) = 0$ for all $e \in \delta(\bar{u})$. Note that $x^{(r)}$ is still an optimum solution.

Lemma D.3. *Let* $r := $ *argmin*$\{c(u, r) \mid r \in N(u)\}$, $H := \{(u, r)\} \cup \{(s, u) \mid s \in N(u) \backslash \{r\}\}$ *and* $\varepsilon := \min\{x^{(r)}(e) \mid e \in H\}$. *Starting from* $x^{(r)}$*, transfer capacity of* ε *from all arcs* $e \in H$ *to* $\bar{e}$ *and term the new capacity reservation* $\bar{x}^{(r)}$*. Then the new capacity vector* $\bar{x}^{(r)}$ *is a feasible optimum solution for* (BCR(r)).

Proof. We first show that the claim holds for *some* $\varepsilon > 0$ which is small enough. Consider any cut $S \subseteq V \backslash \{r\}$ and assume for the sake of a contradiction that $\bar{x}^{(r)}(\delta^+(S)) < 1$. For $\varepsilon > 0$ small enough, this may only happen if S was a tight cut before, i.e. $x^{(r)}(\delta^+(S)) = 1$. Furthermore, any critical cut must contain at least two arcs of the form (s, u), i.e., $|N(u) \cap S| \geq 2$. Pick $r' := \text{argmin}\{c(u, r') \mid r' \in N(u) \cap S\}$. According to Lemma D.2, the flow is $x^{(r)}(e) = 0$ for $e \in (\delta^-(u) \cup \delta^+(u)) \backslash H$. Since $x^{(r)}(\delta^+(S)) = 1$, the unit flow from r' to r needs all capacities on (s, u) arcs for $s \in N(u) \cap S$. Consequently, when relocating the root to r', the flow on all these arcs must be turned around completely. In particular $x^{(r')}(u, s) > 0$ for $s \in (N(u) \cap S) \backslash \{r'\}$ contradicting Lemma D.2.



(a) capacities $x^{(r)}$

(b) capacities $\bar{x}^{(r)}$

Figure 5: Transferring capacity of ε according to Lemma D.3. (a) visualizes capacity in $x^{(r)}$, where newly added edges $\bar{e}$ are dashed. (b) depicts $\bar{x}^{(r)}$ and a potentially critical cut S.

We conclude that we can choose *some* $\varepsilon > 0$ s.t. $\bar{x}^{(r)}$ is feasible. But the argument above shows that no cut S can become tight, thus the only limitation on ε is the arc capacity. The claim then follows. □

We apply Lemma D.3 iteratively to all stars, adding copies of Steiner nodes as required, until we have a solution x^* (with root $r^* \in R$ chosen arbitrarily) where (i) every Steiner node has flow on at most one outgoing arc, and (ii) the flow on all arcs of a star carrying a nonzero amount of flow is the same. Then x^* induces a solution to the *directed component-based relaxation*:

$$\begin{array}{rcll} \min \sum\limits_{C\in\mathcal{K}, s\in C} \operatorname{cost}(C)\cdot y_{C,s} & & & \\ \sum\limits_{C\in\mathcal{K}, s\in C: C\cap S\neq\emptyset, s\notin S} y_{C,s} & \geq & 1 & \forall \emptyset \subsetneq S \subseteq R\backslash\{r^*\} \\ y_{C,s} & \geq & 0 & \forall C\in\mathcal{K}\ \forall s\in C. \end{array}$$

The solution y^* corresponding to x^* is obtained by setting, for each flow-carrying star with terminal set C and outgoing flow on arc (u,s), $y^*_{C,s} = x^*(u,s)$ (the common flow value in the star). All other components of y^* are zero. It is easily checked that y^* is feasible, and has the same cost as x^* (and hence x). Then projecting to the undirected formulation, the vector $(\sum_{s\in C} y_{C,s})_{C\in\mathcal{K}}$ is feasible for (LP) (see [18]), and moreover corresponds precisely to the natural decomposition described earlier.

E NP-hardness for solving the component-based relaxation

It is well-known that there is a PTAS for solving (LP). In other words, for every fixed $\varepsilon > 0$, there is a polynomial time algorithm that computes a feasible fractional solution to the considered hypergraphic relaxation (LP), which is within a $1+\varepsilon$ factor of the optimum fractional value. We argue now, that this is best possible (answering the posed question in [6]).

Theorem E.1. *Solving* (LP) *is strongly* $\mathbf{NP}$*-hard.*

Proof. Let $G = (V, E)$ be a complete graph with terminals $R = \{s_1, \ldots, s_k\} \subseteq V$, edge cost $c(e) \in \{1, 2\}$ for all $e \in E$. Bern and Plassmann [1] showed that it is $\mathbf{NP}$-hard to decide whether the cost OPT of the cheapest Steiner tree is at most a given parameter Z.

We construct another Steiner tree instance $G' = (V', E')$ as follows: For each terminal $s_i \in R$ in the original instance, we add a terminal s'_i and an edge $s_i s'_i$ with cost $c(s_i, s'_i) := M$ with $M := 3n^2$ and $n = |V|$. Furthermore, we downgrade the original terminal to an ordinary vertex, i.e., we define $R' := \{s'_i \mid i = 1, \ldots, k\}$ as terminal set. Let OPT'_f be the value of the optimum fractional solution of (LP) for instance G' (using components of arbitrary size).

First we show that $OPT \leq Z \Rightarrow OPT'_f \leq Z + k\cdot M$. Let S^* be the optimum integral Steiner tree in G. We add all $s_i s'_i$ edges to S^* and consider the emerging tree as component with fractional weight 1 and cost $OPT + k \cdot M$.

Next, we prove that $OPT \geq Z + 1 \Rightarrow OPT'_f \geq Z + 1 + k\cdot M$ (which in turn implies the claim of the theorem). Let x be an optimum solution to (LP) in G'. For a component $C \in \mathcal{K}$, we denote $E(C)$ as the contained edges from the original graph (i.e. without $s_i s'_i$ edges) and by $|C|$ we denote the number of terminals. Either C contains less than k terminals, or $\operatorname{cost}(E(C)) \geq Z + 1$. In any case

$$\frac{cost(E(C)) + M}{|C| - 1} \geq \min\left\{\frac{Z+1+M}{k-1}, \frac{M}{k-2}\right\} \geq \frac{Z+1+M}{k-1}$$

using that $M = 3n^2$, $k \leq n$ and $Z \leq 2n$. Now we can bound the cost of the fractional solution as

$$\begin{aligned} OPT'_f &= \sum_{C\in\mathcal{K}} (\operatorname{cost}(E(C)) + |C|\cdot M)\cdot x_C \\ &= M\sum_{C\in\mathcal{K}}(|C|-1)x_C + \sum_{C\in\mathcal{K}}(\operatorname{cost}(E(C)) + M)x_C \\ &\geq (k-1)M + \sum_{C\in\mathcal{K}} \frac{|C|-1}{k-1}(Z+1+M)x_C = Z+1+kM \end{aligned}$$

exploiting $\sum_{C \in \mathcal{K}} x_C(|C| - 1) = k - 1$. □

Observe that the above reduction in not approximation preserving. This is not surprising, considering the fact that Steiner tree even with edge weights $\{1, 2\}$ is $\mathbf{APX}$-hard (e.g. by a straightforward reduction from Set Cover with sets of size 3).