On the Equivalence of the Bidirected and Hypergraphic Relaxations for Steiner Tree

Andreas Emil Feldmann[1], **Jochen Könemann**[1], **Neil Olver**[2], **and Laura Sanità**[1]

1 **Department of Combinatorics and Optimization, University of Waterloo**
`{andreas.feldmann,jochen,laura.sanita}@uwaterloo.ca`

2 **VU University & CWI, Amsterdam**
`n.olver@vu.nl`

Abstract

The bottleneck of the currently best $(\ln(4)+\varepsilon)$-approximation algorithm for the NP-hard *Steiner tree* problem is the solution of its large, so called *hypergraphic*, linear programming relaxation (HYP). Hypergraphic LPs are NP-hard to solve exactly, and it is a formidable computational task to even approximate them sufficiently well.

We focus on another well-studied but poorly understood LP relaxation of the problem: the *bidirected cut relaxation* (BCR). This LP is compact, and can therefore be solved efficiently. Its integrality gap is known to be greater than 1.16, and while this is widely conjectured to be close to the real answer, only a (trivial) upper bound of 2 is known.

In this paper, we give an efficient constructive proof that BCR and HYP are polyhedrally equivalent in instances that do not have an (edge-induced) claw on Steiner vertices, i.e., they do not contain a Steiner vertex with 3 Steiner neighbors. This implies faster $\ln(4)$-approximations for these graphs, and is a significant step forward from the previously known equivalence for (so called *quasi-bipartite*) instances in which Steiner vertices form an independent set. We complement our results by showing that even restricting to instances where Steiner vertices induce one single star, determining whether the two relaxations are equivalent is NP-hard.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Steiner tree, bidirected cut relaxation, hypergraphic relaxation, polyhedral equivalence, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

In an instance of the well-studied, NP-hard [5, 14] Steiner tree problem one is given an undirected graph $G = (V, E)$, a non-negative cost $\mathrm{cost}(e)$ for each edge $e \in E$, and a set of *terminals* $R \subseteq V$. The goal is to find a minimum-cost tree spanning R. Steiner trees arise in a host of practical applications (e.g., see the survey [12] and the current DIMACS implementation challenge [8]), and therefore have been extensively studied in the network design community.

In this paper, we focus on the problem's efficient approximability. In a recent breakthrough, Byrka et al. [2] presented the currently best $(\ln(4)+\varepsilon)$-approximation algorithm for the problem. The algorithm crucially relies on the repeated solution of a large, so called *hypergraphic* LP relaxation (henceforth abbreviated by HYP) for the problem. It was later shown by Goemans et al. [11] that it is possible to achieve the same approximation guarantee while only solving HYP once. However, solving hypergraphic Steiner tree relaxations is challenging: Goemans et al. [11] also showed that solving them exactly is strongly NP-hard, and even their approximation amounts to (exactly) solving LPs with more than $|R|^k$ variables and constraints (where k is a constant that needs to be ~ 100 in order to yield an approximation to HYP of sufficient quality).

Another well-known formulation for the Steiner tree problem is the *bidirected cut* relaxation (BCR) [6, 20]. BCR is an appealing relaxation as its compactness implies efficient solvability. As one way of obtaining a faster approximation algorithm for the Steiner tree problem, we therefore propose to first compute a solution to HYP *from* a solution to BCR. Then, we apply the algorithm of Goemans et al. [11] in order to compute a Steiner tree with cost at most $\ln(4)$ times that of the given HYP solution. Since HYP has a smaller integrality gap than BCR in general (we present the largest known gap ratio of 12/11 below), we cannot hope to compute a solution to HYP of the same cost as the optimum BCR solution. We therefore ask when these two LPs have the same integrality gap.

The integrality gap of BCR is not well-understood. It is known to be at least $36/31 \approx 1.16$ [2], and while the latter number is widely conjectured to be close to the truth, the only thing known is an almost trivial upper bound of 2. HYP on the other hand has an integrality gap of at least $8/7 \approx 1.14$ [15] and at most $\ln(4) \approx 1.39$ [11]. Hence comparing the gaps of these two LPs can shed some light on the gap of BCR.

Previously it was known that the integrality gaps of BCR and HYP are equal for *quasi-bipartite* instances where no two Steiner vertices (the vertices in $V \setminus R$) are connected by an edge [3, 9, 11]. For these graphs the Steiner tree problem remains NP-hard [18]. In this paper we significantly extend the class of instances where the integrality gaps are identical. In our main result, we show that as long as the input graph G has no Steiner vertex with three Steiner neighbors (we will refer to this as a *Steiner claw*), BCR and HYP are polyhedrally equivalent. Specifically, we will provide a cost-preserving, and efficiently computable map between feasible solutions to BCR and those of HYP. We will also show that our results are nearly best possible by exhibiting instances with a single star on Steiner vertices for which it is NP-hard to decide whether BCR and HYP have the same integrality gap.

In the following we describe the relaxations BCR and HYP in more detail before formally stating our contributions.

1.1 Bidirected and hypergraphic LPs for Steiner trees

In the bidirected cut relaxation one usually considers a directed auxiliary graph that has two arcs (u,v) and (v,u) of cost $\mathrm{cost}(uv)$ for each original edge $uv \in E$. The LP, which we will refer to as BCR*, has a variable for each of these arcs, and its constraints force at least one arc to cross each directed cut that separates a chosen root $r \in R$ from at least one other terminal (see [6, 20]). More concretely, if the set $\vec{E}$ contains the directed arcs (u,v) and (v,u) for all edges $uv \in E$, $\delta^+(S) := \{(u,v) \in \vec{E} \mid u \in S, v \notin S\}$ is the set of arcs crossing a set $S \subseteq V$, and $z(\delta^+(S)) = \sum_{a \in \delta^+(S)} z_a$, the LP is

$$\min \sum_{a \in \vec{E}} z_a \,\mathrm{cost}(a) \qquad \text{s.t.} \qquad\qquad \text{(BCR*)}$$
$$z(\delta^+(S)) \geq 1 \qquad\qquad \forall S \subseteq V \setminus \{r\}, S \cap R \neq \emptyset$$
$$z \geq \mathbb{0}$$

In this paper, we importantly choose to work with an equivalent *undirected* formulation (see [10]) which we will refer to as BCR. We state this LP below, where we associate a variable z_e with each (undirected) edge $e \in E$, and a variable y_v with each vertex $v \in V$. For brevity we use $E(S)$ for the collection of edges with both ends in $S \subseteq V$, $z(E') = \sum_{e \in E'} z_e$, $y(S) = \sum_{v \in S} y_v$, and $y_{max}(S)$ as a shorthand for $\max_{v \in S} y_v$.

$$\min \sum_{e \in E} z_e \operatorname{cost}(e) \quad \text{s.t.} \tag{BCR}$$

$$\begin{aligned} z(E(S)) &\le y(S) - y_{max}(S) && \forall S \subseteq V \\ z(E) &= y(V) - 1 \\ y_t &= 1 && \forall t \in R \\ y, z &\ge \mathbb{0} \end{aligned}$$

We note that the LP becomes Edmonds' famous *subtour* formulation for the spanning tree polyhedron [7] when y is replaced by the vector of ones. Furthermore, BCR can be solved efficiently: simply compute a solution to a compact flow formulation of its directed counterpart BCR*, and observe that it can be mapped to a solution of the same value for BCR (see [10]): set y_v in BCR to the sum of outgoing arc values from $v \in V \setminus \{r\}$ in BCR* (this corresponds to the amount of flow that v can send to the root). The value y_r of the root of BCR* is simply 1 in BCR. The value z_e for an edge in BCR is given by the sum of the corresponding two arc values in BCR*.

Hypergraphic LPs are inspired by the observation that the Steiner tree problem can be equivalently phrased as that of computing a minimum-cost spanning tree in an appropriately defined hypergraph on the terminals. There are multiple equivalent, directed and undirected forms of HYP [3]. Corresponding to our undirected choice of BCR, we will henceforth focus on the hypergraphic subtour relaxation introduced in [19]. The LP has one variable for each *full-component* of the instance. A full-component is a tree all of whose leaves are terminals, and whose internal nodes are Steiner vertices. We let $\mathcal{K}$ be the set of all full-components of the instance, and note that $\mathcal{K}$ may have multiple full-components spanning the same set of terminals, but having different edges. The cost of a full-component is equal to the sum of the cost of its edges. In the following hypergraphic subtour formulation we let $(a)^+$ be a short-hand for $\max\{0, a\}$, and $R(C)$ denote the set of terminals included in C .

$$\min \sum_{C \in \mathcal{K}} x_C \operatorname{cost}(C) \quad \text{s.t.} \tag{HYP}$$

$$\begin{aligned} \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S| - 1)^+ &\le |S| - 1 && \forall S \subseteq R, S \neq \emptyset \\ \sum_{C \in \mathcal{K}} x_C(|R(C)| - 1)^+ &= |R| - 1 \\ x &\ge \mathbb{0} \end{aligned}$$

As mentioned, solving HYP is strongly NP-hard. However, restricting $\mathcal{K}$ to full-components spanning at most k terminals (for some fixed k) renders the LP polynomial-time solvable, and it can be shown that its optimal value increases by at most a factor of $(1 + 1/\lfloor \log k \rfloor)$ [1]. We may therefore choose $k = k(\varepsilon)$ appropriately to obtain a $1 + \varepsilon$ approximation to HYP, for any $\varepsilon > 0$. As mentioned above, to achieve solutions of sufficient quality, k needs to be ~ 100, which implies LPs with more than $|R|^{100}$ variables and constraints.

1.2 Our contributions

We call a Steiner tree instance *Steiner claw-free* if the graph G has no Steiner vertex with at least three Steiner neighbors. Our main result is the following, which implies faster ln(4)-approximations for Steiner claw-free graphs. In particular, our running time is dominated by solving BCR, which in its compact flow formulation has $O(|R||E|)$ variables and constraints.

▶ **Theorem 1.** *In a Steiner claw-free Steiner tree instance, any minimal solution to BCR can be efficiently converted to a solution to HYP of no larger cost.*

As an immediate consequence, we obtain an integrality gap bound of ln(4) for BCR in Steiner claw-free instances via [11], improving the previously known bound of 2. The only class of Steiner tree instances where BCR was previously known to exhibit an integrality gap smaller than 2 is that of quasi-bipartite graphs. Previous work in [4, 11] showed that their integrality gap is at most $73/60 \approx 1.216$.

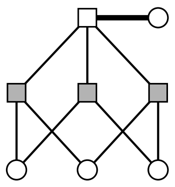


Figure 1 Example instance with HYP $\neq$ BCR. Terminals are circles, Steiner vertices are squares.

Theorem 1 implies that BCR and HYP are equivalent in every instance of the Steiner tree problem where Steiner vertices induce subgraphs in which the maximum degree of each vertex is 2 (i.e. paths and cycles). On the other hand, Figure 1 shows a graph with 4 Steiner vertices inducing a subgraph with only one vertex of degree 3, where BCR and HYP are not equivalent. If we assume all edges to have unit cost, BCR is easily seen to admit a solution of cost 5.5: let $z_e = 1$ for the thick edge, and $z_e = 1/2$ otherwise. All white vertices v in the figure have $y_v = 1$, and others have $y_v = 1/2$. The optimum Steiner tree has cost 6 and this is also the value of HYP. Hence in general the gap ratio between the two LPs is at least 12/11.

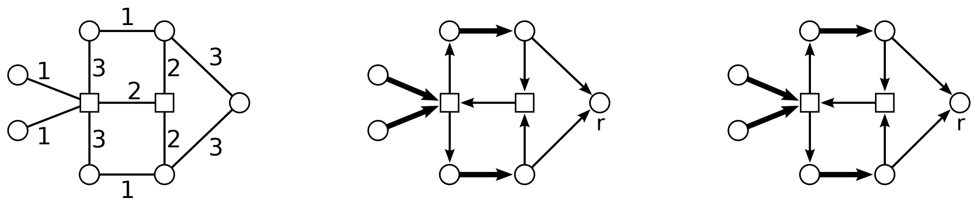
At a high level, our algorithmic proof of Theorem 1 follows the greedy approach taken in [9, 11] for quasi-bipartite instances. Roughly, the above papers first solve the directed version BCR* of BCR, and convert it into a solution for a directed version of HYP, commonly referred to as *directed component relaxation* DCR (see [17]). This directed formulation is equivalent to HYP [3]. For DCR, each full-component is directed, i.e. it is an in-arborescence to one of its terminals, called its *head*. We call the set of all directed full-components $\vec{\mathcal{K}}$. By $\Delta^+(S)$ we denote all full-components $C \in \vec{\mathcal{K}}$ for which the head lies outside S, while some other terminal of C lies inside. Also let $x(\Delta^+(S)) = \sum_{C\in\Delta^+(S)} x_C$. The directed hypergraphic (component) relaxation then is:

$$\min \sum_{C\in\vec{\mathcal{K}}} x_C \,\mathrm{cost}(C) \quad \text{s.t.} \tag{DCR}$$

$$x(\Delta^+(S)) \geq 1 \qquad \forall S \subseteq R \setminus \{r\}, S \neq \emptyset$$

$$x \geq \mathbb{0}$$

The approach of [9, 11] is to iteratively and greedily *shave off* fractional capacity uniformly from the arcs of a directed full-component in the support of the given directed BCR* solution. In the case of quasi-bipartite instances, this approach works and yields a feasible solution for DCR of the same cost as the original BCR* solution. As soon as Steiner vertices are allowed to have Steiner neighbors, the above strategy runs into problems, however. Figure 2(a) shows a Steiner claw-free instance, and two optimal solutions to BCR* in Figures 2(b) and 2(c). One can show that there is no DCR solution whose canonical *projection* yields (or more precisely, is dominated by) the BCR* solution in 2(c). Hence the outlined greedy strategy taken in [9, 11] will not work here. On the other hand, the solution given in 2(b) is the projection of a feasible solution to DCR. The crux appears to be that both solutions in Figure 2(b) and 2(c) *project* to the same undirected solutions of BCR. By considering undirected relaxations we avoid the complication inherent in the directed nature of the LPs.



(a) Edge costs.

(b) Decomposable: one full-component is marked a, the other b, each with value 1/2.

(c) Not decomposable.

Figure 2 An instance with edge costs as given in (a). Some optimal solutions to BCR* are decomposable (b) into a DCR solution, and others (c) are not (we omit the proof). In the BCR* solutions the root is marked r, bold arcs have capacity 1, and the others 1/2.

The results for the quasi-bipartite case [3, 9, 11] at their heart rely on the property that tight sets that intersect in terminals can be uncrossed. To move beyond the quasi-bipartite case, however, we require a deeper understanding of the interaction of tight sets, including those that are not terminal-intersecting. We believe that our techniques may be helpful in the quest for a better-than-2 bound on the integrality gap of the bidirected cut relaxation: while a mapping from BCR to HYP that preserves cost is not possible, one that only loses a small factor may be. In fact, it can readily be seen that the algorithm we present in the rest of this paper can be used to compute an approximate solution to HYP for the example given in Figure 1. For this we set $y_v = 1$ for any one of the gray Steiner vertices v, and also $z_e = 1$ for the edge e connecting v to the white Steiner vertex. This again yields a feasible solution to BCR for which our algorithm computes a solution to HYP of the same cost 6.

Our main result of this paper shows that the property of being Steiner claw-free, which is polynomially checkable, is a sufficient condition for equivalence of the two relaxations. We also show that there is no good characterization of this equivalence. Even if we restrict to instances where the Steiner vertices induce a single star, deciding equivalence is NP-hard. We defer the proof of the following theorem to the full version of the paper.

▶ **Theorem 2.** *It is NP-hard to decide for a given Steiner tree instance whether BCR has the same optimum value as HYP, even if the Steiner vertices induce a single star.*

2 A constructive map between BCR and HYP

In this section, we will give a detailed description of an algorithm that converts a minimal feasible BCR solution into a solution for HYP. At a high level, the arguments are structured similarly to those used in [9, 11]. Crucially, however, we will be using the undirected relaxations BCR and HYP introduced in the previous section instead of their directed analogs.

Our algorithm computes a solution to BCR and in each step identifies a tree C^* in the support of this solution. We then carefully choose $\varepsilon > 0$, and remove it from z_e for all $e \in E(C^*)$ as well as from y_v for all $v \in V(C^*) \setminus R$. Subsequently, we then add ε to the x-variable of a maximal full-component contained in C^*. In order to facilitate our discussion of this

Algorithm 1 Finding a full component.

```
1: Choose an arbitrary Steiner vertex $\ell \in V(H)$ and let $V(C^*) = \{\ell\}$.
2: As long as there is a Steiner vertex $v$ neighboring a vertex $w \in V(C^*)$, add it and the
   edge $vw$ to $C^*$ as long as $C^*[S]$ is connected for all tight sets $S$.
3: As long as there is a terminal $t$ neighboring a Steiner vertex $w \in V(C^*)$, add it and the
   edge $wt$ to $C^*$ as long as $C^*[S]$ is connected for all tight sets $S$.
4: Obtain $C$ from $C^*$ by deleting Steiner leaves as long as these exist.
```

greedy process, we define the following "mixed LP" (M), which is a hybrid of BCR and HYP.

$$\min \sum_{e\in E} z_e \operatorname{cost}(e) + \sum_{C\in\mathcal{K}} x_C \operatorname{cost}(C) \quad \text{s.t.} \tag{M}$$

$$z(E(S)) + \sum_{C\in\mathcal{K}} x_C(|R(C)\cap S| - 1)^+ \le y(S) - y_{max}(S) \qquad \forall S \subseteq V \tag{1}$$

$$z(E) + \sum_{C\in\mathcal{K}} x_C(|R(C)| - 1)^+ = y(V) - 1 \tag{2}$$

$$y_v = 1 \qquad \forall v \in R \tag{3}$$

$$x, y, z \ge \mathbb{0}.$$

Note that if for a feasible solution (x, y, z) to this LP, $z = \mathbb{0}$ and $y_v = 0$ for all $v \in V \setminus R$, then x is a solution to HYP. On the other hand, if $x = \mathbb{0}$, then (y, z) is a solution to BCR. Hence we want to begin with a feasible solution to (M) with $x = \mathbb{0}$, and end with one where $z = \mathbb{0}$ and $y = \chi(R)$, where $\chi(R)$ is the characteristic vector of the terminal set.

Let H be the *support graph* of (y, z), where $V(H) = \{v \in V : y_v > 0\}$ and $E(H) = \{e \in E : z_e > 0\}$. Observe that, whenever there is an edge $uv \in E(H)$ connecting two terminals $u, v \in R$, we may transfer the value z_{uv} to the x variable of the corresponding full component without affecting the feasibility or cost of our solution. We will therefore now assume that H has no edge connecting terminals.

We first describe an algorithm for picking a specific tree C^* in H. Define the *slack* of a vertex set $S \subseteq V$ as

$$\operatorname{sl}(S) := y(S) - y_{max}(S) - z(E(S)) - \sum_{C\in\mathcal{K}} x_C(|R(C)\cap S| - 1)^+,$$

and note that $\operatorname{sl}(S) \ge 0$ in a feasible solution (x, y, z) to (M). We will call a set S *tight* if $\operatorname{sl}(S) = 0$. Furthermore, we denote by $C^*[S]$ the subgraph of C^* induced by the vertices in $S \cap V(C^*)$. We will use Algorithm 1 to compute C^*.

It turns out to be the case that for Steiner claw-free instances, C^* always contains a terminal. We will not assume this for the following analysis however, and so it is convenient to include an "empty" full-component in $\mathcal{K}$, which contains no terminals. Such a component of course has no impact on (M). We also allow full-components containing only a single terminal, which also have no impact on (M). In any case, $C \in \mathcal{K}$ is always a maximal full-component contained in the tree C^*, and $R(C) = R(C^*)$.

Given C^* and full-component C as computed by Algorithm 1, and some $\varepsilon > 0$, we obtain $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ from (x, y, z) by adding ε to x_C, and subtracting ε from y_v and z_e for $v \in V(C^*) \setminus R$ and $e \in E(C^*)$, respectively. Note that this does not increase the cost of the solution. We will argue that if our input instance has no Steiner claw, $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ is feasible for (M) for some choice of $\varepsilon > 0$ small enough. This leads to Algorithm 2.

Note that at the end of the algorithm, $y = \chi(R)$ but also $z = \mathbb{0}$. This is because for every edge uv with $v \notin R$, Constraint (1) on the set $S = \{u, v\}$ implies $z_{uv} \le y_v = 0$. Moreover, we explicitly moved all z-value from edges between terminals. Hence if the algorithm succeeds, we computed a solution to HYP of the same cost as the solution to BCR we started from.

Algorithm 2 can be implemented in polynomial time. The details of this can be found in Section 4. It is necessary to show that (i) C^* can be efficiently computed for a given solution to (M), (ii) the correct choice of ε in Step 5 of Algorithm 2 can be efficiently found, and (iii) the number of iterations in Algorithm 2 is polynomially bounded. Roughly speaking, (i) follows by reducing the problem of finding a set of minimum slack (under certain restrictions) to a flow problem, and (ii) then follows by applying parametric search methods to this reduction. For point (iii), we obtain a bound of $O(|V|^2)$ on the number of iterations. This is done by arguing via uncrossing techniques that the number of “independent” tight constraints of a certain form for a solution to (M) cannot be too large. A new constraint becomes tight at the start of each iteration, with all previous ones remaining tight, and the bound follows.

In order for Algorithm 2 to produce a feasible HYP solution of no larger cost than the initial BCR solution, we need to show that it is always possible to select $\varepsilon > 0$ at Step 5, while maintaining feasibility for (x, y, z). If ε is small enough, all variables in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ have non-negative values, because C^* is a subgraph of the support graph H of (y, z). Furthermore, going from (x, y, z) to $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ does not change the value of y_v for any terminal $v \in R$, and thus (3) is unaffected by this change. It remains to check that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ satisfies (1), and moreover that the constraint remains tight for $S = V$, so that (2) is also satisfied. We begin by characterizing when a tight set in (x, y, z) remains feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.

▶ **Lemma 3.** *Let $S \subseteq V$ be tight in a feasible solution (x, y, z) to (M), C^* be a tree of the support graph H of (y, z), $V(C^*) \cap S \neq \emptyset$, and $\varepsilon > 0$ small enough. Then S is feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ if and only if (i) $C^*[S]$ is connected, and (ii) $\{v \in S : y_v = y_{max}(S)\} \subseteq V(C^*)$ if $S \cap R = \emptyset$, or $R(C^*) \cap S \neq \emptyset$ if $S \cap R \neq \emptyset$. Moreover, S remains tight in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.*

Proof. First consider the case when $S \cap R = \emptyset$. Let $S_m = \{v \in S : y_v = y_{max}(S)\}$, and define $\rho = 1$ if $S_m \subseteq V(C^*)$, and $\rho = 0$ otherwise. We use $\mathrm{sl}_\varepsilon(S)$ for the slack of set S in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, and obtain

$$\begin{aligned}\mathrm{sl}_\varepsilon(S) = \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S])| + \rho + |E(C^*[S])| - (|R(C^*) \cap S| - 1)^+\big)\\ = \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S])| + \rho + |E(C^*[S])|\big).\end{aligned}$$

But since $C^*[S]$ is a forest, $|E(C^*[S])| \le |V(C^*[S])| - 1$, with equality only if $C^*[S]$ is connected. The result follows.

Note that if there was a Steiner vertex v with $y_v > 1$, then Constraint (1) for the set V would contradict Constraint (2). In particular this means that in a feasible solution to (M),

Algorithm 2 Converting a BCR solution to an HYP solution.

```
1: Start with a solution $(x, y, z)$ feasible for (M) with $x = \mathbb{0}$.
2: For any $z_{vw} > 0$ with $v, w \in R$, move all weight to the corresponding $x$ variable.
3: **while** $y \neq \chi(R)$ **do**
4:     Apply Algorithm 1 to compute a tree $C^*$ and a full-component $C$.
5:     Choose $\varepsilon > 0$ maximally such that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ remains feasible for (M), and
       replace $(x, y, z)$ with this new solution.
6: **end while**
7: Output $(x, y, z)$.
```

the terminals have maximal y-values. Hence in the case where $S \cap R \neq \emptyset$ we obtain

$$\mathrm{sl}_\varepsilon(S) = \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S \setminus R])| + |E(C^*[S])| - (|R(C^*) \cap S| - 1)^+\big).$$

Thus S stays feasible if and only if $|V(C^*[S \setminus R])| + (|R(C^*) \cap S| - 1)^+ \leq |E(C^*[S])|$. Let ρ' be 1 if $R(C^*) \cap S \neq \emptyset$ and 0 otherwise. Then, simplifying further, S stays feasible if and only if $|V(C^*[S])| - \rho' \leq |E(C^*[S])|$. Again since $C^*[S]$ is a forest, $|V(C^*[S])| \geq |E(C^*[S])| + 1$, with equality if and only if $C^*[S]$ is connected. So the inequality is satisfied if and only if $C^*[S]$ is connected and $R(C^*) \cap S \neq \emptyset$, in which case it is satisfied with equality. ◀

Note that the constraint corresponding to V is tight in (x, y, z). Thus if it is feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, by Lemma 3 it will remain tight, and (2) will be satisfied. The goal is now to apply Lemma 3 to show that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ satisfies (1) for some $\varepsilon > 0$ whenever there is no Steiner claw. By construction (see Algorithm 1), $C^*[S]$ is connected for all tight sets $S \subseteq V$. Thus we can shift $\varepsilon > 0$ of the value of the y and z variables associated with C^* to x_C, unless there is a tight set *demanding* the inclusion of (some of) its vertices with maximum y-value (e.g. terminals) in C^*.

▶ **Definition 4.** A tight set S for which $V(C^*) \cap S \neq \emptyset$ is called a *demanding set* if $R(C^*) \cap S = \emptyset$ in case S contains a terminal, or if there is some vertex $v \in S \setminus V(C^*)$ for which $y_v = y_{max}(S)$ in case S has no terminals.

3 Analysis of the algorithm

In this section, we show that if a demanding set S exists, then we can identify a Steiner claw in H. This implies that for Steiner claw-free instances, Algorithm 2 will always find an $\varepsilon > 0$ at Step 5, and therefore terminates successfully. As for the algorithm for quasi-bipartite instances, we will rely on uncrossing arguments of tight sets. However, it will not be sufficient to only uncross tight sets intersecting in terminals. Therefore we develop more advanced uncrossing techniques in our arguments.

Demanding sets and blocked edges.

Let S be a demanding set for the tree C^* chosen by Algorithm 1. Thus $S \cap V(C^*) \neq \emptyset$ and $C^*[S]$ is connected.

▶ **Lemma 5.** *Let U be a tight set of a feasible solution (x, y, z) to* (M)*, and let H be the support graph of (y, z). If $U \cap R \neq \emptyset$, then every connected component of $H[U]$ contains a terminal. If $U \cap R = \emptyset$, then $H[U]$ is connected.*

Proof. Assume the statement is wrong. Regardless of whether U contains terminals or not, there must then be a connected component in $H[U]$ with vertex set U_1, such that $U_1 \cap R = \emptyset$ and $U_2 := V(H[U]) \setminus U_1$ is non-empty. In particular, $E(U) = E(U_1) \cup E(U_2)$, $|R(C) \cap U_1| = 0$

for every full component $C \in \mathcal{K}$, and $y_{max}(U_2) > 0$. Thus,

$$\begin{aligned}
\mathrm{sl}(U_1 \cup U_2) &= y(U) - y_{max}(U) - z(E(U)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U| - 1)^+ \\
&= y(U_1) + y(U_2) - y_{max}(U_1 \cup U_2) - z(E(U_1)) - z(E(U_2)) \\
&\quad - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ \\
&> y(U_1) - y_{max}(U_1) - z(E(U_1)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_1| - 1)^+ \\
&\quad + y(U_2) - y_{max}(U_2) - z(E(U_2)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ \\
&= \mathrm{sl}(U_1) + \mathrm{sl}(U_2).
\end{aligned}$$

By feasibility of U_1 and U_2, $U_1 \cup U_2$ cannot be tight, a contradiction. ◀

Consider a path P_1 in $H[S]$ for the demanding set S, that connects $S \cap V(C^*)$ to some vertex $v \in S \setminus V(C^*)$ with $y_v = y_{max}(S)$ (e.g. a terminal). By Lemma 5 this path exists, whether or not S contains terminals. Traversing the path from v, let b_1 be the first vertex of C^*, and let a_1 be its immediate predecessor (see Figure 3). Note that b_1 must be a Steiner vertex, otherwise S would not be a demanding set. We will in fact be able to show later that a_1 must also be a Steiner vertex.

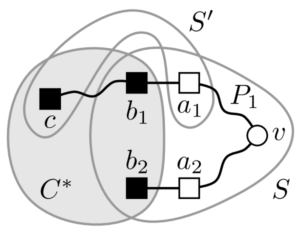


Figure 3 Interaction of a demanding set S and a blocking set S'.

Edge a_1b_1 is called a *blocked* edge: its endpoint b_1 is part of C^*, but a_1 was not added to C^* by Algorithm 1. Thus, there must be a tight set S' for which $C^* \cup \{a_1b_1\}$ is disconnected in S', and thus *blocks* the addition of a_1b_1; we call S' a *blocking set.* S' contains a_1, not b_1, but some other vertex $c \in V(C^*)$. The following technical lemma helps us to argue that a demanding set must have two distinct blocked edges. In its statement, we use $\delta_H(A, B)$ for the collection of edges of H with one endpoint in vertex set A, and the other in B.

▶ **Lemma 6.** *Let U be a tight set with a partition $\{U_1, U_2\}$ such that $|\delta_H(U_1, U_2)| = 1$, $y_{max}(U_1) \geq y_{max}(U_2)$, and $U_2 \cap R = \emptyset$. If $u_1u_2 \in \delta_H(U_1, U_2)$ where $u_2 \in U_2$, then $z_{u_1u_2} = y_{u_2}$. Moreover, U_1 and U_2 are tight sets.*

Proof. Let $\mathcal{K}_{12} = \{C \in \mathcal{K} : R(C) \cap U_1 \neq \emptyset \wedge R(C) \cap U_2 \neq \emptyset\}$ be those full-components that intersect both U_1 and U_2. Noting that $y_{max}(U) = y_{max}(U_1)$, consider the slack of U_1:

$$\begin{aligned}
\mathrm{sl}(U_1) &= \mathrm{sl}(U_1) + \mathrm{sl}(U_2) - \mathrm{sl}(U_2) \\
&= y(U_1) + y(U_2) - y_{max}(U_1) - y_{max}(U_2) - z(E(U_1)) - z(E(U_2)) \\
&\quad - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_1| - 1)^+ - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ - \mathrm{sl}(U_2) \\
&= \mathrm{sl}(U) - y_{max}(U_2) + z_{u_1u_2} + \sum_{C\in\mathcal{K}_{12}} x_C - \mathrm{sl}(U_2).
\end{aligned}$$

We know that U is tight so that $\mathrm{sl}(U) = 0$, and our solution is feasible which means $\mathrm{sl}(U_2) \geq 0$. Also there are no terminals in U_2 so that $\sum_{C\in\mathcal{K}_{12}} x_C = 0$. From Constraint (1) on the set $\{u_1, u_2\}$ we get $z_{u_1u_2} \leq y_{u_2} \leq y_{max}(U_2)$. Hence

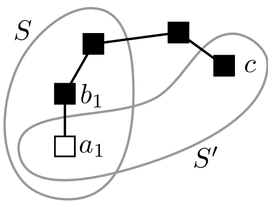
$$\mathrm{sl}(U_1) = z_{u_1u_2} - y_{max}(U_2) - \mathrm{sl}(U_2) \leq 0.$$

By feasibility, $\mathrm{sl}(U_1) \geq 0$ and therefore $\mathrm{sl}(U_1) = 0$. Moreover, the above inequality can only be satisfied with equality if $\mathrm{sl}(U_2) = 0$ and $z_{u_1u_2} = y_{max}(U_2)$. This means that also U_2 is tight, and $z_{u_1u_2} = y_{u_2} = y_{max}(U_2)$, which proves the claim. ◀

The above lemma enables us to prove that there must be a second blocked edge $a_2b_2 \in H[S]$, $a_1b_1 \neq a_2b_2$, that crosses from $S \setminus V(C^*)$ to $S \cap V(C^*)$.

▶ **Lemma 7.** *Every demanding set S has at least two blocked edges.*

Proof. Suppose that there is a single edge $a_1b_1 \in \delta_H(S \setminus V(C^*), S \cap V(C^*))$, for the sake of contradiction. Since S is a demanding set, we have $y_{max}(S \setminus V(C^*)) \geq y_{max}(S \cap V(C^*))$ and $S \cap V(C^*) \cap R = \emptyset$. Thus, by Lemma 6, $z_{a_1b_1} = y_{b_1}$.



Now consider the set $S' \cup \{b_1\}$ where S' is the blocking set for a_1b_1. Since $a_1 \in S'$ and $b_1 \notin S'$, this set includes the edge a_1b_1, while S' does not. We know that S' is tight and therefore Constraint (1) on $S' \cup \{b_1\}$ can only be feasible if a_1b_1 is the *only* edge in H added to this set, i.e. $E(S' \cup \{b_1\}) \setminus E(S')$ contains only a_1b_1. Moreover, for the same reason $S' \cup \{b_1\}$ must be tight. Also, S' contains some other vertices of $V(C^*)$, which cannot be adjacent to b_1 as otherwise $E(S' \cup \{b_1\}) \setminus E(S')$ would contain more than one edge. Hence $S' \cup \{b\}$ is a tight set for which $C^*[S' \cup \{b_1\}]$ is disconnected. This contradicts our construction of C^*. ◀

To show that the vertices a_1, a_2, b_1, b_2, and c that we have found can be used to construct a Steiner claw, we need to show that they are Steiner vertices. For this we will analyze the intersections of demanding sets and their blocking sets. In particular we show next that the intersection does not contain any terminal. Note that for our main result we can assume w.l.o.g. that the considered demanding set is inclusion-wise minimal.

▶ **Lemma 8.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. Then $S \cap S' \cap R = \emptyset$.*

To prove this lemma we need the following standard fact about tight sets sharing terminals.

▶ **Lemma 9.** *For any feasible solution (x, y, z) to (M), suppose U_1 and U_2 are tight sets, such that $U_1 \cap U_2 \cap R \neq \emptyset$. Then $U_1 \cap U_2$ and $U_1 \cup U_2$ are also tight. Also, (i) $\delta_H(U_1 \setminus U_2, U_2 \setminus U_1) = \emptyset$, where H is the support graph of (y, z), and (ii) for all $C \in \mathcal{K}$ with $x_C > 0$ and $R(C) \cap U_i \neq \emptyset$ for both $i \in \{1, 2\}$, $R(C) \cap U_1 \cap U_2 \neq \emptyset$.*

Proof. Since $U_1 \cap U_2 \cap R \neq \emptyset$, we have that

$$y_{max}(U_1) = y_{max}(U_2) = y_{max}(U_1 \cap U_2) = y_{max}(U_1 \cup U_2) = 1.$$

We also have that $S \to z(E(S))$ and $S \to (|R(C) \cap S| - 1)^+$ are both supermodular functions, which means that

$$z(E(U_1)) + z(E(U_2)) \leq z(E(U_1 \cup U_2)) + z(E(U_1 \cap U_2)), \tag{4}$$

and, for any $C \in \mathcal{K}$,

$$\begin{aligned}(|R(C) \cap U_1| - 1)^+ + (|R(C) \cap U_2| - 1)^+ \leq \\ (|R(C) \cap (U_1 \cap U_2)| - 1)^+ + (|R(C) \cap (U_1 \cup U_2)| - 1)^+.\end{aligned} \tag{5}$$

Hence

$$\mathrm{sl}(U_1 \cup U_2) + \mathrm{sl}(U_1 \cap U_2) \leq \mathrm{sl}(U_1) + \mathrm{sl}(U_2) = 0. \tag{6}$$

Each term on the left-hand side is non-negative by feasibility, and thus $U_1 \cap U_2$ and $U_1 \cup U_2$ are tight as well.

For the second part, since (6) holds with equality, (4) must hold with equality as well. This can only be if $z(\delta(U_1 \setminus U_2, U_2 \setminus U_1)) = 0$. Whenever $x_C > 0$, also (5) must hold with equality. If $R(C) \cap U_i \neq \emptyset$ for both $i \in \{1, 2\}$ then this is only possible if $R(C) \cap U_1 \cap U_2 \neq \emptyset$. ◀

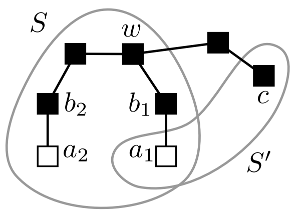
Proof of Lemma 8. Assume the claim is wrong and the intersection of S and S' contains a terminal. Consider the case when S and S' do not intersect in $V(C^*)$. Note however that both sets contain vertices of $V(C^*)$. Let $U_1 = S \cap V(C^*)$ and $U_2 = S' \cap V(C^*)$. Since $\delta_H(S \setminus S', S' \setminus S) = \emptyset$ by Lemma 9, no vertex in U_1 is adjacent to a vertex in U_2. But by the same lemma $S \cup S'$ is a tight set in which C^* is disconnected. This contradicts our construction of C^*.

Hence it must be that $S \cap S' \cap V(C^*) \neq \emptyset$. In this case we consider the set $S \cap S'$, which we know is tight by Lemma 9. We also know that one of the vertices incident to the edge that S' blocks in S is not in S', i.e. there is a vertex $b \in S$ such that $b \notin S \cap S'$. Hence $S \cap S'$ is a strict subset of S, which contains no terminal of C^*. However it does contain some terminal and a vertex from $V(C^*)$, and is therefore a demanding set. This contradicts the minimality of S. ◀

Using the insight of Lemma 8 we can finally prove that a demanding set implies the existence of a Steiner claw. We distinguish the cases of whether the demanding set and its blocking set intersect in C^*.

▶ **Lemma 10.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. If $S \cap S' \cap V(C^*) = \emptyset$ then there is a Steiner claw.*

Proof. By Lemma 7, S has two blocked edges a_1b_1 and a_2b_2 with $a_i \notin V(C^*)$ and $b_i \in V(C^*)$ for $i \in \{1, 2\}$. We know that for $i \in \{1, 2\}$, b_i must be a Steiner vertex since $S \cap R(C^*) = \emptyset$, and the same is true for a_i by Lemma 8. Let S' be a blocking set for a_1b_1. Recall that in Algorithm 1 we first add Steiner vertices and only then terminals. Since S' blocks the addition of the Steiner vertex a_1, it was already a blocking set in Step 2 of the algorithm, before any terminals had been added to C^*. Thus there exists some Steiner vertex $c \in V(C^*) \cap S'$. Also note that by the assumptions of the lemma, $c \notin S$.

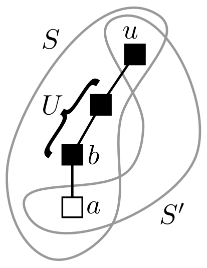


Since $C^*[S]$ is connected, there must be a path P on the Steiner vertices of $C^*[S]$ from b_1 to b_2. Every Steiner vertex of P has at least two Steiner neighbors, since a_1 and a_2 also are Steiner vertices and are neighbors to the endpoints of P. Note that c cannot be part of P since c is not in S. However it is in the component C^*. Let w be the first vertex of P reached by the unique path Q in C^* from c to P. The path Q has non-zero length, and since a_1 and a_2 are not in C^*, Q contains neither a_1 nor a_2. Moreover, Q contains only Steiner vertices since c also is a Steiner vertex. Hence w has at least three Steiner neighbors: two since it is in P and an additional one from Q. ◀

We are left with the case where demanding and blocking sets intersect in C^*. Hence the following lemma completes the proof of correctness for Algorithm 2, and therefore also that of Theorem 1.

▶ **Lemma 11.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. If $S \cap S' \cap V(C^*) \neq \emptyset$ then there is a Steiner claw.*

Proof. Assume the claim is false, so that the intersection of S and S' contains a vertex of C^* and every Steiner vertex has at most two Steiner neighbors. Let ab be an edge blocked by S'. Consider a path P in $H[S]$ starting with ab and continuing from b along vertices of C^* until a vertex $u \in S \cap S'$ is reached, i.e. $V(P) \cap S' = \{a, u\}$ and $V(P) \setminus V(C^*) = \{a\}$. Such a path exists since $C^*[S]$ is connected.



Let $U = V(P) \setminus S'$ be the vertices on P exclusively in S. Note that this set is non-empty since $b \in U$, and it contains only Steiner vertices since $S \cap R(C^*) = \emptyset$. Our goal is to show that the z-values of $E(P)$ are small compared to the y-values of U. As a consequence we will see that removing U from S gives a demanding set, thus contradicting the minimality of S.

The edge set $E(S' \cup U)$ is a superset of $E(S') \cup E(P)$. Thus from Constraint (1) on the set $S' \cup U$ we can conclude that

$$\begin{aligned} &z(E(S')) + z(E(P)) + \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ \\ &\leq z(E(S' \cup U)) + \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ \\ &\qquad\qquad \leq y(S' \cup U) - y_{max}(S' \cup U) = y(S') + y(U) - y_{max}(S' \cup U). \end{aligned}$$

Note that $\sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ = \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S'| - 1)^+$ since U contains only Steiner vertices. Hence substituting $z(E(S'))$ from Constraint (1) on the tight set S' in the above inequality and eliminating superfluous terms gives $z(E(P)) \leq y(U) + y_{max}(S') - y_{max}(S' \cup U) \leq y(U)$.

We now remove U from S and bound $z(E(S \setminus U))$. Consider an edge $vw \in E(S)$ where $v \in U$ and $w \in S \setminus (U \cup \{u, a\})$. Since any Steiner vertex from U has at most two Steiner neighbors, w must be a terminal, i.e. a maximum valued vertex in S. However S does not contain any terminals of C^* and therefore vw must be a blocked edge for S. This also means that there is a blocking set S'' preventing w to be part of C^*. However by Lemma 8, S and S'' do not share terminals. Hence such an edge vw cannot exist. This means that the edges in $E(S)$ can be partitioned into $E(S \setminus U)$ and $E(P)$ and therefore $z(E(S \setminus U)) = z(E(S)) - z(E(P))$.

Some vertex v with maximum value in S lies outside of C^* and is therefore not contained in U, and thus $y_{max}(S) = y_{max}(S \setminus U)$. Note that $\sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S \setminus U)| - 1)^+ = \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S| - 1)^+$, as above. Hence the tightness of S together with the inequality $z(E(P)) \leq y(U)$ gives

$$\begin{aligned} z(E(S)) - z(E(P)) &\geq y(S) - \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S| - 1)^+ - y_{max}(S) - y(U) \\ &= y(S \setminus U) - y_{max}(S \setminus U) - \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S \setminus U)| - 1)^+. \end{aligned}$$

Due to Constraint (1) on $S \setminus U$ of our feasible solution, this implies that $S \setminus U$ must be tight. However this set is a strict subset of S, intersects $V(C^*)$ (for instance at u), and it contains the maximum valued vertex v, which is not in C^*. Hence S was not an inclusion-wise minimal demanding set, which is a contradiction. ◀

4 Algorithmic issues

In order to show that Algorithm 2 can be implemented efficiently, we need to show that (i) the number of iterations of the algorithm is polynomial, and (ii) that we can compute the correct choice of C^* in each iteration, and the amount that we should extract.

4.1 Bounding the number of iterations

We prove the following:

▶ **Theorem 12.** *Given a Steiner tree instance with n nodes and m edges, the number of iterations of Algorithm 2 is $O(n^2)$.*

Let the Steiner tree instance be described by $G = (V, E)$ and terminal set R. Let (y^0, z^0) be the initial solution to BCR, which we extend to a solution (x^0, y^0, z^0) of HYP with $x^0 = \mathbb{0}$. Let (x^i, y^i, z^i) denote the solution obtained after i iterations, i.e., i components have been maximally extracted. Let i_{max} denote the index of the final iteration, so $y^{i_{max}} = \chi(R)$ and $z^{i_{max}} = \mathbb{0}$.

We will first observe that once a set becomes tight, it remains tight from then on.

▶ **Lemma 13.** *For all $i \leq j$, if $S \subseteq V$ is tight in iteration i, then it is tight in iteration j.*

Proof. An immediate corollary of Lemma 3. ◀

It is also clear that if $i \leq j$, then $z^i_e = 0$ implies that $z^j_e = 0$, and $y^i_v = 0$ implies that $y^j_v = 0$. At the end of each iteration, a new constraint must become tight, and this constraint must be independent of, i.e. not implied by, the previously tight constraints. So in order to bound the number of iteration, it is enough to show that the number of independent tight constraints can never be too large. This we will show via standard combinatorial uncrossing arguments, albeit with some technicalities.

Let $\mathcal{K}' = \{C \in \mathcal{K} : x_C^{i_{max}} > 0\}$. Let $\mathcal{R} = 2^V \,\dot\cup\, E$ denote the set of constraints of (M) corresponding to (1) and the nonnegativity constraints for z. For any $\ell \in \mathcal{R}$, let $\Gamma(\ell)$ denote row ℓ of the constraints matrix of (M); so $\Gamma(\ell)$ is a vector in $\mathbb{R}^{V \cup E \cup \mathcal{K}'}$.

▶ **Lemma 14.** *For any $i \in [i_{max}]$, and any two sets S_1, S_2 with $S_1 \cap S_2 \cap R \neq \emptyset$ that are tight in iteration i,*

$$\Gamma(S_1) + \Gamma(S_2) - \Gamma(S_1 \cup S_2) - \Gamma(S_1 \cap S_2) \in \operatorname{span}(\{\Gamma(e) : z^i_e = 0\}).$$

Proof. First, since S_1 and S_2 remain tight in the final iteration, and $x_C^{i_{max}} > 0$ for all $C \in \mathcal{K}'$, we may deduce from Lemma 9 applied to $(x^{i_{max}}, y^{i_{max}}, z^{i_{max}})$ that there are no components "crossing" S_1 and S_2, meaning that if $R(C)$ intersects both S_1 and S_2, it must intersect $S_1 \cap S_2$. It follows that for any $C \in \mathcal{K}'$,

$$f_C(S_1) + f_C(S_2) = f_C(S_1 \cup S_2) + f_C(S_1 \cap S_2),$$

where $f_C(S) := (|R(C) \cap S| - 1)^+ = |R(C) \cap S| - [R(C) \cap S \neq \emptyset]$ is the coefficient of x_C for the constraint corresponding to S in (M).

Let $F = \{e \in \delta(S_1 \setminus S_2, S_2 \setminus S_1) : z^i_e = 0\}$. We may deduce from Lemma 9, this time applied to (x^i, y^i, z^i), that $z^i(\delta(S_1 \setminus S_2, S_2 \setminus S_1)) = 0$. Hence

$$\chi(E(S_1)) + \chi(E(S_2)) = \chi(E(S_1 \cup S_2)) + \chi(E(S_1 \cap S_2)) + \chi(F).$$

Since also $\chi(S_1) + \chi(S_2) = \chi(S_1 \cup S_2) + \chi(S_1 \cap S_2)$, the lemma follows. ◀

▶ **Lemma 15.** *Fix any $i \in [i_{max}]$. Let $\mathcal{R}_{tight} \subseteq \mathcal{R}$ be the subset of $\mathcal{R}$ that are tight constraints in (x^i, y^i, z^i). Then*

$$\dim \operatorname{span}(\{\Gamma(\ell) : \ell \in \mathcal{R}_{tight}\}) = O(n^2).$$

Proof. Let $\mathfrak{F} = \{\Gamma(e) : e \in E, z^i_e = 0\}$. Fix any $r \in R$, and let

$$\mathfrak{R}_r = \{\Gamma(S) : r \in S \text{ and } \operatorname{sl}(S) = 0\} \cup \mathfrak{F}.$$

Let $\mathcal{L}_r$ be a maximal laminar family of tight sets in (x^i, y^i, z^i) containing r; so in fact, $\mathcal{L}_r$ is a chain. We claim that

$$\operatorname{span}(\{\Gamma(S) : S \in \mathcal{L}_r\} \cup \mathfrak{F}) = \operatorname{span}(\mathfrak{R}_r).$$

This follows immediately from Lemma 14 by an argument of Jain [13]. If for some tight set U with $r \in U$, $\Gamma(U)$ was not in $\operatorname{span}(\{\Gamma(S) : S \in \mathcal{L}_r\} \cup \mathfrak{F})$, we could uncross S w.r.t. $\mathcal{L}_r$ to obtain a strictly larger laminar family, a contradiction.

Applying this reasoning for each $r \in R$, we conclude that

$$\operatorname{span}\left(\left\{\Gamma(S) : S \in \bigcup_{r \in R} \mathcal{L}_r\right\} \cup \mathfrak{F}\right) = \operatorname{span}(\Gamma(\ell) : \ell \in \mathcal{R}_{\text{tight}}).$$

Since $|\mathcal{L}_r| = O(n)$ and $|\mathfrak{F}| = O(m)$, the result follows. ◀

Theorem 12 is proven.

4.2 Determining the minimal tight sets, and the duration of each iteration

The main observation here will be that checking if a solution (x, y, z) is feasible for (M), as well as checking for tight sets under certain constraints, can be reduced to solving certain maximum flow problems. This will allow for the efficient determination of the component C^* for each iteration, as well as the duration of each iteration using parametric search methods. The construction extends one for HYP described in [11] (as well as classical results for separation over the forest polytope); no major new ideas are needed, though for convenience some aspects of the construction are different.

We construct the directed graph $D = (W, A)$ with capacities ξ as follows. Let $W = V \cup \{r_C : C \in \mathcal{K}, x_C > 0\} \cup \{s, t\}$, where r_C is a new vertex for each component C, and s and t will be source and sink vertices. Let $M = \sum_{C \in \mathcal{K}} x_C$. For each $e \in \operatorname{supp}(z)$, add both orientations of the edge to A, giving both arcs capacity $\frac{1}{2}z(e)$; for each $r_C \in W \setminus V$, add an arc of capacity x_C from r_C to t, and infinite capacity arcs from each terminal in $R(C)$ to r_C. For each $v \in V$, add the arc sv with capacity $M + \frac{1}{2}z(\delta(v))$, and the arc vt with capacity $M + y_v - \sum_{C \in \mathcal{K} : v \in R(C)} x_C$. The role of M is solely to ensure that all capacities are nonnegative.

▶ **Theorem 16.** *Let S, T be two disjoint subsets of V, with S nonempty and satisfying $\max_{w \in S} y_w = \max_{w \in V \setminus T} y_w$. Given a (feasible or infeasible) solution (x, y, z) to (M), a set $U^* \subseteq V$ is of minimal slack under the constraint $S \subseteq U^* \subseteq (V \setminus T)$ if and only if $U^* \cup \{r_C \in W \setminus V : R(C) \cap S \neq \emptyset\}$ is a minimum capacity $(\{s\} \cup S)$-$(\{t\} \cup T)$-cut in D.*

Note that, for example, in order to find an overall minimal slack set U^*, one may first guess $w \in V$ s.t. $y_w = y_{max}(U^*)$. Then apply the above theorem with $T = \{v \in V : y_v > y_w\}$ and $S = \{w\}$. Trying all possibilities for w, U^* can be found with n maximum flow computations.

Proof. Observe that if Q is an $(\{s\}\cup S)$-$(\{t\}\cup T)$-cut in D, but with $r_C \notin Q$ for some $C \in \mathcal{K}$ where $R(C) \cap Q \neq \emptyset$, then $\xi(\delta^+_D(Q)) = \infty$. Conversely, if $r_C \in Q$ but $R(C) \cap Q = \emptyset$, then removing r_C from Q yields a cut of strictly smaller capacity.

So consider any $(\{s\} \cup S)$-$(\{t\} \cup T)$-cut Q satisfying $\{C \in \mathcal{K} : r_C \in Q\} = \{C \in \mathcal{K} : R(C) \cap Q \neq \emptyset\}$. Let $U = Q \cap V$. Then

$$\begin{aligned}
\xi(\delta^+_D(Q)) &= \sum_{v\in U}\Big(M + y_v - \sum_{C\in\mathcal{K}:v\in R(C)} x_C\Big) + \tfrac{1}{2}z(\delta_G(U)) \\
&\quad + \sum_{v\in V\setminus U}\Big(M + \tfrac{1}{2}z(\delta_G(\{v\}))\Big) + \sum_{C\in\mathcal{K}:C\cap R(U)\neq\emptyset} x_C \\
&= M\cdot|V| + y(U) + z(E) - z(E(U)) - \sum_{C\in\mathcal{K}} x_C(|R(C)\cap U| - 1)^+ \\
&= \mathrm{sl}(U) + M\cdot|V| + y_{max}(U) + z(E).
\end{aligned}$$

By the conditions on S and T, $y_{max}(U) = \max_{w\in S} y_w$. Thus all terms in the above aside from $\mathrm{sl}(U)$ are independent of U. The result follows. ◀

Choosing C^*.

Given a solution (x, y, z) to (M) and any Steiner vertex ℓ with $y_\ell > 0$, we will now show how the choice of C^* described in Section 2 can be efficiently computed.

Suppose we are considering adding $v \in V$ to our current C^*, with $vu \in \mathrm{supp}(z)$ and $u \in V(C^*) \setminus R$. (Here, v could be either a Steiner node, if we are in step 2, or a terminal if we are in step 3.) Let C' be the component obtained by adding v and vu to C^*. The only reason to not add v is that there is some tight set U for which C' would be disconnected in U. By assumption, C^* is connected in U. Thus $u \notin U$, and $v \in U$. By trying all possibilities for w which might be a maximizer of y in U, and hence applying Theorem 16 with $S = \{w, v\}$ and $T = \{u\} \cup \{v' \in V : y_{v'} > y_w\}$, we can determine whether such a tight set U exists or not, and hence whether v should be added to C^*.

The choice of ε in an iteration.

What remains is to determine what value ε should take in a particular iteration. Let (x, y, z) denote the solution at the start of the iteration, and let $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ denote the solution after an amount ε of the current component C^* has been extracted. As before, let $\mathrm{sl}_\varepsilon(S)$ denote the slack of set S in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.

It is of course easy to determine the maximum value of ε such that all nonnegativity constraints remain satisfied. So the main challenge is to determine ε such that a new tight set U forms (which would then be violated if a larger value of ε was chosen). It is clearly sufficient to compute, for each $w \in V$, the maximum value of ε such that $\min_{U\subseteq V: y_w = y_{max}(U)} \mathrm{sl}_\varepsilon(U) \geq 0$. (We may then simply take the minimum over all the values of ε obtained).

The maximum flow problem we have constructed has capacities that are linear functions of (x, y, z). Moreover, $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ is a linear function of ε. Thus a parametric maximum flow algorithm can be applied [16].

References

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

2 J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, 60(1):6:1–6:33, 2013.

3 D. Chakrabarty, J. Könemann, and D. Pritchard. Hypergraphic LP relaxations for Steiner trees. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 383–396, 2010.

4 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*, pages 567–570, 2010.

5 M. Chlebík and J. Chlebíková. Approximation hardness of the Steiner tree problem on graphs. In *Proceedings, Scandinavian Workshop on Algorithm Theory*, pages 170–179, 2002.

6 J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71B:233–240, 1967.

7 J. Edmonds. Matroids and the greedy algorithm. *Math. Programming*, 1:127–136, 1971.

8 DIMACS Center for Discrete Mathematics and Theoretical Computer Science. 11th DIMACS implementation challenge in collaboration with ICERM: Steiner tree problems. http://dimacs11.cs.princeton.edu/, 2014.

9 I. Fung, K. Georgiou, J. Könemann, and M. Sharpe. Efficient algorithms for solving hypergraphic steiner tree relaxations in quasi-bipartite instances. *CoRR*, abs/1202.5049, 2012.

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

11 M. X. Goemans, N. Olver, T. Rothvoß, and R. Zenklusen. Matroids and integrality gaps for hypergraphic steiner tree relaxations. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1161–11762, 2012.

12 F.K. Hwang, D.S. Richards, and P. Winter. *The Steiner tree problem.* Monograph in Annals of Discrete Mathematics, 53. Elsevier, 1992.

13 K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 448–457, 1998.

14 R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, NY, 1972.

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

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

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

18 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.

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

20 R. T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Math. Programming*, 28:271–287, 1984.