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

Andreas Emil Feldmann*[2], Jochen Könemann*[1], Neil Olver†[3], and Laura Sanità*[1]

[1]Department of Combinatorics and Optimization, University of Waterloo
{jochen,laura.sanita}@uwaterloo.ca
[2]SZTAKI, Hungarian Academy of Sciences & KAM, Charles University in Prague
feldmann.a.e@gmail.com
[3]Vrije Universiteit Amsterdam & CWI
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 strongly 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 article, 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 three Steiner neighbours. 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.

1 Introduction

In an instance of the well-studied, NP-hard [5, 15] Steiner tree problem one is given an undirected graph $G=(V,E)$, a non-negative cost $\text{cost}(e)$ for each edge $e \in E$, and a set of *terminals* $R \subseteq V$. The goal is to find a minimum-cost tree in G spanning R. Steiner trees arise in a host of practical applications (e.g., see the survey [13] and the recent DIMACS implementation challenge [8]), and therefore have been extensively studied in the network design community.

In this article, we focus on the problem's efficient approximability. It is well-known that computing a minimum spanning tree in the metric closure of the graph gives a 2-approximation [10, 26]. A number of papers give algorithms with improved running times while maintaining this approximation ratio [18, 20, 25, 28, 30]. Later the approximation ratio was improved in a sequence of papers [16, 22, 24, 31], culminating in the recent breakthrough by Byrka et al. [2] who present 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. [12] 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. [12] also showed that solving them exactly is strongly NP-hard, and known approaches to obtain a good approximation require 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, 29]. BCR is an appealing relaxation as its compactness implies efficient solvability. As one way of

*Supported by an NSERC Discovery grant.
†Supported by an NWO Veni grant.

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. [12] in order to compute a Steiner tree with cost at most $\ln(4)$ times that of the given HYP solution. It is known that HYP is in general a strictly stronger formulation than BCR, so we cannot hope to always compute a solution to HYP of the same cost as the optimum BCR solution. We therefore ask when these two LPs have the same strength.

A second motivation for considering this question is that 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 upper bound known is an almost trivial bound of 2. HYP on the other hand has an integrality gap of at least $8/7 \approx 1.14$ [17] and at most $\ln(4) \approx 1.39$ [12]. Hence bounding the integrality gap of BCR in terms of the smaller gap of HYP can help to better understand the former gap.

Previously it was known that BCR and HYP are equally strong for *quasi-bipartite* instances where no two Steiner vertices (the vertices in $V \setminus R$) are connected by an edge [4, 9, 12]. For these graphs the Steiner tree problem remains NP-hard [23]. In this article we significantly extend the class of instances where the two relaxations are equivalent. In our main result, we show that as long as the input graph G has no Steiner vertex with three Steiner neighbours (we will refer to this as a *Steiner claw*), BCR and HYP are polyhedrally equivalent. Specifically, we will provide an efficiently computable cost-preserving 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 $\text{cost}(uv)$ 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, 29]). 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 $z(A) = \sum_{a \in A} z_a$, the LP is

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

In this article, we importantly choose to work with an equivalent *undirected* formulation (see [11]) 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 $\max_{v \in S} y_v$.

$$\begin{aligned} \min \quad & \sum_{e \in E} z_e \, \text{cost}(e) \qquad \text{s.t.} && && \text{(BCR)} \\ & z(E(S)) \leq y(S) - y_{max}(S) && \forall S \subseteq V \\ & z(E) = y(V) - 1 \\ & y_t = 1 && \forall t \in R \\ & y, z \geq \mathbb{0} \end{aligned}$$

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, i.e. $R = V$. In a feasible integral solution, y_v denotes whether vertex v is part of the Steiner tree or not. Note that in any feasible fractional solution, $y_v \leq 1$ for every $v \in V$, just by considering the constraint for $S = V$, which would otherwise contradict the equality constraint. So y_v intuitively represents "how much" node v is in the fractional solution.

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 [11]).

The value z_e for an edge in BCR is given by the sum of the corresponding two arc values in BCR*. For $v \in V \setminus R$, 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).

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 [4]. Corresponding to our undirected choice of BCR, we will henceforth focus on the hypergraphic subtour relaxation introduced in [27]. The LP has one variable for each *component* of the instance. A component is a tree in G containing at least one terminal, and in which every terminal contained in the tree is a leaf. We let $\mathcal{K}$ be the set of all components of the instance. The cost of a component is equal to the sum of the cost of its edges. In the following hypergraphic subtour formulation we use $(a)^+$ as shorthand for $\max\{0, a\}$, and use $R(C)$ to denote the set of terminals included in the component C.

$$\begin{aligned} \min \quad & \sum_{C \in \mathcal{K}} x_C \operatorname{cost}(C) \qquad \text{s.t.} && && \text{(HYP)} \\ & \sum_{C \in \mathcal{K}} x_C (|R(C) \cap S| - 1)^+ \leq |S| - 1 && \forall S \subseteq R, S \neq \emptyset \\ & \sum_{C \in \mathcal{K}} x_C (|R(C)| - 1)^+ = |R| - 1 \\ & x \geq \mathbb{0} \end{aligned}$$

To interpret this LP, we should first note that it is usually defined using a smaller set than $\mathcal{K}$: normally only *full components*, which are components where in addition every leaf is a terminal, are used. Observe that there is always an optimal solution to HYP supported only on full components, since any component can be replaced by a full component containing the same set of terminals, which will only decrease the cost without affecting any of the constraints. We allow these non-minimal components for convenience in the definition of the algorithm, and to obtain a clean polyhedral correspondence. This will be discussed later; for now, it may be helpful to think only about full components.

A feasible integral solution x to HYP corresponds to the Steiner tree T obtained by taking the union of all edges in components C with $x_C = 1$ (these components will be edge-disjoint in any optimal integral feasible solution). Any Steiner tree can be uniquely described in this way, by its decomposition into components. The coefficients $(|R(C) \cap S| - 1)^+$ can be interpreted as the amount of connectivity that component C contributes to the set $S \subseteq R$. The main constraint can then be seen as saying that if one considers just a set $S \subseteq R$, the total amount of connectivity should be at most $|S| - 1$. The equality constraint simply says that in addition, the terminal set itself is connected by the components. Also notice that if all components contain at most two terminals, this collapses to just the standard formulation of the spanning tree polytope.

As mentioned, solving HYP exactly is strongly NP-hard [12]. 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 $\varepsilon > 0$. The number of variables and constraints will consequently be more than $|R|^{2^{1/\varepsilon}}$, i.e., doubly exponential in $1/\varepsilon$.

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 neighbours. In other words, the subgraph induced by the Steiner nodes does not contain $K_{1,3}$ as a subgraph; we emphasize that this is different from saying that this graph is "claw-free" in the usual sense, which only prohibits $K_{1,3}$ as an *induced* subgraph. 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 $O(|R| \cdot |E|)$ variables and constraints.

Theorem 1. *In a Steiner claw-free instance, any solution to BCR can be efficiently converted to a solution to HYP of equal cost.*

Our proof also shows a polyhedral result. Let $P_{\text{BCR}} \subseteq \mathbb{R}_+^{V \dot\cup E}$ and $P_{\text{HYP}} \subseteq \mathbb{R}_+^{\mathcal{K}}$ be the polytopes of feasible solutions to BCR and HYP respectively. There is a natural projection map π from $\mathbb{R}_+^{\mathcal{K}}$ to $\mathbb{R}_+^{V \dot\cup E}$

defined by

$$\begin{aligned}\pi(x)_t &= 1 && \forall t \in R\\ \pi(x)_v &= \sum_{C\in\mathcal{K}: v\in V(C)} x_C && \forall v \in V \setminus R\\ \pi(x)_e &= \sum_{C\in\mathcal{K}: e\in E(C)} x_C && \forall e \in E.\end{aligned} \tag{1}$$

Clearly π preserves the cost, i.e., $\sum_{C\in\mathcal{K}} x_C \operatorname{cost}(C) = \sum_{e\in E} \pi(x)_e \operatorname{cost}(e)$. We will show that our proof of Theorem 1 yields the following.

Theorem 2. *For any Steiner claw-free instance, $\pi(P_{HYP}) = P_{BCR}$.*

As an immediate consequence of Theorem 1, we obtain an integrality gap bound of $\ln(4) \approx 1.39$ for BCR in Steiner claw-free instances via [12], 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 [3, 12] showed that their integrality gap is at most $73/60 \approx 1.216$.

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 an instance with 4 Steiner vertices inducing a subgraph with only one vertex of degree 3 where BCR and HYP are not equivalent. The optimum Steiner tree has cost 6 and this is also the value of HYP; the BCR optimum (shown) has cost 5.5. We prove a larger gap of 8/7 in Section 6.

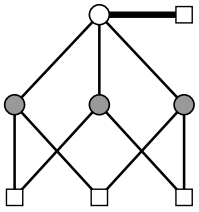
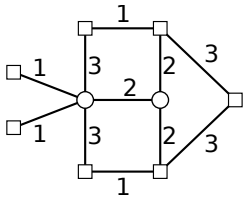


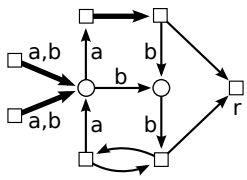
Figure 1: Example instance with $\pi(P_{\text{HYP}}) \neq P_{\text{BCR}}$. Terminals are squares, Steiner vertices are circles. All edges have unit cost. BCR admits a solution of cost 5.5: let $z_e = 1$ for the thick edge, and $z_e = 1/2$ otherwise. All white vertices v in the figure have $y_v = 1$, and others have $y_v = 1/2$.

At a high level, our algorithmic proof of Theorem 1 follows the greedy approach taken in [9, 12] for quasi-bipartite instances. Roughly, these 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 the *directed component relaxation* (see [21]). This directed formulation is equivalent to HYP [4], and we will refer to it as HYP*. For HYP*, each 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 components $\vec{\mathcal{K}}$. By $\Delta^+(S)$ we denote all components $C \in \vec{\mathcal{K}}$ for which the head lies outside S, while some other terminal of C lies inside. Also let $x(\Delta^+(S)) = \sum_{C\in\Delta^+(S)} x_C$. Given a root $r \in R$, the directed hypergraphic relaxation then is:

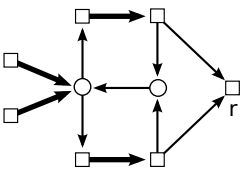
$$\begin{aligned}\min \quad & \sum_{C\in\vec{\mathcal{K}}} x_C \operatorname{cost}(C) \quad \text{s.t.} && && \text{(HYP*)}\\ & x(\Delta^+(S)) \geq 1 && \forall S \subseteq R \setminus \{r\}, S \neq \emptyset\\ & x \geq \mathbb{0}\end{aligned}$$



(a) Edge costs.



(b) Decomposable: one 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 HYP* solution, and others (c) are not (we omit the proof). In the BCR* solutions the root is marked r, bold arcs have capacity $z_e = 1$, and the others $z_e = 1/2$.

The approach of [9, 12] is to iteratively and greedily *shave off* fractional capacity uniformly from the arcs of a directed 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 HYP* of the same cost as the original BCR* solution. As soon as Steiner vertices are allowed to have Steiner neighbours, the above strategy runs into problems, however. Figure 2(a) shows a Steiner claw-free instance, and two optimal solutions to BCR* in Figure 2(b) and 2(c). One can show that there is no HYP* solution whose canonical projection (the directed analogue of the map π discussed earlier) yields the BCR* solution in 2(c). Hence the outlined greedy strategy taken in [9, 12] will not work here. On the other hand, the solution given in 2(b) is the projection of a feasible solution to HYP*. The difficulty is that BCR* and HYP* have many extreme points that do not correspond to extreme points in their undirected counterparts. Indeed, the polytope described by BCR* is simply the standard relaxation of the *directed* Steiner tree problem (which is known to have a large gap [32]); the additional fact that opposing arcs have equal cost must be put in "by hand" in the objective function. So while the solutions Figure 2(b) and 2(c) are different solutions to BCR*, and in fact both are extreme point solutions, they project to the same undirected solution of BCR.

The results for the quasi-bipartite case [4, 9, 12] 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. A more general uncrossing lemma will be a key technical tool in our analysis.

Theorem 2 shows that the property of being Steiner claw-free, which is polynomially checkable, is a sufficient condition for equivalence of BCR and HYP. We also show that there is no good characterization of this equivalence, even if we try to go very slightly beyond the Steiner claw-free case.

Theorem 3. *It is NP-hard to decide for a given Steiner tree instance whether BCR has the same optimum value as HYP, even if we restrict to instances where the Steiner vertices induce a single star.*

Outline of the article. We first describe our algorithm for turning a solution of BCR into a solution of HYP in Section 2. We then prove correctness of this algorithm in Section 3, and its efficiency in Section 4. The proof of Theorem 3 can be found in Section 5, and the 8/7 lower bound instance between BCR and HYP is in Section 6. Finally we end with some concluding remarks in Section 7.

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, 12]. Crucially, however, we will be using the undirected relaxations BCR and HYP introduced in Section 1 instead of their directed analogs.

Our algorithm begins by computing a solution to BCR. It then gradually transforms this into a solution to HYP, by repeatedly picking components to "extract". To facilitate the discussion of this process, we define the following "mixed LP" MIX, which is a hybrid of BCR and HYP.

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

$$z(E(S)) + \sum_{C\in\mathcal{K}} x_C(|R(C)\cap S| - 1)^+ \le y(S) - y_{max}(S) \qquad \forall S \subseteq V \tag{2}$$

$$z(E) + \sum_{C\in\mathcal{K}} x_C(|R(C)| - 1)^+ = y(V) - 1 \tag{3}$$

$$y_v = 1 \qquad \forall v \in R \tag{4}$$

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

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 MIX 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.

The high level algorithm is described in Algorithm 1. The first step is to transfer any edges with non-zero z-value and directly connecting terminals, since they are already components. This does not affect the feasibility or cost of our solution, and it will be convenient in what follows that every edge in

Algorithm 1 Converting a BCR solution to a HYP solution.

1: Start with a solution (x, y, z) feasible for MIX with $x = \mathbb{0}$.
2: For any $z_{vw} > 0$ with $v, w \in R$, move all weight to $x_{\{v,w\}}$.
3: **while** $y \neq \chi(R)$ **do**
4: Apply FindComponent(x, y, z) to compute a component C^*.
5: Choose $\varepsilon > 0$ maximally such that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ remains feasible for MIX.
6: Replace (x, y, z) with $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.
7: **end while**
8: Output x feasible for HYP.

the support of z is adjacent to at least one Steiner node. The main part of the algorithm proceeds by repeatedly identifying a component C^* in the support of (y, z). We delay for now the description of how C^* is chosen. For a carefully chosen $\varepsilon > 0$, we then decrease z_e for $e \in E(C^*)$ and y_v for $v \in V(C^*) \setminus R$ by ε, and simultaneously increase x_{C^*} by ε. The notation $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ refers to the result of this modification; we call this *extracting* the component C^*, and ε is the amount extracted. (We do not explicitly indicate the component being extracted in the notation $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, but it will always be denoted by C^*.) The value ε is chosen to be as large as possible while maintaining a feasible solution to MIX. Note that this extraction procedure does not change the cost of the solution.

Observe that at the end of the algorithm, $y = \chi(R)$ and also $z = \mathbb{0}$. This is because for every edge uv with $v \notin R$, Constraint (2) on the set $S = \{u, v\}$ implies $z_{uv} \leq y_v = 0$. Moreover, we explicitly moved all z-value from edges between terminals. Hence if the algorithm succeeds, it computes a solution to HYP of the same cost as the solution to BCR we started from.

We also see immediately that if the algorithm always succeeds on Steiner claw-free instances, then Theorem 2 holds. With the projection π as defined in (1), the definition of extraction ensures that $\pi(x) + (y, z)$ remains unchanged in each iteration. Hence $\pi(P_{\text{HYP}}) \subseteq P_{\text{BCR}}$. But the reverse containment $P_{\text{BCR}} \subseteq \pi(P_{\text{HYP}})$ also holds, since it is well-known that HYP is always at least as strong as BCR [21].

Let us now investigate the effect of extracting some component C^*, and see why extreme care will be needed in choosing this component. In order to demonstrate that some given C^* is satisfactory, the key thing we must show is that some *strictly positive* amount of C^* can be extracted while maintaining feasibility. Once we know that we make some progress towards shifting the weight from (y, z) to x in each iteration, it is a small further step to show that the number of iterations is finite and hence the algorithm terminates in a feasible solution to HYP.

The component C^* will always be chosen within the support of the current solution, so $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ will be non-negative for sufficiently small $\varepsilon > 0$. Furthermore, extracting C^* does not change the value of y_v for any terminal $v \in R$, and so (4) remains satisfied. So only (2) and (3) are of concern.

Define the *slack* of a vertex set $S \subseteq V$ with respect to some solution (x, y, z) of MIX as

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

Note that a solution (x, y, z) to MIX is feasible precisely if $\text{sl}(S) \geq 0$ for all $S \subseteq V$, and $\text{sl}(V) = 0$. We will call a set S *tight* if $\text{sl}(S) = 0$. The slack of a set $S \subseteq V$ depends linearly, and so certainly continuously, on the solution (x, y, z). Moreover, it turns out that the slack of a set can only decrease (or stay the same) upon extracting a component. This is natural, since HYP is always at least as strong as BCR (see the proof of Lemma 8 in Section 3 for a formal argument). In particular, the slack of V can never become positive. So if it is impossible to extract any strictly positive amount of C^* while remaining feasible, the reason must be that the slack of some currently tight set would immediately become negative. Understanding the structure of tight sets, and choosing C^* in a way that interacts well with them, is thus crucial.

We denote by $C^*[S]$ the subgraph of C^* induced by the vertices in $S \cap V(C^*)$. We say that C^* *is connected in* S if $C^*[S]$ is connected. Let H be the *support graph* of (y, z), where $V(H) = \{v \in V : y_v > 0\}$ and $E(H) = \{e \in E : z_e > 0\}$. The algorithm FindComponent that we use to compute C^* is described in Algorithm 2. It greedily adds vertices to C^* as long as the component is still connected in all tight sets. Importantly, it first adds as many Steiner vertices as possible to C^* before adding terminals.

In order to show that our proposed algorithm works correctly and efficiently, we must answer the following:

1. Why can we extract a positive amount of the component C^* in each iteration?

2. Why does the algorithm terminate, and moreover, involve only polynomially many iterations?

3. How do we efficiently implement FindComponent, and compute the amount of C^* to extract in each iteration?

We will proceed to answer these questions in the following sections, first showing that the algorithm is correct in Section 3, and then that it can be implemented efficiently in Section 4.

3 Correctness of the algorithm

3.1 Some properties of tight sets

A partial uncrossing lemma. It is well known that if S and S' are two tight sets in some solution to BCR which are *terminal-intersecting*, meaning that $S \cap S'$ contains a terminal, then S and S' can be *uncrossed*: $S \cup S'$ and $S \cap S'$ will both be tight sets. This is already a very useful result; as we will see later, uncrossing terminal-intersecting tight sets is already enough to give a proof of the equivalence in the quasi-bipartite setting using the undirected BCR and HYP formulations. (Recall that the previous proofs of this result [9, 12] work with the directed formulations.) We will then build on this observation to show that the same is true for Steiner-claw free graphs. To show our main result for these instances however, it will not be sufficient to only uncross terminal-intersecting tight sets. Here we prove a more general uncrossing result for sets which do not necessarily contain any common terminals. Unlike standard uncrossing results, this lemma may only partially uncross sets. That is, it implies the tightness of two sets that are smaller than the union and larger than the intersection, respectively. (see Figure 3).

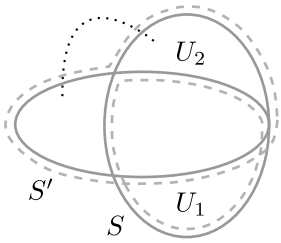


Figure 3: Partial uncrossing of two tight sets S and S'. $S \setminus S'$ can be partitioned into $U_1 \cup U_2$ such that $S \setminus U_2$ contains vertex v with $y_v = y_{max}(S)$, and solution (x, y, z) does not shortcut U_1 and U_2. Then sets $S \setminus U_2$ and $S' \cup U_2$ are tight as well.

Before stating our main lemma, we introduce some useful terminology.

Definition 4. Let S, T be two subsets of V. We say that an edge e *shortcuts* S and T if it has one endpoint in $S \setminus T$ and the other in $T \setminus S$. We say that a component C shortcuts S and T if $R(C) \cap S \neq \emptyset$ and $R(C) \cap T \neq \emptyset$, but $R(C) \cap S \cap T = \emptyset$. Given a solution (x, y, z) to MIX, we say that the solution shortcuts S and T if there is an edge in the support of z or a component in the support of x that shortcuts S and T.

Lemma 5. *For any feasible solution (x, y, z) to MIX, suppose S and S' are tight sets, such that $S \setminus S'$ can be partitioned into two sets U_1 and U_2 with the following properties:*

1. *there is a vertex $v \in S \setminus U_2$ with $y_v = y_{max}(S)$, and*
2. *(x, y, z) does not shortcut U_1 and U_2.*

Then

(i) both $S \setminus U_2$ and $S' \cup U_2$ are tight sets, and

(ii) (x, y, z) does not shortcut $S \setminus U_1$ and S'.

Proof. We will use $[\![X]\!]$ to denote the indicator function of the predicate X, and $\text{supp}(x)$ to denote the support of a vector x. Observing that $(|R(C) \cap S| - 1)^+ = |R(C) \cap S| - [\![R(C) \cap S \neq \emptyset]\!]$, and

Algorithm 2 FindComponent(x, y, z).

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 neighbouring a vertex $w \in V(C^*)$ in H for which $C^* \cup \{vw\}$ is still connected in every tight set, add the edge vw to C^*.
3: As long as there is a terminal t neighbouring a Steiner vertex $w \in V(C^*)$ in H for which $C^* \cup \{tw\}$ is still connected in every tight set, add the edge tw to C^*.
4: Return C^*.

$[\![e \in E(S)]\!] = |e \cap S| - [\![e \cap S \neq \emptyset]\!]$, we can rewrite the slack function as

$$\mathrm{sl}(S) = \underbrace{y(S) - \sum_{e \in E(H)} z_e |e \cap S| - \sum_{C \in \mathrm{supp}(x)} x_C |R(C) \cap S|}_{g(S)} \underbrace{- y_{max}(S) + \sum_{e \in E(H)} z_e [\![e \cap S \neq \emptyset]\!] + \sum_{C \in \mathrm{supp}(x)} x_C [\![R(C) \cap S \neq \emptyset]\!]}_{h(S)}.$$

We now claim that g is modular; i.e., for any $A, B \subseteq V(H)$, we have $g(A) + g(B) = g(A \cap B) + g(A \cup B)$. To see that this is true, note that $y(A) + y(B) = y(A \cap B) + y(A \cup B)$. Furthermore, edge $e \in E(H)$ contributes the same amounts to $|e \cap A| + |e \cap B|$ as it does to $|e \cap (A \cup B)| + |e \cap (A \cap B)|$. Finally,

$$|R(C) \cap A| + |R(C) \cap B| = |R(C) \cap (A \cup B)| + |R(C) \cap (A \cup B)|,$$

for any $C \in \mathrm{supp}(x)$, and therefore $g(S \setminus U_2) + g(S' \cup U_2) = g(S) + g(S')$. Since $y_{max}(S \setminus U_2) = y_{max}(S)$ by assumption, and $y_{max}(S' \cup U_2) \geq y_{max}(S')$, clearly $y_{max}(S \setminus U_2) + y_{max}(S' \cup U_2) \geq y_{max}(S) + y_{max}(S')$.

It remains to show that $h(S \setminus U_2) + h(S' \cup U_2) \leq h(S) + h(S')$, with equality only if (x, y, z) does not shortcut $S' \setminus S$ and U_2. We will then obtain

$$\mathrm{sl}(S \setminus U_2) + \mathrm{sl}(S' \cup U_2) \leq \mathrm{sl}(S) + \mathrm{sl}(S') = 0,$$

so that (by feasibility) indeed $S \setminus U_2$ and $S' \cup U_2$ are tight, and condition (ii) holds.

Claim. *For any two sets A, B, $h(A \cup B) + h(A \cap B) \leq h(A) + h(B)$, with equality if and only if (x, y, z) does not shortcut A and B.*

Proof. Consider the contribution of any component $C \in \mathrm{supp}(x)$ to both sides of the claimed inequality. If $R(C) \cap (S \cup T) = \emptyset$, then it does not contribute at all. If $R(C) \cap S \cap T \neq \emptyset$, then it contributes exactly 2 to both sides. If $R(C) \cap S \cap T = \emptyset$ and $R(C) \cap (S \cup T) \neq \emptyset$, then certainly $R(C)$ intersects at least one of S and T. Moreover if C does not shortcut S and T, then it intersects exactly one of S and T. The argument for the contribution of an edge $e \in \mathrm{supp}(y)$ is similar. □

By the claim, and assumption 2,

$$h(S) + h(S \setminus (U_1 \cup U_2)) = h(S \setminus U_1) + h(S \setminus U_2). \tag{5}$$

Again by the claim,

$$h(S' \cup U_2) + h(S \cap S') \leq h(S \setminus U_1) + h(S'), \tag{6}$$

with equality only if (x, y, z) does not shortcut $S \setminus U_1$ and S'. Subtracting (5) from (6) (using $S \setminus (U_1 \cup U_2) = S \cap S'$) and rearranging, we obtain

$$h(S' \cup U_2) + h(S \setminus U_2) \leq h(S') + h(S),$$

with equality under the same conditions. This completes the proof. □

We obtain the following corollary immediately by choosing $U_1 = \emptyset$ in Lemma 5.

Corollary 6. *For any feasible solution (x, y, z) to MIX, suppose S and S' are tight sets such that $y_{max}(S \cap S') = y_{max}(S)$. Then $S \cap S'$ and $S \cup S'$ are both tight, and (x, y, z) does not shortcut S and S'.*

This corollary in turn is a generalization of the standard uncrossing of terminal-intersecting tight sets referred to earlier, since if $S \cap S'$ contains a terminal, then $y_{max}(S \cap S') = y_{max}(S) = 1$.

Connectivity of tight sets. We continue with one further useful observation about tight sets.

Lemma 7. *Let S be a tight set of a feasible solution (x, y, z) to MIX, and let H be the support graph of (y, z). If $S \cap R \neq \emptyset$, then every connected component of $H[S]$ contains a terminal. If $S \cap R = \emptyset$, then $H[S]$ is connected.*

Proof. Assume the statement is false. Regardless of whether S contains terminals or not, there must then be a connected component in $H[S]$ with vertex set U_1, such that $U_1 \cap R = \emptyset$ and $U_2 := V(H[S]) \setminus U_1$ is

non-empty. In particular, $E(S) = E(U_1) \cup E(U_2)$, $|R(C) \cap U_1| = 0$ for every full component $C \in \mathcal{K}$, and $y_{max}(U_2) > 0$. Thus,

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

By feasibility of U_1 and U_2, $U_1 \cup U_2$ cannot be tight, a contradiction to S being tight. □

3.2 Tight sets and feasibility of extraction

We begin by characterizing when a tight set S will remain feasible upon extracting some small amount of a component C^*. If S does not intersect $V(C^*)$, then clearly $\mathrm{sl}(S)$ is unchanged. Otherwise we have the following.

Lemma 8. *Let $S \subseteq V$ be tight in a feasible solution (x, y, z) to MIX and C^* a component in the support graph H of (y, z) with $V(C^*) \cap S \neq \emptyset$. Then there exists some $\varepsilon > 0$ such that 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 $S_m = \{v \in S : y_v = y_{max}(S)\}$. We will use $[\![X]\!]$ to denote the indicator function of the predicate X. We use $\mathrm{sl}_\varepsilon(S)$ for the slack of set S in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, and obtain

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

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

Now consider the case where $S \cap R \neq \emptyset$. Since $y_{max}(S) = 1$ (and this remains true after extracting C^*), 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 $|V(C^*[S \setminus R])| + (|R(C^*) \cap S| - 1)^+ \leq |E(C^*[S])|$. Then, simplifying further, S stays feasible if and only if $|V(C^*[S])| - [\![R(C^*) \cap S \neq \emptyset]\!] \leq |E(C^*[S])|$. 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. □

The goal is now to apply Lemma 8 to show that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ satisfies (2) for some $\varepsilon > 0$ whenever there is no Steiner claw. By construction (see Algorithm 2), $C^*[S]$ 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 violating the second condition of Lemma 8. Such a set "demands" that at least one of its vertices maximizing y be included in C^* if it is to remain feasible.

Definition 9. 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.

Our goal is now to show that there are no demanding sets. This has two consequences: First, it implies that we can feasibly shift a positive amount of the value of y and z variables associated with C^* to x_{C^*}. Second, it will imply that C^* contains at least one terminal. For consider the set V; by the equality constraint of MIX, it is certainly tight. But for it not to be a demanding set, C^* must contain a terminal.

As a warmup, let us consider the quasi-bipartite special case. Then $V(C^*) \setminus R$ consists of only a single element ℓ. Suppose for a contradiction that S is a demanding set for C^*. Then S contains no terminals of C^*, but intersects C^*, so $\ell \in S$. The set S contains more than just ℓ so that its maximizer is outside of C^*, and it is connected in H by Lemma 7. Hence S contains at least one terminal t adjacent to ℓ. Since we did not include t during the execution of Algorithm 2, it must be that some other tight set S' prevented its addition. So S' is disconnected in $C^* \cup \{\ell t\}$, but not in C^*; hence $t \in S'$ but $\ell \notin S'$, and at least one other terminal t' of $R(C^*)$ adjacent to ℓ is in S'. Since S and S' are terminal-intersecting, we can uncross them due to Corollary 6, and so in particular no edge in H shortcuts S and S'. But the edge $t'\ell$ shortcuts S and S', which is a contradiction.

The proof for the general Steiner claw-free case proceeds along similar lines, but we will need the more general partial uncrossing lemma in order to handle sets that are not terminal-intersecting.

Suppose for a contradiction that there exists at least one demanding set, and let S be an inclusion-wise minimal example. Again, we know that $S \cap V(C^*) \neq \emptyset$ and that $C^*[S]$ is connected. Consider a path P in $H[S]$ that connects $S \cap V(C^*)$ to some vertex $u \in S \setminus V(C^*)$ with $y_u = y_{max}(S)$ (e.g., a terminal). By Lemma 7 this path exists, whether or not S contains terminals. Traversing the path from u, let b be the first vertex of C^*, and let a be its immediate predecessor (see Figure 4). Note that b must be a Steiner vertex, otherwise S would not be a demanding set.

Definition 10. Given an edge $vw \in E(H)$ with $v \in V(C^*) \setminus R$ and $w \notin V(C^*)$, we say that a set $S' \subseteq V$ *blocks* vw if S' is tight and $C^* \cup \{vw\}$ is disconnected, and we then call S' a *blocking set*.

A blocking set provides a reason that a certain edge was not added to C^* by Algorithm 2. Since a was not added, there must be some tight set S' that blocks ab. Note that S' contains a, not b, but some other vertex $c \in V(C^*)$.

The following lemma applied to the blocking set S' shows in particular that a is a Steiner vertex.

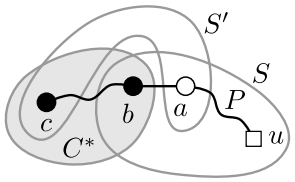


Figure 4: Interaction of a demanding set S and a blocking set S'.

Lemma 11. *For the inclusion-wise minimal demanding set S and any blocking set $\bar{S}$, $y_{max}(S \cap \bar{S}) < y_{max}(S)$.*

Proof. Assume for a contradiction that there exists a vertex $v \in S \cap \bar{S}$ with $y_v = y_{max}(S)$. Consider the case when S and $\bar{S}$ do not intersect in $V(C^*)$. Note however that both sets contain vertices of $V(C^*)$. Let $U_1 = S \cap V(C^*)$ and $U_2 = \bar{S} \cap V(C^*)$. Since $\delta_H(S \setminus \bar{S}, \bar{S} \setminus S) = \emptyset$ by Corollary 6, no vertex in U_1 is adjacent to a vertex in U_2. But by the same lemma $S \cup \bar{S}$ is a tight set in which C^* is disconnected. This contradicts our construction of C^*.

Hence it must be that $S \cap \bar{S} \cap V(C^*) \neq \emptyset$. In this case we consider the set $S \cap \bar{S}$, which we know is tight by Corollary 6. We also know that one of the vertices incident to the edge that $\bar{S}$ blocks is not in $\bar{S}$, i.e., there is a vertex $b \in S$ such that $b \notin S \cap \bar{S}$. Hence $S \cap \bar{S}$ is a strict subset of S, which contains no terminal of C^*. However it does contain the vertex v with $y_v = y_{max}(S) = y_{max}(S \cap \bar{S})$ and a vertex from $V(C^*)$, and is therefore a demanding set, whether or not S contains terminals. This contradicts the minimality of S. □

Now we exploit the fact that Algorithm 2 first adds as many Steiner nodes to C^* as possible before attempting to add terminals. Since a is a Steiner vertex, it follows that S' was a blocking set in Step 2 of the algorithm, i.e., even before any terminals were added to C^*. Thus $S' \cap V(C^*)$ contains at least one Steiner node. Relabelling if necessary, we may thus assume that c is a Steiner node.

Observe that already, we can deduce that b has at least two Steiner neighbours in H; a, and the node adjacent to b on the path from b to c in C^*.

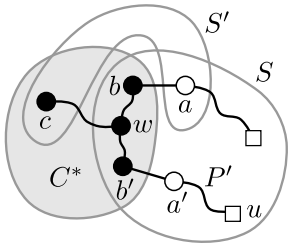


Figure 5: Finding the Steiner-claw given a demanding set.

Lemma 12. *There is a path P' in $H[S \setminus S']$ from b to a vertex u with $y_u = y_{max}(S)$.*

Before giving the proof, let us see how the existence of a Steiner claw follows from this. Since S is a demanding set, along the path P' there must be a blocked edge $a'b'$, where $a' \in S \setminus V(C^*)$ and $b' \in S \cap V(C^*)$ (see Figure 5). Analogous to a and b, we can argue that also a' and b' are Steiner vertices due to Lemma 11 and the fact that a' was not added to C^* by Algorithm 2. Furthermore, $a \neq a'$ since P' lies outside of S'. Starting from b, we can assume that a' is the first vertex not in C^* on P'. Hence we get a path P'' from a to b and then along P' to a', that only consists of Steiner vertices. Again since P' lies outside of S', we know that $c \in V(C^*) \cap S'$ is not part of P''. But there is a path of Steiner vertices from c to P'' in C^*, which ends at a vertex w different from a and a'. This Steiner vertex w therefore has three Steiner neighbours, and Theorem 1 follows.

Proof of Lemma 12. The proof of this lemma roughly follows the same lines as the proof of Lemma 11, but uses Lemma 5 instead of Corollary 6.

Assume the claim is false, and there is no path in $H[S \setminus S']$ from b to any vertex of $M := \{u \in S : y_u = y_{max}(S)\}$. Let U_2 contain the vertices of the connected component of $H[S \setminus S']$ that includes b, and let U_1 contain all other vertices of $S \setminus S'$. By Lemma 11, $M \cap S' = \emptyset$. Hence U_1 must contain a vertex $v \in M$. Since U_2 does not contain any vertex of M, it also does not contain any terminals. Thus there cannot be any full component $C \in \mathcal{K}$ for which $R(C) \cap U_2 \neq \emptyset$. We conclude that all conditions listed in Lemma 5 are fulfilled by U_1 and U_2.

Consider the case when S and S' do not intersect in $V(C^*)$. Note however that both sets contain vertices of $V(C^*)$. Let $W_1 = S' \cap V(C^*)$ and $W_2 = S \cap V(C^*)$. Since $b \in U_2 \cap V(C^*)$ and all vertices of C^* are connected in the tight set S, $W_2 \subseteq U_2$. On the other hand, $W_1 \subseteq S' \setminus S$. Since $\delta_H(S' \setminus S, U_2) = \emptyset$ by Lemma 5, no vertex in W_1 is adjacent to a vertex in W_2. But by the same lemma $S' \cup U_2$ 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 $S \setminus U_2$, which we know is tight by Lemma 5. We also know that $b \in U_2$, which means that $S \setminus U_2$ is a strict subset of S, which contains no terminal of C^*. However $S \setminus U_2$ contains v where $y_v = y_{max}(S) = y_{max}(S \setminus U_2)$, and it contains some vertex of C^* since $S \cap S' \cap V(C^*) \neq \emptyset$ and $U_2 \cap S' = \emptyset$. Thus (whether or not S contains terminals) $S \setminus U_2$ is a demanding set, which contradicts the minimality of S. □

Finally, we observe that in each iteration, either an edge is removed from the support of the current solution y, or a new set becomes tight. By Lemma 8, once a set becomes tight it remains so for the remainder of the algorithm. This yields an exponential bound on the number of iterations, which is sufficient to complete the demonstration of the correctness of the algorithm. A more refined analysis in Section 4 provides a polynomial bound.

4 Efficiency of the algorithm

In order to show that Algorithm 1 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 13. *Given a Steiner tree instance with n nodes, and m edges, the number of iterations of Algorithm 1 is at most $n^2 + m$.*

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 $z^{i_{max}} = \mathbb{0}$.

Each iteration ends because a constraint (which was not tight at the beginning of the iteration) becomes tight. There are two types of constraints that we need to consider: those corresponding to (2), and nonnegativity constraints for z. We need not consider the nonnegativity constraints for y, because in any feasible solution to MIX, $y_v = 0$ if and only if $z(\delta(\{v\}) = 0$.

By Lemma 8, if a set $S \subseteq V$ is tight in some iteration i, then it remains tight in all subsequent iterations. It is also clear that if $z^i_e = 0$ then $z^j_e = 0$ for all $j \geq i$. 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, in particular following an argument of Jain [14], albeit with some technicalities.

Let $\mathcal{K}' = \{C \in \mathcal{K} : x_C^{i_{max}} > 0\}$; all other components have zero value throughout the execution of the algorithm, and can be ignored. So from now on, we think of the columns of the constraint matrix of MIX as being indexed by $V \cup E \cup \mathcal{K}'$. We may index the constraints (2), unpacked as

$$z(E(S)) + \sum_{C \in \mathcal{K}'} x_C(|R(C) \cap S| - 1)^+ \leq y(S - \{v\}) \qquad \forall S \subseteq V, v \in S,$$

by a pair (v, S) where $v \in S$. So we may index the relevant rows (including the nonnegativity constraints for z) of the constraint matrix with

$$\mathcal{R} := \{(v, S) : S \subseteq V, v \in S\} \cup E.$$

Let Γ_ℓ denote the row vector of the constraint matrix indexed by row $\ell \in \mathcal{R}$.

From now on, fix some arbitrary iteration $i \in [i_{max}]$. Let $\mathcal{T} \subseteq \mathcal{R}$ be the set of constraints for which (x^i, y^i, z^i) is tight. Note that $(v, S) \in \mathcal{T}$ precisely when S is tight and $y_v = y_{max}(S)$. Also let $E_0 := \mathcal{T} \cap E$ be the tight nonnegativity constraints, and $\mathcal{T}_v := \{(v, S) : (v, S) \in \mathcal{T}\}$ be the tight constraints involving a given $v \in V$. We will use $\text{span}(\mathcal{S})$ to denote the vector space spanned by the row vectors Γ_ℓ for all $\ell \in \mathcal{S}$. Our goal is to show that

$$\dim \text{span}(\mathcal{T}) \leq n^2 + m.$$

Since this dimension must increase by at least one in each iteration, this suffices to prove the theorem.

We first note that tight constraints in $\mathcal{T}_v$ can be uncrossed, roughly maintaining their span.

Lemma 14. *For any $v \in V$, $S_1, S_2 \subseteq V$ with $(v, S_1) \in \mathcal{T}$ and $(v, S_2) \in \mathcal{T}$, we have $(v, S_1 \cup S_2) \in \mathcal{T}$ and $(v, S_1 \cap S_2) \in \mathcal{T}$, and moreover*

$$\Gamma_{v,S_1} + \Gamma_{v,S_2} - \Gamma_{v,S_1 \cup S_2} - \Gamma_{v,S_1 \cap S_2} \in \text{span}(E_0).$$

Proof. We have that S_1 and S_2 are tight, with $y_v = y_{max}(S_1) = y_{max}(S_2)$. Hence by Corollary 6 $S_1 \cup S_2$ and $S_1 \cap S_2$ are tight, and of course $y_v = y_{max}(S_1 \cup S_2) = y_{max}(S_1 \cap S_2)$. So indeed $(v, S_1 \cup S_2)$ and $(v, S_1 \cap S_2)$ are in $\mathcal{T}$.

To show the required relation between the rows, we consider in turn the columns corresponding to x_C for $C \in \mathcal{K}'$, z_e for $e \in E$, and y_v for $v \in V$. 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 Corollary 6 applied to $(x^{i_{max}}, y^{i_{max}}, z^{i_{max}})$ that there are no components in the support of x that shortcut S_1 and 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 MIX (recall that $[\![X]\!]$ denotes the indicator of predicate X).

Let $F = \delta(S_1 \setminus S_2, S_2 \setminus S_1) \cap E_0$. Corollary 6, this time applied to (x^i, y^i, z^i), implies that $z^i(\delta(S_1 \setminus S_2, S_2 \setminus S_1)) = 0$. Hence

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

Finally since $\chi(S_1) + \chi(S_2) = \chi(S_1 \cup S_2) + \chi(S_1 \cap S_2)$, we have the required dependency between the rows, and the lemma follows. □

Lemma 15. *Fix any $v \in V$. Then there exists $\mathcal{S}_v \subseteq \mathcal{T}_v$ with $|\mathcal{S}_v| \leq n$ and for which*

$$\text{span}(\mathcal{S}_v \cup E_0) = \text{span}(\mathcal{T}_v \cup E_0).$$

Proof. Let $S_1 \subset S_2 \subset \cdots \subset S_t$ be a maximal chain in $\{S : (v, S) \in \mathcal{T}_v\}$. Let $\mathcal{S}_v := \{(v, S_i) : i \in [t]\}$. We will show that this satisfies the requirements of the lemma.

Clearly $|\mathcal{S}_v| \leq n$. Suppose for a contradiction that there is some $U \subseteq V$ such that $\Gamma_{v,U} \in \text{span}(\mathcal{T}_v \cup E_0)$, but $\Gamma_{v,U} \notin \text{span}(\mathcal{S}_v \cup E_0)$. Choose U so that it crosses the fewest number of sets in the chain, i.e., so that $|\{i : S_i \not\subseteq U \text{ and } S_i \not\supseteq U\}|$ is as small as possible. Because of the maximality of the chain, U must certainly cross at least one, say S_i. Now it can easily be shown that $U \cap S_i$ crosses fewer sets in the chain [14], so $\Gamma_{v,U \cap S_i} \in \text{span}(\mathcal{S} \cup E_0)$ by our choice of U. Similarly, $\Gamma_{v,U \cup S_i} \in \text{span}(\mathcal{S}_v \cup E_0)$. But then Lemma 14 implies that $\Gamma_{v,U} \in \text{span}(\mathcal{S}_v \cup E_0)$, a contradiction. □

We can now bound the dimension of span($\mathcal{T}$). Choose $\mathcal{S}_v$ for each $v \in V$ as per the above lemma. Then

$$\text{span}(\mathcal{T}) = \text{span}\Big(\bigcup_{v \in V} \mathcal{S}_v \ \cup \ E_0 \Big),$$

so

$$\dim \text{span}(\mathcal{T}) \leq \sum_{v \in V} |\mathcal{S}_v| \ + \ |E_0| \leq n^2 + m.$$

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 MIX, 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 for the bounding of the duration of each iteration using parametric search methods. The construction extends one for HYP described in [12] (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 with $z_e > 0$, 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 MIX, 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

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

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

Choosing C^*. Given a solution (x, y, z) to *MIX* 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 $z_{vu} > 0$ 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^*.

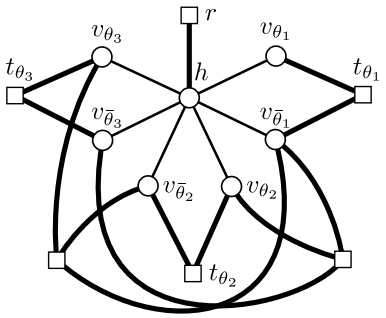


Figure 6: The graph G_φ for $\varphi \equiv (\bar{\theta}_1 \vee \theta_2 \vee \bar{\theta}_3) \wedge (\bar{\theta}_1 \vee \bar{\theta}_2 \vee \theta_3)$. Bold edges have cost $b-1$ and thin edges cost 1.

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 instance 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 [19].

5 Deciding equivalence of BCR and HYP

Our main result of this article shows that the property of being Steiner claw-free, which is polynomially checkable, is a sufficient condition for equivalence of the two relaxations. We show here that there is no good characterization for equivalence. Even if we restrict to instances where the Steiner vertices induce a single star (or alternatively, by splitting Steiner vertices and adding zero-cost edges, a single binary tree), deciding equivalence is NP-hard (Theorem 3).

In this section we consider the equivalent [4, 11] directed versions of BCR and HYP called BCR* and HYP*, as introduced in Section 1. The reduction is from the NP-hard 3-SAT problem (see Figure 6). Given a 3-SAT formula φ with a variables and b clauses we construct the following Steiner tree instance G_φ. We introduce a root terminal r and a Steiner vertex h which we call the *hub* of the instance. The hub h is connected to the root r via an edge of cost $b-1$. For every variable θ of the 3-SAT formula we introduce two Steiner vertices v_θ and $v_{\bar{\theta}}$, one for each possible literal of θ. Each vertex v_θ and $v_{\bar{\theta}}$ is connected to the hub h by an edge of cost 1. Additionally we introduce a terminal t_θ for variable θ and connect the Steiner vertices v_θ and $v_{\bar{\theta}}$ to t_θ by an edge of cost $b-1$ each. For every clause Γ of the 3-SAT formula we introduce a terminal t_Γ which is connected by an edge to each of its literal Steiner vertices. That is, if $\lambda \in \Gamma$ then there is an edge between t_Γ and v_λ. Each such edge has cost $b-1$.

We claim that the optimum solution to BCR* is equal to the optimum of HYP* for G_φ if and only if φ is satisfiable. We first analyze the optimum solution to BCR* in G_φ regardless of whether φ is satisfiable or not. For this we need the dual of BCR* which has a variable β_S for each set $S \subseteq V \setminus \{r\}$ containing at least one terminal.

$$\max \sum_{S \subseteq V \setminus \{r\} : S \cap R \neq \emptyset} \beta_S \qquad \text{s.t.} \qquad \text{(BCR*-dual)}$$

$$\sum_{S \subseteq V \setminus \{r\} : S \cap R \neq \emptyset \wedge a \in \delta^+(S)} \beta_S \leq \mathrm{cost}(a) \qquad \forall a \in \vec{E}$$

$$\beta \geq \mathbb{0}$$

Lemma 17. *Let φ be a 3-SAT formula with a variables and b clauses. The optimum solution to BCR* of G_φ has value $ab + b^2 - 1$.*

Proof. We give a primal and dual solution to BCR*, each of value $ab + b^2 - 1$. The primal solution is given by setting the arc variable as follows.

- (h, r) to 1.
- (t_θ, v_λ) and (v_λ, h) to $1/2$ for each variable θ of φ and each literal λ of θ.
- (t_Γ, v_λ) to $1/3$ for each clause Γ and each literal $\lambda \in \Gamma$.

The dual solution to BCR*-dual is given by setting the variables of the sets as follows.

- $\{t_\theta\}$ and $\{t_\Gamma\}$ to $b - 1$ for each clause Γ and variable θ of φ.
- $\{t_\theta, v_\theta, v_{\bar{\theta}}\}$ to 1 for each variable θ of φ.
- $V \setminus \{r\}$ to $b - 1$.

It is easy to see that both these solutions are feasible and have the claimed value. □

In case φ is satisfiable we now show that the integer solution has the same value as the solution to BCR*. From this it also follows that the solution to HYP* has the same value, since HYP* is stronger than BCR*.

Lemma 18. *Let φ be a satisfiable 3-SAT formula with a variables and b clauses. The optimum integer solution of G_φ has value $ab + b^2 - 1$.*

Proof. Fix a satisfying assignment for φ. We pick the arc hr to be part of the integer solution. For any variable θ the assignment sets one of its literals λ to true. We include the edges $t_\theta v_\lambda$ and $v_\lambda h$ in our solution. Since the assignment is satisfying, for any clause Γ of φ one of its literals λ is set to true. We include the arc $t_\Gamma v_\lambda$ in the solution. It is easy to see that the value of this solution is as claimed. □

In case φ is unsatisfiable we provide a dual solution to HYP* which gives a lower bound on the optimum value. The dual has a variable α_S for each set $S \subseteq R \setminus \{r\}$ that is non-empty.

$$\max \sum_{S \subseteq R \setminus \{r\} : S \neq \emptyset} \alpha_S \qquad \text{s.t.} \qquad \text{(HYP*-dual)}$$

$$\sum_{S \subseteq R \setminus \{r\} : C \in \Delta^+(S)} \alpha_S \leq \text{cost}(C) \qquad \forall C \in \vec{\mathcal{K}} \qquad (7)$$

$$\alpha \geq \mathbb{0}$$

Lemma 19. *Let φ be an unsatisfiable 3-SAT formula with a variables and b clauses. The optimum solution to HYP* for G_φ has value at least $ab + b^2$.*

Proof. The dual solution is given by setting the variables of each singleton set $\{t_\theta\}$ and $\{t_\Gamma\}$ to b for each variable θ and clause Γ of φ. It is easy to see that the solution has the claimed value. We still need to argue its feasibility.

Let $C \in \vec{\mathcal{K}}$ be a component of G_φ that does not contain the hub h. It must be a star with one Steiner vertex v_λ as the center which corresponds to some literal λ. All other vertices of C are terminals. Moreover at most one of the terminals of C corresponds to a variable θ of φ (λ is a literal of θ), while all others correspond to clauses. It cannot be that all clauses of φ contain the literal λ since otherwise φ would be satisfiable, and thus $|R(C)| \leq b$. Since one of these terminals is going to be the root of C, the dual on C (i.e. the left-hand side of (7)) is $(|R(C)| - 1)b$. The cost of C is $|R(C)|(b-1)$, which is at least the dual for $|R(C)| \leq b$, so that (7) is satisfied.

Now let C be a component of G_φ that contains the hub h, and let $V_\lambda(C) = V(C) \setminus (R \cup \{h\})$ denote the literal Steiner vertices of C. Each literal Steiner vertex must be connected to the hub with an edge of cost 1. Hence the cost of C is $|R(C)|(b-1) + |V_\lambda(C)|$. Let R_Γ and R_θ denote the set of clause terminals and variable terminals of G_φ, respectively. Recalling that $[\![X]\!]$ denotes the indicator of predicate X, the dual on C is at most $(|R_\Gamma(C)| + |R_\theta(C)| + [\![r \in V(C)]\!] - 1)b$, since one of the terminals of C is its head.

First consider the case when $|R_\Gamma(C)| \leq b - 1$. Note that each variable terminal needs a literal Steiner vertex to connect to the hub, while a literal Steiner vertex can be used by at most one variable terminal for this purpose. Hence $|V_\lambda(C)| \geq |R_\theta(C)|$ and we can lower-bound the cost of C by

$$\begin{aligned} |R(C)|(b-1) + |R_\theta(C)| = \\ (|R_\Gamma(C)| + |R_\theta(C)| + [\![r \in V(C)]\!])(b-1) + |R_\theta(C)| = \\ (|R_\Gamma(C)| + |R_\theta(C)| + [\![r \in V(C)]\!] - 1)b + (b-1) - |R_\Gamma(C)| + 1 - [\![r \in V(C)]\!]. \end{aligned}$$

Since $|R_\Gamma(C)| \leq b-1$ and $[\![r \in V(C)]\!] \leq 1$, the cost is lower-bounded by the dual on C.

Finally consider the case when $|R_\Gamma(C)| = b$, i.e. all clause terminals are in C. Each clause terminal connects to the hub through a literal Steiner vertex in C. If C would contain at most one of the literal Steiner vertices v_θ and $v_{\bar{\theta}}$ for each variable θ, then the literals of C would induce an assignment of the variables of φ. Since C contains all clause terminals and they connect through the literal Steiner vertices, this assignment would satisfy φ. This contradiction means that for some variable θ both literal Steiner vertices v_θ and $v_{\bar{\theta}}$ are part of C. As before, each variable terminal needs a literal Steiner vertex to connect to the hub, while a literal Steiner vertex can be used by at most one variable terminal for this purpose. That is, t_θ (if it is part of C) can only use one of the vertices v_θ and $v_{\bar{\theta}}$. Therefore $|V_\lambda(C)| \geq |R_\theta(C)| + 1$ and we can lower-bound the cost of C by

$$\begin{aligned}|R(C)|(b-1) + |R_\theta(C)| + 1 = \\ (|R_\Gamma(C)| + |R_\theta(C)| + [\![r \in V(C)]\!])(b-1) + |R_\theta(C)| + 1 = \\ (|R_\Gamma(C)| + |R_\theta(C)| + [\![r \in V(C)]\!] - 1)b + b - |R_\Gamma(C)| + 1 - [\![r \in V(C)]\!].\end{aligned}$$

Since $|R_\Gamma(C)| = b$ and $[\![r \in V(C)]\!] \leq 1$, the cost is lower-bounded by the dual on C. □

From the above lemmas it follows that the optimum solution to BCR* of G_φ is equal to the optimum solution to HYP* if and only if φ is satisfiable. This proves Theorem 3.

6 An 8/7 lower bound example

Figure 1 gives a lower bound of $12/11 \approx 1.09$ on the worst-case ratio between the optima of BCR and HYP. In this section we present a more involved example that gives a ratio of $8/7 \approx 1.14$. It is based on the example given by Byrka et al. [2], which bounds the integrality gap of BCR* by $36/31 \approx 1.16$. The idea is to modify this example so that the optimum solution to HYP* is equal to the integer solution. Byrka et al. [2] use a recursive construction based on Skutella's graph [17]. We will base our construction on the topology of the graph given in Figure 1 instead, which essentially is a smaller version of Skutella's graph. We leave it as an open question whether Skutella's graph can also be used in order to obtain a stronger lower bound.

For any nonnegative integer p, define the graph $\tilde{G}_p$ as follows. The graph will have $p+2$ "levels"; the nodes at level j (for any $1 \leq j \leq p+1$) are labelled by the elements of $\{1,2,3\}^j$. Nodes at level $p+1$ are all terminals, and nodes at level $j \in \{1,2,\ldots,p\}$ are all Steiner nodes. At level 0, there is one Steiner node h called the *hub node*, and also an adjacent terminal node r connected only to h. All nodes at level 1 are connected to h. For any $t \in \{1,2,3\}$, let $b(t)$ denote the binary representation of t, considered as a vector in $\{0,1\}^2$. There is an edge between a node $v = (v_1, \ldots v_i)$ on level $i \in \{1,\ldots,p\}$ and a node $u = (u_1,\ldots,u_{i+1})$ on level $i+1$, if $v_j = u_j$ for all $j \in \{1,\ldots i-1\}$ and $b(v_i) \cdot b(u_i) \equiv_2 1$. All edges of $\tilde{G}_p$ have unit cost.

Now define G_p to be the graph obtained from $\tilde{G}_p$ by removing all terminals $v = (v_1, v_2, \ldots, v_{p+1})$ at level $p+1$ for which $v_{p+1} \neq 1$. These instances (taking a limit over p) will provide the lower bound. Note that G_1 is precisely the graph depicted in Figure 1. An important difference between our construction and the one of Byrka et al. [2] is the additional edge between the hub and the root. This edge essentially forces the optimum solution of HYP* to be equal to the optimum integer solution.

It is easy to see that a feasible solution to BCR* for G_p is to direct all arcs upwards towards the root, and install the following capacities on the arcs (see also [2]): $z_{(h,r)} = 1$ on the arc between the hub h and the root r, $z_a = 1/2$ for all arcs $a = (v,h)$ between Steiner vertices v of level 1 and the hub h, $z_a = 1/2$ for all arcs $a = (u,v)$ between terminals u on level $p+1$ and Steiner vertices v on level p, and $z_a = 1/4$ on all remaining arcs $a = (u,v)$ between Steiner vertices of level $i+1$ and i for all $i \in \{1,\ldots,p-1\}$. In the support graph of this solution the out-degree of any vertex on level $i \in \{2,\ldots,p+1\}$ is 2. Hence the contributed cost of the arcs incident to terminals on level $p+1$ is $2 \cdot 3^p/2$, and for arcs having some Steiner vertex on level $i \in \{2,\ldots,p\}$ as their tail the contribution is $2 \cdot 3^i/4$. The arcs connecting the Steiner vertices on level 1 to the hub contribute $3/2$, and the arc from the hub to the root adds a cost of 1. Hence the total cost of the solution is

$$3^p + \sum_{i=2}^{p} \frac{3^i}{2} + \frac{3}{2} + 1 = \frac{7}{4} \cdot 3^p + \frac{1}{4}.$$

We go on to show that the optimum solution to HYP* has cost at least $2 \cdot 3^p$ (in fact this is also the cost of an optimum integral solution; cf. [2]). Letting p tend to infinity, the ratio between the optimum

values of BCR* and HYP* is thus 8/7, as claimed. We prove the lower bound on the optimum of HYP* by considering the following solution to HYP*-dual: $\alpha_{\{v\}} = 2$ for all $v \in R \setminus \{r\}$, and $\alpha_S = 0$ for all other subsets. Clearly this solution has cost $2 \cdot 3^p$. We need to show feasibility, which we do by proving that constraint (7) is valid.

Since all edge costs of G_p are 1, constraint (7) is valid if and only if any directed full component $\vec{C} \in \vec{\mathcal{K}}$ contains at least as many arcs as the total dual on $\vec{C}$ given by the suggested solution. The only relevance of the orientation of $\vec{C}$ is that the head does not contribute to the dual. So it is sufficient to show that for any undirected component C of G_p, $|E(C)| \geq 2|R(C)| - 2$.

We prove the following strengthened claim by induction. Applying this claim to a component C of G_p (where we think of G_p as a subgraph of $\tilde{G}_p$), and placing 2 tokens on each terminal of C, yields the required inequality.

Claim. *Let p be a nonnegative integer, and C a connected subgraph of $\tilde{G}_p$. Suppose tokens are assigned to the terminals of C, with the property that: i) each $v \in R(C)$ receives 0, 1 or 2 tokens, and ii) the total number of tokens assigned to $\{(v_1, \ldots, v_p, i) \in R(C) : i \in \{1,2,3\}\}$ is at most 3, for any $v_1, \ldots, v_p \in \{1,2,3\}$. Then there are at most $|E(C)| + 2$ tokens in total.*

Proof. The base case $p = 0$ is straightforward to verify directly. So assume $p \geq 1$ and that the claim holds inductively for $p - 1$.

Notice that deleting all nodes of $\tilde{G}_p$ at level $p + 1$, and promoting all nodes at level p to terminals, yields $\tilde{G}_{p-1}$.

Begin with $H = C$. We will modify H until it becomes a connected subgraph of $\tilde{G}_{p-1}$, at the same time redistributing all tokens at level $p + 1$ either to level p, or to deleted edges. This will ensure that the difference between the number of tokens and the number of edges of H never decreases.

Consider each choice of $v_1, \ldots, v_{p-1} \in \{1,2,3\}$ in turn. Let

$$S = \{(v_1, \ldots, v_{p-1}, i) : i \in \{1,2,3\}\} \cup \{(v_1, \ldots, v_{p-1}, i, j) : i, j \in \{1,2,3\}\},$$

which consists of 3 level p nodes and their 9 adjacent level $p + 1$ nodes. Let $\mathcal{H}$ be the set of connected components of $H[S]$. For each $K \in \mathcal{H}$, we do the following:

1. Pick an arbitrary level p node v_K from K.
2. Collect all the tokens on K; for each edge of K, discard one token. If any tokens remain, assign them all to v_K.
3. Remove all edges of K and nodes of level $p + 1$ from H.
4. For every level p node $u \neq v_K$ in K and adjacent level $p - 1$ node w, if $uw \in E(H)$ then replace it with the edge $v_K w$ and discard u. This ensures that H remains connected.

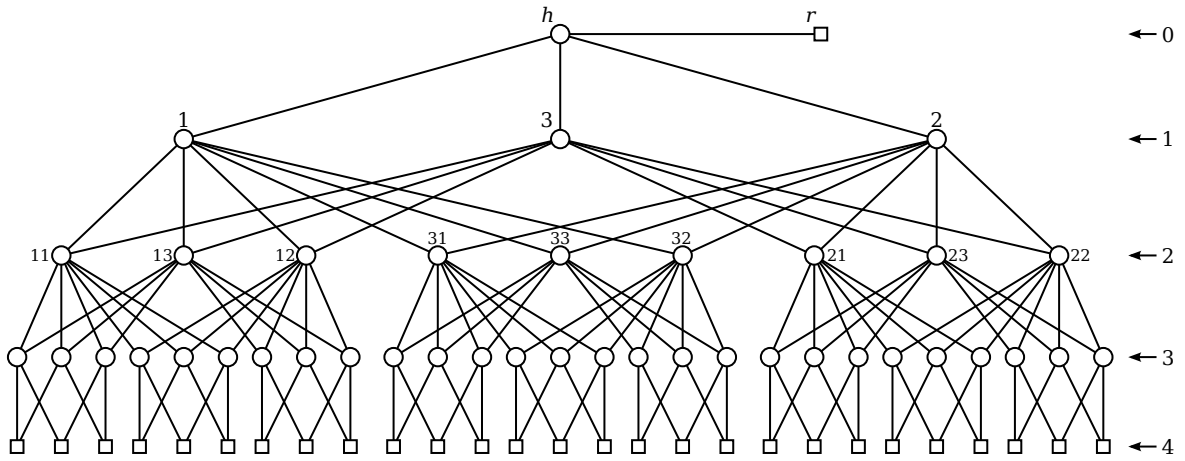


Figure 7: The graph G_3. Notice that removing all level 4 nodes, and promoting those on level 3 to terminals, yields $\tilde{G}_2$.

Let us check that after applying this procedure to all connected components in $\mathcal{H}$, the newly placed tokens on $H[S]$ at level p satisfy the required properties i) and ii). To do this, we will show that for each $K \in \mathcal{H}$, the number of tokens collected and placed at v_K is at most $\min\{\phi(K), 2\}$, where $\phi(K)$ denotes the number of nodes of K which started with two tokens. The properties then follow immediately: i) trivially, and ii) because $\sum_{K\in\mathcal{H}} \phi(K) \leq 3$ by the distribution properties that the tokens had on the level $p+1$ nodes.

So consider any $K \in \mathcal{H}$. For any node w of K at level $p+1$, charge as many of the tokens at w as possible to the edges of K adjacent to w. So w can have at most one extra token remaining, and only if w started with two tokens and had only one adjacent edge. Hence v_K collects at most $\phi(K)$ tokens, and the only case when v_K could potentially collect more than 2 tokens is when K contains three such vertices (which, due to property ii), is the maximum possible). By the topology of $\tilde{G}_p$ (cf. Figure 7) and the connectedness of K, there must be a level $p+1$ vertex in K with at least two adjacent edges. Furthermore, due to property i) this vertex has at most one token. This provides a further edge to charge against, so indeed v_K collects at most $\min\{\phi(K), 2\}$ tokens.

Hence we end up with a connected subgraph H of $\tilde{G}_{p-1}$, as well as an assignment of tokens to the terminals of H satisfying properties i) and ii). Thus by our inductive assumption, there are at most $|E(H)|+2$ tokens remaining at this point. The number of tokens we discarded is at most $|E(C)|-|E(H)|$. So we started with at most $|E(C)|+2$ tokens, as required. □

7 Final remarks

We have shown that for Steiner-claw-free instances, the bidirected cut and hypergraphic relaxations are polyhedrally equivalent, and that there is an efficient map between them. This implies that BCR has an integrality gap of at most $\ln(4) \approx 1.39$ on such instances. Equivalence does not hold in general, but it remains plausible that the gap between BCR and HYP is small. In particular, a positive answer to the following question would immediately yield a better-than-2 bound on the integrality gap of BCR.

Question. Is the ratio between the optimal HYP solution and the optimal BCR solution of a given instance always bounded by a constant strictly smaller than $2/\ln 4$?

We believe that our techniques may be helpful in attacking this question. Consider, for instance, the example given in Figure 1, for which the optimal BCR solution is strictly cheaper than the optimal HYP solution. Our algorithm can in fact be used, if we start with a slightly suboptimal solution to BCR. Namely, we adjust the solution given in Figure 1 by setting $y_v = 1$ for any one of the gray Steiner vertices v, and also setting $z_e = 1$ for the edge e connecting v to the white Steiner vertex. This yields a feasible solution to BCR that costs 6 rather than 5.5, but our algorithm succeeds when applied to this solution, yielding a solution of HYP of the same cost. In general, one approach would be to to let our algorithm extract components (chosen in the way described) until it gets stuck, i.e., a demanding set is identified. At this point it might be possible to adjust the current solution to MIX by augmenting (cheaply) some of the y and z values, in such a way that the demanding set disappears. The algorithm could then be resumed. If this can always be done so that the total augmentation cost is sufficiently small, this would give a positive answer to the above question.

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. 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.

[4] 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.

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

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

[12] 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.

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

[14] 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.

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

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

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

[18] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. *Acta informatica*, 15(2): 141–145, 1981.

[19] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM*, 30(4):852–865, 1983. doi: 10.1145/2157.322410. URL http://doi.acm.org/10.1145/2157.322410.

[20] K. Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters*, 27(3):125–128, 1988.

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

[22] 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.

[23] 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.

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

[25] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Math. Japonica*, 24(6):573–577, 1980.

[26] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.

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

[28] P. Widmayer. On approximation algorithms for Steiner's problem in graphs. In *Graph-Theoretic Concepts in Computer Science*, volume 246, pages 17–28, 1987.

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

[30] Y.-F. Wu, P. Widmayer, and C.-K. Wong. A faster approximation algorithm for the Steiner problem in graphs. *Acta informatica*, 23(2):223–229, 1986.

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

[32] Leonid Zosin and Samir Khuller. On directed Steiner trees. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 59–63, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics. ISBN 0-89871-513-X. URL http://dl.acm.org/citation.cfm?id=545381.545388.