From Directed Steiner Tree to Directed Polymatroid Steiner Tree in Planar Graphs

Chandra Chekuri
University of Illinois, Urbana-Champaign, IL, USA

Rhea Jain
University of Illinois, Urbana-Champaign, IL, USA

Shubhang Kulkarni
University of Illinois, Urbana-Champaign, IL, USA

Da Wei Zheng
University of Illinois, Urbana-Champaign, IL, USA

Weihao Zhu
University of Illinois, Urbana-Champaign, IL, USA

Abstract

In the Directed Steiner Tree (DST) problem the input is a directed edge-weighted graph $G = (V, E)$, a root vertex r and a set $S \subseteq V$ of k terminals. The goal is to find a min-cost subgraph that connects r to each of the terminals. DST admits an $O(\log^2 k/\log\log k)$-approximation in *quasi-polynomial* time [29, 27], and an $O(k^\epsilon)$-approximation for any fixed $\epsilon > 0$ in polynomial-time [45, 7]. Resolving the existence of a polynomial-time poly-logarithmic approximation is a major open problem in approximation algorithms. In a recent work, Friggstad and Mousavi [25] obtained a simple and elegant polynomial-time $O(\log k)$-approximation for DST in *planar* digraphs via Thorup's shortest path separator theorem [41]. We build on their work and obtain several new results on DST and related problems.

- We develop a tree embedding technique for rooted problems in planar digraphs via an interpretation of the recursion in [25]. Using this we obtain polynomial-time poly-logarithmic approximations for Group Steiner Tree [26], Covering Steiner Tree [34] and the Polymatroid Steiner Tree [5] problems in planar digraphs. All these problems are hard to approximate to within a factor of $\Omega(\log^2 n/\log\log n)$ even in trees [33, 29].
- We prove that the natural cut-based LP relaxation for DST has an integrality gap of $O(\log^2 k)$ in planar digraphs. This is in contrast to general graphs where the integrality gap of this LP is known to be $\Omega(\sqrt{k})$ [46] and $\Omega(n^\delta)$ for some fixed $\delta > 0$ [36].
- We combine the preceding results with density based arguments to obtain poly-logarithmic approximations for the multi-rooted versions of the problems in planar digraphs. For DST our result improves the $O(R + \log k)$ approximation of [25] when $R = \omega(\log^2 k)$.

2012 ACM Subject Classification Theory of computation → Discrete optimization; Theory of computation → Routing and network design problems

Keywords and phrases Directed Planar Graphs, Submodular Functions, Steiner Tree, Network Design

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.42

Related Version *Full Version*: https://arxiv.org/abs/2407.01904

Funding Chandra Chekuri and Rhea Jain are partially supported by NSF grants CCF-1907937 and CCF-2402667. Weihao Zhu acknowledges funding support from a graduate fellowship of the CS department.

1 Introduction

We consider several rooted network design problems in *directed* graphs and develop new approximation algorithms and integrality gap results for them in planar digraphs. It is well-known that many problems in directed graphs are harder to approximate than their corresponding undirected graph versions. A canonical example, and the motivating problem for this paper, is the Steiner Tree problem. The input is an undirected graph $G = (V, E)$ with edge costs $c : E \to \mathbb{R}_{\geq 0}$, a root $r \in V$, and a set of terminals $S \subseteq V \setminus \{r\}$. The goal is to find a minimum cost subgraph of G in which each terminal is connected to the root. Steiner Tree is NP-Hard and APX-Hard to approximate. There is a long and rich history on approximation algorithms for this problem. The current best approximation ratio is $\ln 4 + \epsilon$ [4, 28], and it is known that there is no approximation factor better than $\frac{96}{95}$ unless $\mathsf{P} = \mathsf{NP}$ [14]. Steiner Tree admits a PTAS in planar graphs [3]. In this paper we consider the directed version of this problem. Given a directed graph $G = (V, E)$ and a vertex r we use the term r-tree to denote a subgraph of G that is a directed out-tree rooted at r; note that all vertices in the r-tree are reachable from r in G.

Directed Steiner Tree (DST). The input is a directed graph $G = (V, E)$ with non-negative edge costs $c(e)$, a root $r \in V$, and a set of *terminals* $S \subseteq V \setminus \{r\}$. The goal is to find a min-cost r-tree that contains each terminal. We let $k := |S|$.

DST is a natural and fundamental network design problem. Its approximability has been a fascinating open problem. An easy observation shows that DST generalizes Set Cover and hence does not admit a better than $(1 - \epsilon) \log k$ approximation [18]. Via a more sophisticated reduction, it is known to be hard to approximate to an $\Omega(\log^2 k/ \log \log k)$-factor under plausible complexity assumptions [29], and to slightly weaker $\Omega(\log^{2-\epsilon} k)$-factor unless NP is contained in randomized quasi-poly time [33]. There is a quasi-polynomial time $O(\log^2 k/ \log \log k)$-approximation [29, 27, 7], and a polynomial time $O(k^\epsilon)$-approximation for any $\epsilon > 0$ [45]. These results suggest that DST may admit a polynomial-time poly-logarithmic approximation. However, this has not been resolved despite the first quasi-polynomial-time poly-logarithmic approximation being described in 1997 [7]. One reason is that the natural LP relaxation has been shown to have a polynomial-factor integrality gap of $\Omega(\sqrt{k})$ [46], and more recently $\Omega(n^\delta)$ for some fixed $\delta > 0$ [36].

In a recent work, Friggstad and Mousavi [25] considered DST in *planar* digraphs. They give a surprisingly simple and elegant algorithm which yields an $O(\log k)$ approximation in polynomial time. Their algorithm is based on a divide-and-conquer approach building on Thorup's shortest path planar separator theorem [41]. Planar graphs are an important and useful class of graphs from a theoretical and practical point of view, and moreover several results on planar graphs have been extended with additional ideas to the larger class of minor-free families of graphs. Inspired by [25], we address approximation algorithms in planar digraphs for several rooted network design problems that are closely related to DST. We formally define the problems below and then discuss their relationship to DST. In all problems below, the input is a directed graph $G = ((V, E), c)$, where $c : E \to \mathbb{R}_{\geq 0}$ denote edge costs, and a root r; the goal is to find a min-cost subgraph to satisfy some connectivity property from the root.

Directed Group Steiner Tree (DGST). The input consists of $G = ((V, E), c)$, r, and k *groups* of terminals $g_1, \ldots, g_k \subseteq V \setminus \{r\}$. The goal is to find a minimum cost r-tree that contains a terminal from each group g_i.

Directed Covering Steiner Tree (DCST). This is a generalization of DGST in which each of the groups $g_1, \dots, g_q \subseteq V \setminus \{r\}$ has an integer requirement $h_i \geq 1$, $i \in [q]$. The goal is to find a minimum cost r-tree that contains at least h_i distinct terminals from each g_i.

Directed Polymatroid Steiner Tree (DPST). DPST generalizes the aforementioned problems. In addition to G and r, the input consists of an integer valued normalized monotone submodular function (polymatroid) $f : 2^V \to \mathbb{Z}_{\geq 0}$ (see Section 1.4 for a formal definition). The goal is to find a minimum cost r-tree $T = (V_T, E_T)$ such that $f(V_T) = f(V)$.

It is not difficult to see that DST $\preceq$ DGST $\preceq$ DCST $\preceq$ DPST where we use $X \preceq Y$ to indicate that X is a special case of Y. In general directed graphs it is also easy to see that DST and DGST are equivalent, though this reduction does not hold in planar graphs[1]. Further, the known approximation ratios (and the main recursive greedy technique) for DST generalize to these problems [27, 5]. In contrast, the situation is quite different in undirected graphs. The undirected version of these problems, namely Group Steiner Tree (GST) [26], Covering Steiner Tree (CST) [34, 31] and Polymatroid Steiner Tree (PST) [5] have been well-studied, and poly-logarithmic approximation ratios are known. We defer a detailed discussion of the motivations and results on these problems, but we highlight one important connection. The known hardness of approximation for DST that we mentioned earlier is due to the fact that it holds for the special case of GST in trees! Thus, the group covering requirement makes the problem(s) substantially harder even in undirected graphs where Steiner tree has a simple constant factor approximation. We point out that the $O(\log k)$ approximation of [25] separates the approximability of DST and DGST in planar graphs since the latter is hard to a factor of $\Omega(\log^2 k/\log\log k)$ in trees. The positive algorithmic result in [25] naturally motivates the following questions.

- *Are there polynomial-time poly-logarithmic approximation algorithms for DGST, DCST, and DPST in planar digraphs*?
- *Is the integrality gap of the natural LP for DST and DGST and DCST in planar digraphs poly-logarithmic* [2]?

1.1 Results

We provide affirmative answers to the first question and for part of the second question. Before stating our main results we set up some notation. In the setting of DPST we let $S = \{v \mid f(v) > 0\}$ denote the set of terminals and let $N = |S|$. We also let $k = f(V)$. Note that in the case of DST, $N = k$, while in the setting of DGST and DCST, $S = \bigcup_i g_i$ and k is the sum of the requirements.

We obtain poly-logarithmic approximation ratios for DGST, DCST and DPST in planar digraphs. These are the first non-trivial polynomial-time approximations for these problems, and we note that the ratios essentially match the known approximation ratios for these problems in *undirected* planar graphs.

[1] Demaine et al. [15] define planar group Steiner tree in a restricted way where the groups correspond to the nodes of distinct faces of an embedded planar graph. There is a PTAS for this special case in undirected graphs [2], and in fact it is equivalent to DST in planar graphs. However we only restrict the graph to be planar, and not the groups.

[2] It is not straightforward to formulate a relaxation for DPSP. The other problems have known LP relaxations.

▶ **Theorem 1.** *For any fixed $\epsilon > 0$, there exists a polynomial time $O\left(\frac{\log^{1+\epsilon} n \log k \log N}{\epsilon \log\log n}\right)$-approximation algorithm for the Directed Polymatroid Steiner Tree on planar graphs. In the special cases of Directed Group Steiner Tree and Directed Covering Steiner Tree on planar graphs, the approximation ratios can be improved to $O(\log k \log^2 N)$.*

Our second result is on the integrality gap of a natural cut/flow based LP for DST; see DST-LP for a formal description. In contrast to a polynomial-factor lower bound on the gap in general directed graphs, we show the following via a constructive argument.

▶ **Theorem 2.** *The integrality gap of (DST-LP) is upper bounded by $O(\log^2 k)$ in planar digraphs.*

The bound we prove is weaker than the known $O(\log k)$ approximation (in fact the proof is inspired by the same technique), and is unlikely to be tight. However, no previous upper bound was known prior to our work; positive results have been obtained only for quasi-bipartite instances via the primal-dual method [23, 24]. LP based algorithms provide several easy and powerful extensions to other problems, and are of much interest. The integrality gap of DST is also of interest in understanding the power and limitations of routing vs coding in network information theory – we refer the reader to [1] and surveys on network coding [20, 19]. We believe that the integrality gap of the natural LP for DGST and DCST is poly-logarithmic in planar digraphs, however, there are some technical challenges in extending our approach and we leave it for future work.

Multi-root versions. Friggstad and Mousavi [25] also considered the multi-root version of DST and one can extend each of the problems we consider to this more general setting. The input consists of multiple roots $r_1, \ldots, r_R$. The goal is to find a minimum cost subgraph in which the relevant set of terminals is reachable from *at least one* of the roots. Note that multi-root versions arise naturally in some problems including information transmission (see the aformentioned work on network coding). In general digraphs it is trivial to reduce the multi-rooted version to the single root version by adding an auxiliary root vertex, but this reduction does not preserve planarity. Friggstad and Mousavi [25] described an $O(R + \log k)$-approximation for the multi-rooted version of DST. Using *density*-based arguments (see Section 1.4) combined with the aforementioned results, we obtain polylogarithmic approximation ratios for multi-rooted versions of all the considered problems in planar digraphs. For DST, our bound is better than the one in [25] when R is $\omega(\log^2 k)$.

▶ **Theorem 3.** *There is an $O(\log^2 k)$-approximation for the multi-rooted version of DST in planar graphs. For the multi-rooted versions of DGST, DCST there is a polynomial-time $O(\log k \log^2 N)$-approximation, and for DPST a polynomial-time $O\left(\frac{\log^{1+\epsilon} n \log k \log N}{\epsilon \log\log n}\right)$-approximation.*

We note that in DGST, DCST, and DPST, the approximation factors for the multi-root versions actually match those of the single root setting.

▶ Remark 4. It is not difficult to see that the algorithm of Friggstad and Mousavi [25] and ours extends to several other rooted problems involving budget constraints on cost or terminals, and prize-collecting versions. We omit a detailed description of these extensions in this version and instead provide a brief discussion in a full version [11].

▶ Remark 5. [25] observed that their approach extends to the node-weighted case. The standard transformation from edge-weights to node-weights does not necessarily preserve planarity, and hence the extension holds due to the specific technique. Our results also hold for node weights. Even in *undirected* graphs there is no known polynomial-time poly-logarithmic approximation for node-weighted GST – this is because metric tree embeddings do not apply to reduce the problem to trees. Thus, our results are new even for node-weighted undirected planar graphs.

1.2 Overview of Ideas

The $O(\log k)$-approximation for DST on planar graphs given by Friggstad and Mousavi [25] uses a recursive divide-and-conquer structure. We provide a brief overview. The algorithm uses Thorup's shortest path separator theorem applied to directed graphs:

▶ **Lemma 6** ([25, 41]). *Let G be a planar directed graph with non-negative edge costs $c(e)$, non-negative vertex weights $w(v)$, and a root $r \in V$ such that every vertex in V is reachable from r. There exists a polynomial time algorithm to find three shortest dipaths P_1, P_2, P_3 starting at r such that every weakly connected component of $G \setminus (P_1 \cup P_2 \cup P_3)$ has at most half the vertex weight of G.*

The high-level idea in [25] is simple. Suppose we can guess the optimum solution value for a given DST instance, say OPT. Then one can remove all vertices v farther than OPT from r (since they will not be in any optimum solution), and use the preceding theorem to find 3 paths of cost at most 3OPT such that removing the paths yields components, each of which contains at most half the original terminals. We can shrink the paths into r and recurse on the "independent" sub-instances induced by the terminals in each component. The recursion depth is $O(\log k)$ which bounds the total cost to $O(\log k) \cdot \text{OPT}$. The main issue is to implement the guess of OPT in each recursive call. The authors obtain a quasi-polynomial time algorithm by brute force guessing OPT to within a factor of 2. They obtain a polynomial-time algorithm by a refined argument where they folding the guessing into the recursion itself. We take an alternate perspective on this algorithm by *explicitly* constructing the underlying recursion tree of the algorithm. Theorem 7 shows that we can view this recursion tree as a "tree embedding" for directed planar graphs that is suitable for rooted problems. The power of the embedding is that it essentially reduces the planar graph problem to a problem on trees which we know how to solve. A caveat of our tree embedding is that it creates *copies* of terminals. Interestingly, for DGST and DPST this duplication does not cause any issues since the definitions of these problems are rich enough to accommodate copies. For DCST one needs a bit more care to obtain a better bound than reducing it to DPST, and we describe the details in the technical section. This parallels the situation in undirected graphs where probabilistic metric tree embeddings [17] are used to reduce the GST, CST, and PST problems to trees, and furthermore, is the only known method to solve those problems. The formal description of the tree embedding is given below.

▶ **Theorem 7.** *Let $G = (V, E)$ be a directed planar graph with edge costs $c : E \to \mathbb{R}_{\geq 0}$, a root $r \in V$, and a set of terminals $S \subseteq V$. Let $\gamma \leq c(E)$, and let $n := |V|$ and $k := |S|$. There exists an efficient algorithm that outputs a directed rooted out-tree $\mathcal{T} = (V_T, E_T)$ with root r_T, edge costs $c_T : E_T \to \mathbb{R}_{\geq 0}$ and a mapping $M : S \to 2^{V_T}$ that maps each terminal in G to a set of terminals in V_T, that satisfies the following properties:*

1. ***Size:** $|V_T| = O(k^3\gamma)$, and for each terminal $t \in S$, $|M(t)| = O(k\gamma)$. Furthermore, all $M(t)$ are disjoint from each other.*
2. ***Height:** The height of T is at most $O(\log k)$.*
3. ***Projection from Graph:** For any r-tree $G' \subseteq G$ with $c(G') \leq \gamma$ there exists a r_T-tree $T' \subseteq \mathcal{T}$ with $c_T(T') = O(\log k)c(G')$, in which for each terminal $t \in S \cap G'$, $M(t) \cap T' \neq \emptyset$.*

4. ***Projection to Graph:*** *For any r_T-tree $T' \subseteq \mathcal{T}$, there exists a r-tree $G' \subseteq G$ with $c(G') \leq c_T(T')$ and for each terminal $t \in S$, if $M(t) \cap T' \neq \emptyset$ then $t \in G'$. Furthermore, we can compute G' efficiently.*

Our proof of Theorem 2 on the LP integrality gap is inspired by the algorithm of [25]. Instead of guessing OPT we use the LP optimum value as the estimate. This is a natural idea, however, in order to prove an integrality gap we need to work with the original LP solution for the recursive sub-instances. We use a relatively simple trick for this wherein we overpay for the top level of the recursion to construct feasible LP solutions for the sub-instances from the original LP solution; the over payment helps us to argue that the cost of the LP solutions for the sub-instances is only slightly larger and this can be absorbed in the recursion since the problem size goes down.

Finally, for the multi-rooted version we rely on a simple reduction to the single root problem via the notion of density, which is a standard idea in covering problems.

1.3 More on related work

There is extensive literature on algorithms for network design in both undirected and directed networks with more literature on undirected network design. Standard books on combinatorial optimization [40, 21], and approximation algorithms [42, 43] cover many of the classical problems and results. We also point to the surveys [30, 35] on network design. In this section we describe some closely related work and ideas.

Directed Steiner Tree. Zelikovsky [45] was the first to address the approximability of DST. He obtained an $O(k^\epsilon)$-approximation for any fixed $\epsilon > 0$ via two ideas. He defined a recursive greedy algorithm and analyzed its performance as a function of the depth of the recursion. He then showed that one can reduced the problem on a general directed graph to a problem on a depth/height d DAG (via the transitive closure of the original graph) at the loss of an approximation factor that depends on d. Charikar et al [7] refined the algorithm and analysis in [45] and combined it with the depth reduction, they showed that one can obtain an $O(d^2 k^{1/d} \log k)$ approximation in $O(n^{O(d)})$-time; this led to an $O(\log^3 k)$-approximation in quasi-polynomial time. Subsequentaly Grandoni et al [29] improved the approximation to $O(\log^2 k / \log\log k)$ in quasi-polynomial time via a more sophisticated LP-based approach. A different approach that also yields the same bound was given by Ghuge and Nagarajan [27] and this is based on a refinement of the recursive greedy algorithm for walks in graphs [12]. The advantage of [27] is that it yields an $\Omega(\log\log k / \log k)$-approximation in quasi-polynomial time algorithm for the budgeted version of DST; the goal is to maximize the number of terminals in a r-rooted tree with a given budget of B on the cost of the tree.

Zosin and Khuller [46] showed that the natural cut-based LP relaxation has an integrality gap of $\Omega(\sqrt{k})$ for DST. However, their example only showed a gap of $\Omega(\log n)$ as a function of the number of nodes n. There was some hope that the integrality gap is poly-logarithmic in n, however [36] recently showed that the gap is $\Omega(n^\delta)$ for some $\delta > 0$ by modifiying the construction in [46]. Interestingly these lower bound examples are DAGs with $O(1)$-layers for which the recursive-greedy algorithm yields an $O(\log k)$-approximation in polynomial-time! Rothvoss [39] showed that $O(\ell)$-levels of the Lasserre SDP hierarchy when applied to the standard cut-based LP reduces the integrality gap to $O(\ell \log k)$ on DAGs with ℓ layers. This was later refined to show that $O(\ell)$-levels of the Sherali-Adams hierarchy suffices [22]. However, both these approaches also require quasi-polynomial time to obtain a poly-logarithmic approximation.

Group Steiner Tree. The group Steiner tree problem (GST) in undirected graphs was introduced by Reich and Widemeyer [38] and it was initially motivated by an application in VLSI design. Garg, Konjevod and Ravi [26] obtained an $O(d \log k)$ approximation in depth d trees via an elegant randomized rounding algorithm of the fractional solution to a natural LP relaxation; one can reduce the depth to $O(\log N)$ via the fractional solution and hence they obtained an $O(\log N \log k)$-approximation. They obtained an algorithm for general graphs via probabilistic tree embeddings [17]. Zosin and Khuller obtained an alternate deterministic $O(d \log k)$-approximation on trees [46]. The randomized algorithm of [26] can also be derandomized via standard methods [8]. The integrality gap of the natural LP for GST was shown to be $\Omega(\log^2 k/(\log\log k)^2)$ by Halperin et al. [32]. This gap motivated the inapproximability result of Halperin and Krauthgamer who showed that GST in trees is hard to approximate within a factor of $\Omega(\log^{2-\epsilon} k)$. This was further improved to $\Omega(\log^2 k/\log\log k)$ [29] under stronger complexity theoretic assumption. Note that one can consider node-weighted GST. In general directed graphs one can see that node-weighted and edge-weighted problems are typically reducible to each other, however this is not necessarily the case in undirected graphs. The known approaches to approximate GST in general graphs in polynomial time uses probabilistic tree embeddings (or, more recently, oblivious routing trees [37, 6, 13] which are intimately connected to tree embeddings). However, node-weighted problems do not admit such tree embeddings and thus we do not have polynomial-time poly-logarithmic approximation for GST in node-weighted undirected graphs.

There is a strong connection between GST, its directed counterpart DGST and DST. As we remarked, it is easy to see that in directed graphs, DST and DGST are equivalent. One can also reduce GST to DST by adding a dummy terminal t_i for each group g_i and connecting all the vertices in g_i to t_i via directed edges. Thus GST admits an $O(\log^2 k/\log\log k)$-approximation in quasi-polynomial time though the best polynomial-time algorithm loses another log factor due to tree embeddings. On the other hand, via the height reduction approach and path expansion, one can reduce DST to GST in trees in quasi-polynomial-time at the loss of an $O(\log k)$ in the approximation ratio (details of this are essentially folklore but can be seen in [9]). This partially explains the reason why the hardness results for DST are essentially based on the hardness of GST in trees.

Covering Steiner Tree and Polymatroid Steiner Tree. The Covering Steiner Tree problem was first considered by Konjevod, Ravi and Srinivasan [34] as a common generalization of GST and the k-MST problems. They obtained a poly-logarithmic approximation by generalizing the ideas from GST (see also [16]). Gupta and Srinivasan subsequently improved the ratio [31]. Calinescu and Zelikovsky [5] defined the general Polymatroid Steiner Tree problem (PSP) and its directed counterpart (DPSP). They were motivated by both theoretical considerations as well as applications in wireless networks. Submodularity provides substantial power to model a variety of problems. PSP is easily seen to generalize GST and CST. However, unlike GST and CST, even in trees there is no easy LP relaxation for PSP that one can formulate, solve and round. Thus, [5] used a different approach. Chekuri, Even and Kortsarz [10] had shown that the recursive greedy algorithm of [7] can be adapted to run in polynomial-time on trees after preprocessing it to reduce the degree and height. The recursive greedy approach naturally generalizes to PSP/DPSP just as the greedy algorithm for Set Cover generalizes to Submodular Set Cover [44]. Via this generalization, [5] obtained polynomial-time approximation algorithms for PSP in trees and hence in general graphs via tree emebeddings. For DPST they obtained quasi-polynomial-time approximation algorithms.

1.4 Preliminaries

Let $G = (V, E)$ be a directed graph with edge costs $c : E \to \mathbb{R}_+$. For $E' \subseteq E$, we denote $c(E') = \sum_{e \in E'} c(e)$. We assume all edge costs $c(e) \geq 1$ and are polynomially bounded in n. For problems considered in this paper, this is without loss of generality by guessing the cost of the optimal solution OPT, contracting edges with cost much smaller than OPT, and scaling appropriately.

We define the minimum density DST problem.

▶ **Definition 8.** *Given an instance of DST on a graph $G = (V, E)$ with root r, the* density *of a partial solution $F \subseteq E$ is $c(F)/k(F)$ where $k(F)$ is the number of terminals in S that have a path from r in $G[F]$. The minimum-density DST problem is to compute a solution of minimum density in a given instance of DST.*

One can similarly define minimum density versions of DPST (which generalizes DGST and DCST); for a partial solution $F \subseteq E$, we let S_F denote the set of terminals in S that have a path from r in $G[F]$. The density of F is $c(F)/f(S_F)$, where f is the given polymatroid.

Graph notation. We use $V(H)$ and $E(H)$ to refer to the vertices and edges of a graph H when the vertex and edge sets have not been explicitly specified. For $S \subseteq V$, we use $E[S]$ to denote the set of edges of E with both endpoints in S, $G[S]$ to denote the subgraph $(S, E[S])$ *induced by* S in G, and $\delta^+(S) = \{(u, v) \in E : u \in S, v \notin S\}$ to denote the *out-cut* of S. For $u \in V$ we use $\mathbb{1}_{u \in S}$ to denote the *indicator* of the vertex u being in S, i.e. $\mathbb{1}_{u \in S} = 1$ if $u \in S$ and is 0 otherwise. For a path $P \subseteq G$, we define the *length* as the number of edges on the path. For a given edge-cost function c, we denote $d_c(r, t)$ as the length of *shortest r-t path* in G with edge weights c; we drop the subscript c if it is clear from context. For $r \in V$ an *out-tree rooted at r* is a subgraph $T = (V_T, E_T) \subseteq G$ such that there is a unique r-v path for every $v \in V_T$. The *height* of the tree is the maximum length of a r-v path, for $v \in V_T$. The *size* of the tree is the number of vertices in the tree $|V_T|$. For a subgraph $G' \subseteq G$, we use G/G' to denote the graph obtained by contracting every edge of G' and $G - G'$ to denote the graph obtained by deleting every edge in G'. A *weakly connected component* of G is a connected component of the underlying undirected graph obtained from G by ignoring the edges orientations.

Submodular functions. Let $f : 2^V \to \mathbb{R}$ be a set function over ground set V. The function f is *monotone* if $f(X) \leq f(Y)$ for every $X \subseteq Y \subseteq V$, *submodular* if $f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$ for every $X, Y \subseteq V$, and *normalized* if $f(\emptyset) = 0$. An integer-valued, normalized, monotone and submodular function is called a *polymatroid*.

2 Recursive Tree Embeddings for Directed Planar Graphs

We show that we can view the recursion tree given by the algorithm of [25] as a tree embedding by proving Theorem 7. As described in Section 1, the algorithm of [25] starts with an upper bound γ for the cost of an optimal solution (we call this OPT). In order to fold the guessing of OPT into the recursion, the algorithm makes two recursive calls and takes the minimum. For the first recursive call, it applies Lemma 6 to obtain a planar separator, buys the separator, and recurses on the resulting weakly connected components. The second recursive call divides the "guess" γ by two. The polynomial runtime comes from the fact that at each step, we either halve the guess of OPT or halve the number of terminals.

We define a subroutine PruneAndSeparate (Algorithm 1) to describe the first recursive call, which takes as input a graph $(G = (V, E), c)$ with root $r \in V$, terminals $S \subseteq V$, and a guess γ for the cost of the optimal solution. PruneAndSeparate$((G, c), r, S, \gamma)$ removes all vertices further than γ away from r, and uses Lemma 6 on the resulting graph with vertex weights set to 1 on the terminals and 0 elsewhere. This yields a *planar separator* $P := P_1 \cup P_2 \cup P_3$ in which each resulting component of $G \setminus P$ has at most half the terminals. The subroutine contracts P into r; each component of $G \setminus P$ corresponds to a new subinstance induced by the terminals in that component along with r. PruneAndSeparate$((G, c), r, S, \gamma)$ returns P along with the subinstances for each component.

Algorithm 1 Prune and Separate Procedure.

PruneAndSeparate$((G = (V, E), c), r, S, \gamma)$:
  Delete all vertices $v \in V$ with $d_c(r, v) > \gamma$.
  Let $P := P_1 \cup P_2 \cup P_3$ be given by applying Lemma 6 with weights $w(v) = \mathbb{1}_{v \in S}$ for $v \in V$.
  Let G_P be obtained from G by contracting P into r.
  Let $C'_1, \ldots, C'_\ell$ be components of $G \setminus P$, and let $C_i \leftarrow G_P[C'_i \cup \{r\}]$
  return $(P, C_1, \ldots, C_\ell)$

Given this subroutine, the tree embedding is simple. We define a recursive function TreeEmb, which takes as input a graph $(G = (V, E), c)$ with root $r \in V$, terminals $S \subseteq V$, and a "guess" γ. The algorithm instantiates a root node r_T and constructs two trees corresponding to the two recursive calls made by [25]:

(1) Call PruneAndSeparate$((G, c), r, S, \gamma)$ and construct the trees $\mathcal{T}_i$ = TreeEmb(C_i, r, S_i, γ) recursively for each subinstance (C_i, r, S_i). Add an auxiliary node v^* and connect it to the root of each subtree with a zero-cost edge.

(2) Recursively construct the tree $\mathcal{T}_h$ = TreeEmb$((G, c), r, S, \gamma/2)$.

TreeEmb$((G, c), r, S, \gamma)$ connects r_T to the root of $\mathcal{T}_h$ with a zero-cost edge. It also connects r_T to v^* with an edge of cost $c(P)$, where P is the planar separator constructed in 1. These edge costs from r_T correspond to the costs of choosing each recursive path. See Figure 1 for a summary.

As described in Theorem 7, we would like this tree embedding to maintain some representation of the terminals S. It is not immediately clear how one could accomplish this; the first recursive call decomposes G while the second takes in a copy of G, so the same terminals can appear in both corresponding subtrees. Therefore, we need to allow for multiple copies of the same terminal. The algorithm of [25] includes a terminal t in the solution either when t is in some planar separator or when t is the only remaining terminal in S, in which case it buys the shortest path r-t path. To represent this in the tree embedding, we create a copy of a terminal for every separator or base case it is in. We denote copies of t as t^P where P is the separator or shortest path containing t, and let $M(t)$ denote the set of all copies of t.

The full algorithm is described in Algorithm 2. We use a subroutine UpdateTree$((\mathcal{T}, M), (\mathcal{T}', M'), v)$, which updates the tree $\mathcal{T}$ to include $\mathcal{T}'$ via a zero-cost edge from $v \in V(\mathcal{T})$ to the root of $\mathcal{T}'$. This subroutine also updates the mapping M to include M'; that is, $M(t) = M(t) \cup M'(t)$ for all $t \in S$.

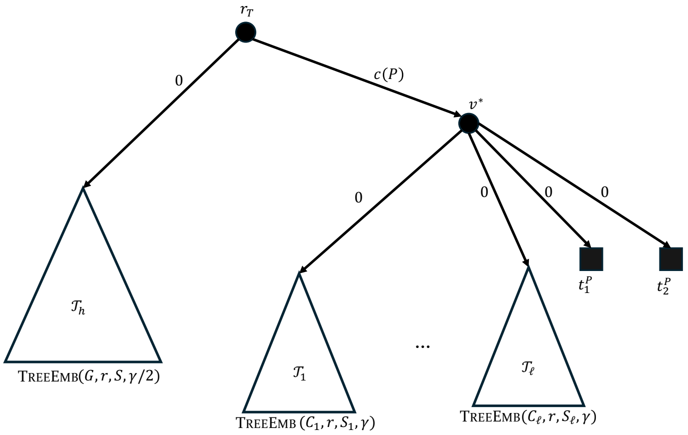

Figure 1 The tree $\mathcal{T}$ given by TreeEmb, where t_1, t_2 are terminals in P.

Algorithm 2 Tree Embedding Construction for Directed Planar Graphs.

TreeEmb $((G = (V, E), c), r, S, \gamma)$:
 if $\gamma < 1$ or $S = \emptyset$ **then return** null
 Initialize r_T, $\mathcal{T} \leftarrow (V_T = \{r_T\}, E_T = \emptyset)$ as tree embedding with empty cost function c_T
 Initialize $M(t) \leftarrow \emptyset$ for all $t \in S$ as the terminal copies
 if $|S| = 1$ **then**
  Let P be a shortest r-t path in G
  Add a new vertex t^P to V_T and to $M(t)$
  Add a new edge $e_t = (r_T, t^P)$ to E_T, and define $c_T(e_t) = c(P)$.
  return $(\mathcal{T}, r_T, M)$
 Recursively construct $(\mathcal{T}_h, r_h, M_h) \leftarrow$ TreeEmb$((G, c), r, S, \gamma/2)$
 UpdateTree$((\mathcal{T}, M), (\mathcal{T}_h, M_h), r_T)$
 $(P, C_1, \ldots, C_\ell) \leftarrow$ PruneAndSeparate$((G, c), r, S, \gamma)$
 Add a new vertex v^* to V_T, edge $e^* = (r_T, v^*)$ to E_T and define $c_T(e^*) = c(P)$.
 for $i \in [\ell]$ **do**
  Recursively construct $(\mathcal{T}_i, r_i, M_i) \leftarrow$ TreeEmb$(C_i, r, S \cap C_i, \gamma)$
  UpdateTree$((\mathcal{T}, M), (\mathcal{T}_i, M_i), v^*)$
 for $t \in S \cap P$ **do**
  Add a new vertex t^P to V_T and $M(t)$
  Add a new edge $e_t = (v^*, t^P)$ to E_T, and define $c_T(e_t) = 0$.
 return $(\mathcal{T}, r_T, M)$

We claim that the tree and mapping $(\mathcal{T}, r_T, M)$ given by TreeEmb(G, r, S, γ) satisfies all properties of Theorem 7. We outline the ideas here; details are deferred to a full version. Properties 1 and 2 follow easily from simple inductive arguments similar to those in [25]. We note that the construction given in Algorithm 2 constructs a tree of height $O(\log(k\gamma))$; this can be improved to $O(\log k)$ (see Remark 9).

For Property 3, let $G' \subseteq G$ be an r-tree. Let i be such that $\gamma/2^{i+1} \leq c(G') \leq \gamma/2^i$. To construct $T' \subseteq \mathcal{T}$, we include the path from r_T to the root of the subtree given by TreeEmb$(G, r, S, \gamma/2^i)$. We then include the planar separator branch, which costs at most $3\gamma/2^i \leq 6c(G')$ and recurse on sub-instances. The cost bounds and terminal copy requirements follow immediately from the feasibility and cost analysis of [25]. For Property 4, let $T' \subseteq \mathcal{T}$ be an r_T-tree. Notice that there are only two types of non-zero cost edges in $\mathcal{T}$; those corresponding to planar separators or those corresponding to shortest paths in the base case. Let $\mathcal{P}(T')$ be the set of all such paths and separators corresponding to non-zero cost edges of T'. We let $G' = \cup_{P\in\mathcal{P}(T')}P$. It is clear that G' can be computed efficiently by traversing T' and including all relevant paths. The cost and terminal copy guarantees are simple; see full version for detailed proofs.

▶ Remark 9. The height of the tree can be reduced to $O(\log k)$ by increasing the degree by a factor of $O(\log \gamma)$. Instead of only making two recursive calls, one can simultaneously make recursive calls $\text{TreeEmb}(G, r, S, \gamma/2^i)$ for $i \in [\log \gamma]$, along with the recursive call using the planar separator. Each recursive call then only considers the "planar separator" branch and proceeds inductively.

3 Group, Covering, and Polymatroid Directed Steiner Tree

In this section we give an overview of the proof of Theorem 1, providing polynomial time polylogarithmic approximation algorithms for DGST, DCST, and DPST. Although DPST generalizes DCST and DGST, we discuss each of the three problems separately in this section since we obtain better approximation ratios for DGST and DCST; moreover, our algorithmic techniques for DGST and DCST are different. For each of these problems, let $G = (V, E)$ denote the input graph, $c : E \to \mathbb{R}_+$ denote the edge costs, $r \in V$ denote the root, and S denote the set of terminals. The embedding theorem given by Theorem 7 allows us to effectively reduce to special cases of the problems in which the input graph is a tree, as described by the following high-level framework:

(a) Use Theorem 7 on inputs $(G, c), r, S$, and $\gamma = c(E)$ to obtain a directed out-tree $\mathcal{T} = (V_T, E_T)$ rooted at r_T with edge costs c_T, and for each terminal $t \in S$ a set of "copies" $M(t) \subseteq V_T$. The new set of terminals S_T is the collection of all copies $\cup_{t\in S}M(u)$.

(b) Compute an approximate solution to a relevant problem on $\mathcal{T}$,

(c) Project the solution on $\mathcal{T}$ to the graph G using Property 4 of Theorem 7.

One challenge in directly applying the above framework is constructing the instance and problem to solve on $\mathcal{T}$ in step (b) above; this is because a terminal in G contains several copies in $\mathcal{T}$. For DGST and DPST, the ability to deal with copies of terminals is quite naturally instilled into the problem definitions themselves. In DGST, we can simply expand the groups to include all copies of a terminal, and in DPST, we can appropriately redefine the underlying submodular function and rely on the diminishing marginal returns property. Thus for DGST and DPST, we can directly solve the equivalent problem on the tree, as explained below:

Directed Group Steiner Tree. The input is a graph (G, c) with a root r and k groups $g_1, \ldots, g_k \subseteq V$. After applying step (a) of the framework, we consider the following instance of DGST. The input graph is the constructed tree $(\mathcal{T}, c_T)$ with root r_T and terminal set S_T. The new groups $g'_1, \ldots g'_k \subseteq V_T$ are defined as $g'_i := \cup_{u\in g_i}M(u)$ for every $i \in [k]$. To obtain an approximate solution on this instance, we use the following result by Zosin and Khuller [46] (also see [26, 8]).

▶ **Theorem 10** ([46]). *There exists a polynomial time $O(d \log k)$-approximation algorithm for Group Steiner Tree when the input graph is a tree with height d.*

Applying this result in conjunction with Property 2 of Theorem 7 regarding the height of the tree $\mathcal{T}$ gives us an $O(\log N \log k)$-approximation for the instance of DGST on $(\mathcal{T}, c_T)$.

Directed Polymatroid Steiner Tree. The input consists of a graph (G, c) with root r and a polymatroid function $f : 2^V \to \mathbb{Z}_{\geq 0}$. After applying step (a) of the framework, we consider the following instance of DPST. The input graph is the constructed tree $(\mathcal{T}, c_T)$ with root r_T. The new polymatroid function $f_T : 2^{V_T} \to \mathbb{Z}$ is defined as $f_T(Z) := f(\{t \in S : M(t) \cap Z \neq \emptyset\})$ for every $Z \subseteq V_T$. It is not hard to see that f_T is a polymatroid, and that an evaluation oracle for f can be used to construct an evaluation oracle for f_T in polynomial time. To obtain an approximate solution for this instance, we directly apply the following result by Calinescu and Zelikovsky [5]:

▶ **Theorem 11** ([5]). *For every $\epsilon > 0$, there exists a polynomial-time $O\left(\frac{\log^{1+\epsilon} n \log k}{\epsilon \log\log n}\right)$-approximation algorithm for the Polymatroid Steiner Problem when the input graph is a tree, assuming a polynomial time oracle for the polymatroid function.*

In both DGST and DPST, we apply step (c) to project the given solutions back to a solution on the original input graphs. By Property 3, this "tree-embedding" framework loses an additional $O(\log N)$ factor in the cost. It is simple to verify that the correctness and approximation guarantees follow from appropriately applying properties of Theorem 7.

3.1 Directed Covering Steiner Tree

We are given a graph (G, c) with root r, and q groups $g_1, \ldots, g_q \subseteq V$, with requirements $h_1, \ldots, h_q$ respectively, where $\sum_i h_i = k$. The algorithm for DCST is more involved than simply instantiating the framework to solve a DCST instance on trees. Technical complications arise because after applying step (a) of the framework to obtain a tree embedding $(\mathcal{T}, c_T)$ and expanding groups to include all copies of a terminal (while keeping the same requirements, say), a solution for DCST on this instance could satisfy the requirement of the i^{th} expanded group by picking multiple copies of a single terminal from g_i. Consequently, it is unclear how to map such a solution on the tree back to a solution of our original DCST instance.

To circumvent this issue, we use an LP-based approach on the tree $(\mathcal{T}, c_T)$ and expanded groups $g'_1, \ldots, g'_q$. The natural flow-based LP relaxation on DCST sends a flow of h_i from the root to each group. We aim to modify this LP to bound the amount of flow reaching each set of copies $M(t)$. To that end, we define an LP with flow variables f_v for every $v \in V_T$ denoting the amount of flow from r_T to v and corresponding capacity variables x_e denoting the amount of flow through e for every $e \in E_T$. The LP constraints guarantee the following:

1. for each $i \in [q]$, f supports a flow of at least h_i from the root r to group g'_i,
2. for each terminal $t \in S$, f supports a flow of at most 1 to the collection of its copies $M(t)$,
3. the capacities given by x support the flow f.

It is not difficult to see that any integral feasible solution to this LP is an r-rooted tree which contains h_i *unique* terminals from each group g_i. While there are known algorithms for DCST on trees that are based on LP-rounding [31], it is not clear if these techniques work for this modified LP. We describe a procedure that iteratively rounds solutions to our LP above by leveraging a connection to the minimum *density* Directed Group Steiner Tree problem (MD-DGST) (defined in Section 1.4).

Let OPT_{LP} denote the cost of a (fractional) optimal solution (x^*, f^*) to the LP. Using our LP constraints and the fact that copies of distinct terminals are disjoint, we observe that a group g_i can be partitioned into $h_i/2$ parts ($g_i = \uplus_{j \in [h_i/2]} g_i^{(j)}$) such that each part

$g_i^{(j)}$ receives at least one unit of flow from the root r_T; here we assume that $h_i/2 \in \mathbb{Z}_+$ to simplify notation. We consider the MD-DGST instance defined on the tree $(\mathcal{T}, c_T)$, root r_T, and groups $g'_{i,j} = \cup_{t \in g_i^{(j)}} M(t)$ for every $i \in [q]$ and $j \in [h_i/2]$. We note that since (x^*, f^*) is feasible for our LP, it is a feasible fractional solution for a natural LP relaxation for MD-DGST (see [46]). Moreover, this fractional solution has density at most $2\mathrm{OPT}_{LP}/k$, since the cost is OPT_{LP} and the number of groups that receive at least one unit of flow is $\sum_{i \in [q]} h_i/2 = k/2$. To obtain a good feasible integral solution, we use the following result by Zosin and Khuller [46].

▶ **Theorem 12** ([46]). *There exists a polynomial time $O(d)$-approximation (w.r.t the LP) for the MD-DGST problem when the input graph is a directed out-tree with height d.*

Thus we can obtain a tree T_1 of density at most $O(d\ \mathrm{OPT}_{LP}/k)$. We remove all terminals $t \in S$ such that $M(t) \cap T_1 \neq \emptyset$. We then repeat this process until we satisfy the requirement that every group i contains at least h_i distinct $t \in g_i$ such that $T' \cap M(t) \neq \emptyset$. This terminates in polynomial time since each iteration removes at least one terminal. Using standard inductive arguments, one can bound the total cost of the edges by $O(d \log k)\mathrm{OPT}_{LP}$. Using Property 2 of Theorem 7, we can bound the height of $\mathcal{T}$ by $O(\log N)$. Thus we obtain a tree T' satisfying the desired conditions with cost $O(\log k \log N)\mathrm{OPT}_{LP}$. The rest of the proof is similar to that of DGST and so we omit the details for brevity.

4 Integrality Gap of Cut-based LP Relaxation

In this section, we prove Theorem 2 on the integrality gap of the LP relaxation for DST on planar graphs, defined as follows. Let $\mathcal{C} := \{U \subseteq V : r \in U \text{ and } U \cap S \neq S\}$.

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ s.t. \quad & \sum_{e \in \delta^+(U)} x_e \geq 1 \quad \forall\, U \in \mathcal{C} \\ & x_e \geq 0 \quad \forall\, e \in E \end{aligned} \tag{DST-LP}$$

This is a relaxation of the integer program where, for every edge $e \in E$, we have a variable $x_e \in \{0, 1\}$ which indicates whether e is contained in the solution. For every subset U of vertices containing root r with $U \cap S \neq S$, a feasible solution must contain at least one edge in $\delta^+(U)$, since it must contain a path from the root r to every terminal in $S \setminus U$. The LP contains an exponential number of constraints but has an efficient separation oracle (an s-t min-cut computation). One can also formulate a compact extended formulation with additional flow variables (see [36]).

We now prove that integrality gap of the LP is at most $O(\log^2 k)$ via a constructive procedure. One can view the algorithm as running the recursive algorithm of [25] by using the LP value as the estimate for the optimum. Given an arbitrary feasible solution x of (DST-LP), Algorithm 3 constructs a directed Steiner tree.

The base case is when there are at most six terminals; the algorithm connects each terminal to the root directly via shortest paths. Otherwise, the algorithm first scales up the x_e values for every $e \in E$ by a factor of $\left(1 + \frac{1}{\log |S|}\right)$. Then, it calls PruneAndSeparate$((G, c), r, S, 2 \log |S| \cdot \sum_{e \in E} c_e x_e)$ and obtains several sub-instances; recall that the separator P is contracted into the root in each sub-instance. The algorithm recursively solves these sub-instances and returns the corresponding solution in the original graph G. We observe that we do not re-solve the LP in the recursion but instead use the induced fractional solution after scaling up x.

Algorithm 3 ROUNDLP(G, r, S, x).

---

if $|S| \leq 6$ **then**
  return T obtained by connecting each terminal t to r via a shortest r-t path
$\bar{x}_e \leftarrow \left(1 + \frac{1}{\log|S|}\right) \cdot x_e$
$(P, C_1, \ldots, C_\ell) \leftarrow$ PRUNEANDSEPARATE$(G, r, S, 2\log|S| \cdot \sum_{e\in E} c_e \cdot x_e)$ (Algorithm 1)
$T \leftarrow P$
for $i = 1, \ldots, \ell$ **do**
  $\bar{x}^{(i)} \leftarrow$ the values of $\bar{x}$ restricted to $E(C_i)$.
  Compute ROUNDLP$(C_i, r, S \cap C_i, \bar{x}^{(i)})$ to obtain tree $T_i = (V_i, E_i)$ with root r_i.
  Augment T with T_i by replacing each edge from r_i with an edge from the corresponding uncontracted node of P.
return (T, r)

We now analyze the correctness and cost of the tree returned by the algorithm with respect to the cost of the LP solution $\sum_e c_e x_e$. In the recursive case we first show that $\bar{x}$ is a feasible LP solution after removing the distant vertices.

▶ **Lemma 13.** *Let $G = (V, E)$ be a directed planar graph with edge costs $c : E \to \mathbb{R}_{\geq 0}$, a root $r \in V$, and a set of terminals $S \subseteq V$ of size k. Let x be a feasible solution of (DST-LP). Let $V' := \{v \in V : d(r, v) > 2\log k \cdot \sum_{e\in E} c_e x_e\}$ and E' be set of edges that are incident to some vertex in V'. Let $\bar{x}$ be a fractional solution where $\bar{x}_e := 0$ if $e \in E'$, and otherwise $\bar{x}_e := \left(1 + \frac{1}{\log k}\right) \cdot x_e$. Then, $\bar{x}$ is a feasible solution to (DST-LP) for the given instance.*

Proof. We note that $\bar{x}_e \geq 0$ since $x_e \geq 0$ for every $e \in E$. It suffices to show that for every $U \in \mathcal{C}$, we have $\sum_{e\in\delta^+(U)} \bar{x}_e \geq 1$. Let $t \in S$ be a terminal such that $t \notin U$. Since x is a feasible solution of (DST-LP), x supports a unit flow from root r to terminal t. Thus $\sum_e c_e x_e \geq d_G(r, t)$. Since every path from root r to a terminal t passing through V' has length more than $2\log k \cdot \sum_{e\in E} c_e x_e$ (by definition of V'), by Markov's inequality, the amount of flow supported by x from r to t passing through V' is smaller than $\frac{1}{2\log k}$. This tells us that after removing all vertices in V', x still supports a flow of at least $1 - \frac{1}{2\log k}$ from r to t. Hence, we have $\sum_{e\in\delta^+(U)} \bar{x}_e = \left(1 + \frac{1}{\log k}\right) \cdot \sum_{e\in\delta^+(U)\setminus E'} x_e \geq \left(1 + \frac{1}{\log k}\right) \cdot \left(1 - \frac{1}{2\log k}\right) \geq 1$. ◀

The lemma below bounds the cost of the tree returned by Algorithm 3, which concludes the proof of Theorem 2.

▶ **Lemma 14.** *Given a directed planar graph $G = (V, E)$ with edge costs $c : E \to \mathbb{R}_{\geq 0}$, a root $r \in V$, and a set of terminals $S \subseteq V$ of size k. Let x be a feasible solution of (DST-LP). ROUNDLP(G, r, S, x), returns a feasible directed Steiner tree with cost $O(\log^2 k) \cdot \sum_{e\in E} c_e x_e$.*

Proof. By induction on the number of terminals, we prove that the cost of the tree output by the algorithm is at most $6(\log k + 1)^2 \cdot \sum_{e\in E} c_e x_e$.

First we consider the base case when $k \leq 6$. We observe that if x is a feasible solution then for every terminal t the length of the shortest r-t path in G is at most the fractional LP cost. Thus the algorithm outputs a feasible solution whose cost is at most $6\sum_{e\in E} c_e x_e$.

Consider the case where $k > 6$. Algorithm 3 finds three paths P_1, P_2, P_3 and contracts their union into r to obtain graph G_P. The contraction creates several independent sub-instances $(C_1, S \cap C_1), \ldots, (C_\ell, S \cap C_\ell)$ where $|S \cap C_i| \leq k/2$ for every $1 \leq i \leq \ell$. By Lemma 13, contraction preserves the feasibility of the LP solution induced on the residual instance. Moreover, it is easy to see that any integer solution to the residual instance together with P is a feasible integer solution to the original instance. It remains to do the cost analysis. Let $COST(G, S)$ be the cost of the tree output by Algorithm 3.

The length of each of the paths P_1, P_2, P_3 is at most $2 \log k \cdot \sum_{e \in E} c_e x_e$, and thus their total cost is at most $6 \log k \cdot \sum_{e \in E} c_e x_e$. Since each edge e of the graph $G - V'$ is in at most one sub-instance, we have

$$\left(1 + \frac{1}{\log k}\right) \cdot \sum_{e \in E} c_e x_e \quad = \quad \sum_{e \in E} c_e \bar{x}_e \quad \geq \quad \sum_{i=1}^{\ell} \sum_{e \in E(C_i)} c_e \bar{x}_e^{(i)}. \tag{1}$$

By the induction hypothesis,

$$\begin{aligned}\sum_{i=1}^{\ell} COST(C_i', S \cap C_i) &\leq \sum_{i=1}^{\ell} 6(\log |S \cap C_i| + 1)^2 \cdot \sum_{e \in E(C_i)} c_e \bar{x}_e^{(i)} \\ &\leq \sum_{i=1}^{\ell} 6 \log^2 k \cdot \sum_{e \in E(C_i)} c_e \bar{x}_e^{(i)} && (\text{since } |C \cap C_i| \leq k/2) \\ &\leq 6 \log^2 k \cdot \left(1 + \frac{1}{\log k}\right) \cdot \sum_{e \in E} c_e x_e && (\text{by inequality (1)})\end{aligned}$$

Thus, $\mathrm{COST}(G, S) \leq 6 \log k \cdot \sum_{e \in E} c_e x_e + \sum_{i=1}^{\ell} \mathrm{COST}(C_i', S \cap C_i) \leq 6(\log k + 1)^2 \cdot \sum_{e \in E} c_e x_e$, completing the induction proof. ◀

5 Multi-Rooted problems via Density Argument

In this section we consider the multi-rooted versions of the DST, DGST, DCST and DPST and prove Theorem 3. As we remarked, we cannot directly reduce the multi-rooted problems to the single-root version while preserving planarity of the graph. We use a simple strategy via density-based arguments. The following Lemma is an easy consequence of iteratively using the min-density algorithm and applying a standard inductive argument; we defer the proof to a full version [11].

▶ **Lemma 15.** *Let $\mathcal{G}$ be a minor-closed family of graphs. Suppose there is an $\alpha(k, n)$-approximation algorithm for the minimum-density DST (MD-DST) problem on instances from graphs in $\mathcal{G}$ containing n nodes and k terminals. Then there is an $O(\alpha(k, n) \log k)$-approximation for the multi-root version of DST on graphs from $\mathcal{G}$ with n nodes and k terminals.*

Thus, in order to approximately solve the multi-root version, it suffices to solve the min-density version of the single root problem. By adapting the approach of [25] we prove the following.

▶ **Theorem 16.** *There is an $O(\log k)$-approximation for MD-DST problem in planar graphs.*

Proof. We are given a graph $(G = (V, E), c)$ with a root $r \in V$ and a set of terminals $S \subseteq V \setminus \{r\}$. Recall that our goal is to find an r-tree T minimizing $c(T)/|T \cap S|$. We can guess the optimal value of $|T \cap S|$ by iterating over all $\ell \in [|S|]$ and choosing the tree with minimum density. Thus we reduce to the problem of finding a tree of minimum cost that contains ℓ terminals; we refer to this problem as ℓ-DST.

When the input graph G is a tree rooted at r, there is a folklore dynamic programming algorithm that solves this problem exactly. We may assume G is a binary tree where non-leaf nodes have out degree at most two[3]. We recursively compute ℓ'-DST for all values $\ell' < \ell$ on both subtrees of G; it is straightforward to combine to get the optimal solution.

[3] If a node in G has out-degree larger than 2, we can create an out arrow to a new node with edge cost 0 that handle all but one of its children. The resulting tree will still have size $O(n)$.

When the input graph G is a directed planar graph, we can use Theorem 7 to construct a tree $\mathcal{T}$. However we cannot naively apply the dynamic programming algorithm as described above to $\mathcal{T}$, since some terminals are duplicated. With a bit of care, we can still easily handle these duplications. In the construction of $\mathcal{T}$, terminals are only duplicated when we make a recursive call corresponding to halving the guest value of the optimum solution γ. Thus in nodes that make this recursive call, we add a constraint that we are restricted to taking an optimum solution in only one of the two subtrees. It is easy to see that the proof of Property 3 of Theorem 7 satisfies this constraint; thus this framework only loses an $O(\log k)$ approximation factor in the cost. ◀

▶ **Corollary 17.** *There is an $O(\log^2 k)$-approximation for multi-root DST in planar graphs.*

▶ Remark 18. Via density-based argument one can also prove that the integrality gap of a natural cut-based LP relaxation for the multi-root version of DST is at most $O(\log^3 k)$. This upper bound is unlikely to be tight; we leave the improvement in the bound to future work.

The density-based argument extends in a natural fashion to multi-rooted versions of DGST, DCST, and DPST. In fact, we are able to attain the approximation ratios that we get in Section 3 for the single-rooted case. This is not surprising, as the algorithms for DGST, DCST, and DPST in trees can all be obtained through the corresponding min-density problems. The tree embedding argument from Section 2 shows that one can reduce the min-density problem in planar graphs to one on trees at the loss of an $O(\log N)$ factor in the approximation ratio. Moreover, the height of the resulting tree can be assumed to $O(\log N)$. For GST on trees with height d there is an $O(d)$ approximation for the min-density problem [46]. Thus, there is an $O(\log^2 N)$-approximation for the min-density DGST in planar graphs. Combining the ingredients, we obtain an $O(\log k \log^2 N)$-approximation for the multi-root version of DGST in planar graphs. We obtain the same approximation factor for DCST in planar graphs, using a similar argument to that of Section 3 to obtain an $O(\log^2 N)$-approximation to the min-density DCST problem in planar graphs. For DPST on trees, implicit in [5] is an algorithm that yields an $O(\frac{\log^{1+\epsilon} n}{\epsilon \log\log n})$-approximation for the min-density problem on trees. Combining it with the tree embedding and the iterative procedure, we obtain an $O(\frac{\log^{1+\epsilon} n \log k \log N}{\epsilon \log\log n})$-approximation for the multi-rooted version of DPST in planar graphs where $k = f(V)$. This proves Theorem 3.

References

1 Amit Agarwal and Moses Charikar. On the advantage of network coding for improving network throughput. In *2004 IEEE Information Theory Workshop, San Antonio, TX, USA, 24-29 October, 2004*, pages 247–249. IEEE, 2004. doi:10.1109/ITW.2004.1405308.

2 MohammadHossein Bateni, Erik D. Demaine, MohammadTaghi Hajiaghayi, and Dániel Marx. A PTAS for planar group steiner tree via spanner bootstrapping and prize collecting. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 570–583. ACM, 2016. doi:10.1145/2897518.2897549.

3 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. An $O(n \log n)$ approximation scheme for steiner tree in planar graphs. *ACM Trans. Algorithms*, 5(3):31:1–31:31, 2009. doi:10.1145/1541885.1541892.

4 Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. Steiner tree approximation via iterative randomized rounding. *J. ACM*, 60(1):6:1–6:33, 2013. doi:10.1145/2432622.2432628.

5 Gruia Călinescu and Alexander Zelikovsky. The polymatroid steiner problems. *J. Comb. Optim.*, 9(3):281–294, 2005. doi:10.1007/S10878-005-1412-9.

6 Parinya Chalermsook, Fabrizio Grandoni, and Bundit Laekhanukit. On survivable set connectivity. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 25–36. SIAM, 2015. doi:10.1137/1.9781611973730.3.

7 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. doi:10.1006/JAGM.1999.1042.

8 Moses Charikar, Chandra Chekuri, Ashish Goel, and Sudipto Guha. Rounding via trees: deterministic approximation algorithms for group steiner trees and k-median. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 114–123, New York, NY, USA, May 1998. Association for Computing Machinery. doi:10.1145/276698.276719.

9 Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Trans. Algorithms*, 7(2):18:1–18:17, 2011. doi:10.1145/1921659.1921664.

10 Chandra Chekuri, Guy Even, and Guy Kortsarz. A greedy approximation algorithm for the group steiner problem. *Discrete Applied Mathematics*, 154(1):15–34, 2006. doi:10.1016/j.dam.2005.07.010.

11 Chandra Chekuri, Rhea Jain, Shubhang Kulkarni, Da Wei Zheng, and Weihao Zhu. From directed steiner tree to directed polymatroid steiner tree in planar graphs, 2024.

12 Chandra Chekuri and Martin Pál. A recursive greedy algorithm for walks in directed graphs. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 245–253. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.9.

13 Q. Chen, B. Laekhanukit, C. Liao, and Y. Zhang. Survivable network design revisited: Group-connectivity. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 278–289, Los Alamitos, CA, USA, November 2022. IEEE Computer Society. doi:10.1109/FOCS54457.2022.00033.

14 Miroslav Chlebík and Janka Chlebíková. The steiner tree problem on graphs: Inapproximability results. *Theor. Comput. Sci.*, 406(3):207–214, 2008. doi:10.1016/J.TCS.2008.06.046.

15 Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Philip N. Klein. Node-weighted steiner tree and group steiner tree in planar graphs. *ACM Trans. Algorithms*, 10(3):13:1–13:20, 2014. doi:10.1145/2601070.

16 Guy Even, Guy Kortsarz, and Wolfgang Slany. On network design problems: fixed cost flows and the covering steiner problem. *ACM Trans. Algorithms*, 1(1):74–101, 2005. doi:10.1145/1077464.1077470.

17 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. doi:10.1016/J.JCSS.2004.04.011.

18 Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998. doi:10.1145/285055.285059.

19 Christina Fragouli and Emina Soljanin. Network coding applications. *Found. Trends Netw.*, 2(2):135–269, 2007. doi:10.1561/1300000013.

20 Christina Fragouli and Emina Soljanin. Network coding fundamentals. *Found. Trends Netw.*, 2(1), 2007. doi:10.1561/1300000003.

21 András Frank. *Connections in combinatorial optimization*, volume 38. Oxford University Press Oxford, 2011.

22 Zachary Friggstad, Jochen Könemann, Young Kun-Ko, Anand Louis, Mohammad Shadravan, and Madhur Tulsiani. Linear programming hierarchies suffice for directed steiner tree. In Jon Lee and Jens Vygen, editors, *Integer Programming and Combinatorial Optimization - 17th International Conference, IPCO 2014, Bonn, Germany, June 23-25, 2014. Proceedings*,

volume 8494 of *Lecture Notes in Computer Science*, pages 285–296. Springer, 2014. doi:10.1007/978-3-319-07557-0_24.

23 Zachary Friggstad, Jochen Könemann, and Mohammad Shadravan. A logarithmic integrality gap bound for directed steiner tree in quasi-bipartite graphs. In Rasmus Pagh, editor, *15th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2016, June 22-24, 2016, Reykjavik, Iceland*, volume 53 of *LIPIcs*, pages 3:1–3:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICS.SWAT.2016.3.

24 Zachary Friggstad and Ramin Mousavi. A constant-factor approximation for quasi-bipartite directed steiner tree on minor-free graphs. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*, 275:13:1–13:18, 2023. doi:10.4230/LIPICS.APPROX/RANDOM.2023.13.

25 Zachary Friggstad and Ramin Mousavi. An $O(\log k)$-approximation for directed steiner tree in planar graphs. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 63:1–63:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.ICALP.2023.63.

26 Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000. doi:10.1006/JAGM.2000.1096.

27 Rohan Ghuge and Viswanath Nagarajan. Quasi-polynomial algorithms for submodular tree orienteering and directed network design problems. *Math. Oper. Res.*, 47(2):1612–1630, 2022. doi:10.1287/MOOR.2021.1181.

28 Michel X. Goemans, Neil Olver, Thomas Rothvoß, and Rico Zenklusen. Matroids and integrality gaps for hypergraphic steiner tree relaxations. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1161–1176. ACM, 2012. doi:10.1145/2213977.2214081.

29 Fabrizio Grandoni, Bundit Laekhanukit, and Shi Li. $O(\log^2 k/\log\log k)$-approximation algorithm for directed steiner tree: A tight quasi-polynomial time algorithm. *SIAM Journal on Computing*, 52(2):298–322, 2022. doi:10.1137/20M1312988.

30 A. Gupta and J. Könemann. Approximation algorithms for network design: A survey. *Surveys in Operations Research and Management Science*, 16(1):3–20, 2011.

31 Anupam Gupta and Aravind Srinivasan. An improved approximation ratio for the covering steiner problem. *Theory Comput.*, 2(3):53–64, 2006. doi:10.4086/TOC.2006.V002A003.

32 Eran Halperin, Guy Kortsarz, Robert Krauthgamer, Aravind Srinivasan, and Nan Wang. Integrality ratio for group steiner trees and directed steiner trees. *SIAM J. Comput.*, 36(5):1494–1511, 2007. doi:10.1137/S0097539704445718.

33 Eran Halperin and Robert Krauthgamer. Polylogarithmic inapproximability. In Lawrence L. Larmore and Michel X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 585–594. ACM, 2003. doi:10.1145/780542.780628.

34 Goran Konjevod, R. Ravi, and Aravind Srinivasan. Approximation algorithms for the covering steiner problem. *Random Struct. Algorithms*, 20(3):465–482, 2002. doi:10.1002/RSA.10038.

35 G. Kortsarz and Z. Nutov. Approximating minimum cost connectivity problems. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010.

36 Shi Li and Bundit Laekhanukit. Polynomial integrality gap of flow LP for directed steiner tree. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 3230–3236. SIAM, 2022. doi:10.1137/1.9781611977073.126.

37 Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 255–264. ACM, 2008. doi:10.1145/1374376.1374415.

38 Gabriele Reich and Peter Widmayer. Beyond steiner's problem: A VLSI oriented generalization. In Manfred Nagl, editor, *Graph-Theoretic Concepts in Computer Science, 15th International Workshop, WG '89, Castle Rolduc, The Netherlands, June 14-16, 1989, Proceedings*, volume 411 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 1989. doi:10.1007/3-540-52292-1_14.

39 Thomas Rothvoß. Directed steiner tree and the lasserre hierarchy. *CoRR*, abs/1111.5473, 2011. arXiv:1111.5473.

40 Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*. Springer, 2003.

41 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, November 2004. doi:10.1145/1039488.1039493.

42 Vijay V Vazirani. *Approximation algorithms*, volume 1. Springer, 2001.

43 David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.

44 Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Comb.*, 2(4):385–393, 1982. doi:10.1007/BF02579435.

45 Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997. doi:10.1007/BF02523690.

46 Leonid Zosin and Samir Khuller. On directed steiner trees. In David Eppstein, editor, *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA*, pages 59–63. ACM/SIAM, 2002. URL: http://dl.acm.org/citation.cfm?id=545381.545388.