Polylogarithmic Approximation Algorithm for \( k \)-Connected Directed Steiner Tree on Quasi-Bipartite Graphs

Chun-Hsiang Chan  
Department of Computer Science, University of Michigan, Ann Arbor, MI, USA  
kenhchan@umich.edu

Bundit Laekhanukit  
Institute for Theoretical Computer Science, Shanghai University of Finance & Economics, China  
http://itcs.shufe.edu.cn/~blaekh  
bundit@sufe.edu.cn

Hao-Ting Wei  
Department of IEOR, Columbia University, New York, NY, USA  
hw2738@columbia.edu

Yuhao Zhang  
Department of Computer Science, The University of Hong Kong, China  
https://i.cs.hku.hk/~yhzhang2/  
yhzhang2@cs.hku.hk

Abstract  
In the \( k \)-Connected Directed Steiner Tree problem (\( k \)-DST), we are given a directed graph \( G = (V, E) \) with edge (or vertex) costs, a root vertex \( r \), a set of \( q \) terminals \( T \), and a connectivity requirement \( k > 0 \); the goal is to find a minimum-cost subgraph \( H \) of \( G \) such that \( H \) has \( k \) edge-disjoint paths from the root \( r \) to each terminal in \( T \). The \( k \)-DST problem is a natural generalization of the classical Directed Steiner Tree problem (DST) in the fault-tolerant setting in which the solution subgraph is required to have an \( r, t \)-path, for every terminal \( t \), even after removing \( k - 1 \) vertices or edges. Despite being a classical problem, there are not many positive results on the problem, especially for the case \( k \geq 3 \). In this paper, we present an \( O(\log k \log q) \)-approximation algorithm for \( k \)-DST when an input graph is quasi-bipartite, i.e., there is no edge joining two non-terminal vertices. To the best of our knowledge, our algorithm is the only known non-trivial approximation algorithm for \( k \)-DST, for \( k \geq 3 \), that runs in polynomial-time Our algorithm is tight for every constant \( k \), due to the hardness result inherited from the Set Cover problem.

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

Keywords and phrases  Approximation Algorithms, Network Design, Directed Graphs

Digital Object Identifier  10.4230/LIPIcs.APPROX/RANDOM.2020.63

Category  APPROX


Funding  Bundit Laekhanukit: Partially supported by Science and Technology Innovation 2030 – “New Generation of Artificial Intelligence” Major Project No.(2018AAA0100903), NSFC grant 61932002, Program for Innovative Research Team of Shanghai University of Finance and Economics (IRTS SHUFE) and the Fundamental Research Funds for the Central Universities. Also, partially supported by the 1000-talent award by the Chinese Government.

Acknowledgements  The works were initiated while all the authors were at the Institute for Theoretical Computer Science at the Shanghai University of Finance and Economics, and the work were done while Chun-Hsiang Chan and Hao-Ting Wei were in bachelor and master programs in Computer Science at the Institute of Information Science, Academia Sinica, Taipei.

1 Introduction

Designing a network that can operate under failure conditions is an important task for Computer Networking in both theory and practice. Many models have been proposed to capture this problem, giving rise to the area of survivable and fault-tolerant network design. In the past few decades, there have been intensive studies on the survivable network design problems; see, e.g., [55, 30, 35, 20, 14, 46, 32]. The case of link-failure is modeled by the Edge-Connectivity Survivable Network Design problem (EC-SNDP), which is shown to admit a 2-approximation algorithm by Jain [35]. The case of node-failure is modeled by the Vertex-Connectivity Survivable Network Design problem (VC-SNDP), which is shown to admit a polylogarithmic approximation algorithm by Chuzhoy and Khanna [14]. Nevertheless, most of the known algorithmic results pertain to only undirected graphs, where each link has no prespecified direction. In the directed case, only a few results are known as the general case of Survivable Network Design is at least as hard as the Label-Cover problem [16], which is believed to admit no sub-polynomial approximation algorithm [44, 2].

This paper studies the special case of the Survivable Network Design problem on directed graphs, namely the \( k \)-Connected Directed Steiner Tree problem (\( k \)-DST), which is also known as the Directed Root \( k \)-Connectivity. In this problem, we are given an \( n \)-vertex directed graph \( G = (V, E) \) with edge-costs \( c : E \rightarrow \mathbb{R}_+^+ \), a root vertex \( r \), a set of \( q \) terminals \( T \subseteq V - \{r\} \) and a connectivity requirement \( k \in \mathbb{Z}^+ \); the goal is to find a minimum-cost subgraph \( H \subseteq G \) that has \( k \) edge-disjoint \( r,t \)-paths for every terminal \( t \in T \). This problem was mentioned in [19] and have been subsequently studied in [12, 41, 8, 43, 32]. The only known non-trivial approximation algorithms for \( k \)-DST is for the case \( k = 2 \) due to the work of Grandoni and Laekhanukit [32], and for the case of \( \gamma \)-shallow instances due to the work of Laekhanukit [43]. To the best of our knowledge, for \( k \geq 3 \), there were only a couple of positive results on \( k \)-DST: (1) Laekhanukit [43] devised an approximation algorithm whose running time and approximation ratios depend on the diameter of the optimal solution, and (2) Chalermsook, Grandoni and Laekhanukit [8] devised a bi-criteria approximation algorithm for a special case of \( k \)-DST, namely the \( k \)-Edge-Connected Group Steiner Tree (\( k \)-GST), where the solution subgraph is guaranteed to be an \( O(\log^2 n \log k) \)-approximate solution, whereas the connectivity is only guaranteed to be at least \( \Omega(k/\log n) \). We focus on the case of \( k \)-DST where an input graph is quasi-bipartite, i.e., there is no edge joining any pair of non-terminal (Steiner) vertices, which generalizes the works of Hibi and Fujito [34], and Friggstad, Könemann and Shadravan [26] for the classical directed Steiner tree problem (the case \( k = 1 \)).

The main contribution of this paper is an \( O(\log q \log k) \)-approximation algorithm for \( k \)-DST on quasi-bipartite graphs, which runs in polynomial-time regardless of the structure of the optimal solution. Our result can be considered the first true polylogarithmic approximation algorithm whose running time is independent of the structure (i.e., diameter) of the optimal solution, albeit the algorithm is restricted to the class of quasi-bipartite graphs. Our technique is different from all the previous works on \( k \)-DST [32, 43, 8]; all these results rely on the tree-rounding algorithm for the Group Steiner Tree problem by Garg, Konjevod and Ravi [27], and thus require either an LP whose support is a tree or a tree-embedding technique (e.g., Räcke’s decomposition [50] as used in [8]). Our algorithm, on the other hand, employs the Halo-Set decomposition devised by Kortsarz and Nutov [39] and further developed in a series

\[1\] While we define the problem here as an edge-connectivity problem, our algorithm itself works for both edge and vertex connectivity variants, and can handle both edge and vertex costs.
of works [17, 11, 46, 48, 42, 47]. It is worth noting that the families of subsets decomposed from our algorithm are not uncrossable. We circumvent this difficulty by reducing it to the Set Cover problem. Our algorithm can be seen as a variant of the spider decomposition method developed by Klein and Ravi [36], and Nutov [45].

Lastly, we remark that it was discussed in [32] that the tree-embedding approach reaches the barrier as soon as \( k > 2 \), and this holds even for quasi-bipartite graphs. Please see Appendix B for discussions. While our algorithm exploits the structure of quasi-bipartite graphs, we hope that our technique using the Halo-Set decomposition would be an alternative method that sheds some light on developing approximation algorithms for the general case of \( k \)-DST for \( k > 2 \).

### 1.1 Related Works

Directed Steiner tree has been a central problem in combinatorial optimization. There have been a series of work studying this problem; see, e.g., [56, 9, 54, 25, 33, 29]. The best approximation ratio of \( O(q\epsilon) \), for any \( \epsilon < 0 \), in the regime of polynomial-time algorithms, is known in the early work of Charikar et al. [9]\(^2\), which leads to an \( O(\log^2 q) \)-approximation algorithm that runs in quasi-polynomial-time. Very recently, Grandoni, Laekhanukit and Li [33] developed a framework that gives a quasi-polynomial-time \( O(\log^2 q / \log \log q) \)-approximation algorithm for the Directed Steiner Tree problem, and this approximation ratio is the best possible for quasi-polynomial-time algorithms, assuming the Projection Games Conjecture and \( \text{NP} \subseteq \bigcup_{\epsilon > 0} \text{ZPTIME}(2^{n^{\epsilon}}) \). The same approximation ratio was obtained in an independent work of Ghuge and Nagarajan [29].

The study of Steiner tree problems on quasi-bipartite graphs was initiated by Rajagopalan and Vazirani [51] in order to understand the bidirected-cut relaxation of the (undirected) Steiner tree problem. Since then the special case of quasi-bipartite graphs has played a central role in studying the Steiner tree problem; see, e.g., [52, 6, 53, 37, 5, 31]. For the case of directed graphs, Hibi and Fujito [34], Friggstad, Könemann and Shadravan [26] independently discovered \( O(\log n) \)-approximation algorithms for the directed Steiner tree problem on quasi-bipartite graphs. Assuming \( P \neq \text{NP} \), this matches to the lower bound of \((1 - \epsilon) \ln n\), for any \( \epsilon > 0 \), inherited from the Set Cover problem [18, 15].

The generalization of the Steiner tree problem is known as the Survivable Network Design problem, which has been studied in both edge-connectivity [55, 30, 35], vertex-connectivity [14] and element-connectivity [20] settings. The edge and element connectivity Survivable Network Design problems admit factor 2 approximation algorithms via the iterative rounding method, while the vertex-connectivity variant admits no polylogarithmic approximation algorithm [38, 7, 41] unless \( \text{NP} \subseteq \text{DTIME}(n^{\text{polylog}(n)}) \). To date, the best approximation ratio known for the Vertex-Connectivity Survivable Network problem is \( O(k^3 \log n) \) due to the work of Chuzhoy and Khanna [14]. The single-source \( k \)-vertex-connectivity variant, which is closely related to the problem considered in this paper, has been studied in [7, 13, 10, 46, 49], culminating in the best approximation ratio of \( O(k \log k) \) due to Nutov [46].

In vertex-connectivity network design, one of the most common techniques is the Halo-Set decomposition method, which has been developed in a series of works [39, 17, 11, 48]. The main idea is to use the number of minimal deficient sets as a notion of progress. Here a deficient set is a subset of vertices that needs at least one incoming edge to satisfy the connectivity requirement. The minimal deficient sets in [39, 17, 11, 48], called cores, are

---

\(^2\) The same result can be obtained by applying the algorithm by Kortsarz and Peleg in [40]
independent and have only polynomial number, while the total number of deficient sets is exponential on the number of vertices. The families of deficient sets defined by these cores allow us to keep track of how many deficient sets remain in a solution subgraph. The early version of this method can be traced back to the seminal result of Frank [22] and that of Frank and Jordan [23]; please see [24] for reference therein.

The spider decomposition method was introduced by Klein and Ravi [36] to handle the Vertex-Weighted Steiner Tree problem. This technique gives a tight approximation result (up to constant factor) to the problem. Later, Nutov generalized the technique to deal with the Minimum Power-Cover problems [45] and subsequently for the Vertex-Weighted Element-Connectivity Survivable Network Design problem [46].

1.2 Our Result

The main result in our paper is an $O(\log q \log k)$-approximation algorithm for $k$-DST on quasi-bipartite graphs. To keep the flow, our algorithm is presented as a randomized algorithm. The derandomization is provided in Appendix A. Since our algorithm is LP-based, it also gives an upper bound on the integrality gap of the standard LP-relaxation.

▶ Theorem 1. Consider the $k$-Connected Directed Steiner Tree problem where an input graph consists of an $n$-vertex quasi-bipartite graph and a set of $q$ terminals. There exists a polynomial-time $O(\log q \log k)$-approximation algorithm. Moreover, the algorithm gives an upper bound on the integrality gap of $O(\log q \log k)$ for the standard cut-based LP-relaxation of the problem.

2 Preliminaries

We use standard graph terminologies. Given a graph $G$, we denote by $V(G)$ and $E(G)$ the vertex set and edge set of $G$, respectively. For any subset of vertices $U \subseteq V(G)$, we denote by $\delta^+_G(U)$ the set of edges in $G$ entering the set $U$ and denote by $\deg^+_G(U)$ its cardinality. We denote by $E_G(U)$ the set of edges that have both head and tail in $U$. That is,

$$\delta^+_G(U) = \{vw \in E(G) : v \in U, w \notin U\}, \quad \deg^+_G(U) = |\delta^+_G(U)|,$$

and

$$E_G(U) = \{vw \in E(G) : v, w \in U\}.$$  

We will omit the subscript $G$ if the graph $G$ is known in the context, and we may replace $E_G$ with another edge-set, e.g., $E_+$. Then, for any subset of edges $E'$, we denote the total cost of edges in $E'$ by $\text{cost}(E') = \sum_{e \in E'} c_e$.

2.1 Problem Definitions

$k$-Edge-Connected Directed Steiner Tree ($k$-DST)

In the $k$-Edge-Connected Directed Steiner Tree problem (k-DST), we are given a graph $G$ with non-negative edge-costs $c : E \rightarrow \mathbb{R}_0^+$, a root vertex $r$ and a set of $q$ terminals $T \subseteq (V(G) - \{r\})$, and the goal is to find a minimum-cost subgraph $H \subseteq G$ such that $H$ has $k$ edge-disjoint $r \rightarrow t$-paths for every terminal $t \in T$.

Rooted Connectivity Augmentation (Rooted-Aug)

In Rooted-Aug, we are given a graph $G$ with the edge-set $E(G) = E_0 \cup E_+$, where $E_0$ is the set of zero-cost edges and $E_+$ is the set of positive-cost edges, a root vertex $r$ and a set of terminals $T \subseteq V(G) - r$ such that $E_0$ induces a subgraph $G_0 \subseteq G$ that has $\ell$ edge-disjoint
r → t-paths for every terminal t ∈ T. The goal in this problem is to find a minimum-cost subset of edges E′ ⊆ E+ such that E0 ∪ E′ induces a subgraph H ⊆ G that has ℓ + 1 edge-disjoint r → t-paths for every terminal t ∈ T.

We may phrase Rooted-Aug as a problem of covering deficient sets as follows. We say that a subset of vertices U ⊆ V(G) is a deficient set if U separates the root vertex r and some terminal t ∈ T, but U has less than ℓ + 1 incoming edges (which means that U has exactly ℓ incoming edges); that is, U is a deficient set if r /∈ U, U ∩ T ≠ Φ and deg_G^in(U) = ℓ. These subsets of vertices need at least one incoming edge to satisfy the connectivity requirement. We say that an edge e ∈ E+ covers a deficient set U if deg_{G_e,U}^in(U) ≥ ℓ, which means that adding e to G0 satisfies the connectivity requirement on U.

Let F denote the set of all deficient sets in the graph G0. Then Rooted-Aug may be phrased as the problem of finding a minimum-cost subset of edges E′ ⊆ E+ that covers all the deficient sets, which can be described by the following optimization problem:

\[ \min \{ E' \subseteq E_+ : \text{deg}_{G_e,U}^in(U) \geq 1 \ \forall U \in F \} . \]

### 2.2 Deficient Sets, Cores and Halo-families

This section discusses subsets of vertices called deficient sets that certify that the current solution subgraph in Rooted-Aug (and also in k-DST) does not meet the connectivity requirement. To be formal, a subset of vertices U ⊆ V(G) is called a deficient set in the graph G if T ∩ U ≠ Φ, r /∈ U and deg_G^in(U) < k; that is, (V(G) − U, U) induces an edge-cut of size < k that separates some terminal t ∈ U ∩ T from the root vertex r. We say that an edge vw /∈ E(G) covers a deficient set U if deg_{G_0 + vw}^in(U) ≥ k, i.e., the set U is not a deficient set after adding the edge vw. Similarly, we say that a subset of edges E′ covers a deficient set or a collection of deficient sets F if deg_{G + E'}^in(U) ≥ k, for every deficient set U ∈ F.

Let F be a family of deficient sets. A core C ∈ F is a deficient set such that there is no deficient set in F properly contained in C. The Halo-family Halo(C) of a core C is the collection of all deficient sets in F that contain C but no other core C′ ≠ C. The Halo-set H(C) of C is the union of all the deficient sets in Halo(C), i.e., H(C) = \bigcup_{U ∈ Halo(C)} U.

### 2.3 LP-relaxations

Throughout this paper, we will use the following standard (cut-based) LP-relaxation for k-DST and Rooted-Aug. Our LP-relaxations will be written in terms of deficient sets. We denote by Val(z) the cost of the optimal solution to an LP z.

#### LP for k-DST

Here we present the standard cut-based LP-relaxation for k-DST, denoted by LP(k). The collection of deficient sets in this LP is defined by F(k) = \{ U ⊆ V − \{ r \} : U ∩ T ≠ Φ \}.

\[
\text{LP(k) = } \begin{cases} 
\min & \sum_{e \in E} c_e x_e \\
\text{s.t.} & \sum_{e \in E^+_G(U)} x_e \geq k \quad \forall U \in F(k) \\
& 0 \leq x_e \leq 1 \quad \forall e \in E(G)
\end{cases}
\]
LP for Rooted-Connectivity Augmentation

Here we assume that the initial graph \( G_0 \) is already \( \ell \)-rooted-connected, and the goal is to add edges to increase the connectivity of the solution subgraph by one. Thus, the collection of deficient sets in this problem is defined by \( \mathcal{F}(\ell) = \{ U \subseteq V : U \cap T \neq \emptyset, \deg^{in}_{G_0}(U) = \ell \} \). Below is the standard cut-based LP-relaxation for the problem of increasing the rooted-connectivity of a graph by one.

\[
\text{LP}^{\text{aug}}(\ell) = \begin{cases} 
\min & \sum_{E(G) - E(G_0)} c_e x_e \\
\text{s.t.} & \sum_{e \in \delta^0_{E(G) - E(G_0)}(U)} x_e \geq 1 \quad \forall U \in \mathcal{F}(\ell) \\
& 0 \leq x_e \leq 1 \quad \forall e \in E(G) - E(G_0)
\end{cases}
\]

3 Properties of Deficient Sets in Rooted Connectivity Augmentation

This section presents the basic properties of deficient sets, cores and Halo-families in a Rooted-Aug instance, which will be used in the analysis of our algorithm. Readers who are familiar with these properties may skip this section. Similar lemmas and proofs can be seen, e.g., in [11]. Our proofs are rather standard. The readers who are familiar with these properties may skip to the next section.

The first property is the uncrossing lemma for deficient sets of Rooted-Aug.

- **Lemma 2 (Uncrossing Properties).** Consider an instance of Rooted-Aug. Let \( G_0 \) be a rooted \( \ell \)-connected graph, and let \( A, B \) be deficient sets in \( G_0 \) that have a common terminal, i.e., \( A \cap B \cap T \neq \emptyset \). Then both \( A \cup B \) and \( A \cap B \) are deficient sets.

**Proof.** We prove the lemma by using Menger’s theorem and the submodularity of the indegree function \( \deg^{in} \). First, since \( G_0 \) is rooted \( \ell \)-connected, we know from Menger’s Theorem that \( \deg^{in}(A) = \deg^{in}(B) = \ell \). We also know that \( \deg^{in}(A \cup B) \geq \ell \) and \( \deg^{in}(A \cap B) \geq \ell \) because the root \( r \) is not contained in either \( A \) or \( B \) and that \( A \cap B \cap T \neq \emptyset \). By the submodularity of \( \deg^{in} \), it holds that

\[ 2\ell = \deg^{in}(A) + \deg^{in}(B) \geq \deg^{in}(A \cup B) + \deg^{in}(A \cap B) \geq 2\ell. \]

Therefore, \( \deg^{in}(A \cup B) = \deg^{in}(A \cap B) = \ell \), implying that both \( A \cup B \) and \( A \cap B \) are deficient sets in the Rooted-Aug instance. \hfill \Box

The next lemma gives an important property of the cores arose from deficient sets in directed graphs; that is, two cores may have non-empty intersection on Steiner vertices, but they are disjoint on terminal vertices.

- **Lemma 3 (Members of Two Halo-families are Terminal Disjoint).** Let \( C \) and \( C' \) be two distinct cores. Then, for any deficient sets \( U \in \text{Halo}(C) \) and \( U' \in \text{Halo}(C) \), it holds that \( U \cap U' \cap T = \emptyset \), i.e., any members of two distinct Halo-families have no common terminals.

**Proof.** We prove the lemma by contradiction. Let \( U \) and \( U' \) be deficient sets \( U \in \text{Halo}(C) \) and \( U' \in \text{Halo}(C) \) such that \( U \) and \( U' \) share a terminal \( t \in U \cap U' \cap T \). We may assume that \( U \) and \( U' \) are minimal such sets, i.e., there are no deficient sets \( W \in \text{Halo}(C) \) and \( W' \in \text{Halo}(C) \) such that (1) \( W \) is properly contained in \( U \), (2) \( W' \) is properly contained in \( U' \) and (3) \( t \in W \cap W' \). By Lemma 2, \( U \cap U' \) must be a deficient set properly contained in both \( U \) and \( U' \) (because \( C \neq C' \)). This contradicts the minimality of \( U \) and \( U' \). \hfill \Box
Then we combine Lemma 3 and that there is no edge joining any pair of Steiner vertices in quasi-bipartite graphs, we have the Internally Edge-Disjoint Lemma.

\textbf{Lemma 4 (Internally Edge-Disjoint).} Consider a quasi-bipartite graph \( G \). For any edge \( e \in E(G) \), there is at most one core \( C \in \mathcal{C} \) such that \( e \in E(H(C)) \).

\textbf{Proof.} Consider any edge \( uv \in E(G) \). Since \( G \) is a quasi-bipartite graph, one of \( u \) and \( v \) must be a terminal. By Lemma 3, we know that there can be at most one Halo-family \( \text{Halo}(C) \), for some \( C \in \mathcal{C} \), whose member contains both \( u \) and \( v \). Hence, the lemma follows. \( \blacksquare \)

The next lemma shows that both the union and the intersection of any two deficient sets in a Halo-family \( \text{Halo}(C) \) are also deficient sets in \( \text{Halo}(C) \). This is a crucial property for computing the halo-set \( H(C) \) as we are unable to list all the deficient sets in a Halo-family.

\textbf{Lemma 5 (Union and Intersection of Halo-Family Members).} Let \( \mathcal{F} \) be a family of all deficient sets in \( G_0 \), and let \( C \) be any core w.r.t. \( \mathcal{F} \). Then, for any two deficient sets \( A, B \in \text{Halo}(C) \), both \( A \cap B \) and \( A \cup B \) are also deficient sets in \( \text{Halo}(C) \).

\textbf{Proof.} Consider any deficient sets \( A, B \in \text{Halo}(C) \). Since both \( A \) and \( B \) contain \( C \), they share at least one terminal. Thus, Lemma 2 implies that both \( A \cup B \) and \( A \cap B \) are deficient sets. Clearly, \( A \cap B \) contains \( C \) and no other core \( C' \neq C \). Thus, \( A \cap B \) is a member of \( \text{Halo}(C) \).

Next consider \( A \cup B \). Assume for a contradiction that \( A \cup B \) is not a member of \( \text{Halo}(C) \). Then \( A \cup B \) must contain a core \( C' \neq C \). This means that at least one of the sets, say \( A \), contains some terminal \( t \in C' \). By Lemma 2, since \( A \) and \( C' \) have a common terminal, it holds that \( A \cap C' \) is a deficient set. Since \( C' \subseteq A \) (because \( A \) is a member of \( \text{Halo}(C) \)), we have that \( A \cap C' \) is a deficient set that is strictly contained in \( C' \), a contradiction. \( \blacksquare \)

It follows as a corollary that \( H(C) = \bigcup_{U \in \text{Halo}(C)} U \) is a also deficient set in \( \text{Halo}(C) \).

\textbf{Corollary 6 (Halo-set is deficient).} Let \( \mathcal{F} \) be a family of all deficient sets in \( G_0 \), and let \( C \) be any core w.r.t. \( \mathcal{F} \). Then the Halo-set \( H(C) = \bigcup_{U \in \text{Halo}(C)} U \) is also a deficient set in \( \text{Halo}(C) \).

Corollary 6 implies that \( H(C) \) can be computed in polynomial-time using an efficient maximum-flow algorithm. Such an algorithm can be seen in [11].

\textbf{Corollary 7.} For any core \( C \), its Halo-set \( H(C) = \bigcup_{U \in \text{Halo}(C)} U \) can be computed in polynomial-time.

\section{Our Algorithm and Its Overview}

This section provides the overview of our algorithm, which is based on the connectivity augmentation framework plus the Halo-set decomposition method. To be specific, our algorithm starts with an empty graph called \( H_0 = (V, \emptyset) \). Then we add edges from \( G \) to the graph \( H_0 \) to form a graph \( H_1 \) that has at least one path from the root vertex \( r \) to each terminal \( t \in T \). We keep repeating the process, which produces graphs \( H_2, \ldots, H_k \) such that each graph \( H_\ell \), for \( \ell \in [k] \), has \( \ell \) edge-disjoint \( r, t \)-paths for every terminal \( t \in S \). In each iteration \( \ell \in [k] \), we increase the rooted-connectivity of a graph by one using the Halo-set decomposition method.

We discuss the connectivity augmentation framework in Section 4.1 and discuss the algorithm based on the Halo-set decomposition method for Rooted-Aug in Section 4.2. We devote Section 5 to present a key subroutine for solving the the problem of covering Halo-families via a reduction to the Set Cover problem.
4.1 Connectivity Augmentation Framework

A straightforward analysis of the connectivity augmentation framework incurs a factor $k$ in the approximation ratio. Nevertheless, provided that the approximation algorithm for Rooted-Aug is based on the standard LP for $k$-DST, the cost incurred by this framework is only $\sum_{\ell=1}^k 1/(k - \ell + 1) = O(\log k)$. This is known as the LP-scaling technique, which has been used many times in literature; see, e.g., [30, 39, 12].

\begin{lemma}[LP-Scaling] Let $G$ be the input graph in the $k$-DST instance. Let $H^*$ be an optimal integral solution to $k$-DST (and thus LP($k$)), and let $G_0 \subseteq G$ be the initial solution subgraph of Rooted-Aug where we wish to increase the connectivity of $G_0$ from $\ell$ to $\ell + 1$ by adding edges from $E(G) - E(G_0)$. Then we can define the following LP solution $\{x_e\}_{e \in E(G)}$ to LP$_{\text{aug}}(\ell)$:

$$x_e = \begin{cases} \frac{1}{k-\ell} & \text{if } e \in E(H^*) - E(G_0) \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathcal{F}$ be the family of deficient sets in the Rooted-Aug instance. Then we know by Menger’s theorem that any deficient set $U \in \mathcal{F}$ has at least $k$ incoming edges in $H^*$, and at most $\ell$ of them are in $G_0$ (because deg$_{G_0}\{U\} = \ell$ by the definition of the deficient set). Consequently, we have

$$\sum_{e \in E_{G_0}\{U\}} x_e \geq (k - \ell) \cdot \frac{1}{k-\ell} = 1.$$ 

This means that $\{x_e\}_{e \in E(G) - E(G_0)}$ is a feasible solution to LP$_{\text{aug}}(\ell)$ whose cost is at most $(1/(k - \ell)) \text{Val}(\text{LP}(k))$. The lemma then follows by taking the summation over all $\ell = 0, 1, \ldots, k - 1$.

4.2 Algorithm for Rooted-Aug via Halo-set Decomposition

The algorithm for rooted-connectivity augmentation is built on the Halo-set Decomposition framework. In detail, we decompose vertices in the graph $G_0$ into a collection of subsets of vertices, each is defined by a Halo-family Halo($C$), which is in turn defined by its core $C$. Then we add edges to cover all the deficiencies that are contained in any of these families. However, the collection of Halo-families does not include all the deficient sets in the graph because a deficient set that contain two distinct cores are not recognized by any Halo-families. Thus, after we cover all these Halo-families (i.e., we add edges covering all its members), we need to recompute the deficient sets remaining in the graph and form the system of Halo-families again.

Following the above method, our algorithm runs in multiple iterations. In each iteration, we first compute all the cores and their corresponding Halo-set in the current solution subgraph, which can be done in polynomial time. (We recall that it is not possible to compute a Halo-family explicitly because it may contain exponential number of deficient sets.) These cores define a collection of Halo-families. Our goal is then to find a subset of edges $E'$ that covers Halo-families in this collection. To be formal, by covering a Halo-family, we mean that we find a subset of edges that covers every deficient set in its family.
our algorithm departs from the previous application of the Halo-set decomposition as we are not aiming to cover all the Halo-families. We cover only a constant fraction of Halo-families from the collection, which is sufficient for our purposes. Once we find the subset of edges $E'$, we add it to the solution subgraph and recompute the cores and their Halo-sets.

To find a set of edges $E'$, we need to compute an optimal solution to the LP for augmenting the connectivity of a graph from $\ell$ to $\ell + 1$ (i.e., $\text{LP}^{\text{aug}}(\ell)$), denoted by $\{x_e\}_{e \in E_+}$, where $E_+$ is the set of edges not in the initial solution subgraph $H_\ell$, which is $\ell$-rooted-connected. Using this LP-solution, we can find a set of edges $E'$ that covers at least $1/9$ fraction of the collection of Halo-families whose cost is at most $4 \sum_{e \in E_+} c_e x_e$ via a reduction to the Set Cover problem. This subroutine is presented in Section 5. Note that the mentioned subroutine is a randomized algorithm that has a constant success probability; thus, we may need to run the algorithm for $O(\log n)$ times to guarantee that it successes with high probability. The derandomization of our subroutine is presented in Appendix A. Our algorithm for the rooted-connectivity augmentation is presented in Algorithm 1.

**Algorithm 1** Rooted-Connectivity Augmentation.

**Require:** : An input graph $G$ and an $\ell$-rooted-connected graph $H_\ell$

**Ensure:** : An $(\ell + 1)$-rooted-connected graph $H_{\ell+1}$

1: Initialize $H_{\ell+1} := H_\ell$
2: repeat
3: Find an optimal solution $x$ to $\text{LP}^{\text{aug}}(\ell)$.
4: Compute cores and their corresponding Halo-sets in $H_{\ell+1}$.
5: Find a subset of edges $E'$ that covers at least $1/9$ fraction of the Halo-families whose cost is at most $4 \sum_{e \in E_+} c_e x_e$.
6: Update $H_{\ell+1} := H_{\ell+1} + E'$.
7: until The graph $H_{\ell+1}$ has no deficient set (and thus has no core).
8: return $H_{\ell+1}$

One may observe that the covering problem in our setting is different from that in the usual Set Cover problem as after we add edges to cover $\gamma$ fraction of the Halo-families, it is not guaranteed that the number of Halo-families will be decreased by a factor $\gamma$. This is because some of the deficient sets in the previous iterations may become new cores in the solution subgraph. Fortunately, we have a key property that any new core that was not contained in any Halo-families must contain at least two old cores. As a result, we can promise a factor $(1 - \gamma/2)$ decrease. Please see Figure 1 for illustration. The subsets $C_1$ and $C_2$ are two cores covered by $e_1$ and $e_2$, respectively. After adding two edges, $C_1$ and $C_2$ are no longer a deficient set. Now the deficient set $C_3 \supseteq C_1 \cup C_2$ becomes a new core, which contains two old cores.

**Figure 1** After adding edges $e_1$ and $e_2$ to cover $C_1$, $C_2$, a new core $C_3$ appear. The new core $C_3$ must contain at least two old cores.
Lemma 9 (The number of cores decreases by a constant factor). Let $H$ be the current solution subgraph whose number of cores is $\nu$, and let $E'$ be a set of edges that covers at least $\gamma$ fraction of the Halo-families in $H$. Then the number of cores in $H \cup E'$ is at most $(1 - \gamma/2)\nu$. In particular, the number of cores in the graph $H \cup E'$ decreases by a constant factor, provided that $\gamma$ is a constant.

**Proof.** Let us count the number of cores in the graph $H \cup E'$. Consider any core $C$ in $H \cup E'$. If $C$ is a member of some Halo-families Halo$(C')$ in $H$, then we know that Halo$(C')$ is not contained by $E'$. Thus, there can be at most $(1 - \gamma/2)\nu$ cores of this type.

Next assume, otherwise, that $C$ is not a member of any Halo-family in $H$. Then, by definition, $C$ must contain at least two cores in $H$. Notice that, for every core $C'$ in $H$ that is contained in $C$, all of the deficients in Halo$(C')$ must be covered by $E'$. Suppose not. Then there exists a deficient set $U$ in Halo($C''$) that is not covered by $E'$. Since $U$ intersects $C$ on the terminal set, Lemma 2 implies that $U \cap C$ is also a deficient set. By Lemma 3, any two cores are disjoint on the terminal set, which means that $U \cap C$ is strictly contained in $C$ (because $C$ contains another core $C''$ distinct from $C'$). The existence of $U \cap C$ contradicts the fact that $C$ is a core in $H \cup E'$. Thus, we conclude that $H \cup E'$ has at most $(1 - \gamma/2)\nu$ cores of this type.

Summing it up, the total number of cores in $H \cup E'$ is at most $(1 - \gamma/2)\nu$ as claimed.

It follows as a corollary that our algorithm terminates within $O(\log q)$ iterations.

**Corollary 10.** The number of iterations of our algorithm is at most $O(\log q)$, where $q$ is the number of terminals.

By Corollary 10, our algorithm for rooted-connectivity augmentation terminates with in $O(\log q)$, and each round, we buy a set of edges whose cost is at most $4 \sum_{x \in E^+} c_x x_c$; see Section 5. Therefore, the total cost incurred by our algorithm is at most $O(\log q)$ times the optimal LP solution, implying an LP-based $O(\log q)$-approximation algorithm as required by Lemma 8. The following lemma then follows immediately.

**Lemma 11.** Consider the problem of augmenting the rooted-connectivity of a directed graph from $t$ to $t + 1$ when an input graph is quasi-bipartite. There exists a polynomial-time algorithm that gives a feasible solution whose cost at most $O(\log q)$ that of the optimal solution to the standard LP-relaxation. In particular, there exists a polynomial-time LP-based $O(\log q)$-approximation algorithm for the problem.

**Remark.** Lastly, we remark that one may simply cover all the Halo-families in each iteration. However, the number of rounds the randomized algorithm required will be at least $O(\log q)$, meaning that the total number of iterations is $O(\log^2 q)$. Consequently, this implies that the algorithm has to pay a factor $O(\log^2 q)$ in the approximation ratio. We avoid the extra $O(\log q)$ factor by covering only a constant fraction of the Halo-families.

### 4.3 Correctness and Overall Analysis

First, to prove the feasibility of the solution subgraph, it suffices to show that the rooted-connectivity of the solution subgraph increases by at least one in each connectivity augmentation step. This simply follows by the stopping condition of the Halo-set decomposition method runs until there exists no core in the graph (and thus no deficient sets). It then follows by Menger’s theorem that the number of edge-disjoint paths from the root vertex $r$ to each terminal $t \in T$ must be increased by at least one.
Next we analyze the cost. By Lemma 11, the approximation factor incurred by Algorithm 1 is $O(\log q)$, and it also bounds the integrality gap of $LP^{aug}(\ell)$. Consequently, letting $OPT_k$ denote the cost of an optimal solution to $k$-DST, by Lemma 8, the total expected cost incurred by the algorithm is then

$$
\sum_{\ell=1}^{k} O(\log q) \cdot Val(LP^{aug}(\ell)) = O(\log q) \cdot \left( \sum_{\ell=1}^{k} \frac{1}{k - \ell + 1} \right) \cdot Val(LP(k)) = O(\log q \log k) \cdot OPT_k.
$$

This completes the proof of Theorem 1.

## 5 Covering Halo-Families via Set Cover

In this section, we present our subroutine for covering the Halo-families that arose from the Rooted-Aug problem. As mentioned in the introduction, the key ingredient in our algorithm is the reduction from the problem of covering Halo-families to the Set Cover problem. However, our instance of the Set Cover problem has an exponential number of subsets, which more resembles to an instance of the Facility Location problem. To prove our result, one route would be using Facility Location as an intermediate problem in the presentation. However, we prefer to directly apply a reduction to the Set Cover problem to avoid confusing the readers.

### 5.1 The Reduction to Set Cover and Algorithm

As an overview, our reduction follows from simple observations.

- (P1) For any minimal subset of edges that covers a Halo-family $Halo(C)$, there is only one edge $e$ that has head in $Halo(C)$ and tail outside. Let us say $e$ is outer-cover $Halo(C)$ since it is coming from the outside of the family.

- (P2) Any edge can be contained in at most one $Halo(C)$. i.e., there is at most one halo-family $Halo(C)$ such that both head and tail of $e$ are contained in $H(C)$. (From Lemma 4.)

- (P3) An LP for covering a single Halo-family is integral.

We remark that while Properties (P1) and (P3) hold in general instances of $k$-DST, Property (P2) holds only in quasi-bipartite graphs.

Now an instance of the Set Cover problem can be easily deduced. We define each Halo-family $Halo(C)$ as an element, and we define each edge $e$ as a subset. However, we may have multiple subsets corresponding to the same edge $e$ as it may serve as an “outer-cover” for many Halo-families. Thus, we need to enumerate all the possible collections of Halo-families that are outer-covered by $e$. We avoid getting exponential number of subsets by using the solution from an LP (for the connectivity augmentation problem) as a guideline.

Before proceeding, we need to formally define some terminologies. Let $\tilde{G}$ be the current solution subgraph. We say that an edge $e$ outer-covers a Halo-family $Halo(C)$ if the head of $e$ is in $H(C)$ and the tail is not in $H(C)$ and that there exists a subset of edges $E' \subseteq E_+ - E(\tilde{G})$ such that (1) both endpoints of every edge in $E'$ are contained in $H(C)$ and (2) the set of edges $E' \cup \{e\}$ covers $Halo(C)$.

For each Halo-family $Halo(C)$, we define the set of edges $I_C$ to be the minimum-cost subset of edges $E' \subseteq E_+ - E(\tilde{G})$ whose both endpoints are in $H(C)$ and that $E' \cup \{e\}$ covers $Halo(C)$, and we denote the cost of $I_C$ by $\sigma_C$. We may think that $\sigma_C$ is the cost for covering...
Halo$(C)$ given that $c$ has been taken for free. We use the notation $E[C]$ to mean the set of edges whose both endpoints are contained in the Halo-set $H(C)$. We denote the cost of the fractional solution restricted to $E[C]$ by $\text{cost}_x(E(C)) = \sum_{e \in E[C]} c_e x_e$.

Our reduction is as follows. Let $H$ be the current solution subgraph. For each core $C$ in $H$, we define an element $C$. For each edge $e \in E_+ - E(H)$, we define a subset $S_e$ by adding to $S_e$ an element $C$ if $\sigma_C^e \leq \text{cost}_x(E[C])$. This completes a reduction. Notice that the resulting instance of the Set Cover problem has polynomial size. To show that our reduction runs in polynomial-time, we need to give a polynomial-time algorithm for computing $\sigma_C^e$, which we defer to Section 5.4. Here we leave a forward reference to Lemma 15. Our algorithm covers a constant fraction of the collection of Halo-families by simply picking each edge $e$ with probability $x_e$ and add all the edges $I_C^e$, for all cores $C \in S_e$, to the solution subgraph; if a core $C$ is outer-covered by two picked edges, then we add only one edge-set $I_C^e$. We claim that the set of edges chosen by our algorithm covers at least $1/9$ fraction of the Halo-families, while paying a cost of at most four times the optimum (with a constant probability). In particular, we prove the following lemma.

\begin{lemma} \label{lemma:partial_covering}
With constant probability, the above algorithm covers at least $1/9$ fraction of the collection of Halo-families, and the cost of the of the edges chosen by the algorithm has cost at most $4 \sum_{e \in E_+} c_e x_e$. In particular, the algorithm partially covers the collection of the Halo-families, while paying the cost of at most constant times the optimum.
\end{lemma}

To prove Lemma 12, we need to show that the fractional solution defined by \( \{x_e\}_{e \in E_+} \) is (almost) feasible to the Set Cover instance, which then implies that the set of edges we bought covers a constant fraction of the Halo-families with probability at least $2/3$. Then we will show that the cost of the fractional solution to the Set Cover instance is at most twice that of the optimal solution to LP\textsuperscript{avg}(\ell), thus implying that we pay at most six times the optimum with probability $2/3$.

To be more precise, we show in Section 5.2 that our algorithm covers at least $1/3$ fraction of the Halo-families in expectation, meaning that we cover less than $1/9$ fraction with probability at most $1/3$. Then we show in Section 5.3 that the expected cost incurred by our algorithm is $2 \sum_{e \in E_+} c_e x_e$, thus implying that we pay more than six times that of the LP with probability at most $1/3$. Applying the union bound, we conclude that our algorithm covers at least $1/9$ fraction of the Halo-families, while paying the cost of at most six times the optimal LP solution with probability at least $1/3$. (Note that in Section 5.3, we show a slightly stronger statement that the cost incurred by our algorithm is $4 \sum_{e \in E_+} c_e x_e$ with probability at least $2/3$.) To finish our proof, we proceed to prove the above two claims and then prove the structural properties used in the forward references.

### 5.2 Partial Covering

We show in this section that our algorithm covers at least $1/9$ fraction of the Halo-families with probability at least $1/3$.

First, we show that the LP variable defined by $x_e$ is almost feasible to the LP-relaxation of the Set Cover problem. We note that our proof will need a forward reference to Lemma 14.

\begin{lemma} \label{lemma:lp_feasibility}
The LP variable \( \{y_e\}_{e \in E_+} \), where $y_e = \min\{1, 2x_e\}$ for all edges $e \in E_+$ is feasible to the Set Cover instance. That is, for any core $C$ in the graph,
\[
\sum_{e \in E_+, \sigma_C^e \in S_e} x_e \geq 1/2.
\]
\end{lemma}
Proof. Consider a core $C$, which corresponds to an element in the Set Cover instance. We take the set of edges incident to its Halo-set $H(C)$, and find a minimal vectors $\{x'_e\}_{e \in E_+}$ such that $\{x'_e\}_{e \in E_+}$ fractionally covers the Halo-family Halo($C$) and $x'_e \leq x_e$ for all edges $e \in E_+$. (Note that by minimality we mean that, for any edge $e$ and any $\epsilon > 0$, decreasing the value of $x'_e$ by $\epsilon$ results in an infeasible solution.) By Lemma 14, we have $\sum_{e \in \delta^\min(H(C))} x'_e = 1$, i.e., the total weight of the LP value of edges incoming to $H(C)$ is exactly one.

Next consider the following LP.

$$\text{LP}^{\text{halo}} = \left\{ \begin{array}{ll}
\min & \sum_{e' \in E_+(H(C))} c_{e'} x_{e'} \\
\text{s.t.} & \sum_{e' \in \delta^\min(H(C))} x_{e'} \geq 1 \quad \forall U \in \text{Halo}(C) \\
& 0 \leq x_{e'} \leq 1 \quad \forall e' \in E_+(H(C))
\end{array} \right.$$

By Lemma 2, we know that both the intersection and union of any two deficient sets in Halo($C$) are also deficient sets in Halo($C$). This means that the Halo-family Halo($C$) is an intersecting family. It then follows from the result of Frank [21] that the above LP is Totally Dual Integral, which means that any convex point of its polytope is an integral solution (including the optimal one). Since $\{x'_e\}_{e \in E_+}$ is a feasible solution to $\text{LP}^{\text{halo}}$, it can be written as a convex combination of integral vectors in the polytope, i.e.,

$$x = \sum_{i=1}^w \lambda_i z^i, \text{ where } \sum_{i=1}^w \lambda_i = 1.$$

Let $F_i$ be the set of edges induced by each integral vector $z^i$ (i.e., $F_i$ is the support of $z^i$). Since the LP requires $H(C)$ to have at least one incoming edge, we deduce that, for each $F_i$, there exists one edge $e_i \in F_i$ entering $H(C)$.

Now we compare the cost of $\sigma_C^{\text{glob}}$ to the cost of $F_i - \{e_i\}$. By minimality of $\sigma_C^{\text{glob}}$, we know that $\sigma_C^{\text{glob}} \leq \text{cost}(F_i - \{e_i\})$ for all $i = 1, \ldots, w$. We recall that we add a core $C$ to the set $S_e$, only if $\sigma_C^{\text{glob}} \leq \text{cost}_x(E[C])$. Since $\text{cost}_x(E[C])$ is the convex combination of $Z^i$, at least half of the $F_i$ (w.r.t. to the weight $\lambda_i$) must have $\sigma_C^{\text{glob}} \leq \text{cost}(F_i - \{e_i\}) \leq \text{cost}_x(E[C])$; that is, $\sum_{i: \sigma_C^{\text{glob}} \leq \text{cost}(F_i - \{e_i\})} \lambda_i \geq 1/2$. Therefore, we conclude that the sum of $y_{e_i}$ over all $e_i$ such that $\sigma_C^{\text{glob}} \leq \text{cost}_x(E[C])$ is at least one, thus proving the lemma.

We remark that we may define the Set Cover instance so that $\{x_e\}_{e \in E_+}$ is exactly a feasible solution to the LP for the Set Cover problem by using the integer decomposition as in the proof of Lemma 13. However, we choose to present it this way to keep the reduction simple.

Now we finish the proof of our claim. Consider a core $C$. Note that by construction, every time we pick an edge $e$, we also add the set of edges $I_C^e$, for each $C \in S_e$ (recall that $I_C^e \cup \{e\}$ covers Halo($C$)). Thus, the probability that the algorithm picks no edges $e$ such that $C \in S_e$ is

$$\Pi_{e \in E_+, C \in S_e} (1 - x_e) \leq \exp \left( - \sum_{e \in E_+} x_e \right) \leq \exp(-1/2) \leq \frac{2}{3}.$$

The first inequality follows because $1 - x \leq \exp(-x)$, for $0 < x \leq 1$. That is, the probability that the algorithm does not cover a core $C$ is at most $2/3$, which means that the expected fraction of Halo-families covered by our algorithm is at least $1/3$. Applying Markov’s inequality, we conclude that with probability at least $2/3$ our algorithms covers at least $1/9$ fraction of the Halo-families. Our algorithm can be derandomized using the method of conditional expectation. Please see Appendix A for details.
5.3 Cost Analysis

Now we analyze the expected cost of the edges we add to the solution subgraph. We classify the cost incurred by our algorithm into two categories. The first case is the set of edges \( e \) that we pick with probability \( x_e \). The expected cost of this case is \( \sum_{e \in E_e} c_e x_e \). Applying Markov’s inequality, we have that with probability at least 2/3 the cost incurred by the edges of this case is at most \( 3 \sum_{e \in E_e} c_e x_e \).

The second case is the set of edges corresponding to each subset \( S_e \) whose edge \( e \) is added to the solution. By construction, a core \( C \) is added to \( S_e \) only if \( \text{cost}_x(E[C]) > \sigma^C_x \) (i.e., the cost of the set of edges \( I^C_x \)). We recall that we also add one set of edges \( I^C_x \) to the solution if there are more than one edges \( e \) such that \( C \in S_e \) are chosen. As the set of edges \( E[C] \) and \( E[C'] \) are disjoint for any two cores \( C \neq C' \) (please see the forward reference to Lemma 4), we conclude that the cost incurred by the edges of this case is at most \( \sum_{e \in E_e} c_e x_e \) (regardless of the choices of the edges randomly picked in the previous step). Therefore, with probability at least 2/3 the cost of edges chosen by our algorithm is at most \( 4 \sum_{e \in E_e} c_e x_e \).

5.4 Structural Properties of the LP solution

We devote this last subsection to prove properties (P1) to (P3) and all the forward references as discussed earlier. Property (P3) simply follows from the fact that the intersection and union of any two members of a Halo-family Halo\((C)\) are also members of Halo\((C)\), which means that the polytope of the problem of covering Halo\((C)\) is integral due to the result of Frank [21]. Thus, we are left to prove the property (P1) and (P2) and to present a polynomial-time algorithm for computing \( \sigma^C_x \), which thus complete the proof that our reduction can be done in polynomial time.

First, we prove Property (P1), which allows us to reduce the instance of the problem of covering Halo-families to a Set Cover instance.

Lemma 14 (Unique Entering Edge in Minimal Cover). Consider a minimal fractional cover \( x \) of a Halo-family Halo\((C)\). That is, \( x \) is a feasible solution to LP\(_\text{halo} \) whose collection of deficient sets is defined by Halo\((C)\), and decreasing the value \( x_e \) of any edge \( e \in E_e \) results in an infeasible solution. It holds that \( \sum_{e \in \delta^\text{in}_{E_e}(H(C))} x_e = 1 \). Thus, for an integral solution \( E' \), there is exactly one edge \( e \in E' \) entering the Halo-set Halo\((C)\).

Proof. Assume for a contradiction that \( \sum_{e \in \delta^\text{in}_{E_e}(H(C))} x_e > 1 \). By the minimality of \( x \), for any edge \( e \in \delta^\text{in}_{E_e}(H(C)) \), there exists a deficient set \( W_e \in \text{Halo}(C) \) such that \( \sum_{e \in \delta^\text{in}_{E_e}(W_e)} x_e = 1 \). We choose \( W_e \) to be the maximum inclusionwise such set and call it the witness set of \( e \).

Now we take two distinct witness sets \( W_e \) and \( W_{e'} \), for \( e \neq e' \). By Lemma 5, both \( W_e \cap W_{e'} \) and \( W_e \cup W_{e'} \) are deficient sets in Halo\((C)\). Let us abuse the notation of \( x \). For any subset of vertices \( S \subseteq V(G) \), let \( x(S) = \sum_{e \in \delta^\text{in}_{E_e}(S)} x_e \). The function \( x(S) \) is known to be submodular [24], meaning that

\[
2 = x(W_e) + x(W_{e'}) \geq x(W_e \cap W_{e'}) + x(W_e \cup W_{e'}) \geq 2.
\]

The last inequality follows because \( \{x\}_{e \in E_e} \) fractionally covers Halo\((C)\), which then implies that \( x(W_e \cap W_{e'}) = x(W_e \cup W_{e'}) = 1 \). But, this contradicts the choice of \( W_e \) (and also \( W_{e'} \)) because \( W_e \cup W_{e'} \) is a deficient set in Halo\((C)\) strictly containing \( W_e \) in which the conditions \( x(W_e \cup W_{e'}) = 1 \) and \( e \in \delta^\text{in}_{E_e}(W_e \cup W_{e'}) \) hold. ▶
Next we prove Property (P2), which allows us to upper bound the cost incurred by the main algorithm.

Finally, we show that $\sigma_C^e$ can be computed in polynomial time.

**Lemma 15.** For any core $C \in C$ and an edge $e \in E(G)$, the set of edges $I_C^e$ and, thus, its cost $\sigma_C^e$ can be computed in polynomial time. Moreover, the value of $\sigma_C^e$ is equal to the optimal value of the corresponding covering LP given below.

$$\text{LP}^{\text{cover}} = \min \left\{ \sum_{e' \in E_+} x_{e'} : \sum_{e' \in \delta_H^e(U)} x_{e'} \geq 1 \quad \forall U \in \text{Halo}(C) \\
0 \leq x_{e'} \leq 1 \quad \forall e' \in E_+ \left( H(C) \right) \\
x_e = 1 \right\}$$

**Proof.** Consider the Halo-family $\text{Halo}(C)$. By Lemma 5, the union and intersection of any deficient sets $U, W \in \text{Halo}(C)$ are also deficient sets in $\text{Halo}(C)$. This means that $\text{Halo}(C)$ is an **intersecting family**. It is known that the standard LP for covering an intersecting family is integral (see, e.g., [21]), which implies that we can compute $\sigma_C^e$ and its corresponding set of edges $I_C^e$ in polynomial time by solving $\text{LP}^{\text{cover}}$.

Alternatively, we may compute $\sigma_C^e$ combinatorially using an efficient minimum-cost $(\ell + 1)$-flow algorithm. In particular, we construct an $s^*, t^*$-flow network by setting the costs of edges in $\delta_H^e(H(C)) \cup \{e\}$ to zero, adding a source $s^*$ connecting to $\ell + 1$ edges entering $\text{Halo}(C)$ (which consists of $\ell$ edges from $\delta_H^e(H(C))$ plus the edge $e$) and then picking an arbitrary terminal $t^* \in C$ as a sink. All the edges not in $E(H(C))$ except $\delta_H^e(H(C)) \cup \{e\}$ are removed. Applying Manger’s theorem, it can be seen that every $(\ell + 1)$-flow in this $s^*, t^*$-flow network corresponds to a feasible solution to the covering problem with the same cost. This gives a polynomial-time algorithm for computing $\sigma_C^e$ and $I_C^e$ as desired. ▶

6 Conclusion and Open Problems

We have presented our $O(\log q \log k)$-approximation algorithm for $k$-DST when an input graph is quasi-bipartite. This is the first polylogarithmic approximation algorithm for $k$-DST for arbitrary $k$ that does not require an additional assumption on the structure of the optimal solution. In addition, our result implies that $k$-DST in quasi-bipartite graphs is equivalent to the Set Cover problem when $k = O(1)$.

Lastly, we conclude our paper with some open problems. A straightforward question is whether there exists a non-trivial approximation algorithm for $k$-DST for $k \geq 3$ in general case or for a larger class of graphs (perhaps, in quasi-polynomial-time). Another interesting question is whether our randomized rounding technique, which consists of dependent rounds of a randomized rounding algorithm for the Set Cover problem, can be applied without connectivity augmentation. If this is possible, it will give $O(\log k)$ improvements upon the approximation ratios for approximating many problems whose the best known algorithms are based on the Halo-Set decomposition technique.

References


Polylogarithmic Approximation for $k$-DST on Quasi-Bipartite Graphs


Polylogarithmic Approximation for $k$-DST on Quasi-Bipartite Graphs


A Derandomization

In this section, we present a derandomization of our algorithm in Section 5 using the method of conditional expectation [1]. We will mostly follow the proof presented in the work of Bertsimas and Vohra [4] who gave a derandomized technique for the randomized scheme for the Set Cover problem.

In more detail, first observe that the cost incurred by our algorithm comes from two parts. The first part is the cost of edges $e$ that we pick with probability $x_e$, and the second part is the cost of edges $I^e_C$ in which the edge $e$ is chosen. For the second part, our algorithm guarantees that, for each core $C$, only one set of edges $I^e_C$ will be added to the solution. Thus, by the construction of $S_e$ and Lemma 4, the cost incurred by this part is $\sum_{e \in E_+} c_e x_e$ regardless of the choices of the edges $e$ added to the solution from the first part.

Hence, it suffices to show that there exists a deterministic algorithm that picks a set of edges $E'$ that outer-covers at least $1/3$ fraction of the Halo-families, while paying the cost at most $\sum_{e \in E_+} c_e x_e$.

Let $C$ be the collection of all the cores in the current solution subgraph. For a given set of edges $E' \subseteq E_+$, we define a function $\tau_C \in \{0, 1\}$ for each Halo-family Halo($C$) to indicate whether Halo($C$) is covered by some edge in $E'$, and we define a function $I(E')$ to indicate whether $E'$ outer-covers at least $1/9$ fraction of the Halo-families. The formal definition of these two functions are given below.

$$\tau_C(E') = \begin{cases} 1 & \text{if } E' \text{ outer-covers Halo}(C) \\ 0 & \text{Otherwise} \end{cases}$$

$$I(E') = \begin{cases} \sum_{C \in C} \tau_C(E') < \frac{|C|}{9} \\ 0 & \text{Otherwise} \end{cases}$$

Next we define the potential function:

$$\Phi(E') = \sum_{e \in E'} c_e + M \cdot I(E'), \text{ where } M = 3 \sum_{e \in E_+} c_e x_e.$$
Observe that $\Phi(E') \leq M$ if $E'$ outer-covers at least $1/9$ fraction of the Halo-families, while having the cost at most three times that of the LP solution; otherwise, $\Phi(X) > M$. Notice that, by Lemma 12, if we add each edge $e \in E_+$ to $E'$ with probability $x_e$, then $\mathbb{E}[\Phi(E')] \leq M$. Thus, there exists an event that $\Phi(X) \leq M$, which will give us the desired integer solution.

We then follow the method of conditional expectation (see, e.g., [1]). That is, we order edges in $E_+$ in an arbitrary order, say $e_1, e_2, \ldots, e_{|E_+|}$. Let $E''$ be the set of edges that we try to simulate the set of randomly chosen edges $E'$. Initially, $E_{det} = \emptyset$. Then we decide to add each edge $e_i$, for $i = 1, 2, \ldots, |E_+|$ to $E'$ if $\mathbb{E}[\Phi(E')|E_{det} \cup \{e_i\} \subseteq E''] \leq \mathbb{E}[\Phi(E')|E_{det} \subseteq E'']$. This way the resulting set of edges $E_{det}$ outer-covers at least $1/9$ fraction of the Halo-families, while having the cost at most $3 \sum e \in E_+ c_e x_e$. Therefore, after adding the set of edges $I^C_e$ for each core outer-covered by some edge $e \in E_{det}$, we have a set of edges that covers at least $1/9$ fraction of the Halo-families with cost at most $4 \sum e \in E_+ c_e x_e$, i.e., with the same guarantee as desired in Lemma 12.

## B Bad Example for Grandoni-Laekhanukit Tree-Embedding Approach

In [32], Grandoni and Laekhanukit proposed an approximation scheme for $k$-DST based on the decomposition of an optimal solution into $k$ divergent arborescences [28, 3]. Their approach results in the first non-trivial approximation algorithm for 2-DST, and the algorithm achieves polylogarithmic approximation ratio in quasi-polynomial-time. Nevertheless, this technique meets a barrier as soon as $k \geq 3$ as it was shown in [3] that the decomposition of an optimal solution into $k$ divergent arborescences does not exist for general graphs when $k \geq 3$. One would hope that the decomposition is still possible for some classes of graphs, e.g., quasi-bipartite graphs. We show that, unfortunately, even for the class of quasi-bipartite graphs the divergent arborescences decomposition does not exist for $k \geq 3$. The counter example of a 3-rooted-connected graph that has no 3 divergent arborescences is shown in Figure 2.

---

**Figure 2** This figure shows an example 3-rooted-connected quasi-bipartite graph that cannot be decomposed into 3 divergent arborescences.