Improved Approximation Algorithms for Directed Steiner Forest

Moran Feldman[*] Guy Kortsarz[†] Zeev Nutov[‡]

Abstract

We consider the k-Directed Steiner Forest (k-DSF) problem: Given a directed graph $G = (V, E)$ with edge costs, a collection $\mathcal{D} \subseteq V \times V$ of ordered node pairs, and an integer $k \leq |D|$, find a minimum cost subgraph H of G that contains an st-path for (at least) k pairs $(s, t) \in \mathcal{D}$. When $k = |\mathcal{D}|$, we get the Directed Steiner Forest (DSF) problem. The best known approximation ratios for these problems are: $\tilde{O}(k^{2/3})$ for k-DSF by Charikar et al. [6], and $O(k^{1/2+\varepsilon})$ for DSF by Chekuri et al. [7].

Our main result is achieving the first sub-linear in terms of $n = |V|$ approximation ratio for DSF. Specifically, we give an $O(n^{\varepsilon} \cdot \min\{n^{4/5}, m^{2/3}\})$-approximation scheme for DSF.

For k-DSF we give a simple greedy $O(k^{1/2+\varepsilon})$-approximation algorithm. This improves upon the best known ratio $\tilde{O}(k^{2/3})$ by Charikar et al. [6], and (almost) matches, in terms of k, the best ratio known for the undirected variant [18]. This algorithm uses a new structure called *start-junction tree* which may be of independent interest.

[*]Technion. `moranfe@cs.technion.ac.il`. Part of this work was done as a part of author's M.Sc. Thesis at The Open University of Israel.

[†]Rutgers University, Camden. `guyk@camden.rutgers.edu`. Partially supported by NSF support grant award number 0829959

[‡]The Open University of Israel. `nutov@openu.ac.il`

1 Introduction

Network design problems seek to find a minimum cost subgraph of a given (directed or undirected) graph, that satisfies some prescribed properties, often connectivity requirements. These problems are among the most studied problems in the fields of Combinatorial Optimization and Approximation Algorithms. We hereby list some classic network design problems on *undirected* graphs. One of the most basic network design problems is the Steiner Tree problem: given a graph $G = (V, E)$ with edge costs, and a set $T \subseteq V$ of terminals, find a minimum cost subtree of G that spans T. This classic NP-hard problem was extensively studied with respect to approximation (see [34] and the references therein). A classic generalization is the Steiner Forest problem: Given a graph $G = (V, E)$ with edge costs and a collection $D \subseteq V \times V$ of (unordered) node pairs, find a minimum cost subgraph H of G that connects all pairs in $\mathcal{D}$ (namely, contains an st-path for every $\{s, t\} \in \mathcal{D}$). The best approximation ratio known for Steiner Forest is 2 [1] (see also [17] for a more general algorithm and a simpler proof). In the more general k-Steiner Forest problem, we are also given an integer $k \leq |\mathcal{D}|$, and the goal is to connect at least k (arbitrary) pairs from D. Here a significant obstacle lies in the way of achieving a good (e.g. polylogarithmic) approximation ratio, even for undirected graphs. It was observed in [19] that k-Steiner Forest is harder than a variant of the Densest k-Subgraph problem, which is commonly believed not to admit a polylogarithmic approximation. The best ratio known for the Densest k-Subgraph problem is $O(n^{1/4+\varepsilon})$ in a recent paper by Bhaskara et al. [3]; see also a slightly weaker polynomial ratio result by [13]. Thus even for undirected k-Steiner Forest we probably could expect only polynomial ratios. The best known approximation ratio for k-Steiner Forest is $O(\min\{\sqrt{n}, \sqrt{k}\})$; see a recent paper by Gupta et al. [18]. In [18] the k-Steiner Forest problem is shown to have further significance due to its relation to the Dial a Ride problem.

In this paper we consider the *directed* variant of the k-Steiner Forest problem, namely:

k-Directed Steiner Forest (k-DSF)
Instance: A directed graph $G = (V, E)$, edge costs $\{c(e) : e \in E\}$, a set $\mathcal{D} \subseteq V \times V$ of ordered pairs, and an integer $k \leq |\mathcal{D}|$.
Objective: Find a min-cost subgraph H of G that contains an st-path for (at least) k pairs $(s, t) \in \mathcal{D}$.

When $k = |\mathcal{D}|$ we get the Directed Steiner Forest (DSF) problem. Another particular case of k-DSF is the k-Directed Steiner Tree (k-DST) problem, where $\mathcal{D} = \{s\} \times T$ for some $s \in V$ and a terminal set $T \subseteq V - \{s\}$.

Remark: The name "Directed Steiner Forest" is used to relate the problem to the undirected version. In the undirected version, any minimal feasible solution is a forest, but in the directed case, the structure of a solution may be more complicated (e.g., it may contain cycles). For example, if all costs are 1 and $\mathcal{D} = V \times V$, then a directed Hamiltonian cycle is the best solution one can expect.

1.1 Directed and undirected Steiner problems

Usually, directed variants of network design problems are much harder to approximate than the undirected ones (we shall later see that for k-DSF this is not the case). For example, while the undirected Steiner Tree and the k-Steiner Tree problems both admit a constant approximation ratio (see [35, 15]), even a very special case of DST – the Group Steiner Tree problem on trees, is unlikely to admit a

$\log^{2-\varepsilon} n$ ratio for any $\varepsilon > 0$ [20]. In fact, the best known ratio for DST is much worse than its proved lower bound. Extending and simplifying the recursive greedy method introduced by Zelikovsky [36] and Kortsarz and Peleg [29], Charikar et al. [6] gave a combinatorial $O(\ell^3 k^{2/\ell})$-approximation algorithm for k-DST that runs in $O(k^{2\ell} n^\ell)$ time (where $k = |T|$). Substituting $\ell = 2/\varepsilon$ gives an $O(k^\varepsilon)$-approximation scheme, namely, an $O(k^\varepsilon/\varepsilon^3)$-approximation algorithm that runs in $O(k^{4/\varepsilon} n^{2/\varepsilon})$ time for any fixed $\varepsilon > 0$. Substituting $\ell = \log k$ gives an $O(\log^3 k)$-approximation in quasi-polynomial time.

For the Steiner Forest problem, the gap between the directed and undirected variants is even wider. The problem admits a constant approximation for undirected graphs [1, 17]. However, for DSF strong lower bounds are known [11]. Dodis and Khanna [11] showed that DSF is at least as hard as the LABEL-COVER$_{\max}$ problem [33]. This implies that DSF cannot be approximated within $O(2^{\log^{1-\varepsilon} n})$ for any fixed $\varepsilon > 0$, unless NP-hard problems can be solved in quasi-polynomial time [33].

Perhaps the most extreme example of the difference between undirected and directed network design problems is the Steiner Network problem. In this problem each pair $(s,t) \in \mathcal{D}$ has a connectivity requirement $r(s,t)$, namely, $r(s,t)$ edge disjoint st-paths are required for every $(s,t) \in \mathcal{D}$. On undirected graphs, this problem admits a 2-approximation algorithm due to Jain [23]. As far as we know no non-trivial ratio is known for the Steiner Network problem on directed graphs.

Table 1 summarizes the best known approximation ratios for Steiner Network problems, prior to our work. For other related problems, including the closely related Group Steiner Tree problem see [8, 16, 25, 20, 22, 14, 28] and surveys in [12] and [24, 27].

Problem	**Undirected**		**Directed**	
	In terms of n	*In terms of k*	*In terms of n*	*In terms of k*
Steiner Tree	1.39 [4]	1.39 [4]	$O(n^\varepsilon)$ [6]	$O(k^\varepsilon)$ [6]
k-Steiner Tree	2 [15]	2 [15]	$O(n^\varepsilon)$ [6]	$O(k^\varepsilon)$ [6]
Steiner Forest	2 [1]	2 [1]	$O(n^{1+\varepsilon})$	$O(k^{1/2+\varepsilon})$ [7]
k-Steiner Forest	$O(\sqrt{n})$ [18]	$O(\sqrt{k})$ [18]	$\tilde{O}(n^{4/3})$ [6]	$\tilde{O}(k^{2/3})$ [6]
Steiner Network	2 [23]	2 [23]	n^2	k

Table 1: Best known approximation ratios for Steiner Network problems, prior to our work. Both ratios are given in terms of n and in terms of k when relevant.

1.2 Our results

1. Our main result is breaking the $\Omega(n)$ ratio for Steiner Forest on directed graphs. Our approximation ratio is $O(n^{4/5+\varepsilon})$ for any constant ε. This is the first sub-linear ratio for any special case of Directed Steiner Network, except for the Directed Steiner Tree problem. Previously the best known ratio was $O(n^{1+\varepsilon})$, which can be easily derived from the algorithm of [6] for DST.

Theorem 1.1 DSF *admits an $O(n^\varepsilon \cdot \min\{n^{4/5}, m^{2/3}\})$-approximation scheme (that runs in polynomial time for any $\varepsilon > 0$).*

2. For k-DSF, the previous best known result was by a classic paper of Charikar et al. [6] that gave an approximation algorithm with ratio $\tilde{O}(k^{2/3})$, which in terms of n is $\tilde{O}(n^{4/3})$ if $k = \Theta(n^2)$. We improve this result to $O(k^{1/2+\varepsilon})$ ratio.

   Theorem 1.2 *k-DSF admits an $O(k^{1/2+\varepsilon})$-approximation scheme.*

   We remark on the relation of our paper to [7]. Our $O(n^\varepsilon \cdot \min\{n^{4/5}, m^{2/3}\})$-approximation scheme for DSF uses the main result of [7] as a black box, combined with new ideas. Our $O(k^{1/2+\varepsilon})$-approximation for DSF uses new techniques unrelated to [7].

   A possibly interesting feature of the state of the art of the k-Steiner Forest problem is that the approximation ratios known for the directed and undirected cases are not that different in terms of k: $O(\sqrt{k})$ for undirected graphs [18] versus $O(k^{1/2+\varepsilon})$ in our paper. However, in terms of n, the difference $O(\sqrt{n})$ versus $O(n^{4/5+\varepsilon})$ is still quite large.

3. A *set pair* is an ordered pair (S, T) of disjoint nonempty subsets of V. Chekuri et al. [7] gave an $O(\log^2 n \log^2 |\mathcal{D}|)$-approximation algorithm for the following generalization of Group Steiner Tree:

   Group Steiner Forest (GSF)
   Instance: An (undirected) graph $G = (V, E)$, edge costs $\{c(e) : e \in E\}$, and a set $\mathcal{D}$ of set pairs in V.
   Objective: Find a min-cost subgraph H of G such that for every set pair $(S, T) \in \mathcal{D}$, H contains an st-path for some $s \in S, t \in T$.

   In the more general k-Group Steiner Forest (k-GSF) problem, we are also given an integer $k \leq |\mathcal{D}|$, and it is only required that for (at least) k set pairs $(S, T) \in \mathcal{D}$, H contains an st-path for some $s \in S, t \in T$. Note that k-GSF also generalizes the (undirected) k-Steiner Forest problem, which is the particular case when every set pair in $\mathcal{D}$ is just a pair of nodes. The polylogarithmic approximation of [7] does not extend to k-GSF. Furthermore, k-GSF is unlikely to admit a polylogarithmic approximation ratio; otherwise, we would obtain a polylogarithmic approximation for (undirected) k-Steiner Forest. Recall that the best known ratio for the latter is $O(\min\{\sqrt{n}, \sqrt{k}\})$, and that a polylogarithmic ratio for it implies a polylogarithmic ratio for the Densest k-Subgraph problem [18]. k-GSF admits an easy (up to constants) approximation ratio preserving reduction to k-DSF. Thus by Theorem 1.2 we obtain the following extension of the $O(\sqrt{k})$-approximation for the (undirected) k-Steiner Forest problem:

   Corollary 1.3 *k-GSF admits an $O(k^{1/2+\varepsilon})$-approximation scheme.*

   In fact, our results can be used to show that if the k-Group Steiner Tree problem admits an α-approximation algorithm, then k-GSF admits an $O(\alpha\sqrt{k})$-approximation algorithm, implying an $\tilde{O}(\sqrt{k})$-approximation [16].

Remark: Antonakopoulos [2], used our results, with additional ideas to get the first approximation algorithm for the Directed Buy-At-Bulk problem (with non uniform subadditive functions over the edges). His ratio is $O(\min\{k^{1/2+\varepsilon}, n^{4/5+\varepsilon}\})$. For more details on the definition of this problem see [2].

1.3 Main new techniques

1.3.1 The new techniques used in the $O(n^{4/5+\varepsilon})$ ratio algorithm for DSF

Intuitively, a pair $st \in D$ is "good" if there are "many" nodes r so that a "cheap" st-path via r exists; otherwise, the pair is "bad". There are three main procedures in our sub-linear algorithm for DSF:

1. The first procedure uses randomization to find a relatively small "junction subset" $R \subset V$ through which all good pairs can be connected. As R is small, and the paths are cheap, we can show that the cost incurred in connecting all good pairs via R is $\tilde{O}(n^{4/5}) \cdot \mathsf{opt}$. After all good pairs are connected, they are excluded from $\mathcal{D}$, and we remain with bad pairs only.

2. If in some optimal solution at least half of bad pairs are connected by "costly path" then we prove, by standard averaging, the existence of a low density junction-tree (as defined in [7]). A sub-graph with density close to the density of such a tree is found using the procedure of [7].

3. The difficult case is when the optimal solution connects most of the bad pairs by "cheap" paths. To handle this case, we formulate a novel LP-relaxation which asks to connect pairs by cheap paths only. This LP assigns capacity x_e to every edge e so that it will be possible to send a unit of s_it_i-flow (separately for every i) along cheap paths, and so that $\sum_{e \in E} c(e)x_e$ is minimized. We show how to find approximate solutions for this LP in polynomial time, and that rounding up entries x_e of large enough value gives a low density sub-graph.

Similarly, we can also think of a "good pair" as a pair of nodes such that many edges are involved in "cheap" paths connecting them. This idea leads to an approximation guarantee in terms of m.

1.3.2 Junction star-trees and their advantages

The algorithm of [6] for k-DST accumulates *low density* directed trees until enough terminals are connected to the root. The density of a tree is its cost over the number of new terminals it connects. The idea of low-density *junction trees* was first invented for approximating Non-Uniform Multicommodity Buy-at-Bulk problems [9], where it was applied to undirected graphs.

In the approximation of the of Non-uniform Multicommodity Buy-at-Bulk, on undirected graphs, studied in [9], a junction tree is defined as a collection of paths, all going via the same node, and sending a (possibly fractional) unit of s_it_i-flow for "many" s_i, t_i pairs at a "low" cost. The reason this definition suffices is that there are known methods to round such fractional solutions into trees of low cost with only polylogarithmic loss compared to the fractional value (see [31, 10]).

For problems on *directed* graphs, it does not seem that we can easily use fractional flow methods to achieve a low ratio approximation algorithm. Even for DST, it is not clear if it is possible to round a fractional flow solution into a low cost out-branching. The only tool we have for tasks of finding low cost out- and in-branchings is the recursive greedy algorithm of [6]. Hence, in the directed setting a more careful definition of junction tree is used in [7].

Definition 1.1 *The* density *of a (directed) graph J w.r.t. a set $\mathcal{D}$ of ordered node pairs is its cost over the number of pairs from $\mathcal{D}$ it connects. A graph J is called a* junction tree *if it is the union of* *an in-going tree and an outgoing tree (not necessarily edge disjoint), both rooted at the same node r.*

We present a central idea of [7]. We may assume that no single s_it_i-path has low density (namely, no s_it_i admits an s_i to t_i path of very low cost). Given that, a simple averaging argument shows that in any optimal solution there is a node r so that at least $\sqrt{k}$ s_it_i-paths go via it. This implies the existence of a low density junction tree.[1] The question is how to find such a low density junction tree. The non-trivial challenge is to "match" each "source" s_i with the proper terminal t_i. A "naive" approach is guessing the root r and the number k' of sources in the junction-tree (which equals the number of terminals in it), and finding an in-branching with k' sources and an out-branching with k' terminals using [6]. This method fails because the sources in the in-branching and the terminals in the out-branching found may not match. In [7], this difficulty was overcome by increasing the number of nodes in the graph to $\Omega(n^{1/\varepsilon})$ and then using "density type" linear program that "forces" the sources to match the sinks. See a simpler application of a density type LP in [9] for undirected graphs.

We overcome the above difficulty of matching s_i and t_i by working with the metric completion graph of G, in which the cost of an edge between every two nodes is the cost of the shortest path between them. A *junction star-tree* is an in-star with leaves s_i entering a root r joined to an out-branching covering the respective t_i. We force the s_i, t_i to match by "attaching" each s_i as a child of t_i with a directed edge t_is_i of cost $c(s_ir)$. Thus, s_i becomes the terminal, and s_i belongs to the solution if and only if t_i does (see Section 4). We show that the metric completion graph of G always contains a junction star-tree of good density. Hence the problem is reduced to k'-DST problem; we still need to guess the root r and the number k' of pairs in the junction star-tree, but we do not need to use LP methods. Obviously, a drawback is that it is harder to prove the existence of a low density junction star-tree. See Section 4, in which the existence of a low density junction tree is proved.

Another disadvantage of the LP method used by [7] is that it is unable to deal with the k-DSF problem. The LP may connect an arbitrary number of pairs, possibly many more than k. The use of junction star-trees allows us to use the algorithm of [6] for k-DST (by solving the k'-DST problem, for all $k' \leq k$) instead of the LP method, which in turn allows us to control the number of pairs connected.

2 Preliminaries

2.1 The Greedy Algorithm

We use a known result about the performance of a *Greedy Algorithm* for the following type of problems:

Covering Problem
Instance: A ground-set E and non-negative functions ν, c on 2^E, given by an evaluation oracle.
Objective: Find an $F \subseteq E$ with $\nu(F) = 0$ and $c(F)$ minimized.

We call ν the *deficiency function* (it measures how far is F from being a feasible solution) and c the *cost function.*

Definition 2.1 *Let $F \subseteq E$ be a partial solution (partial cover) for an instance of* Covering Problem

[1]Technically speaking, a union of s_it_i-paths all going via r is not a junction tree as defined in [7]. However, is easy to see that it contains a junction tree which can be found in linear time.

and let $J \subseteq E$. Let $\rho(x)$ be a positive function, and let opt *be the optimal solution value for* $\mathsf{Covering\ Problem}$. *We say that $J \subseteq E$ obeys the $\rho(x)$-*Density Condition *if:*

$$\sigma_F(J) = \frac{c(J)}{\nu(F) - \nu(F \cup J)} \leq \mathsf{opt} \cdot \frac{\rho(\nu(F))}{\nu(F)} \tag{1}$$

The quantity $\sigma_F(J)$ in (1) is the *density of J* (w.r.t. F). The *Greedy Algorithm* starts with $F = \emptyset$ and iteratively adds to F a subset $J \subseteq E$ obeying (1). A set-function f on 2^E is *decreasing* if $f(F_2) \leq f(F_1)$ for any $F_1 \subseteq F_2 \subseteq E$, and *subadditive* if $f(F_1 \cup F_2) \leq f(F_1) + f(F_2)$ for any $F_1, F_2 \subseteq E$. The following statement is well known (e.g., see a slightly weaker version in [6]).

Theorem 2.1 *If ν is decreasing, c is subadditive, and $\rho(x)/x$ is a decreasing function, then the Greedy Algorithm computes a solution F with:*

$$c(F) \leq \mathsf{opt} \cdot \int_0^{\nu(\emptyset)} \frac{\rho(x)}{x} dx \ . \tag{2}$$

In our setting, the ground-set is the set E of edges of the graph. For every partial solution $F \subseteq E$, the deficiency $\nu(F)$ of F is the number of ordered pairs not connected by F. Formally, $\nu(F) = \max\{k - |\mathcal{D}(F)|, 0\}$, where $\mathcal{D}(F)$ denotes the set of pairs from $\mathcal{D}$ connected by F. Clearly, ν is decreasing, and c is subadditive.

2.2 Three simple reductions

We briefly describe three well known reductions to be used later that we can apply with negligible loss (in time complexity or approximation ratio) on a given k-DSF instance.

Reduction 1 *We may assume that the ratio between the maximum cost of an edge and non-zero minimum cost of an edge is $O(n^4)$; this reduction invokes a factor of 2 in the approximation ratio.*

This is achieved as follows. Let $\mathcal{I} = (G = (V, E), c, \mathcal{D}, k)$ be a k-DFS instance. Let $q = \max\{c(e) : e \in H^*\}$ be a maximum c-cost of an edge in some optimal solution H^* to $\mathcal{I}$. Let $E_q = \{e \in E : c(e) > qn^2\}$, and for $e \in E_q$ let $c_q(e) = 0$ if $c(e) \leq q/2n^2$ and $c_q(e) = c(e)$ otherwise. Note that the ratio between the maximum c_q-cost of an edge and non-zero minimum c_q-cost of an edge is at most $2n^4$. Let opt be the minimum c-cost of a solution to the instance $\mathcal{I}$, and let opt_q be the minimum c_q-cost of a solution to the instance $\mathcal{I}_q = (G_q = (V, E_q), c_q, \mathcal{D}, k)$. Note that:
(i) $\mathsf{opt}_q \leq \mathsf{opt}$, since H^* is a feasible solution to $\mathcal{I}_q$ and $c_q(H^*) \leq c(H^*)$.
(ii) If H is feasible for $\mathcal{I}_q$ then H is feasible for $\mathcal{I}$, and $c(H) \leq 2c_q(H)$, since $c(E \setminus E_q) \leq q/2 \leq \mathsf{opt}/2$.
Now let H be a ρ-approximate solution for $\mathcal{I}_q$, so $c_q(H) \leq \rho \cdot \mathsf{opt}_q$. Then H is also feasible for $\mathcal{I}$ and by (i) and (ii) above $c(H) \leq 2c_q(H) \leq 2\rho \cdot \mathsf{opt}_q \leq 2\rho \cdot \mathsf{opt}$. Hence H is a 2ρ-approximate solution for $\mathcal{I}$. Consequently, existence of a ρ-approximation algorithm for $\mathcal{I}_q$ implies a 2ρ-approximation algorithm for $\mathcal{I}$. Although q is not known, there are (at most) $|E| \leq n^2$ distinct choices of q, so we can try all of them while keeping the time polynomial, and output the best outcome.

Reduction 2 *Let S and T be the sets of first and second nodes in pairs of $\mathcal{D}$, respectively. We may assume that $S \cap T = \emptyset$ and that no edge enters S or leaves T.*

This is achieved by adding for every node v two new nodes s_v, t_v with edges $s_v v, vt_v$ of cost 0 each, and replacing every ordered pair $(u, v) \in \mathcal{D}$ by the pair (s_u, t_v). Note that in the obtained instance the total number of nodes is at most $3n$.

Reduction 3 *We may assume that G is transitively closed and that the costs are metric.*

This is achieved by applying metric completion.

Note that Reductions 1 and 2 can be applied simultaneously, and so are Reductions 2 and 3.

3 A sub-linear algorithm for DSF (Proof of Theorem 1.1)

In this section we describe an $O(n^{4/5+\varepsilon})$-approximation scheme for DSF. The $O(m^{2/3+\varepsilon})$-approximation scheme uses a similar method, and is shortly described in Section 3.4.

Our algorithm uses a parameter τ which is an estimate of the optimum. More precisely, the algorithm either returns a feasible solution H of cost $c(H) = O(n^{4/5+\varepsilon}) \cdot \tau$, or declares correctly that $\tau < \mathsf{opt}$. Note that the algorithm may return a solution of cost at most $O(n^{4/5+\varepsilon}) \cdot \tau$ also if $\tau < \mathsf{opt}$. Using binary search we find τ such that a solution of cost $O(n^{4/5+\varepsilon}) \cdot \tau$ is returned, while for $\tau/2$ the algorithm declares that $\tau/2 < \mathsf{opt}$. Then $\tau \leq 2\mathsf{opt}$, and hence $c(H_\tau) = O(n^{4/5+\varepsilon}) \cdot \mathsf{opt}$. To perform this binary search in strongly polynomial time we apply Reduction 1 from Section 2.2; this invokes a constant factor in the approximation ratio, which is negligible in our context. Assuming that the edges of cost 0 do not form a feasible solution, the initial range of the binary search is between $\tau_{\min}$ being the minimum non-zero cost of an edge and $\tau_{\max}$ being n^2 times the maximum cost of an edge. We have $\tau_{\max}/\tau_{\min} = O(n^6)$ by applying Reduction 1, hence the number of iterations in the binary search is $O(\log n^6) = O(\log n)$. Thus in the rest of the proof, we show existence of a polynomial time algorithm that given a DSF instance and a parameter τ either returns a feasible solution H of cost $c(H) \leq O(n^{4/5+\varepsilon}) \cdot \tau$, or declares correctly that $\tau < \mathsf{opt}$.

Given a DSF instance, assume that Reduction 2 from Section 2.2 is implemented. Recall that in DSF $k = |D|$. In what follows, let p, α, ℓ be parameters eventually set to:

$$p = 2 \ln k / n^{2/5}, \quad \alpha = n^{2/5}, \quad \ell = \tau / \alpha^2 \ .$$

Definition 3.1 *For a graph H, let $\mathsf{dist}_H(u, v)$ denote the minimum cost of a uv-path in H. A path P is* short *if $c(P) \leq \ell$, and P is* long *otherwise. For $(s, t) \in D$, let $U(s, t) = \{u \in V : \mathsf{dist}_G(s, u), \mathsf{dist}_G(u, t) \leq \ell\}$. A pair $(s, t) \in D$ is* good *if $|U(s, t)| \geq \alpha$, and (s, t) is* bad *otherwise.*

3.1 Connecting the good pairs

Lemma 3.1 *There exists a polynomial time algorithm that given a DSF instance finds an edge set F of cost $c(F) \leq 4pn^2\ell = \tilde{O}(n^{4/5}) \cdot \tau$ that connects all good pairs.*

Proof: Form a set $R \subseteq V$ by picking every node $v \in V$ into R with probability p. For a given good pair (s, t) we have:

$$\Pr[R \cap U(s, t) = \emptyset] \leq (1 - p)^\alpha \leq \frac{1}{k^2}$$

By the union bound, the probability that $R \cap U(s,t) \neq \emptyset$ for every good pair (s,t) is at least $1-1/k$. Notice that $|R|$ is a random variable with binomial distribution $B(n,p)$, thus $E(|R|)=pn$. Using the Chernoff Bound we get:

$$\Pr[|R| \leq 2pn] = \Pr[|R| \leq 2 \cdot E(|R|)] > 1 - e^{-pn/4} \ .$$

By definition, $pn/4 \geq \ln k$ thus we get that with high probability both $|R| \leq 2pn$ and $R \cap U(s,t) \neq \emptyset$ for every good pair (s,t) (this procedure can be derandomized using the method of conditional probabilities). We connect by a short path every node $s \in S$ to every node $v \in R$, if such path exists. Similarly, we connect by a short path every node $v \in R$ to every node $t \in T$, if such path exists. Let H be the sub-graph constructed by the above procedure. Clearly, H connects all good pairs. As $|S|+|T| \leq 2n$, we get that $c(H) \leq |R| \cdot 2n \cdot \ell \leq 2pn \cdot 2n \cdot \ell = 4pn^2\tau/\alpha^2 = \tau \cdot \tilde{O}\left(n^{4/5}\right)$. □

3.2 Connecting the bad pairs

After all good pairs are connected using the algorithm of Lemma 3.1, they are excluded from $\mathcal{D}$, and we are left with bad pairs only.

Lemma 3.2 *There exists a polynomial time algorithm that given a* DSF *instance without good pairs and a constant $\varepsilon > 0$, computes in polynomial time an edge set $J \subseteq E$ of density $O(n^{4/5+\varepsilon}) \cdot \tau/|\mathcal{D}|$, if $\tau \geq \mathsf{opt}$.*

In the rest of this subsection we prove Lemma 3.2. We compute two edge sets using two different algorithms, and choose among them the one with lower density. Let $L = \{(s,t) \in D : \mathsf{dist}_H(s,t) \geq \ell\}$. We will consider two cases: $|L| \geq |\mathcal{D}|/2$ and $|\mathcal{D} - L| > |\mathcal{D}|/2$.

The following two statements handle the case $|L| \geq |\mathcal{D}|/2$.

Lemma 3.3 ([7]) *The problem of finding a minimum density junction tree admits an $O(k^\varepsilon)$-approximation scheme.*

Proposition 3.4 *Any feasible solution H contains a junction tree J of density at most $\frac{c(H)}{\ell} \cdot \frac{c(H)}{|L|}$. In particular, if $\tau \geq c(H)$ then*

$$\sigma(J) \leq \frac{\tau}{\ell} \cdot \frac{\tau}{|L|} = n^{4/5} \cdot \frac{\tau}{|L|}$$

Hence if $|L| \geq |\mathcal{D}|/2$ and if $\tau \geq \mathsf{opt}$, then the algorithm of [7] *from Lemma 3.3 finds a junction tree J of density $O(n^{4/5+\varepsilon}) \cdot \tau/|\mathcal{D}|$.*

Proof: Let $\Pi(L)$ be a set of paths in H corresponding to the pairs in L. The sum of the costs of the paths in $\Pi(L)$ is at least $|L| \cdot \ell$. Since the paths of $\Pi(L)$ are in H, there must be an edge of H belonging to at least $|L| \cdot \ell/c(H)$ paths. This implies that there is a junction-tree in H connecting at least $|L| \cdot \ell/c(H)$ pairs from $\Pi(L)$. The density of this junction-tree is at most $c(H)/(|L| \cdot (\ell/c(H)))$, as claimed. the other statements are straightforward. □

Now suppose that $|L| < |\mathcal{D}|/2$, so $|\mathcal{D} - L| > |\mathcal{D}|/2$. Consider the following LP-relaxation (LP1) for the problem of connecting at least $k' \leq |\mathcal{D} - L|$ pairs from $\mathcal{D} = \{(s_1,t_1),\ldots,(s_k,t_k)\}$. Here k' can be any number $k' \leq |\mathcal{D} - L|$. Intuitively, (LP1) decides on a capacity x_e for every $e \in E$ and an amount y_i of s_it_i-flow. The sum of the y_i's is at least k'. The main restriction is that the flow has to be delivered on (simple) paths of cost $\leq \ell$. This is done as follows. Let $\Pi(i)$ be the set of (simple) s_it_i-paths in G of cost $\leq \ell$, and let $\Pi = \bigcup_i \Pi(i)$. For every i, decompose the final s_it_i-flow in the graph into flow paths. For every $P \in \Pi(i)$, the variable f_P is the amount of s_it_i-flow through P. The total s_it_i-flow equals the sum of the flows on the paths in $\Pi(i)$, namely, $y_i = \sum_{P\in\Pi(i)} f_P$. For every i and $e \in E$, the capacity constraint is $\sum_{\Pi(i)\ni P\ni e} f_P \leq x_e$; namely, the total s_it_i-flow through e is at most x_e. Note that it holds for every pair separately.

$$
\begin{array}{llrcll}
\text{(LP1)} & \min & \sum_{e\in E} c(e)x_e & & & \\
& \text{s.t.} & \sum_i y_i & \geq & k' & \\
& & \sum_{\Pi(i)\ni P\ni e} f_P & \leq & x_e & \forall\ i,\ e\in E \\
& & \sum_{P\in\Pi(i)} f_P & = & y_i & \forall\ i \\
& & y_i,\ x_e & \leq & 1 & \forall\ i,\ e\in E \\
& & y_i,\ f_P,\ x_e & \geq & 0 & \forall\ i,\ P\in\Pi,\ e\in E
\end{array}
$$

For solving this LP we use an idea that as far as we know was first introduced in [5]. This paper introduced the technique of using an approximation algorithm for the separating oracle needed for the dual. The corresponding dual LP is:

$$
\begin{array}{llrcll}
\text{(LP2)} & \max & \sum_{e\in E} a_e + \sum_i b_i - W\cdot k' & & & \\
& \text{s.t.} & \sum_i z_{i,e} + c(e) & \leq & a_e & \forall\ e\in E \\
& & b_i + w_i & \geq & W & \forall\ i \\
& & w_i & \leq & \sum_{e\in P} z_{i,e} & \forall\ i,\ P\in\Pi(i) \\
& & W,\ a_e,\ b_i,\ z_{i,e} & \geq & 0 & \forall\ i,\ e\in E
\end{array}
$$

Lemma 3.5 *For any $k' \leq |D-L|$ the optimal value of* (LP1) *is at most* opt*. Furthermore, a solution for* (LP1) *of value $\leq (1+\varepsilon)\cdot \mathsf{opt}$ can be found in polynomial time.*

Proof: The first statement is obvious, as (LP1) is a relaxation for the problem. We will show how to find an approximate solution in polynomial time. Although the number of variables in (LP1) might be exponential, any basic feasible solution to it has $O(|\mathcal{D}|\cdot m)$ non-zero variables. Now, if we had a polynomial time separation oracle for (LP2), we could compute an optimal solution to (LP1) (the non-zero entries) in polynomial time. The number of non-zero entries in such a computed solution is polynomial in $O(|\mathcal{D}|\cdot m)$. There is a polynomial number of dual constraints of all types, except for the constraints of the form $w_i \leq \sum_{e\in P} z_{i,e}$. Unfortunately, for these constraints, a polynomial time separation oracle may not exist, since the separation problem defined by a specific pair (s_i,t_i) is equivalent to the following problem, which is NP-hard, see [30]:

Restricted Shortest Path (RSP)
Instance: A directed graph $G=(V,E)$, transition times $\{z(e) : e\in E\}$, lengths $\{\ell(e) : e \in E\}$, a pair (s,t), and an integer Z.
Objective: Find a minimum length st-path P such that $\sum_{e\in P} z(e) \leq Z$.

RSP admits an FPTAS (see [21] and an improved result in [30]), therefore we can compute an approximate separation oracle, which for any $\varepsilon > 0$ checks whether there exists a path $P\in\Pi(i)$ so that $w_i \leq \sum_{e \in P} z_{i,e}/(1+\varepsilon)$. This implies that we can solve the following linear program in time polynomial in $1/\varepsilon$ and the size of the original DSF problem:

$$
\begin{array}{llrcll}
\text{(LP3)} & \max & \multicolumn{4}{l}{\sum_{e \in E} a_e + \sum_i b_i - W \cdot k'} \\
& \text{s.t.} & \sum_i z_{i,e} + c(e) & \leq & a_e & \forall\, e \in E \\
& & b_i + w_i & \geq & W & \forall\, i \\
& & w_i & \leq & \sum_{e \in P} z_{i,e}/(1+\varepsilon) & \forall\, i,\ P \in \Pi(i) \\
& & W,\ a_e,\ b_i,\ z_{i,e} & \geq & 0 & \forall\, i,\ e \in E
\end{array}
$$

Thus we can also solve the dual of (LP3), which is:

$$
\begin{array}{llrcll}
\text{(LP4)} & \min & \multicolumn{4}{l}{\sum_{e \in E} c(e) x_e} \\
& \text{s.t.} & \sum_i y_i & \geq & k' & \\
& & \sum_{\Pi(i) \ni P \ni e} f_P & \leq & x_e \cdot (1+\varepsilon) & \forall\, i,\ e \in E \\
& & \sum_{P \in \Pi(i)} f_P & = & y_i & \forall\, i \\
& & y_i,\ x_e & \leq & 1 & \forall\, i,\ e \in E \\
& & y_i,\ f_P,\ x_e & \geq & 0 & \forall\, i,\ P \in \Pi,\ e \in E
\end{array}
$$

Let $\mathsf{opt}(\varepsilon)$ denote the optimal value of (LP4). Clearly, $\mathsf{opt}(\varepsilon) \leq \mathsf{opt}$. Note that if $x(\varepsilon)$ is a feasible solution to (LP4), then by replacing the value of every variable x_e in $x(\varepsilon)$ by $\min\{1, x_e \cdot (1+\varepsilon)\}$ we get a new solution x which is a feasible solution to (LP1). The value of such x is at most $(1+\varepsilon) \cdot \mathsf{opt}(\varepsilon) \leq (1+\varepsilon) \cdot \mathsf{opt}$. □

Lemma 3.6 *Let x, y be a feasible solution to* (LP1) *and let β be any number obeying* $0 \leq \beta < k'/|\mathcal{D}|$. *Then at most $(|\mathcal{D}| - k')/(1-\beta)$ pairs in $\mathcal{D}$ have flow $y_i < \beta$. Thus, the number of pairs in $\mathcal{D}$ that have flow $y_i \geq \beta$ is at least:*

$$|\{i : y_i \geq \beta\}| \geq |\mathcal{D}| - \frac{|\mathcal{D}| - k'}{1-\beta} = \frac{k' - \beta|\mathcal{D}|}{1-\beta}$$

Proof: If more than $(|\mathcal{D}| - k')/(1-\beta)$ pairs in $\mathcal{D}$ have flow strictly less than β, then the sum of the flows between all pairs must be strictly less than:

$$\frac{|\mathcal{D}| - k'}{1-\beta} \cdot \beta + \left(|\mathcal{D}| - \frac{|\mathcal{D}| - k'}{1-\beta}\right) \cdot 1 = |\mathcal{D}| + (\beta - 1) \cdot \frac{|\mathcal{D}| - k'}{1-\beta} = k'$$

This is a contradiction, since in any feasible solution of (LP1), the sum of the flows between all pairs must be at least k'. □

Lemma 3.7 *Let (x, y) be any feasible solution to* (LP1) *and let β be any number obeying* $0 \leq \beta < 1$. *If $y_i \geq \beta$ for some i then $J = \{e \in E : x_e \geq 4\beta/\alpha^2\}$ contains an $s_i t_i$-path.*

Proof: We claim that $C \cap J \neq \emptyset$ for every $s_i t_i$-cut C. Suppose to the contrary that $C \cap J = \emptyset$ for some $s_i t_i$-cut C, namely, $x_e < 4\beta/\alpha^2$ for every $e \in C$. Thus $|C| \geq \alpha^2/4$, since $\sum_{e \in C} x_e \geq y_i \geq \beta$. Every edge $e = uv \in C$ that carries a positive amount of $s_i t_i$-flow belongs to some short $s_i t_i$-path, thus $u, v \in U(s_i, t_i)$. Consequently, $U(s_i, t_i)$ contains end nodes of at least $\alpha^2/4$ edges of the cut C, implying that $U(s_i, t_i)$ contains at least $2\sqrt{\alpha^2/4} = \alpha$ nodes. Thus (s_i, t_i) is a good pair, contradicting our assumption that all the pairs are bad. □

Corollary 3.8 *Assuming $k' \leq |\mathcal{D} - L|$, let (x,y) be any feasible solution to* (LP1) *found using Lemma 3.5. Then for any $0 \leq \beta < k'/|\mathcal{D}|$, the edge set $J = \{e \in E : x_e \geq 4\beta/\alpha^2\}$ has density at most:*

$$\frac{\alpha^2 \mathsf{opt} \cdot (1+\varepsilon)}{4\beta} \cdot \frac{(1-\beta)}{k' - \beta|\mathcal{D}|}$$

In particular, for $k' = |\mathcal{D}|/2 \leq |\mathcal{D} - L|$ and $\beta = 1/4$, the density of J is at most $3\alpha^2 \cdot \mathsf{opt} \cdot (1+\varepsilon)/\mathcal{D} = O(n^{4/5}) \cdot \mathsf{opt}/|\mathcal{D}|$.

Proof: Since $k' \leq |D - L|$, the value of (LP1) is at most $\mathsf{opt} \cdot (1+\varepsilon)$, by Lemma 3.5. Thus $c(J) \leq \mathsf{opt} \cdot (1+\varepsilon)/(4\beta/\alpha^2) = \mathsf{opt} \cdot (1+\varepsilon)\alpha^2/(4\beta)$. By Lemmas 3.6 and 3.7, $|\mathcal{D}(J)| \geq (k' - \beta|\mathcal{D}|)/(1-\beta)$. Thus:

$$\sigma(J) = \frac{c(J)}{|\mathcal{D}(J)|} \leq \frac{\mathsf{opt} \cdot (1+\varepsilon)\alpha^2/(4\beta)}{(k' - \beta|\mathcal{D}|)/(1-\beta)} = \frac{\alpha^2 \mathsf{opt} \cdot (1+\varepsilon)}{4\beta} \cdot \frac{(1-\beta)}{k' - \beta|\mathcal{D}|}$$

□

Proof of Lemma 3.2: We execute two algorithms to compute edge sets J', J'' and choose among them the one with the better density. The set J' is computed using the algorithm of Lemma 3.3. The set J'' is computed using the algorithm of Corollary 3.8 with parameters $k' = |\mathcal{D}|/2$ and $\beta = 1/4$. Now suppose that $\tau \geq \mathsf{opt}$. If $|L| \geq |\mathcal{D}|/2$ then the density of J' is $O(n^{4/5+\varepsilon}) \cdot \tau/|\mathcal{D}|$. Otherwise, since $|\mathcal{D} - L| \geq |\mathcal{D}|/2$, the density of J'' is $O(n^{4/5}) \cdot \mathsf{opt}/|\mathcal{D}| = O(n^{4/5}) \cdot \tau/|\mathcal{D}|$. In both cases, one of J', J'' has density $O(n^{4/5+\varepsilon}) \cdot \tau/|\mathcal{D}|$. □

3.3 Putting everything together

Recall that we need to show existence of a polynomial time algorithm that given a DSF instance and a parameter τ either returns a feasible solution H of cost $c(H) = O(n^{4/5+\varepsilon}) \cdot \tau$, or declares correctly that $\tau < \mathsf{opt}$. The solution H is computed by the following procedure.

```
1. Implement Reduction 2.
2. Find an edge set $F$ as in Lemma 3.1, and exclude all good pairs from $\mathcal{D}$.
3. While $F$ is not a feasible solution do:
        - Find an edge set $J$ as in Lemma 3.2;
        - $F \leftarrow F + J$;
        - $\mathcal{D} \leftarrow \mathcal{D} - \mathcal{D}(J)$.
   EndWhile
4. Return $H = (V, F)$.
```

The total cost of the edges added at Step 2 is $A \cdot n^{4/5+\varepsilon} \cdot \tau$ for some constant A (for any τ), by Lemma 3.1. Step 3 is essentially the Greedy Algorithm with $\rho = O(n^{4/5+\varepsilon})$, by Lemma 3.2. Thus by Theorem 2.1, if $\tau \geq \mathsf{opt}$ then the total cost of the edges added at Step 3 is $B \cdot n^{4/5+\varepsilon} \cdot \tau$, for an appropriate constant B. Consequently, either $c(H) \leq (A+B) \cdot n^{4/5+\varepsilon} \cdot \tau$ or $\tau < \mathsf{opt}$ must hold.

This concludes the proof of Theorem 1.1.

3.4 An $O(m^{2/3+\varepsilon})$-approximation scheme for DSF

In this subsection we show how to modify the above algorithm, to achieve an approximation ratio of $O(m^{2/3+\varepsilon})$. We assume that the input graph is connected, and therefore $n = O(m)$, otherwise we can process each connected component separately. Let us update the values of the parameters of the algorithm, as follows:

$$p = 2\ln k/m^{2/3}, \quad \alpha = m^{2/3}, \quad \ell = \tau/\alpha \ .$$

In order to reflect the fact that we are now interested in edges, rather than in nodes, we change the definition of $U(s,t)$ to:

Definition 3.2 *For each pair s, t of nodes, let $U(s,t)$ be the set of edges involved in any short path from s to t.*

Notice that we keep the definition of good and bad pairs – a pair (s,t) is a good pair if and only if $|U(s,t)| \geq \alpha$. The following Lemma is the equivalent of Lemma 3.1:

Lemma 3.9 *There exists a polynomial time algorithm that given an instance of* DSF *finds an edge set F of cost $c(F) \leq 4pnm\ell = \tilde{O}(n/m^{1/3}) \cdot \tau = \tilde{O}(m^{2/3}) \cdot \tau$ that connects all the good pairs.*

Proof: Form a set $R \subseteq E$ by picking every edge e such that $c(e) \leq \ell$ into R with probability p. Then, connect by a short path every source $s \in S$ to the beginning of every edge $e \in R$, if such a path exists. Similarly, connect by a short path the end of every edge $e \in R$ to every terminal $t \in T$, if such a path exists. Let H be the sub-graph constructed by the above procedure. The same arguments used in the proof of Lemma 3.1 show that with high probability H connects all the good pairs and $|R| \leq 2pm$. Notice that an edge e with $c(e) > \ell$ cannot be in $U(s,t)$ for any pair s,t), therefore, the total cost of the edges in R is at most $|R| \cdot \ell \leq 2pm\ell$. As $|S| + |T| \leq 2n$, the cost of the paths we add to H after R is determined is no more than $|R| \cdot 2n \cdot \ell \leq 4pnm\ell$. □

To complete the algorithm we need the following Lemma, which is equivalent to Lemma 3.2:

Lemma 3.10 *There exists a polynomial time algorithm that given a* DSF *instance without good pairs and a constant $\varepsilon > 0$, computes in polynomial time an edge set $J \subseteq E$ of density $O(m^{2/3+\varepsilon}) \cdot \tau/|\mathcal{D}|$, if $\tau \geq$ opt.*

Again, we make a distinction between the two cases: $|L| \geq |\mathcal{D}|/2$ and $|\mathcal{D} - L| \geq |\mathcal{D}|/2$, where L is defined as before. The next statement tells us that in the first case we can find a junction tree with the required density; the proof is exactly the same as the proof of Proposition 3.4, up to the changes we made to the parameters, and thus is omitted.

Proposition 3.11 *Any feasible solution H contains a junction tree J of density at most $\frac{c(H)}{\ell} \cdot \frac{c(H)}{|L|}$. In particular, if $\tau \geq c(H)$ then*

$$\sigma(J) \leq \frac{\tau}{\ell} \cdot \frac{\tau}{|L|} = m^{2/3} \cdot \frac{\tau}{|L|}$$

Hence if $|L| \geq |\mathcal{D}|/2$ and if $\tau \geq$ opt, then the algorithm from Lemma 3.3 of [7] *finds a junction tree J of density $O(m^{2/3+\varepsilon}) \cdot \tau/|\mathcal{D}|$.*

Now consider the case $|\mathcal{D} - L| \geq |\mathcal{D}|/2$. Looking back at (LP1), after the changes implied by the new set of short paths, we notice that Lemmas 3.5 and 3.6 still hold.

Lemma 3.12 (Equivalent of Lemma 3.7) *Let (x, y) be any feasible solution to* (LP1) *and let $0 < \beta \leq 1$. If $y_i \geq \beta$ for some i then $J = \{e \in E : x_e \geq \beta/\alpha\}$ contains an s_it_i-path.*

Proof: We claim that $C \cap J \neq \emptyset$ for every s_it_i-cut C. Suppose to the contrary that $C \cap J = \emptyset$ for some s_it_i-cut C, namely, $x_e < \beta/\alpha$ for every $e \in C$. Thus $|C|$ contains at least α edges of positive s_it_i-flow, since $\sum_{e \in C} x_e \geq y_i \geq \beta$. Every edge e carrying a positive amount of s_it_i-flow belongs to some short s_it_i-path, thus $e \in U(s_i, t_i)$. Consequently, $U(s_i, t_i)$ contains at least α edges. Thus (s_i, t_i) is a good pair, contradicting our assumption that all pairs are bad. □

Corollary 3.13 (Equivalent of Corollary 3.8) *Assuming $k' \leq |\mathcal{D} - L|$, let (x, y) be any feasible solution to* (LP1) *found using Lemma 3.5. Then for any $0 < \beta < k'/|\mathcal{D}|$, the edge set $J = \{e \in E : x_e \geq \beta/\alpha\}$ has density at most:*

$$\frac{\alpha \cdot \mathsf{opt} \cdot (1+\varepsilon)}{\beta} \cdot \frac{(1-\beta)}{k' - \beta|\mathcal{D}|}$$

In particular, for $k' = |\mathcal{D}|/2 \leq |\mathcal{D} - L|$ and $\beta = 1/4$, the density of J is at most $3\alpha \cdot \mathsf{opt} \cdot (1+\varepsilon)/(4\mathcal{D}) = O(m^{2/3}) \cdot \mathsf{opt}/|\mathcal{D}|$.

Proof: Since $k' \leq |\mathcal{D} - L|$, the value of (LP1) is at most $\mathsf{opt} \cdot (1+\varepsilon)$, by Lemma 3.5. Thus $c(J) \leq \mathsf{opt} \cdot (1+\varepsilon)/(\beta/\alpha) = \mathsf{opt} \cdot (1+\varepsilon)\alpha/\beta$. By Lemmas 3.6 and 3.12, $|\mathcal{D}(J)| \geq (k' - \beta|\mathcal{D}|)/(1-\beta)$. Thus:

$$\sigma(J) = \frac{c(J)}{|\mathcal{D}(J)|} \leq \frac{\mathsf{opt} \cdot (1+\varepsilon)\alpha/\beta}{(k' - \beta|\mathcal{D}|)/(1-\beta)} = \frac{\alpha \mathsf{opt} \cdot (1+\varepsilon)}{\beta} \cdot \frac{(1-\beta)}{k' - \beta|\mathcal{D}|}$$

□

Proof of Lemma 3.10: The proof is the same as that of Lemma 3.2, when Corollary 3.8 is replaced by Corollary 3.13, and Proposition 3.4 is replaced by Proposition 3.11. □

In conclusion, the approximation scheme we get is the same as the one described in Section 3.3, with two differences:

- The good pairs are connected in step 1 as in Lemma 3.9.
- The edge set J in step 2 is found as in Lemma 3.10.

Using a similar analysis to the one in Section 3.3, one can show that this scheme has an approximation ratio of $O(m^{2/3+\varepsilon})$. This completes the proof of Theorem 1.1.

4 Algorithm for k-DSF (Proof of Theorem 1.2)

This section is organized as follows: Section 4.1 defines the notation of "junction star-trees" and proves the "The Junction Star-Tree Theorem" which ensures the existence of a good density junction star-tree in the metric completion of any graph. Section 4.2 describes our algorithm for k-DSF which is based on this theorem.

4.1 Junction star-trees

Definition 4.1 *Let G be a directed graph with a set $\mathcal{D} = \{(s_1, t_1), \ldots, (s_{|\mathcal{D}|}, t_{|\mathcal{D}|})\}$ of ordered pairs; $S = \{s_1, \ldots, s_{|\mathcal{D}|}\}$ are* sources *and $T = \{t_1, \ldots, t_{|\mathcal{D}|}\}$ are* terminals. *A subgraph J of G is a* junction star-tree *if it is the union of an out-branching J_T rooted at r in $(G - S) \cup \{r\}$ and a star J_S in-going to r in $(G - T) \cup \{r\}$.*

See Section 1.3.2 for intuition about junction star-trees. Our main interest will be in the case where the leaves of J_S are the set of sources corresponding to the terminals of J_T. In the rest of this subsection we will prove the following statement, which is used in the algorithm presented in the next subsection, and which we believe is of independent interest. For the rest of the section let g be a fixed parameter (to be determined later).

Theorem 4.1 (The Junction Star-Tree Theorem) *Let $H = (V, E)$ be a graph with edge costs $\{c(e) : e \in E\}$ containing a set Π of k paths connecting a set $\mathcal{D} \subseteq S \times T$ of k node pairs, so that $S \cap T = \emptyset$ and so that no edge enters S or leaves T. If $c(P) \geq c(H)/g$ for every $P \in \Pi$ then the metric completion of H contains a junction star-tree J of density at most:*

$$\frac{c(J)}{|\mathcal{D}(J)|} \leq c(H) \cdot \left(\frac{g}{k} + \frac{2}{g}\right) \tag{3}$$

For every st-path $P \in \Pi$, the *truncated path* $\bar{P}$ of P is the maximal sv-sub path of P so that $c(\bar{P}) < c(H)/g$. The vertex v is the last vertex so that the distance between s and v on the path is at most $c(H)/g$. Let e_P be the edge (going out of v) in $P - \bar{P}$ leaving the last node of $\bar{P}$. Since $c(P) \geq c(H)/g$, then by the definition of $\bar{P}$: e_P always exists, and $c(\bar{P} + e_P) \geq c(H)/g$. Let $\bar{\Pi} = \{\bar{P} : P \in \Pi\}$.

Definition 4.2 *We say that two (not necessarily different) truncated paths in $\bar{\Pi}$* collide *if they have a node in common.*

Lemma 4.2 *There exists a partition $\bar{\mathcal{P}}_1, \ldots, \bar{\mathcal{P}}_q$ of $\bar{\Pi}$ into $q \leq g$ parts, and a set of pairwise non-colliding paths $\{\bar{P}_i \in \bar{\mathcal{P}}_i : i = 1, \ldots, q\}$, such that $\bar{P}_i$ collides with every path in $\bar{\mathcal{P}}_i$, $i = 1, \ldots, q$. Thus there is a path $\bar{P} \in \bar{\Pi}$ colliding with at least $\ell \geq k/g$ paths in $\bar{\Pi}$.*

Proof: We will construct the partition iteratively. Assuming that at the end of iteration $i - 1$ we constructed a sub partition $\{\bar{\mathcal{P}}_1, \ldots, \bar{\mathcal{P}}_{i-1}\}$ of $\bar{\Pi}$, which is not yet a partition of $\bar{\Pi}$, in iteration i perform two steps:

1. Pick a path $\bar{P}_i \in \bar{\Pi}$ which does not belong to any part yet, and place it in a new part $\bar{\mathcal{P}}_i$.
2. Add to $\bar{\mathcal{P}}_i$ every path that collides with $\bar{P}_i$ and does not belong to any other part yet.

By the construction, it is clear that eventually we will get a partition of $\bar{\Pi}$, such that $\bar{P}_i$ collides with every path in $\bar{\mathcal{P}}_i$ for every i, and that $\{\bar{P}_i\}_{i=1}^{q}$ are pairwise non-colliding. Hence we only need to show that the number q of parts is bounded by g. Let $e_i = e_{P_i}$, $i = 1, \ldots, q$. Note that since $\bar{P}_1, \ldots, \bar{P}_q$ are pairwise node disjoint, the paths $\bar{P}_1 + e_1, \ldots, \bar{P}_q + e_q$ are pairwise edge-disjoint. Thus their total cost is at most $c(H)$. Since $c(\bar{P}_i + e_i) \geq c(H)/g$ for every i, the statement follows. $\square$

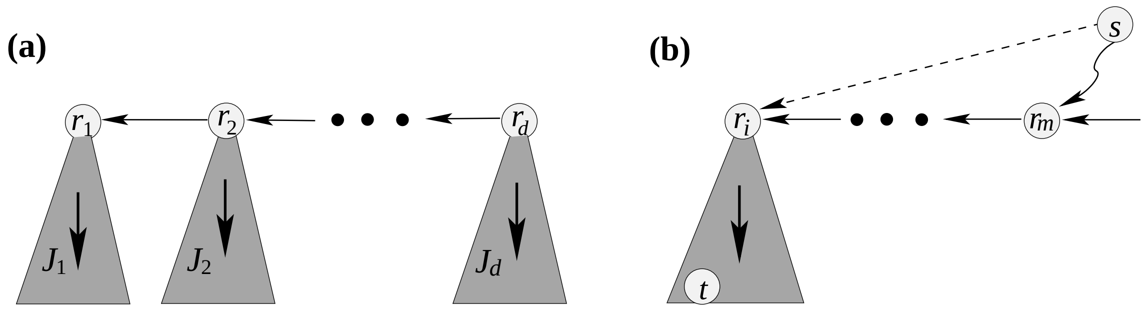

Figure 1: (a) The trees $J_1, \ldots, J_d$ hanged on the path $\bar{P}$; the trees are edge disjoint, but might not be node disjoint; some of the trees might consist of the root only. (b) Illustration of property 3 in Lemma 4.3 and the "shortcut" in the proof of Corollary 4.4.

Focus on a pair of a path $\bar{P} \in \bar{\Pi}$ and a set $\bar{\mathcal{P}} = \{\bar{P}_1, \ldots, \bar{P}_\ell\}$ of $\ell \geq k/g$ truncated paths colliding with $\bar{P}$ ($\bar{P} \in \bar{\mathcal{P}}$), whose existence is guaranteed by Lemma 4.2. Let $\mathcal{P} = \{P_1, \ldots, P_\ell\} \subseteq \Pi$ be the set of corresponding non-truncated paths. Let $\bar{S} = \{s_1, \ldots, s_\ell\}$ and $\bar{T} = \{t_1, \ldots, t_\ell\}$ be the sets of sources and terminals of the paths in $\mathcal{P}$, respectively. Let $r_1, \ldots, r_d$ be the sequence of nodes of $\bar{P}$ arranged in reverse order; r_d is the first node of $\bar{P}$, r_{d-1} is the second, and so on; the last node of $\bar{P}$ is r_1 (see Fig. 1(a)).

Lemma 4.3 *There exists in H a family $J_1, \ldots, J_d$ of pairwise edge disjoint trees so that (see Fig. 1(b)):*

1. *Every J_i is rooted at r_i, $i = 1, \ldots, d$.*
2. *Every $t \in \bar{T}$ belongs exactly one tree J_i, $1 \leq i \leq d$.*
3. *If $t \in \bar{T} \cap J_i$ and $(s,t) \in \mathcal{D}$, then there is $m \geq i$ so that r_m belongs to a path in $\bar{\mathcal{P}}$ starting at s.*

Proof: We construct the trees iteratively. J_1 is any inclusion minimal tree in H rooted at r_1 that contains the set T_1 of all the terminals in $\bar{T}$ that are reachable in H from r_1. J_2 is any inclusion minimal tree in H rooted at r_2 that contains the set T_2 of all the terminals in $\bar{T} - T_1$ that are reachable in H from r_2. And, in general, J_i is any inclusion minimal tree in H rooted at r_i that contains the set T_i of all the terminals in $\bar{T} - T_1 \cup \cdots \cup T_{i-1}$ that are reachable in H from r_i. By the construction, and since every path in $\bar{\mathcal{P}}$ collides with $\bar{P}$ and no edge leave the terminals, it is clear that the three properties given in the lemma hold. We explain why the trees $J_1, \ldots, J_d$ are pairwise edge disjoint. Otherwise, there are $1 \leq m < i \leq d$ so that J_m and J_i have an edge uv in common. By the minimality of J_i, there is a terminal $t \in \bar{T}_i$ reachable from v in J_i (possibly $v = t$). But then t is also reachable from r_m, hence, by the construction, t should have appeared in J_m and not in J_i, contradiction. $\square$

Using Lemma 4.3, we show that the metric completion of H contains a low density junction star-tree as a subgraph. For a subgraph J of H let $k(J) = |V(J) \cap \bar{T}|$ denote the number of terminals from $\bar{T}$ in J .

Corollary 4.4 *There exists a junction star-tree J in the metric completion of H, such that* (3) *holds.*

Proof: Let $J_1, \ldots, J_d$ be the decomposition of H into trees as in Lemma 4.3. We will extend these rooted trees to junction star-trees by adding for every st path in $\mathcal{P}$ an edge sr_i from s to the root r_i of the tree J_i which includes t (see Fig. 1(b), if $s = r_i$ we need not add this edge). The cost of each new edge is at most $2c(H)/g$, since it shortcuts a path that is obtained by joining two sub paths of truncated paths (recall that each truncated path has cost less than $c(H)/g$). Let $J_1^+, \ldots, J_d^+$ denote the resulting junction star-trees. Every junction star-tree connects all its sources to the corresponding terminals, and therefore $\sum_{i=1}^d k(J_i^+) = \ell$. On the other hand we can bound the sum of the costs of the junction star-trees as follows:

$$\sum_{i=1}^{d} c(J_i^+) < \sum_{i=1}^{d} c(J_i) + \ell \cdot \frac{2c(H)}{g} \leq c(H) + \ell \cdot \frac{2c(H)}{g}$$

The last inequality holds because $J_1, \ldots, J_d$ are subgraphs of H that are pairwise edge disjoint. Using an averaging argument we get that there must be a junction star-tree $J = J_i^+$ whose density is bounded by:

$$\frac{c(J)}{k(J)} \leq \frac{c(H) + \ell \cdot 2c(H)/g}{\ell} = \frac{c(H)}{\ell} + \frac{2c(H)}{g} \leq c(H) \cdot \left(\frac{g}{k} + \frac{2}{g}\right)$$

Where the last inequality holds because $\ell \geq k/g$. □

4.2 The algorithm

Given a k-DSF instance assume that Reduction 2 and 3 are implemented.

Lemma 4.5 *For any k-DSF instance (after applying Reductions 2, 3), there exists a junction star-tree J so that $c(J)/|\mathcal{D}(J)| \leq \mathsf{opt} \cdot \sqrt{8/k}$.*

Proof: Let $g = \sqrt{2k}$. If $c(P) \leq c(H)/g$ for some st-path P with $(s,t) \in \mathcal{D}$, then P is the required junction-star tree. Otherwise from Theorem 4.1, by choosing H, as an optimal solution of the k-DSF instance (after applying Reductions 2, 3), we get that H's metric completion contains a junction star-tree J of density $c(J)/|\mathcal{D}(J)| \leq \sqrt{8/k} \cdot c(H)$. □

Example: This example shows that the bound in Lemma 4.5 is tight up to a constant factor. Consider the graph in Fig. 2, where $\mathcal{D} = \{(s_i, t_j) : 1 \leq i, j \leq q\}$. Here $k = q^2$, and the lowest possible density of a junction star-tree is $(q+1)/q > 1$, while the density of the optimal solution (which is the entire graph) is $2q/q^2 = 2/q$.

Lemma 4.6 *Suppose that there exists an algorithm that given an instance of k-DSF finds an edge set J of density $\sigma \leq \mathsf{opt} \cdot \rho(k)/k$ and the set $\mathcal{D}(J)$ of demand pairs that J connects in $T'(n,k)$ time. Then the $\rho(x)$-Greedy Algorithm for k-DSF can be implemented in $O(kT'(n,k))$ time.*

Proof: We need to show how to find a low density edge set J for every instance $G, c, \mathcal{D}$ of k-DSF and every partial cover F. With that aim in mind, set $\mathcal{D}' \leftarrow \mathcal{D} - \mathcal{D}(F)$ to get an instance $G, c, \mathcal{D}'$ of $(k - |\mathcal{D}(F)|)$-DSF. Then use the given algorithm for finding an edge set J of density at most $\mathsf{opt}' \cdot \rho(k - |\mathcal{D}(F)|)/(k - |\mathcal{D}(F)|) = \mathsf{opt}' \cdot \rho(\nu(F))/\nu(F) \leq \mathsf{opt} \cdot \rho(\nu(F))/\nu(F)$, where opt and opt' denote the optimum solution values of the instances $G, c, \mathcal{D}$ and $G, c, \mathcal{D}'$, respectively. The number of

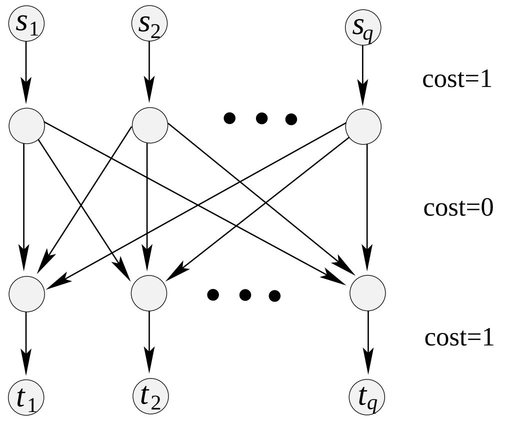

Figure 2: An example showing that the bound in Lemma 4.5 is tight.

iterations is at most k, since in each iteration at least one more demand pair is satisfied. Hence the time complexity is $O(kT'(n,k))$. $\square$

If we could find a low-density junction star-tree as in Lemma 4.5 in polynomial time, then we would obtain an $O(\sqrt{k})$-approximation algorithm for k-DSF, by Theorem 2.1 and Lemma 4.6. We will show how to find a junction star-tree of approximately optimal density using any approximation algorithm for k-DST; in particular, we can use the algorithm of [6].

Corollary 4.7 *If k-DST admits an α-approximation in $T(n,k)$ time then there exists an algorithm that given an instance of k-DSF finds a junction star-tree J satisfying $\sigma(J) \leq \mathsf{opt} \cdot \alpha \cdot \sqrt{8k}/k$ and $\mathcal{D}(J)$ in $O(nkT(2n+k,k))$ time.*

Proof: We may assume that we know the root r of some optimal density junction star-tree, as we may try every $r \in V$. For every demand pair $(s,t) \in D$, add a new node t' and the edge tt' of cost $c(sr)$ (if $s = r$ let the cost of the edge be 0). Let T' be the set of nodes added. For every $1 \leq k' \leq k$ apply the α-approximation algorithm on the obtained instance of k'-DST with root r and terminal set T'. From the solutions computed, output the one J' with minimum density. The junction star-tree J is obtained from J' by replacing every terminal t' of J' by the corresponding edge sr. It is easy to see that J' is as required, and that it is possible to calculate $\mathcal{D}(J')$ without increasing the time complexity. The graph on which we call the algorithm for k'-DST has $n + |T| + |S| + k$ nodes due to Reduction 2 and the addition of the nodes of T'. However, $|S|$ of these nodes are sources (into which no edge enters) and can be removed before the algorithm for k'-DST is called. The time complexity follows. $\square$

Combining Corollary 4.7 with the result of [6], Theorem 2.1, and Lemma 4.6, gives Theorem 1.2.

Remark: When using the algorithm of [6] for k-DST, the time complexity in Corollary 4.7 is in fact $O(nT(2n+k,k))$, since this algorithm approximates the minimum density augmentation tree in a k-DST instance within the same time bound as approximating k-DST.

5 Conclusions and open problems

We presented the first sub-linear, in terms of n, approximation algorithm for the DSF problem. Due to a reduction from LABEL-COVER$_{\mathsf{max}}$ [11], obtaining, say, a polylogarithmic approximation for DSF is unlikely. But the problem may admit a better polynomial approximation. We also presented a simple combinatorial $O(k^{1/2+\varepsilon})$-approximation scheme for k-DSF, which matches the best known LP-based algorithm of Chekuri et al. [7] for the less general problem DSF. Our result also (almost) matches the best know ratio in terms of k for the undirected version of the problem by Gupta et al. [18]. It is interesting to note that the situation is completely different in terms of n, as there is no known non-trivial approximation ratio for k-DSF in terms of n, while the undirected version admits an $O(\sqrt{n})$-approximation [18]. It is an open question whether the asymmetry between the parameters n and k can be reduced.

Almost every aspect of the more general Directed Steiner Network problem is still an open problem. No non-trivial approximation ratio is known for this problem, even for the simple case when the maximum requirement is 2. In contrast, the Undirected Steiner Network problem was studied extensively, and admits a 2-approximation algorithm due to Jain [23].

Note that the Directed Steiner Network problem can be trivially solved using min-cost flow techniques when there is only a single positive requirement pair. This fact can be used to achieve a k approximation for the Directed Steiner Network problem, where k is the number of positive requirement pairs: Simply solve independently for every positive requirement pair and combine the resulting graphs. A similar algorithm also extends to the more general problem of k-Directed Steiner Network, where we are only required to connect k positive requirement pairs. Again, we can solve the problem separately for each positive requirement pair and then combine the k cheapest resulting graphs.

We also note that on directed graphs, there is an approximation ratio preserving reduction between the edge-weighted and the node-weighted versions, but this is not so for undirected graphs. On undirected graphs, the best known ratio for the Node-Weighted Steiner Forest problem is $O(\log |U|)$ due to Klein and Ravi [26] and this is tight (up to a constant factor), where U is the set of nodes involved in a positive requirement pair. Recently, an $r_{\max} \cdot O(\ln |U|)$-approximation algorithm for the *undirected* Node-Weighted Steiner Network problem was presented in [32], where $r_{\max} = \max_{u,v \in V} r(u,v)$ is the largest requirement.

Acknowledgment: We thank an anonymous referee whose remarks considerably helped in improving the presentation of the paper.

References

[1] A. Agrawal, P. Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM J. Computing*, 24(3):440–456, 1995.

[2] S. Antonakopoulos. Approximating directed buy-at-bulk network design. In *WAOA*, pages 13–24, 2010.

[3] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *STOC*, pages 201–210, 2010.

[4] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. An improved LP-based approximation for Steiner tree. In *STOC*, pages 583–592, 2010.

[5] R. Carr and S. Vempala. Randomized metarounding. *Random Struct. Algorithms*, 20(3):343–352, 2002.

[6] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed Steiner problems. *Journal of Algorithms*, 33:73–91, 1999.

[7] C. Chekuri, G. Even, A. Gupta, and D. Segev. Set connectivity problems in undirected graphs and the directed Steiner network problem. In *SODA*, pages 532–541, 2008.

[8] C. Chekuri, G. Even, and G. Kortsarz. A greedy approximation algorithms for the group Steiner problems. *Discrete applied Math.*, 154(1):15–34, 2006.

[9] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM J. Comput.*, 39(5):1772–1798, 2010.

[10] C. Chekuri, S. Khanna, and J. Naor. A deterministic algorithm for the cost-distance problem. In *SODA*, pages 232–233, 2001.

[11] Y. Dodis and S. Khanna. Design networks with bounded pairwise distance. In *STOC*, pages 750–759, 1999.

[12] G. Even. *Recursive greedy methods,* Ch. 5 in *Approximation Algorithms and Metahueristics,* T. F. Gonzales ed. Chapman and Hall/CRC, 2007.

[13] U. Feige, G. Kortsarz, and D. Peleg. The dense k-Subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[14] J. Feldman and M. Ruhl. The directed Steiner network problem is tractable for a constant number of terminals. *SIAM J. Comput.*, 36(2):543–561, 2006.

[15] N. Garg. Saving an epsilon: a 2-approximation for the k-MST problem in graphs. In *STOC*, pages 396–402, 2005.

[16] N. Garg, N. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group Steiner tree problem. *J. Algorithms*, 66(1):66–84, 2000.

[17] M. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Computing*, 24(2):296–317, 1995.

[18] A. Gupta, M. T. Hajiaghayi, V. Nagarajan, and R. Ravi. Dial a ride from -forest. *ACM Transactions on Algorithms*, 6(2), 2010.

[19] M. T. Hajiaghayi and K. Jain. The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In *SODA*, pages 631–640, 2006.

[20] E. Halperin and R. Krauthgamer. Polylogarithmic inapproximability. In *STOC*, pages 585–594, 2003.

[21] R. Hassin. Approximation schemes for the restricted shortest path problem. *Math. Oper. Res.*, 17(1):36–42, 1992.

[22] C. H. Helvig, G. Robins, and A. Zelikovsky. Improved approximation scheme for the group Steiner problem. *Networks*, 37(1):8–20, 2001.

[23] K. Jain. Factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[24] S. Khuller. *Approximation algorithms for finding highly connected subgraphs,* Chapter 6 in *Approximation Algorithms for NP-hard problems,* D. S. Hochbaum Ed., pages 236–265. PWS, 1995.

[25] S. Khuller and L. Zosin. On directed Steiner trees. In *SODA*, pages 59–63, 2002.

[26] P. N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *Journal of Algorithms*, 19(1):104–115, 1995.

[27] G. Kortsarz and Z. Nutov. *Approximating minimum cost connectivity problems,* Ch. 58 in *Approximation Algorithms and Metahueristics,* T. F. Gonzales ed. Chapman & Hall/CRC, 2007.

[28] G. Kortsarz and Z. Nutov. Tight approximation for connectivity augmentation problems. *J. Comput. Syst. Sci.*, 74(5):662–670, 2008.

[29] G. Kortsarz and D. Peleg. Approximating the weight of shallow Steiner trees. *Discrete Applied Math.*, 93(2–3):265–285, 1999.

[30] D. H. Lorenz and D. Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, 28(5):213–219, 2001.

[31] A. Meyerson, K. Munagala, and S. A. Plotkin. Cost-distance: Two metric network design. *SIAM J. Comput.*, 38(4):1648–1659, 2008.

[32] Z. Nutov. Approximating Steiner networks with node-weights. *SIAM J. Comput.*, 39(7):3001–3022, 2010.

[33] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

[34] G. Robins and A. Zelikovsky. Improved Steiner tree approximation in graphs. In *SODA*, pages 770–779, 2000.

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

[36] A. Zelikovsky. A series of approximation algorithms for the acyclic directed Steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.