# Improved Approximation Algorithm for Steiner $k$-Forest with Nearly Uniform Weights

**Michael Dinitz[1], Guy Kortsarz*[2], and Zeev Nutov[3]**

1 **Johns Hopkins University**
mdinitz@cs.jhu.edu
2 **Rutgers University, Camden**
guyk@camden.rutgers.edu
2 **The Open University of Israel**
nutov@openu.ac.il

## Abstract

In the Steiner $k$-Forest problem we are given an edge weighted graph, a collection $D$ of node pairs, and an integer $k \leq |D|$. The goal is to find a minimum cost subgraph that connects at least $k$ pairs. The best known ratio for this problem is $\min\{O(\sqrt{n}), O(\sqrt{k})\}$ [8]. In [8] it is also shown that ratio $\rho$ for Steiner $k$-Forest implies ratio $O(\rho \cdot \log^2 n)$ for the Dial-a-Ride problem: given an edge weighted graph and a set of items with a source and a destination each, find a minimum length tour to move each object from its source to destination, but carrying at most $k$ objects at a time. The only other algorithm known for Dial-a-Ride, besides the one resulting from [8], has ratio $O(\sqrt{n})$ [4]. We obtain ratio $n^{0.448}$ for Steiner $k$-Forest and Dial-a-Ride with unit weights, breaking the $O(\sqrt{n})$ ratio barrier for this natural special case. We also show that if the maximum weight of an edge is $O(n^{\epsilon})$, then one can achieve ratio $O(n^{(1+\epsilon)\cdot 0.448})$, which is less than $\sqrt{n}$ if $\epsilon$ is small enough. To prove our main result we consider the following generalization of the Minimum $k$-Edge Subgraph (M$k$-ES) problem, which we call Min-Cost $\ell$-Edge-Profit Subgraph (MC$\ell$-EPS): Given a graph $G = (V, E)$ with edge-profits $p = \{p_e : e \in E\}$ and node-costs $c = \{c_v : v \in V\}$, and a lower profit bound $\ell$, find a minimum node-cost subgraph of $G$ of edge profit at least $\ell$. The M$k$-ES problem is a special case of MC$\ell$-EPS with unit node costs and unit edge profits. The currently best known ratio for M$k$-ES is $n^{3-2\sqrt{2}+\epsilon}$ (note that $3 - 2\sqrt{2} < 0.1716$) [5]. We extend this ratio to MC$\ell$-EPS for arbitrary node weights and edge profits that are polynomial in $n$, which may be of independent interest.

**1998 ACM Subject Classification** G.2.2 Graph Theory: Graph algorithms

**Keywords and phrases** $k$-Steiner Forest, Uniform weights, Densest $k$-Subgraph, Approximation algorithms

**Digital Object Identifier** 10.4230/LIPIcs.APPROX-RANDOM.2014.115

## 1 Introduction

We consider the following problem, originally introduced by Hajiaghayi and Jain [9]:

Steiner $k$-Forest
*Instance:* A graph $G = (V, E)$ with edge-weights $w = \{w_e : e \in E\}$, a collection $D$ of node pairs (called the *demand pairs*), and an integer $k \leq |D|$.
*Objective:* Find a minimum weight subgraph of $G$ that connects at least $k$ pairs from $D$.

* Partially supported by NSF award number 1218620.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 115–127

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Steiner $k$-Forest generalizes several well known problems, among them the following:

- For $k = |D|$ we get the Steiner Forest problem, which admits a 2-approximation algorithm [1].
- When the demand pairs form a star (namely, when there is a node that belongs to all the demand pairs) and $|D| = n - 1$ we get the $k$-MST problem, which admits a 2-approximation algorithm [7].
- When $k = |D|$ and the demand pairs form a star we get the Steiner Tree problem, which admits a $(\ln 4 + \epsilon)$-approximation scheme [3].
- When $G$ contains a spanning star and all edges have unit weights, we get the Minimum $k$-Edge Subgraph (M$k$-ES) problem: given a graph $G$ and an integer $k$, find a subgraph of $G$ with $k$ edges and minimum number of nodes (see [9] for the reduction details). This problem admits an $n^{3-2\sqrt{2}+\epsilon}$-approximation scheme [5] (note that $3 - 2\sqrt{2} < 0.1716$).

The best known ratio for Steiner $k$-Forest is $\min\{O(\sqrt{n}), O(\sqrt{k})\}$ [8], even for the case of unit weights. For $k = O(n)$ this ratio is essentially no better than the best known ratio $k^{1/2+\epsilon}$ for the directed version of the problem [6], even though undirected network design problems are usually much easier to approximate than their directed variants. We prove the following.

▶ **Theorem 1.** Steiner $k$-Forest *with unit weights admits an* $n^{\frac{1}{3}(7-4\sqrt{2})+\epsilon}$*-approximation scheme.*

Note that $\frac{1}{3}(7 - 4\sqrt{2}) < 0.44772$, so this is a polynomial improvement over the previous $O(\sqrt{n})$-approximation.

To prove Theorem 1 we consider the following generalization of the M$k$-ES problem, which we call Min-Cost $\ell$-Edge-Profit Subgraph, or MC$\ell$-EPS for short.

| Min-Cost $\ell$-Edge-Profit Subgraph (MC$\ell$-EPS) |
|---|
| *Instance:* A graph $G = (V, E)$ with edge-profits $p = \{p_e : e \in E\}$ and node-costs $c = \{c_v : v \in V\}$, and a profit lower bound $\ell$. |
| *Objective:* Find a minimum node-cost subgraph of $G$ of profit at least $\ell$. |

MC$\ell$-EPS with unit node costs and unit edge profits (and $\ell = k$) is the M$k$-ES problem. As was mentioned, the currently best known ratio for M$k$-ES is $n^{3-2\sqrt{2}+\epsilon}$ [5]. We extend this ratio to MC$\ell$-EPS by modifying the algorithm of [5] to handle weights and profits (essentially by adding an extra preprocessing step). Our extension can handle general node weights and profits bounded by a polynomial in $n$. Thus the node costs can be exponential in $n$ or beyond. When the edge profits are exponential in $n$ we can only give a bicriteria approximation: the algorithm will find a subgraph in which the total node weight is at most $n^{3-2\sqrt{2}+\epsilon}$ worse than the optimum, but it only covers edges with at least $\ell(1 - 1/poly(n))$ profit rather than the desired profit of $\ell$ (where $poly(n)$ is any polynomial in $n$). However, in the application to Steiner $k$-Forest edge profits are at most $n^2$, and hence we do not have to resort to the bicriteria approximation.

▶ **Theorem 2.** MC$\ell$-EPS *with edge profits that are at most polynomial in* $n$ *(but with arbitrary node costs) admits an* $n^{3-2\sqrt{2}+\epsilon}$*-approximation scheme.*

The following theorem establishes a relation between Steiner $k$-Forest and MC$\ell$-EPS, and it implies Theorem 1 by substituting the value of $\gamma = 3 - 2\sqrt{2} + \epsilon$ from Theorem 2.

▶ **Theorem 3.** *If* MC$\ell$-EPS *admits approximation ratio* $\rho = n^{\gamma}$, $0 \le \gamma \le 1/4$, *then* Steiner $k$-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$.

This theorem forms our core technical contribution, and most of the rest of the paper is devoted to proving it.

Another problem closely related to Steiner $k$-Forest is the following:

Dial-a-Ride
*Instance:* A graph $G = (V, E)$ with edge-lengths $w = \{w_e : e \in E\}$, a collection of items with a source and a destination each, and an integer $k$.
*Objective:* Move every item from its source to its destination using a vehicle that can carry at most $k$ items, minimizing total travel length.

Charikar and Raghavachari [4] showed that this problem admits ratio $O(\sqrt{n})$, while Gupta et al. [8] showed that ratio $\rho$ for Steiner $k$-Forest implies ratio $O(\rho \cdot \log^2 n)$ for Dial-a-Ride. Note that in Theorem 9 of [8], the Dial-a-Ride problem is approximated using the approximation for Steiner $k$-Forest as a *black box*. Thus if the Dial-a-Ride problem has uniform edge costs, the black box can be replaced by our approximation for Steiner $k$-Forest. This implies the same approximation (up to $polylog(n)$ factors) for uniform edge cost Dial-a-Ride.

▶ **Corollary 4.** *There is an* $n^{\frac{1}{3}(7-4\sqrt{2})+\epsilon}$*-approximation scheme for* Dial-a-Ride *with unit edge weights.*

We note that the unit weight versions of Steiner $k$-Forest and Dial-a-Ride are natural special cases to consider. In addition, the M$k$-ES problem is a special case of Steiner $k$-Forest with unit weights.

It is easy to see (and is a relatively standard observation) that the input to Steiner $k$-Forest can be replaced by a graph spanner with $O(\log n)$ stretch and $O(n)$ edges, while only paying an extra $O(\log n)$ in the approximation ratio [2]. Thus as long as the average edge weight in the spanner is at most $n^{\epsilon}$ we can obtain an $O(n^{(1+\epsilon)0.448})$ approximation ratio by replacing every edge by a path. This ratio is better than $\sqrt{n}$ if $\epsilon$ is small enough. This is true, for example, if the maximum cost of an edge in the original graph is at most $O(n^{\delta})$ for small enough $\delta$.

Our techniques may be the first step towards breaking the $O(\sqrt{n})$ barrier for both Steiner $k$-Forest and Dial-a-Ride with general edge weights. If this is not possible, then we get the somewhat rare case in which the unweighted version of a network design problem admits an approximation ratio that is better by a polynomial factor than what is possible in the weighted case.

## 2 A High Level Overview of the Main Ideas

We now turn to proving Theorem 3, which as discussed suffices to prove Theorem 1. Since it is rather involved, in this section we give a quick outline of the main ideas. We assume that we know the optimal solution value, which we denote by $\tau$. This is without loss of generality, since we can just run our algorithm on every possible $\tau$.

At a high level, our algorithm is a two-step process. In the first step we find a set of trees which together contain $k$ demand pairs, and in the second step we connect *all* pairs of terminals in different trees. In order to bound the cost of the second step, we will make sure that we use very few trees, and that the distance between any two trees is at most $\tau$. This will let us connect the trees very cheaply, since we can just arbitrarily connect them together and pay at most the number of trees times $\tau$, losing us at most the number of trees in our approximation ratio.

So the problem boils down to finding a small set of trees that together contain many demand pairs, and which are of pairwise distance at most $\tau$. We first build a set of candidate trees, and from them select a small set (possibly with some modifications to the trees). A cluster decomposition is a collection of clusters, and a cluster is a collection $\mathcal{T}$ of trees with some particular properties. We will guarantee that every terminal belongs to exactly one tree of exactly one cluster. Moreover, trees in the same cluster are node-disjoint (not just terminal-disjoint). Each tree has diameter at most $2d$ for some parameter $d$, and the distance between every two terminals in different trees of the same cluster is at least $2d$.

While the trees from the cluster decomposition cover the demand, there is no upper bound on the number of trees or on the cost of any particular tree. We divide the trees into light trees (those with few edges compared to $\tau$) and heavy trees (those with many edges). If we choose to take a light tree we can simply take the entire tree, since it is small by definition. But heavy trees have to be handled differently: we might choose to take part of a heavy tree, rather than the entire thing.

It turns out because of how we construct the clusters, we can prove that at most $\tau/d$ trees from any cluster can intersect the optimum solution, and that there are always two clusters which together contain a significant amount of demand pairs. This means that there is a way of covering at least $k$ demand pairs using few trees, since the optimum solution does. Thus we can use the algorithm we develop for MC$\ell$-EPS to select few trees which together contain at least $k$ demand pairs. This is fine if they are light trees, but if they are heavy trees we instead must select a subset of terminals in the tree to actually connect without using too many edges. Fortunately this can be handled simultaneously using our MC$\ell$-EPS algorithm.

## 3 Proof of Theorem 3

Fix some optimal solution $J$ and a set $D_J$ of $k$ demand pairs connected by $J$. We will use the following notation.

- $\tau = |J|$ is the optimum solution value, namely, the number of edges in $J$.
- $R_J$ is the union of the pairs in $D_J$, and $q = |R_J|$ is the number of nodes in $R_J$.
- Recall that $\rho = n^\gamma$ denotes the best known ratio for MC$\ell$-EPS.

In what follows, we may "guess" the right values of $\tau$ and $q$, by applying any of our algorithms for all possible values of $\tau = 1, \ldots, |E|$ and $q = 1, \ldots, n$, and among the edge sets computed return the best one. Note that $q = |R_J| \leq 2|J| \leq 2\tau$, since every node in $R_J$ is an endnode of some edge in $J$.

We first give some bounds on $q$ and $\tau$ by showing easy algorithms for special cases.

▶ **Lemma 5.** *For any $0 \leq \theta \leq 1/2$ the following holds: unless $n^{1/2-\theta} \leq q \leq 2\tau \leq n^{1/2+\theta}$,* Steiner $k$-Forest *with unit weights admits approximation ratio $O(n^{1/2-\theta})$.*

**Proof.** Any maximal forest of $G$ is a feasible solution that has at most $n-1$ edges, so if $\tau \geq n^{1/2+\theta}$ then simply returning a maximal forest guarantees an approximation ratio $(n-1)/n^{1/2+\theta} < n^{1/2-\theta}$. If $q < n^{1/2-\theta}$, then we claim that the ratio $O(n^{1/2-\theta})$ follows from the ratio $O(\sqrt{k})$ of [8]. This is since $k \leq q(q-1)/2$, and thus $\sqrt{k} \leq q < n^{1/2-\theta}$. ◀

▶ **Corollary 6.** *For any $0 \leq \gamma \leq 1/4$, the following holds: if $\tau\sqrt{q} > n^{1-\gamma}$, then* Steiner $k$-Forest *with unit weights admits approximation ratio $O\left(n^{1/3+2\gamma/3}\right)$.*

**Proof.** Let $\theta = \frac{1}{6} - \frac{2}{3}\gamma$. Since $\gamma \leq \frac{1}{4}$, $\theta \geq 0$. We claim that $\tau > n^{1/2+\theta}$, and then ratio $O\left(n^{1/2-\theta}\right) = O\left(n^{1/3+2\gamma/3}\right)$ follows from Lemma 5. To see this, note that if $\tau \leq n^{1/2+\theta}$ then also $q \leq n^{1/2+\theta}$, so $\tau\sqrt{q} \leq n^{1/2+\theta}n^{1/4+\theta/2} = n^{3/4+3\theta/2} = n^{1-\gamma}$, which is a contradiction. ◀

The next lemma forms the technical heart of our paper, and is proved in Section 4. It says that we can do step 1 from the overview: there is a way of finding a subgraph (i.e. the collection of trees discussed in Section 2) that is relatively cheap, contains few components, and contains a lot of demand pairs.

▶ **Lemma 7.** *There exists a polynomial time algorithm that when given a* Steiner $k$-Forest *instance and integers $1 \leq d, h \leq n/2$, computes a subgraph $G' = (V', E')$ of $G$ such that the following holds: $G'$ has $\tilde{O}(\rho\tau/d + n/h)$ connected components, $V'$ contains $\tilde{\Omega}(k)$ pairs from $D$, and $|E'| = \tilde{O}(\rho h\tau/d + \rho qd)$.*

We now want to give an algorithm that uses this lemma. We first need an easy lemma: an algorithm for Steiner Forest where we bound the total cost by the number of nodes and the maximum distance.

▶ **Lemma 8.** Steiner Forest *(with arbitrary weights) admits a polynomial time algorithm that computes a solution $F'$ of size at most $L(|R| - 1)$, where $L = \max_{uv \in D} \text{dist}_G(u, v)$ and $R$ is the union of the demand pairs.*

**Proof.** Let $(V, D')$ be a spanning forest of the demand graph $(V, D)$. The connected components of $(V, D')$ coincide with those of $(V, D)$. For every $\{u, v\} \in D'$ let $P_{uv}$ be the edge set of a shortest $uv$-path, and let $F'$ be the union of these edge sets. It is easy to see that the graph $(V, F)$ connects every pair in $D$, and clearly its weight is at most $L(|R| - 1)$. ◀

We can now give our algorithm: we simply connect each of the components we are given to each other by using the above Steiner Forest algorithm. We write this a little more formally as Algorithm 1.

**Algorithm 1:** PARTIAL-CONNECT$(G, D, \tau, h, d)$

1. Remove from $D$ every pair $u, v$ with $\text{dist}_G(u, v) > \tau$.
2. Compute a graph $G' = (V', E')$ using Lemma 7.
3. Contract every connected component of $G'$ into a single node and update the set of demand pairs accordingly. For the obtained instance of Steiner Forest, compute an edge set $F'$ as in Lemma 8.
4. **return** $E' \cup F'$

▶ **Theorem 9.** *Given integers $1 \leq d, h \leq n/2$, Algorithm 1 returns a graph that covers $\tilde{\Omega}(k)$ demand pairs and has size at most $\tau \cdot \tilde{O}(f(d, h))$, where $f(d, h) = \rho\tau/d + n/h + \rho h/d + \rho qd/\tau$.*

**Proof.** We know from Lemma 7 that the number of components of $G'$ is at most $\tilde{O}(\rho\tau/d + n/h)$, and since each component of $G'$ is contracted to a single terminal the total number of terminals in the Steiner Forest instance is also at most $\tilde{O}(\rho\tau/d + n/h)$. Thus by Lemma 8, we know that $|F'| \leq \tau \cdot \tilde{O}(\rho\tau/d + n/h)$. We also know from Lemma 7 that $|E'| \leq \tilde{O}(\rho h\tau/d + \rho qd)$ and that the algorithm covers $\tilde{\Omega}(k)$ demand pairs. This proves the theorem. ◀

The attentive reader will note that we only proved the ability to cover $\tilde{\Omega}(k)$ demand pairs, rather than $k$ pairs. This is a minor technicality, though, as the next lemma shows.

▶ **Lemma 10.** *Suppose that* Steiner $k$-Forest *admits a bicriteria approximation algorithm that returns a subgraph of weight* $\leq f \cdot \tau$ *that connects at least* $k/p$ *demand pairs, where* $1 < p < k$. *Then* Steiner $k$-Forest *admits an* $f \cdot \lceil \ln k / \ln \alpha \rceil$*-approximation algorithm, where* $\alpha = 1 + \frac{1}{p-1}$. *In particular, if* $k = n^{\epsilon}$ *for some* $\epsilon > 0$ *and* $p = \text{polylog}(n)$, *then* Steiner $k$-Forest *admits a* $\tilde{O}(f)$*-approximation algorithm.*

**Proof.** We run the bicriteria algorithm iteratively, as follows. Let $k_i$ denote the residual demand (the number of pairs we still need to connect) at the beginning of iteration $i$, where $k_1 = k$. While $k_i \geq 1$, we run the bicriteria algorithm, remove from $D$ the pairs connected in the current iteration, and set $k_{i+1} = k_i - p_i$, where $p_i$ is the number of pairs connected at iteration $i$. Clearly, at the beginning of each iteration $i$ there exists a solution to the residual problem (namely, a subgraph that connected $k_i$ pairs from the remaining pairs) of weight at most $\tau$, where $\tau$ is the optimal solution value to the original problem. Hence the weight of the bicriteria solution computed at each iteration is at most $f \cdot \tau$.

We have $k_i = k_{i-1} - p_{i-1} \leq k_{i-1} - (1 - 1/\alpha)k_{i-1} = k_{i-1}/\alpha$, hence $k_i \leq k/\alpha^i$. The least integer $i$ such that $\alpha^i \geq k$ is $i = \lceil \ln k / \ln \alpha \rceil$, and it bounds the number of iterations. Consequently, the overall weight of the solution computed is at most $f \cdot \lceil \ln k / \ln \alpha \rceil \cdot \tau$, as claimed.

The last statement follows from the observation that for $p = \text{polylog}(n)$ we have $\ln \alpha = \ln(1 + \frac{1}{p-1}) \approx \frac{1}{p-1}$ and hence $\ln k / \ln \alpha \approx p \ln k = \text{polylog}(n)$. ◀

We now instantiate some parameters to show that for certain ranges of values, our algorithm gives a good approximation ratio.

▶ **Lemma 11.** *If* $\tau\sqrt{q} \leq n/\rho$ *then* Steiner $k$-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(\left(\frac{\rho^2 nq}{\tau}\right)^{1/3}\right)$.

**Proof.** Let $f(d, h) = \rho\tau/d + n/h + \rho h/d + \rho q d/\tau$ be as in Theorem 9 and let

$$d = \left(\frac{n\tau^2}{\rho q^2}\right)^{1/3} \qquad \text{and} \qquad h = \left(\frac{n^2\tau}{\rho^2 q}\right)^{1/3} = d \cdot \left(\frac{nq}{\rho\tau}\right)^{1/3}.$$

Elementary computations show that

$$\frac{\rho\tau}{d} = \left(\frac{\rho^4 q^2 \tau}{n}\right)^{1/3} \qquad \text{and} \qquad \frac{n}{h} = \frac{\rho h}{d} = \frac{\rho q d}{\tau} = \left(\frac{\rho^2 nq}{\tau}\right)^{1/3}.$$

The statement follows from Theorem 9 and Lemma 10, since the condition $\tau\sqrt{q} \leq n/\rho$ implies $d, h \leq n/2$ and $\frac{\rho^4 q^2 \tau}{n} \leq \frac{\rho^2 nq}{\tau}$. ◀

▶ **Corollary 12.** *For any* $0 \leq \gamma \leq 1$, *the following holds: if* $\rho = n^{\gamma}$ *and* $\tau\sqrt{q} \leq n^{1-\gamma}$, *then* Steiner $k$-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$.

**Proof.** This follows from Lemma 11, since $\frac{\rho^2 nq}{\tau} = \frac{q}{\tau}\rho^2 n \leq 2\rho^2 n = 2n^{1+2\gamma}$. ◀

From Corollaries 12 and 6 it follows that Steiner $k$-Forest with unit weights admits approximation ratio $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$, as claimed in Theorem 3. It only remains to prove Lemma 7.

# 4 Proof of Lemma 7

As discussed in Section 2, we will prove Lemma 7 by first constructing a clustering and then selecting a carefully chosen subset of that clustering. We begin by defining a clustering and proving a few simple results about them. We will then show how to use an algorithm for MC$\ell$-EPS to prove Lemma 7.

## 4.1 Clustering

▶ **Definition 13.** A $(d, r)$-*cluster* of a subset $S$ of nodes in a graph $G$ is a collection $\mathcal{T}_S$ of node-disjoint rooted subtrees of $G$ of radius at most $r$ each, such that $dist_G(u, v) > d$ for any two nodes $u, v \in S$ that belong to distinct trees. A $(d, r)$-*cluster-decomposition* of $S$ is a collection of $(d, r)$-clusters $\{\mathcal{T}_A : A \in \mathcal{A}\}$ where $\mathcal{A}$ is a partition of $S$.

▶ **Lemma 14.** *There exists a polynomial time algorithm that given a graph $G = (V, E)$, a node subset $S \subseteq V$, and an integer $1 \leq d \leq n/2$ (called the* clustering parameter*), constructs a $(d, d(\lg |S| + 1))$-cluster $\mathcal{T}_A$ of a subset $A \subseteq S$ with $|A| \geq |S|/2$, where $\lg i = \log_2 i$.*

**Proof.** For a subtree $T$ of $G$ let $B_d(T) = \{v \in S \setminus T : \text{dist}_G(T, v) \leq d\}$ denote the set of nodes in $S \setminus T$ of distance at most $d$ from $T$. The algorithm is as follows.

**Algorithm 2:** CLUSTER-CONSTRUCT$(G, S, d)$

```
initialize 𝒯 ← ∅, A ← ∅
while S ≠ ∅ do
    Choose root s ∈ S and set T ← ({s}, ∅)
    while |B_d(T)| ≥ |S ∩ T| do
        EXPAND(T): For each v ∈ B_d(T), add to T the shortest path from T to v.
UPDATE(𝒯, S, A): Add T to 𝒯, move T ∩ S from S to A, and remove B_d(T) from S.
return 𝒯
```

The lines in the loop add nodes to the trees as long as the number of terminals in the "boundary" is at least equal to the number of terminals inside the tree. When this is no longer the case, the update line removes the boundary of the new tree from the graph.

Each time we expand $T$, the radius of $T$ increases by at most $d$ while $|T \cap S|$ is at least doubled. Thus the radius of $T$ is bounded by $d(\lg |S| + 1)$.

Note that at the update step, the set $B_d(T)$ of nodes within distance $d$ from $T$ is removed from $S$, and thus none of them will belong to $A$. This implies that at the end of the algorithm, $\text{dist}_G(u, v) > d$ for any two nodes $u, v \in A$ that belong to distinct trees. Note also that the number of nodes moved from $S$ to $A$ and included in $T$ is at least half the number of nodes removed from $S$ (since at this point $|B_d(T)| \leq |T \cap S|$). This implies that $|A| \geq |S|/2$.

It remains to prove that the trees in $\mathcal{T}$ are pairwise node-disjoint. Suppose to the contrary that there is $v \in V$ that belongs to two trees $T_1, T_2 \in \mathcal{T}$, where $T_2$ was constructed after $T_1$. Let $T_2'$ denote the tree stored in $T_2$ right before the expansion step when $v$ was added to $T_2$. When $v$ was added to $T_2'$, this was because there was a path of length $\leq d$ that goes through $v$ from $T_2'$ to some $t \in S$. In particular, $\text{dist}_G(v, t) \leq d$. Now let $T_1'$ denote the tree stored in $T_1$ right after the expansion step when $v$ was added to $T_1$. At this point, $t$ was not added to $T_1'$, hence we must have $\text{dist}_G(v, t) > d$. This is a contradiction. ◀

▶ **Corollary 15.** *There exists a polynomial time algorithm that given a graph $G = (V, E)$, a node subset $S \subseteq V$, and an integer $1 \leq d \leq n/2$, returns a $(d, d(\lg |S| + 1))$-cluster-decomposition of $S$ with at most $\lg |S| + 1$ clusters.*

**Proof.** We construct the clusters in the decomposition sequentially, using the algorithm from Lemma 14. After construction of each cluster $\mathcal{T}_A$ we remove from $S$ the corresponding set $A$ of nodes and add $A$ to $\mathcal{A}$. Clearly, at the end $\mathcal{A}$ is a partition of $S$. After each cluster construction the number of nodes in $S$ decreases by a factor of at least 2, hence $|\mathcal{A}| \leq \lg |S| + 1$. ◀

Lemma 14 and Corollary 15 extend to edge-weighted graphs by an elementary construction of replacing every edge $e$ of weight $w_e$ by a path of length $w_e$.

▶ **Lemma 16.** *Given a* Steiner $k$-Forest *instance, let $\{\mathcal{T}_A : A \in \mathcal{A}\}$ be a cluster-decomposition of the set $R$ of terminals as in Corollary 15 (so $|\mathcal{A}| \leq \lg |R| + 1$), and let $D'$ be an arbitrary set of demand edges. Then there exist $A, B \in \mathcal{A}$ (possibly $A = B$) such that $\Omega(|D'|/\lg^2 |D|)$ pairs in $D$ have one node in $A$ and the other node in $B$.*

**Proof.** Let $D'(A, B)$ denote the set of pairs in $D'$ with one node in $A$ and the other in $B$. The statement follows by an averaging argument from the observations that $\sum_{\{A,B\} \subseteq \mathcal{A}} |D'(A, B)| = |D'|$, and that $|\{\{A, B\} : \{A, B\} \subseteq \mathcal{A}\}| = |\mathcal{A}|(|\mathcal{A}| + 1)/2$. ◀

For simplicity of exposition, let us assume that we know the sets $A, B$ as in the above corollary (we can try all possible choices) and that $A \neq B$ (the analysis of the case $A = B$ is similar). Furthermore, by Lemma 16, we lose only a polylogarithmic factor by replacing $D$ by $D(A, B)$; hence we assume that $D = D(A, B)$ (and that an optimal solution connects $k$ pairs from $D$), and denote by $\mathcal{T}_A, \mathcal{T}_B$ the corresponding pair of clusters.

## 4.2 Choosing Trees

Now that we have two clusters which contain a lot of demand pairs, we want to find a cheap way of connecting much of it. Recall that $J$ denotes an optimal solution, $\tau = |J|$ is the number of edges in $J$, $D_J$ is a set of $k$ demand pairs connected by $J$, $R_J$ is the union of all pairs in $D_J$, and $q = |R_J|$. The two parameters $d$ and $h$ from Lemma 7 are related to the cluster decomposition, and have the following meaning:

- $d$ is the cluster decomposition parameter as in Corollary 15.
- $h$ is a threshold on tree size in a cluster; a tree $T$ is *heavy* if it has more that $h$ edges, and $T$ is *light* otherwise.

Let us consider the case that $\text{dist}_J(u, v) \geq 2d$ holds for at least half of the pairs in $D_J$. In this case, we can remove from $D$ all pairs $\{u, v\}$ with $\text{dist}_J(u, v) < 2d$, losing only a constant factor of 2 in the number of connected pairs. For simplicity of exposition, we will assume that $\text{dist}_J(u, v) \geq 2d$ holds for all pairs in $D_J$.

▶ **Lemma 17.** *Suppose that $\text{dist}_J(u, v) \geq 2d$ for every $\{u, v\} \in D_J$. Then at most $\tau/d$ trees in $\mathcal{T}_A$ contain a node from a pair in $D_J$.*

**Proof.** For every tree $T \in \mathcal{T}_A$ that intersects the optimum, fix some pair $\{u_T, v_T\} \in D_J$, where $u_T \in T$. Let $P_T$ be the set of the first $d$ nodes on the $u_T v_T$-path in $J$. Note that this path is completely unrelated to the trees in the cluster. Its a path in the optimum, and so we can not know what the path is. Nevertheless we get the following property: The sets $P_T$ are disjoint, since the distance between any two terminals that belong to distinct trees is

larger than $2d$. Thus every tree that intersects $J$ is associated with its own path of length $d$. Since the paths are edge disjoint and the number of total edges is at most $\tau$, there are at most $\tau/d$ trees that intersect $J$. ◀

Let us partition $D_J$ into three sets: $D_{LL}$ of $LL$-pairs with both nodes in light trees, $D_{HH}$ of $HH$-pairs with both nodes in heavy trees, and $D_{LH} = D_J \setminus (D_{LL} \cup D_{HH})$ of $LH$-pairs with one node in a light tree and the other in a heavy tree. At least one of these sets has size at least $k/3$. We execute three different algorithms, and choose the outcome of one of them. Intuitively, in each algorithm, we have the following three procedures.

1. Construct: This procedure constructs a MC$\ell$-EPS instance from the graph $(R, D)$.
2. Compute: This procedure computes a $\rho$-approximate solution to the obtained MC$\ell$-EPS instance, which determines a certain set $R'$ of terminals.
3. Connect: This procedure returns a graph $G' = (V', E')$ obtained by connecting some pairs of chosen terminals.

**Algorithm 3:** Heavy-Heavy$(G, D, \mathcal{T})$

1 Construct a MC$\ell$-EPS instance with $\ell = k/3$ by removing from the graph $(R, D)$ nodes that belong to light trees.
2 Compute a $\rho$-approximate solution $R'$ for the obtained MC$\ell$-EPS instance (in fact, here we get unit node-costs and unit edge-profits, so just a M$k$-ES instance with $k = \ell$).
3 Connect: $G' = (V', E')$ is the union of the shortest paths from each terminal in $R'$ to the root of the tree it belongs to.

**Algorithm 4:** Light-Light$(G, D, \mathcal{T})$

1 Construct a MC$\ell$-EPS instance with $\ell = k/3$ from the graph $H = (R, D)$ as follows.
- Remove nodes that belong to heavy trees.
- For every light tree, shrink its terminals into a single node.
- For every node pair $u, v$, replace the set $D_{uv}$ of parallel $uv$-edges by a single edge of profit $|D_{uv}|$.

2 Compute a $\rho$-approximate solution $R'$ for the obtained MC$\ell$-EPS instance (with unit node-costs and with edge profits).
3 Connect: $G' = (V', E')$ is the union of the light trees in $\mathcal{T}$ that correspond to $R'$.

**Algorithm 5:** Light-Heavy$(G, D, \mathcal{T})$

1 Construct a MC$\ell$-EPS instance with $\ell = k/3$ from the graph $(R, D)$ as follows: for every light tree, shrink its terminals into a single node of cost $dq/\tau$.
2 Compute a $\rho$-approximate solution $R'$ for the obtained MC$\ell$-EPS instance (with node-costs in $\{1, dq/\tau\}$ and unit edge-profits).
3 Connect: $G' = (V', E')$ is the union of the light trees that correspond to nodes in $R'$ and shortest paths from each terminal in $R'$ that belongs to a heavy tree to the root of the heavy tree it belongs to.

▶ **Lemma 18.** *Suppose that* $\mathrm{dist}_J(u, v) \geq 2d$ *holds for every* $\{u, v\} \in D_J$.

**(i)** *If* $|D_{HH}| \geq k/3$ *then Algorithm 3 computes a graph* $G' = (V', E')$ *with* $O(n/h)$ *connected components,* $|E'| = \tilde{O}(\rho q d)$, *and* $V'$ *contains* $\tilde{\Omega}(k)$ *pairs from* $D$.

**(ii)** *If* $|D_{LL}| \geq k/3$ *then Algorithm 4 computes a graph* $G' = (V', E')$ *with* $O(\rho\tau/d)$ *connected components,* $|E'| = O(\rho h\tau/d)$, *and* $V'$ *contains* $\tilde{\Omega}(k)$ *pairs from* $D$.

**(iii)** *If* $|D_{LH}| \geq k/3$ *then Algorithm 5 computes a graph* $G' = (V', E')$ *with* $O(\rho\tau/d + n/h)$ *connected components,* $|E'| = \tilde{O}(\rho h\tau/d + \rho q d)$, *and* $V'$ *contains* $\tilde{\Omega}(k)$ *pairs from* $D$.

**Proof.** Suppose that $|D_{HH}| \geq k/3$. The MC$\ell$-EPS instance in Algorithm 3 has a feasible solution of size at most $q$. Hence a $\rho$-approximate solution $R'$ has size at most $|R'| \leq \rho q$. We connect each terminal in $R'$ by a path of length $O(d \lg n)$, hence $|E'| = \tilde{O}(\rho q d)$. The heavy trees in each of $\mathcal{T}_A, \mathcal{T}_B$ are node disjoint, and each of them has at least $h$ nodes. Thus $G'$ has at most $2n/h$ connected components.

Suppose that $|D_{LL}| \geq k/3$. By Lemma 17, the MC$\ell$-EPS instance in Algorithm 4 has a feasible solution of size at most $\tau/d$. Hence a $\rho$-approximate solution $R'$ has size $|R'| \leq \rho\tau/d$, which bounds the number of trees from $\mathcal{T}$ and connected components that we include in $G'$. As the number of edges in each tree is at most $h$, $|E'| \leq \rho h\tau/d$.

Suppose that $|D_{LH}| \geq k/3$. Recall that in this case, in the obtained MC$\ell$-EPS instance, the cost of each node that corresponds to a light tree is $d \cdot q/\tau$, while the other nodes have unit costs and their number is at most $q$. By Lemma 17, at most $\tau/d$ trees contain a node from a pair in $D_J$. Hence the obtained MC$\ell$-EPS instance admits a solution of node cost $(\tau/d) \cdot d \cdot q/\tau = O(q)$. The returned $\rho$-approximate solution $R'$ has node cost $O(\rho q)$, and thus $O(\rho q)$ nodes. The number of light trees obtained is therefore bounded by $\frac{O(\rho q)}{d \cdot q/\tau} = \frac{\tau \cdot \rho}{d}$, which coincides with our bound on the number light trees returned in the $D_{LL}$ case. The number of connected components of heavy terminals is $O(\rho \cdot q)$ which coincides with our bound in the $D_{HH}$ case. This proves the claim. ◀

This completes the analysis of the case that $\text{dist}_J(u, v) \geq 2d$ holds for at least half of the pairs in $D_J$.

Now let us consider the case when $\text{dist}_J(u, v) < 2d$ holds for at least half of the pairs in $D_J$. This is the case that most pairs can be connected by a relatively short path. In this case, we remove from $D$ all pairs $\{u, v\}$ with $\text{dist}_J(u, v) \geq 2d$ in the graph. The algorithm in this case is essentially identical to Algorithm 3 as in case (i) of Lemma 18 (the $|D_{HH}| \geq k/3$ case). We construct a MC$\ell$-EPS instance as in Algorithm 3 with $m = k/2$. The obtained MC$\ell$-EPS instance admits a solution with $q$ nodes, and the returned $\rho$-approximate solution $R'$ has $O(\rho q)$ nodes. The same analysis as in case (i) of Lemma 18 finishes the proof of this case. Alternatively, we may also argue that since $\text{dist}_J(u, v) < 2d$ for all pairs in $D$, then by Lemma 8, we can connect $k/2$ pairs by cost $O(\rho q d)$, a term that already appears in Lemma 18(iii). Summarizing, the case when $\text{dist}_J(u, v) < 2d$ holds for at least half of the pairs in $D_J$ appears in the analysis of Lemma 18(iii), and thus does not change the approximation ratio.

This ends the proof of Lemma 7, and thus proves Theorem 1.

## 5 Proof of Theorem 2

We first make the node costs bounded by a polynomial in $n$. We remove nodes of cost more than $\tau$ and zero the edges of cost at most $\tau/n^2$. The cost we ignore due to the zeroing of the node costs is less than $\tau$ and is negligible in our context. Then we divide all the weights by the minimum weight and round the value down. Note that the cost of any edge over the cost of the minimum weight is at least 1. Hence the rounding down loses a negligible factor of 2: the worse case is that we may round a number that is at most 2 to 1. If the profits are exponential in $n$ or larger, we give a bicriteria approximation in which we have the same ratio but we cover only $\ell - \ell/poly(n)$ profit where $poly(n)$ is an arbitrary polynomial function of $n$. Thus our generalization of [5] is really for the case when node weights are arbitrary and edge profits are polynomial in $n$.

We call an instance of MC$\ell$-EPS *simple* if all the edge-profits are the same and there are at most two distinct node costs (say $c_1$ and $c_2$) such that every edge has exactly one endpoint of each cost (note that it might be the case that $c_1 = c_2$).

▶ **Lemma 19.** *If* MC$\ell$-EPS *admits an $f$-approximation algorithm on simple instances, then* MC$\ell$-EPS *admits a bicriteria approximation algorithm that returns a graph of node-cost $O(f)$ times the optimal and edge-profit $\Omega(\ell/\log^3 n)$.*

**Proof.** Let $\langle G=(V,E), c, p, \ell\rangle$ be an instance of MC$\ell$-EPS. Recall that we may assume that the node costs are polynomial in $n$ because of the reduction described above. Also, by assumption, the edge profits are bounded by a polynomial in $n$.

Partition $E$ into $O(\log n)$ sets $E_h=\{e\in E: 2^h \le p_e < 2^{h+1}\}$. Each $e\in E_h$ is given profit $2^h$. Partition the nodes similarly: $V_i=\{v\in E: 2^h \le c_v < 2^{h+1}\}$. according to powers of 2. Let $E_{ijh}$ be the set of edges in $E_h$ with one end in $V_i$ and the other in $V_j$. The edge sets $E_{ijh}$ partition $E$, and there are $O(\log^3 n)$ such sets. Each graph $G_{ijh}=(V_i\cup V_j, E_{ijh})$ gives a simple instance of MC$\ell$-EPS, and one of them contains $\Omega(\ell/\log^3 n)$ profit of the optimum. We run the algorithm for simple instances on each graph $G_{ijh}$ with $\Omega(\ell/\log^3 n)$ instead of $\ell$, and return the one of minimum node cost. The returned subgraph has node cost $O(f)$ times the optimal and $\Omega(\ell/\log^3 n)$ edge-profit, as required. ◀

By the same argument as in Lemma 10 we have the following.

▶ **Lemma 20.** *Suppose that* MC$\ell$-EPS *admits a bicriteria approximation algorithm that returns a graph of node-cost $f$ times the optimal and edge-profit at least $(1-1/\alpha)\cdot\ell$, where $1<\alpha<\ell$. Then* MC$\ell$-EPS *admits an $f\cdot\lceil \ln \ell/\ln\alpha\rceil$-approximation algorithm.*

From Lemmas 19 and 20 we have the following.

▶ **Corollary 21.** *Suppose that* MC$\ell$-EPS *on simple instances admits a bicriteria approximation algorithm that returns a graph of node-cost $f$ times the optimal and edge-profit $\tilde{\Omega}(\ell)$. Then* MC$\ell$-EPS *admits a $\tilde{O}(f)$-approximation algorithm.*

In the rest of this section we prove the following statement, which together with Corollary 21 implies Theorem 2.

▶ **Lemma 22.** MC$\ell$-EPS *on simple instances admits a bicriteria approximation algorithm that returns a graph of node-cost $f$ times the optimal and edge-weight $\tilde{\Omega}(\ell)$, where $f=\tilde{O}\left(n^{3-2\sqrt{2}+\epsilon}\right)$ for arbitrarily small constant $\epsilon>0$.*

We need some definitions and results from [5].

▶ **Definition 23** ([5])**.** A bipartite graph $G=(V_1\cup V_2, E)$ is called $(n_1,d_1,n_2,d_2)$-*nearly regular* if for every $i=1,2$ we have $|V_i|=n_i$ and the following condition on the degrees holds:

$$d_i \ge \max_{v\in V_i} d(v) \ge \min_{v\in V_i} d(v) = \Omega(d_i/\log n).$$

▶ **Lemma 24** ([5])**.** *Any graph $H=(V,E)$ contains an $(n_1,d_1,n_2,d_2)$-nearly regular subgraph with $\Omega(|E|/\log^2 n)$ edges, for some $n_1,d_1,n_2,d_2$.*

A key step in [5] was the following lemma:

▶ **Lemma 25** ([5])**.** *For any $\epsilon>0$ there exists a randomized polynomial time algorithm that given a bipartite graph $G$ on $n$ nodes that contains an $(n_1,d_1,n_2,d_2)$-nearly regular subgraph, returns a subgraph $G'=(V',E')$ of $G$ such that $|V'|\le f\cdot(n_1+n_2)$ (with probability 1) and $\mathbf{E}[|E'|]=\tilde{\Omega}(n_1d_1)$, where $f=n^{3-2\sqrt{2}+\epsilon}$.*

We prove the following refinement of Lemma 25, which gives a more "balanced" guarantee.

▶ **Lemma 26.** *For any $\epsilon > 0$ there exists a randomized polynomial time algorithm that given a bipartite graph $G$ on $n$ nodes that contains an $(n_1, d_1, n_2, d_2)$-nearly regular subgraph, returns a subgraph $G' = (V', E')$ of $G$ such that $|V' \cap V_1| \leq f n_1$ and $|V' \cap V_2| \leq f n_2$ (with high probability) and $\mathbf{E}[|E'|] = \tilde{\Omega}(n_1 d_1)$, where $f = n^{3-2\sqrt{2}+\epsilon}$.*

**Proof.** We will assume that $n_1 \geq n_2$; otherwise we just switch indices. Note that if $n_1 \leq 2n_2$, then the algorithm from Lemma 25 produces a subgraph that satisfies the new stronger requirement on the chosen nodes. So suppose that $n_1 > 2n_2$. For simplicity, let us also assume that $p = n_1/n_2$ is an integer.

Let $\hat{G} = (V_1 \cup \hat{V}_2, \hat{E})$, where $\hat{V}_2$ consists of $p$ copies of $V_2$ and $\hat{E}$ is obtained by putting between $V_1$ and each copy of $V_2$ a copy of $E$. For a subgraph $G' = (V_1' \cup V_2', E')$ of $G$ let $\hat{G}' = (V_1' \cup \hat{V}_2', \hat{E}')$ denote the corresponding subgraph of $\hat{G}$, i.e. where between each copy of $V_1'$ and $V_2'$ we include a copy of $E'$. Note that $|\hat{V}_2'| = p|V_2'|$, that $d_{\hat{G}'}(v) = p d_{G'}(v)$ if $v \in V_1'$, and that if $\hat{v} \in \hat{V}_2$ is a copy of $v \in V_2$ then $d_{\hat{G}'}(\hat{v}) = d_{G'}(v)$. This implies that if $G'$ is $(n_1, d_1, n_2, d_2)$-nearly regular then $\hat{G}'$ is $(n_1, d_2, n_1, d_2)$-nearly regular.

We run the algorithm from Lemma 25 on the instance $\langle \hat{G}, (n_1, d_2, n_1, d_2)\rangle$ independently $\tilde{O}(n^2)$ times, and among the subgraphs computed take one $\hat{G}' = (V_1' \cup \hat{V}_2', \hat{E}')$ with maximum number of edges. For each $v \in V_2$, let $T_v$ denote the number of copies of $v$ in $\hat{V}_2'$. We build $V_2'$ by sampling each $v \in \hat{V}_2'$ independently with probability $T_v/p$. Let $E'$ be the set of edges between $V_1'$ and $V_2'$. We will return the graph $G' = (V_1' \cup V_2', E')$.

We now prove the bounds on the sizes of $V_1'$, $V_2'$, and $E'$. Since we run the algorithm as in Lemma 25, $|V_1'| \leq 2f n_1$ and $\sum_{v \in V_2} T_v \leq |\hat{V}_2'| \leq 2f n_1$. By linearity of expectations, we get that the expected size of $V_2'$ is at most $\frac{n_2}{n_1} \sum_{v \in V_2} T_v \leq 2f n_2$. Since each node in $V_2$ was chosen independently, a simple Chernoff bound implies that $|V_2'| = \tilde{O}(f n_2)$ with high probability.

To bound $|E'|$, note that a Chernoff bound implies that with high probability $|\hat{E}'| = \tilde{\Omega}(n_1 d_2)$ (since we ran Lemma 25 a polynomial number of times and took the best, and each run was independent). An edge $uv \in E$ with $u \in V_1'$ is included in our subgraph with probability $T_v/p = T_v n_2/n_1$. Thus

$$\begin{aligned}\mathbf{E}[|E'|] &= \sum_{u \in V_1'} \sum_{v \in V_1' : uv \in E} T_v/p = \frac{n_2}{n_1} \sum_{u \in V_1'} \sum_{v \in V_2' : uv \in E} T_v \\ &= \frac{n_2}{n_1} \cdot \tilde{\Omega}(n_1 d_2) = \tilde{\Omega}(n_2 d_2) = \tilde{\Omega}(n_1 d_1),\end{aligned}$$

proving the lemma. ◀

Now we finish the proof of Lemma 22. Let $\langle G = (V_1 \cup V_2, E), m, (c_1, c_2)\rangle$ be a simple MC$\ell$-EPS instance. Let $G^* = (V_0^* \cup V_1^*, E^*)$ be an optimal subgraph. Applying Lemma 24 to $G^*$ implies that there exist values of $n_1, d_1, n_2, d_2$ such that there is a $(n_1, d_1, n_2, d_2)$-nearly regular subgraph of $G$ of cost at most $c(V^*) = c_1|V_1^*| + c_2|V_2^*|$ that contains at least $\tilde{\Omega}(\ell) = \tilde{\Omega}|E^*|$ edges (note that up to polylogs $n_1 d_1 = n_2 d_2 = \ell = |E^*|$). So when we run the algorithm from Lemma 26, we get a graph $G' = (V_1' \cup V_2', E')$ with the properties that with high probability $|V_1'| = \tilde{O}(f n_1)$ and $|V_2'| = \tilde{O}(f n_2)$ and in expectation $|E'| = \tilde{\Omega}(n_1 d_1) = \tilde{\Omega}(\ell)$. The node-cost of this subgraph is $\tilde{O}(f n_1 c_1 + f n_2 c_2) = \tilde{O}(f) \cdot c(V^*)$. This proves Lemma 22, and thus also the proof of Theorem 2 is complete.

## 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** I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

**3** Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. An improved LP-based approximation for Steiner tree. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, STOC'10, pages 583–592, 2010.

**4** M. Charikar and B. Raghavachari. The finite capacity dial-a-ride problem. In *FOCS*, pages 458–467, 1998.

**5** E. Chlamtac, M. Dinitz, and R. Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *FOCS*, pages 758–767, 2012.

**6** M. Feldman, G. Kortsarz, and Z. Nutov. Improved approximation algorithms for directed Steiner forest. *J. Comput. Syst. Sci.*, 78(1):279–292, 2012.

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

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

**9** 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.